Aguila Documentation

Release trunk

Kor de Jong

March 22, 2011

CONTENTS

CHAPTER

ONE

INTRODUCTION

Aguila is visualisation software for spatio-temporal data. It can be used to visualise temporal, spatial and/or uncertain data and contains features to aid exploratory data analysis.

Aguila is the primary visualisation tool of the PCRaster environmental modelling software, but since it can read data in various *formats*, it can be used in combination with other modelling and GIS software.

These pages will get you started using Aguila as quickly as possible:

- *Installation*
- *Quick start*

Subsequent pages describe types of visualisations (views) that can be used, the dialogs, and the types of data that can be visualised.

- *Views*
- *Dialogs*
- *Data*

CHAPTER

TWO

INSTALLATION

Note: The information in this section is related to the stand-alone binary Aguila distribution and is not valid when Aguila is distributed as part of a PCRaster distribution. The source code and information about building Aguila can be obtained from the PCRaster Open Source Tools project website on SourceForge.

Aguila installation packages can be found at the files section of the SourceForge project page.

2.1 Microsoft Windows

The software is distributed as a Windows installer. Install Aguila by executing the installer. After installation there is an option to uninstall Aguila again.

The following directories are created within the installation directory:

```
bin    # Contains the executable and the required shared libraries.
demos  # Contains demo data.
doc    # Contains this manual, in pdf format.
share  # Contains support files.
```

When you install Aguila as a user with Administrator rights, the search path for executables is automatically updated to include the path to the Aguila binary. Otherwise, you need to update the search path yourself:

1. Right-click on `My Computer` and select `Properties`.
2. Go to `Advanced` tab and select `Environment Variables`.
3. Edit the `PATH` variable from the `User variables for user` list, or add it if it not already exists.
4. Insert the path to the Aguila binary into the edit field (separate paths using a semicolon).
5. Open a new command prompt which will have the new environment setting.

2.2 GNU Linux

Warning: The binary distribution of Aguila for Linux is created on the PCRaster development machines. This version of Aguila will run on systems with similar properties (currently Debian testing on i686 and x86_64). On other machines this version may not run [a]. Here is a list of Linux platforms for which Aguila is reported to run:

- i686 / Debian testing
- i686 / Ubuntu 8.04
- i686 / Ubuntu 9.04
- x86_64 / Debian testing
- x86_64 / Ubuntu 8.04

[a] We are in the process of making Aguila LSB-compliant.

The software is distributed as a compressed tar file which can be unzipped as follows:

```
tar zxf Aguila-<architecture>-<version>-Linux.tar.gz
```

This will result in a directory with the following subdirectories:

```
bin     # Contains the executable.
demos   # Contains demo data.
doc     # Contains this manual, in pdf format.
lib     # Contains the required shared libraries.
share   # Contains support files.
```

To be able to use Aguila you need to add the path to the `bin` directory to the search path for executables (`PATH`).

Note: It is not necessary anymore to add the `lib` directory to the search path for shared libraries (`LD_LIBRARY_PATH`).

Tip: To debug issues when Aguila fails to start you can look at the output of the following command:

```
ldd path_to_aguila/aguila | less
```

This command will report whether certain required shared libraries cannot be found.

2.3 Mac OS X

Aguila compiles on Mac OS X too, but since we currently do not have a development machine available with this operating system installed, we cannot provide binary packages, unfortunately.

CHAPTER

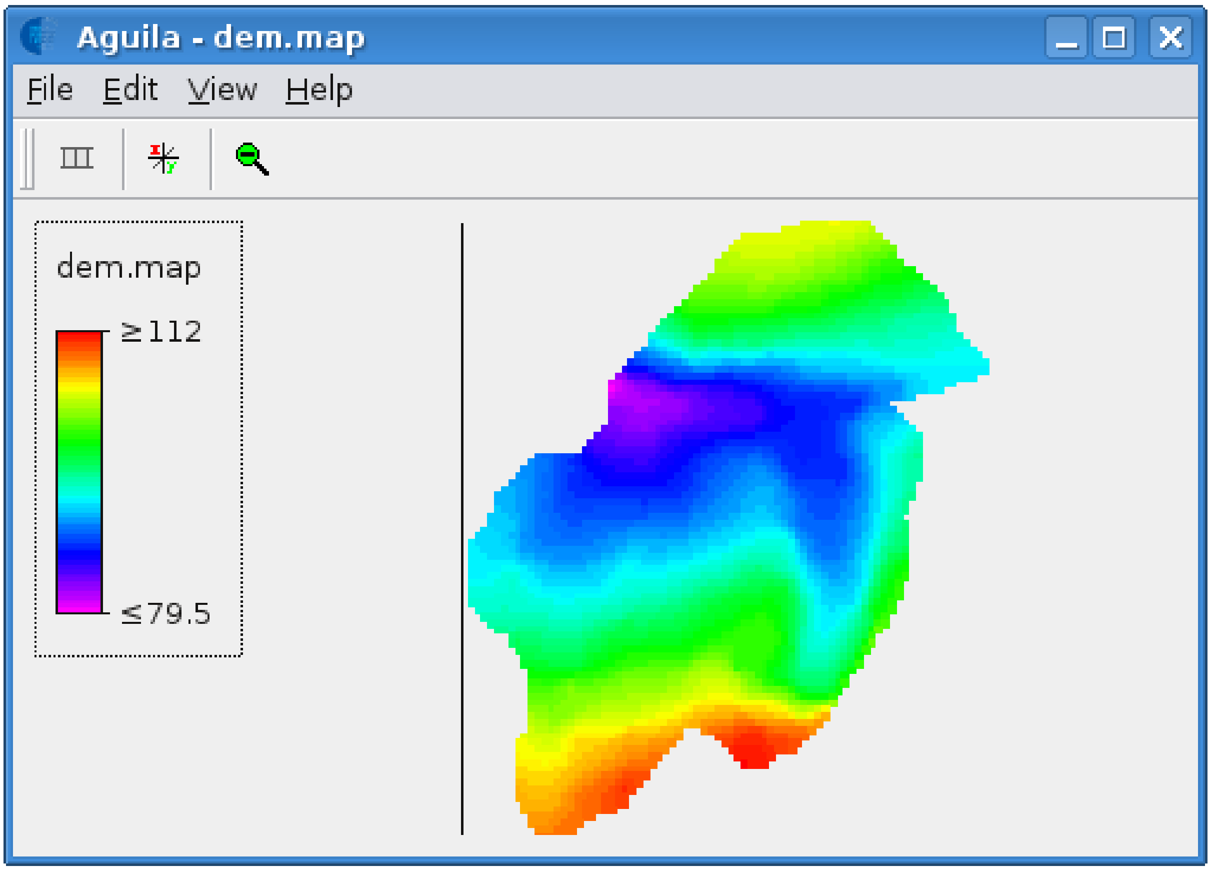
THREE

QUICK START

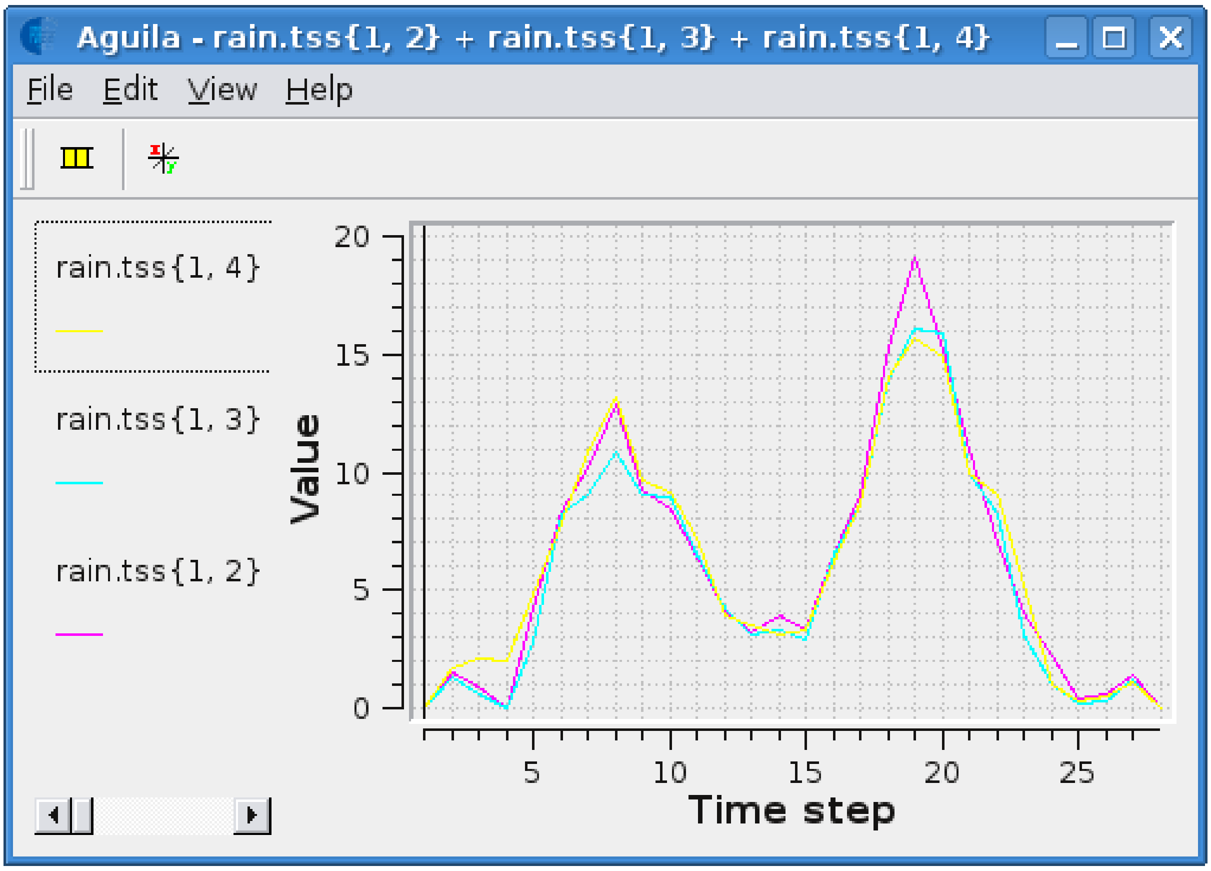
3.1 Displaying attributes

An attribute that is stored in a single file can be displayed by simply specifying the file name on the command line (see section *Program options*). Basic commands for maps and time series are:

```
# Show a map.
aguila dem.map
```



```
# Show time series.
aguila rain.tss
```

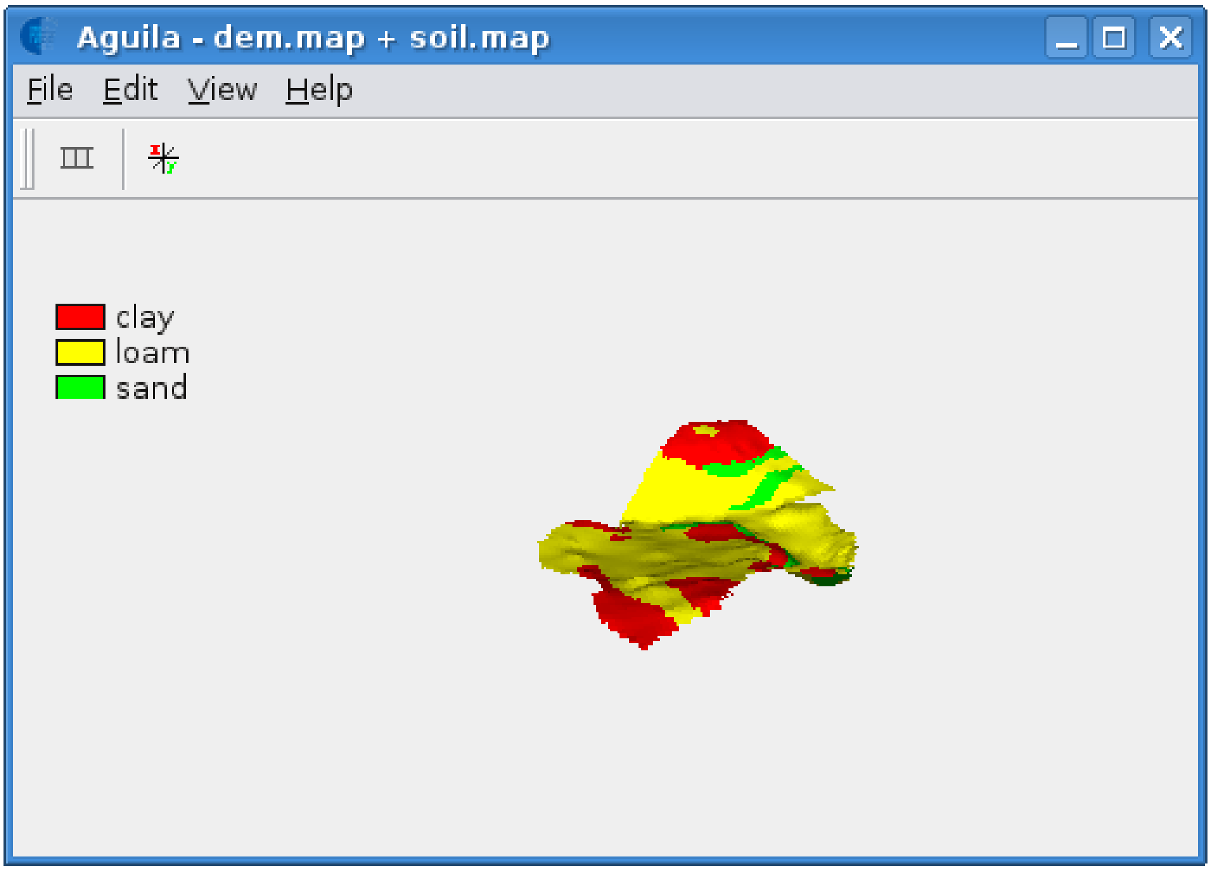


```
# Show some maps and a time series.
aguila dem.map soil.map rain.tss

# Show a map in 2.5D.
aguila --drapeView dem.map
```

Each attribute is displayed in a separate window, here called a view (see section *Views*). To display multiple attributes in a single view insert a +-symbol between the names, with spaces surrounding the +-symbol.

```
# Multiple data items as drape.
aguila --drapeView dem.map + soil.map
```



Spatio-temporal attributes stored in the PCRaster raster format, is not stored in a single file but as a sequence of numbered rasters. This is true for most raster formats. To show such a spatio-temporal data set, two notations are allowed:

```
# Old style: show discharge and rainfall map stack.
aguila dis00000.001+36 rainfall.001+36

# New style: show discharge and rainfall map stack.
aguila --timesteps [1,36] dis rainfall
```

That is all information you need to display spatio-temporal raster attributes and timeseries. This manual will further instruct you how to create more sophisticated and combined views of data.

3.2 Required information

There are three types of information Aguila has to know about before it can visualise your data: 1) the name(s) of the attributes you want to visualise, 2) the dimensions of the attributes in which it is defined (see section *Dimensions*), and 3) the views you want to use.

Of course, the name of the data is required. Telling Aguila about the dimensions of the data may be required or not, depending on the data format (see *Data formats*). For each type of data there is a default view which may be fine or has to be chosen otherwise by configuring specific start up options. See *Program options* for information about using program options to pass information to Aguila.

Once Aguila has started visualising data it maintains links between data properties and the views, and between the views. For example, if two spatial data sets are visualised with 2D map views and in one of them the map shown is zoomed into, the other view will also zoom into the same area.

3.3 Program options

Aguila can be configured by passing it program options on the command line. For example, the command

```
aguila dem.map
```

starts Aguila with the name of a dataset to visualise. If Aguila can read the data it will show it in the default view. Type

```
aguila --help
```

for a list of all command line options supported.

Since typing long commands with a lot of options can be tedious, some program options can also be read from a configuration file. For example, the command

```
aguila --config settings.cfg
```

tells Aguila to look for program options in the file settings.cfg.

Aguila can also be started from an XML specification file, which allows for more fine grained control over the Aguila configuration:

```
aguila --xml settings.xml
```

See section *Xml Startup Configuration* for details.

Command line options and configuration file options can be used at the same time, so

```
aguila -f settings.cfg dem.map
```

starts Aguila with options on the command line and in a configuration file. Some options will be combined when given more than once and others will be overridden (this is documented in the tables below or obvious). When an option is overridden because it is given both on the command line and in a configuration file, the one on the command line is given precedence. One way to use configuration files is to put common options in them which can be used together with the command line (see also the examples below).

Some options take a range or a set of values as an argument. The syntax for a *range* of values is `[first, last, step]` where `first` is the first value, `last` is the last value and `step` is the interval between individual values between first and last. For example, the range `[1,6,2]` consists of the values `1`, `3`, `5` (note that `6` is not used in this case because `5 + step` equals `7`, which is considered outside the range). The step is optional and the default value is dependent on the kind of range values. The range of timesteps `[1,6]` consists of the values `1`, `2`, `3`, `4`, `5`, `6`. The syntax for a *set* of values is `{value1, value2, ..., valuen}`.

Warning: The range and set notations can lead to suprising effects when used in a Unix (or Cygwin) shell (eg: bash). If you use such a shell, you must escape the `{`, `}`, `[` and `]` characters, because these have special meaning to the shell interpreter. Or you may quote the whole argument:

```
aguila --scenarios \{a,b,c\} concentration
aguila --scenarios "{a, b, c}" concentration
```

When quoting the argument, you can use spaces between the values, otherwise you cannot.

Some options take the name of a data set as an argument. Aguila supports certain naming conventions which depend on the format the data is stored in. For example, a table stored in an ASCII column file is named by its filename, but the same table stored by a database management system might be named as `myname(mypasswd)@mydbmsserver:mydatabase/mytable` or just `mydatabase/mytable`. For more information about these naming conventions see section *Data formats*.

Command line options which can also occur in a configuration file must be named by the long option name, without the leading double dash. So, while `-n` and `--scenarios` are equivalent when used on the command line, only `scenarios` can be used in a configuration file. Furthermore, the value of an option given on the command line is put immediately after the option name or character, optionally separated by white space. In a configuration file the option name and value are separated by an equals sign, optionally surrounded by white space. See also section Examples.

Table 3.1: Command line options

Option	Description
-f [–config] arg	Read options from the configuration file named `arg`.
-x [–xml] arg	Read options from the xml file named `arg`, see section *Xml Startup Configuration*.
-h [–help]	Show the command synopsis and exit.
-l [–lock] arg	Create a lock file named `arg`. If the file does not already exists it is created. When Aguila exits the file is deleted again. This can be useful if Aguila is started by another application which wants to be able to check whether Aguila is still running.
–license	Show the software license and exit.
-v [–version]	Show the software version and exit.
-2 [–mapView] arg	Show attribute named `arg` in a 2D map view. `arg` can contain the names of more than one attribute. When the names of two attributes are separated by `whitespace + whitespace`, they are stacked on top of each other in the same view. Otherwise each attribute is visualised in its own view.
-3 [–drapeView] arg	Show attribute named `arg` in a 3D map view. `arg` can contain the names of more than one attribute. When the names of two attributes are separated by `whitespace + whitespace`, they are stacked on top of each other in the same view. Otherwise each attribute is visualised in its own view. The first attribute is used for the height values. This attribute must contain scalar values.
-t [–timeGraphView] arg	Show attribute named `arg` in a time graph view. `arg` can contain the names of more than one attribute. The optional selection specification in the attribute name must contain 2 column numbers of which the first one is regarded as the time step column and the second as the attribute column.
-p [–probabilityGraphView] arg	Show attribute named `arg` in a probability graph view. `arg` can contain the names of more than one attribute.
–valueOnly arg	Show attribute named `arg` only in the value cursor matrix. `arg` can contain the names of more than one data set.

Table 3.2: Command line and configuration file options

Option	Description
-n [–scenarios] arg	Configures the scenario dimension using the set of scenarios in `arg`. Multiple scenarios options are merged into one scenario dimension.
-s [–timesteps] arg	Configures the time dimension using the range or set of time steps in `arg`. Time steps must be larger than `0`. Multiple time steps options are merged into one time dimension.
-q [–quantiles] arg	Configures the cumulative probability dimension using the range or set of quantiles in `arg`. Quantiles must be larger than `0` and smaller than `1`. Multiple quantiles options are merged into one cumulative probability dimension.
–cursorValueMonitorFile arg	Tells Aguila to append an `aguilaCursorValue` element to the value monitor file named `arg` each time `Save` is pressed in the Cursor Value Window. On start up, `arg` is created with 0 `aguilaCursorValue` sub-elements. The file is written in XML conforming to the Aguila XML Schema.
-m [–multi] arg	When visualising scenarios of a spatial attribute, this option can be used to put all scenario's side by side in one 2D map view. `arg` should be formatted as `<number or rows>x<number of columns>`, eg: `2x3`.

3.4 Examples

These examples assume `dem`, `ldd` and `erosion` are valid names of raster attributes and `discharge` is a valid name of a time series attribute. These attributes are presented in a format supported by Aguila. Note that attributes in different formats can be combined.

3.4.1 2D raster

Visualise a raster in 2D. Default view for rasters is 2D map.

```
aguila dem
```

See also section *Map view*.

3.4.2 2D rasters on top of each other

Stack rasters on each other. Spaces around the +-sign.

```
aguila dem + ldd
```

See also section *Map view*.

3.4.3 2.5D raster draped

2.5D is also possible.

```
aguila -3 dem + ldd
```

See also section *Drape View*.

3.4.4 2D raster stack

Raster attribute might be temporal. `dem00000.001+100` is deprecated. Separate the name of the dataset from the dimension information.

```
aguila --timesteps [1,100] dem
```

See also sections *Map view*, *Time Graph View*.

3.4.5 Time series

Visualise all time series in one time series plot.

```
aguila discharge
```

See also section *Time Graph View*.

3.4.6 Time series selection

Select some time series (the fifth and seventh columns) from discharge.

```
aguila discharge{1,5} + discharge{1,7}
```

See also section *Time Graph View*.

3.4.7 2D raster draped and time series

Combine rasters and time series data.

```
aguila dem + ldd discharge
```

See also sections *Map view*, *Time Graph View*.

3.4.8 Scenarios of temporal quantiles

Select some more dimensions. For each scenario one 2D map view of the median value of `erosion`.

```
aguila --scenarios {simple,complex} --quantiles [0.01,0.99] --timesteps [1,100] erosion
```

This example assumes that two erosion models where created, a simple one and a more complex one. Each of these models was used in a Monte Carlo analysis resulting in a distribution of erosion outcomes. For each timestep the distribution in erosion outcomes was summarised by a range of percentiles, for example the `0.01`, `0.05`, `0.1`, `0.25`, `0.5`, `0.75`, `0.9`, `0.95`, `0.99` percentiles. Aguila starts by presenting the median value of `erosion` for each scenario as a 2D map view. Given this attribute it is possible to get a time series plot and a cumulative distribution plot for each location. All views can be animated. Aguila interpolates the data for percentiles that are not provided.

See also sections *Map view*, *Time Graph View*, *Probability Graph View*.

3.5 Environment variables

Aguila has support for many data set formats. While searching for data, Aguila tries each format driver in turn to see whether it is able to read the data. If you only use a fixed set of formats for your data sets, you can decrease startup time by limiting the number of potential data set formats Aguila considers. For this, define the environment variable called `PCRASTER_DAL_FORMATS`. Its contents should be a comma separated list of format names. Most of them correspond to the names used in the GDAL and OGR data I/O libraries used by Aguila (*Data set types*).

Example:

```
$ export PCRASTER_DAL_FORMATS="CSF, HDF4, WCS, ESRI Shapefile"
```

Note: Although GDAL comes with a PCRaster driver built in, Aguila makes use of its own PCRaster driver and skips GDAL's one. The name of the PCRaster raster format driver is `CSF`.

Unknown format names are skipped. If none of the format names are known, Aguila will use all formats it knows of.

Warning: Whenever Aguila is unable to read your attributes, make sure the setting of PCRASTER_DAL_FORMATS is not excluding a format driver required for reading the attribute.

CHAPTER

FOUR

VIEWS

Views are attribute visualisations. Every type of attribute supported by Aguila has a default view type which will be shown when no view is explicitly selected when starting up Aguila. For example, a raster attribute will be shown in the map view and a time series in a time graph view. Some attribute types can be shown in multiple view types. A spatio-temporal raster attribute can be shown in a map view and a time graph view at the same time, for example.

Aguila uses the notion of a cursor to enable the views to decide which part of an attribute data set to visualise. The cursor consists of a set of coordinates that the user can set, like the current time step and the current spatial x- and y-coordinates. The current time step determines which map of a spatio-temporal attribute is shown, for example. And the current spatial coordinate determines for which spatial location of a spatio-temporal attribute the time graph is shown.

Views enable the user to change the cursor. For example, in a map view, the spatial coordinates of the cursor can be changed by clicking in the map. And in a time graph view, the time coordinate of the cursor can be changed by clicking in the graph. All views will respond to a change of the cursor. See section *Cursor And Values View* for a view for viewing and changing the current cursor coordinates.

Apart from the cursor there are more global properties used by Aguila to maintain links between the views. Examples of those are the scale of a map and the pan position. Because of this, all map views will always show the same area.

Attributes have draw properties, for example the palette used to select colours used to visualise the attribute. These properties are shared among the views, so each attribute always uses the same properties, no matter in how many views the attribute is shown. Changing the properties of the attribute in one view, also changes the properties of the attribute in the other view(s).

All views have a similar layout. The largest part is taken up by the visualisation itself, and on the left there is a legend for each attribute visualised. Between the legend and the main visualisation there is a splitter which can be dragged to make the legend larger or smaller relative to the main visualisation. At the top there are a menu bar and a toolbar.

Right-clicking the legend of an attribute brings up a context menu with actions specific for the selected attribute. Some actions are always available and some actions are available depending on the characteristics of the selected attribute. For example, the action to show an attribute in a time graph view is only available for a temporal attribute.

Here is a list of common attribute context menu actions:

Table 4.1: Attribute context menu actions

Menu item	Effect
`Edit draw properties`	Shows the `Draw Properties` dialog, see section *Draw Properties Dialog*.

Double-clicking the attribute legend shows the draw properties dialog also.

For spatio-temporal attributes, the attribute context menu has a `Show Time Series` action. This will show the attribute in a new, linked, time graph view.

4.1 Cursor And Values View

All attributes shown by Aguila are put in the cursor and values view too. This view is normally not visible when the application starts (unless requested with the `valueOnly` program option (see section *Program options*).

This view shows the current global cursor coordinates and the attribute values at the current cursor position.

The view consists of an upper part with the current cursor coordinates and a lower part with the values of all attributes loaded. The cursor sliders allows navigating through the dimensions such as time and space (see section *Dimensions*). Each slider has a knob for the current position.

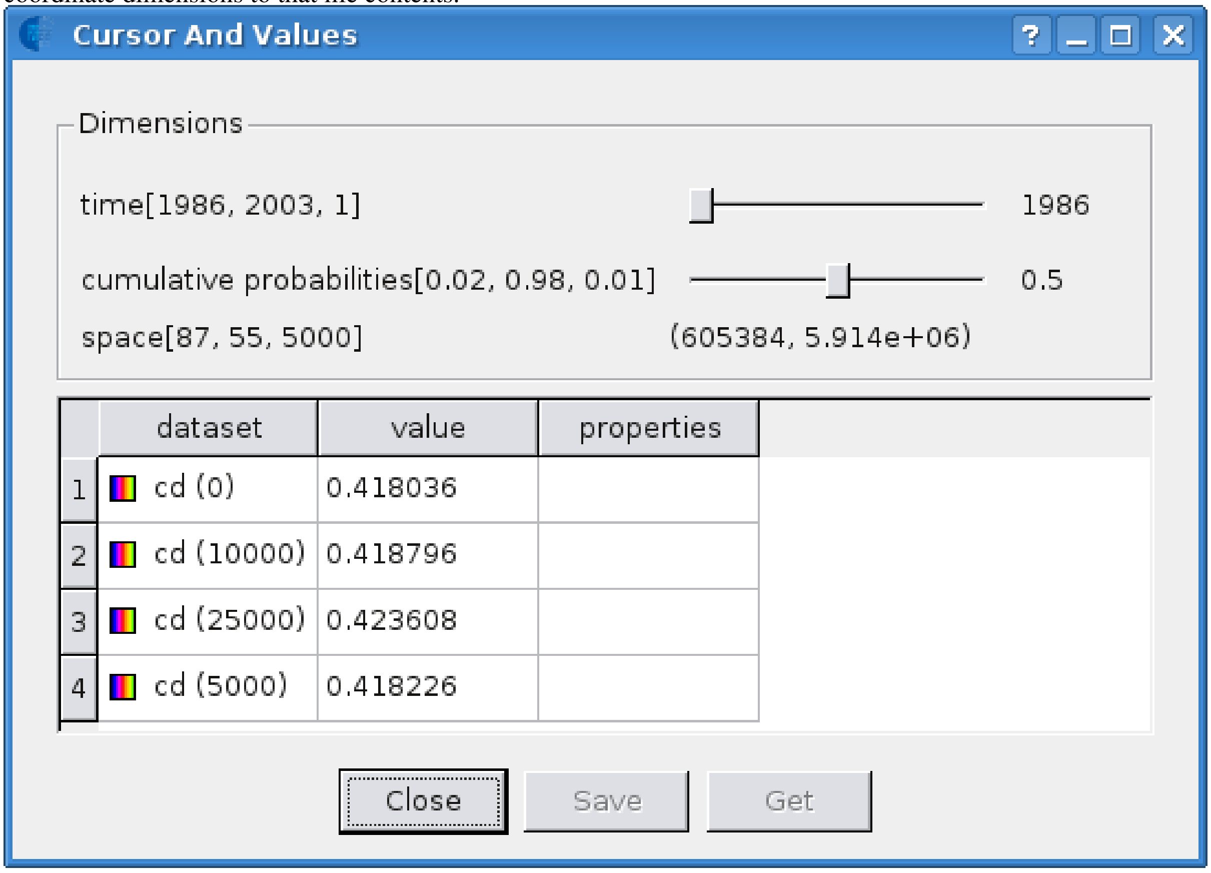
Table 4.2: Cursor slider actions

Change position	Slider action
Set slider one position backwards	Click on the slider part left from the knob
Set slider one position forward	Click on the slider part right from the knob
Set slider to specific position	Drag knob to position

In the value list at the bottom the data names and their values at the current cursor position are shown.

If Aguila started with the `cursorValueMonitorFile` program option, the `Save` button is enabled to append the current data in that file.

If Aguila started with the `fileToGetCursorValue` program option, the `Get` button is enabled to set the coordinate dimensions to that file contents.



time[1986, 2003, 1]	1986
cumulative probabilities[0.02, 0.98, 0.01]	0.5
space[87, 55, 5000]	(605384, 5.914e+06)

	dataset	value	properties
1	cd (0)	0.418036	
2	cd (10000)	0.418796	
3	cd (25000)	0.423608	
4	cd (5000)	0.418226	

4.2 Map view

A map view shows spatial attributes by a map. This map visualises the spatial variation of the attribute, given the current cursor position.

4.2.1 Map Controls

Table 4.3: Map controls.

Effect	Control
change spatial cursor	left mouse click
pan	left mouse drag
continuously change spatial cursor (prevents panning)	Ctrl+left mouse drag
move map to the right	h / left mouse drag right
move map to the left	l / left mouse drag left
move map to the bottom	k / left mouse drag down
move map to the top	j / left mouse drag up
zoom in	Ctrl+k / left mouse double click / scroll mouse
zoom out	forward Ctrl+j / scroll mouse backward
Zoom by rectangle	Shift+left mouse drag
reset	r

4.3 Time Graph View

A time graph view shows temporal attributes by a single line. This line visualises the temporal variation of the attribute, given the current cursor position. The current time can be changed by clicking or dragging in the graph.

The attribute context menu has a `Save graph data as...` action. This enables one to export the data of a single graph line.

4.4 Drape View

The drape view shows a scalar spatial raster attribute as a surface (or sheet) which floats in space. Additional spatial raster attributes can be shown on top of this surface.

You can change some properties of the drape view by right clicking your mouse in the map view (the one which shows the sheet) and selecting `Properties`. The properties dialogue for the map view will be shown.

4.4.1 Drape Controls

There's a difference between controlling the camera ('your head') and the 3D scene. You can control the position and aim of the camera (see the table below). You can only control the orientation of the scene (see the second table below). Together these controls enable you to look at every part of the scene from everywhere.

Table 4.4: Controls for changing the position and orientation of the camera.

Effect	Control
look left	h / left mouse drag left / right mouse drag left
look right	l / left mouse drag right / right mouse drag right
look up	k / right mouse drag forwards
look down	j / right mouse drag backwards
roll clockwise	n
roll counter-clockwise	m
move left	Shift+h / left-right mouse drag left
move right	Shift+l / left-right mouse drag left
move up	Shift+k / left-right mouse drag forwards
move down	Shift+j / left-right mouse drag backwards
move forward	Shift+Up / Ctrl+k / left mouse drag forwards
move backwards	Shift+Down / Ctrl+j / left mouse drag backwards
reset	r

Table 4.5: Controls for changing the orientation of the scene.

Effect	Control
rotate clockwise around z-axis	Left
rotate counter-clockwise around z-axis	Right
rotate clockwise around x-axis	Up
rotate counter-clockwise around x-axis	Down
reset	r

The camera and the scene have their own coordinate system which can be moved and rotated independently from each other. You should interpret the above controls relative to the direction of the coordinate system of the camera or the scene. For example, moving the camera forward means moving the camera in the direction of the camera. This might be in a direction away from the scene!

4.4.2 Camera's

Apart from the camera whose position and orientation can be changed, there are 5 more camera's in the scene. These static camera's are positioned in such a way that it is possible to see the relative position of the scene and the mobile camera. The table below gives a list of camera's you can choose from (note the correspondence between the shortcuts and the layout of the numeric keypad on your keyboard).

Table 4.6: Camera's.

Camera	Shortcut
Top	5
Front	2
Left	4
Back	8
Right	6
Mobile	0

4.4.3 Shortcuts

Table 4.7: Shortcuts.

Effect	Shortcut
exaggerate heights	Plus (+)
understate heights	Minus (-)
enlarge quad length	Shift+q
decrease quad length	q

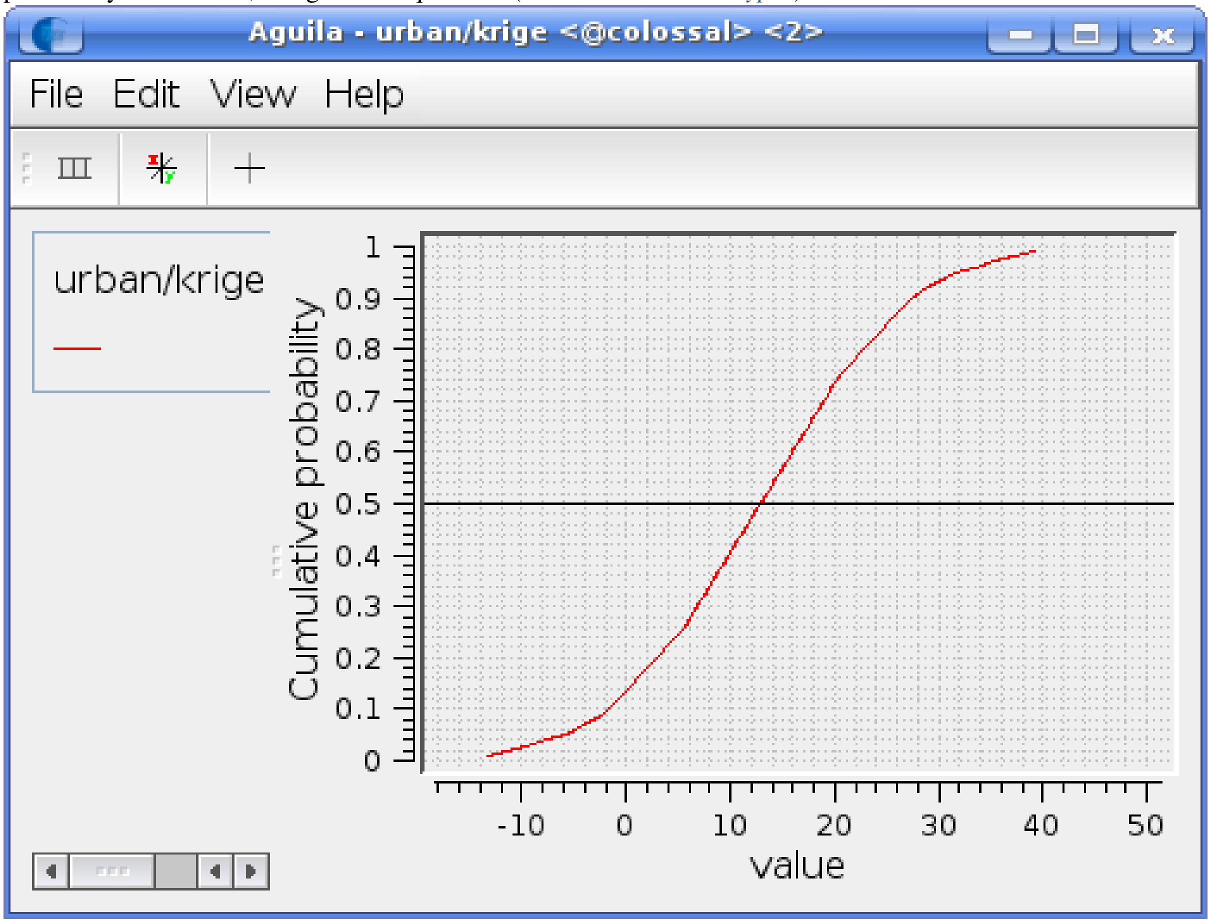
4.5 2D Multimap View

2D Multimap mode can be enabled when multiple scenarios are provided, that need to be compared side by side. Without duplication of window borders, legends, etc., in this mode, multiple maps are organized in panels over a regular lattice, and shown in a single window. Zoom/pan/identify actions, as well as legend modifications are now reflected automatically over all panels.

Aguila does not provide interactive setting of the panel layout; it needs to be set on the command line or in the configuration file specified on startup. If it is not specified, all maps (scenarios) will be shown in different windows, and e.g. legend changes will only reflect to the window at which it is applied.

4.6 Probability Graph View

Aguila can show probability distribution functions (cumulative probabilities and exceedance probabilities) for continuous random variables. The probability graph view is available only when the data are specified as a probability distribution, using a set of quantiles (see section *Data set types*).



For uncertain attributes, the attribute context menu has a `Show Probability Plot` action. This will show the attribute in a new, linked, probability graph view. This view shows, for the current location and time step, the distribution function for the data, or the set of distribution functions for the set of scenarios.

The view shows a horizontal line at distribution value `0.5`, indicating that the map currently shows the median value. The horizontal line can be dragged to show other quantile values in the map view or time plot.

As an alternative view, the distribution function values for a given threshold value (one minus the exceedance probability) can be shown by pressing the tool button with the + symbol, or through menu items `View | Toggle Marker`. This will flip the horizontal line into vertical position, allowing modification of the threshold (attribute) value for which distribution function values (probabilities) will be shown.

Pressing the + tool button again returns to quantile views.

Dragging the line may, for larger data sets, be slower in vertical mode than in horizontal mode. The reason is that in vertical mode, besides a linear interpolation, a lookup needs to take place, because the function is specified for a known set of (constant) probability values, and not as probabilities for a known set of attribute (data) values.

CHAPTER

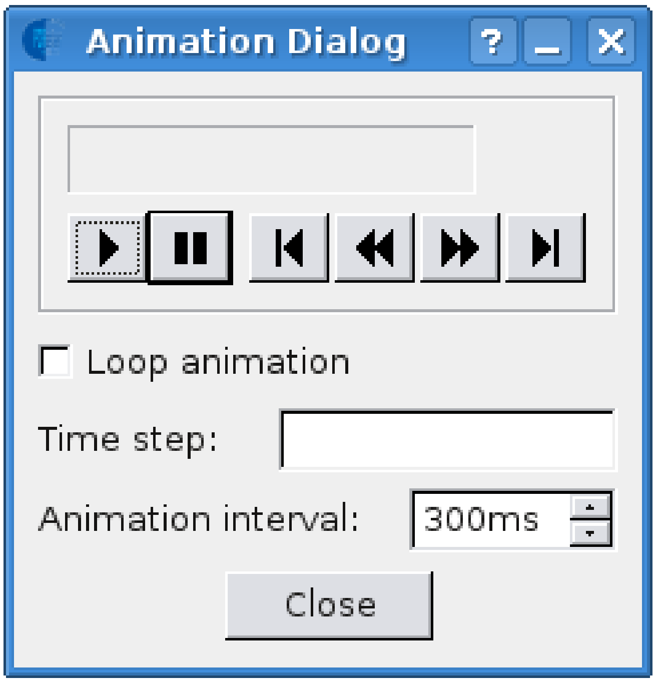
FIVE

DIALOGS

In this section the various Aguila dialogs are described.

5.1 Animation Dialog

If you loaded a temporal attribute, than you can start an animation by selecting the `Animate` menu item from the `View` menu, or by pressing the `Animate` toolbutton from the toolbar. A dialog will be shown from which you can control the animation. This will trigger an update of all views that show temporal attributes.



5.2 Draw Properties Dialog

What properties are show in the properties dialog depends on the attribute type, not all properties are applicable for all types of data.

Palette The colours used in the view

Exact legend borders Should the minimum and maximum value of the legend, exactly match the data minimum and maximum. The values are rounded otherwise

Number of colours Numbers of colours sampled from the colour palette.

Maximum cutoff This value will set the upper range used in the colour assignment. Data values above this value are drawn using the same colour as the one used for the maximum cutoff value. `Reset` will set this back to the default case of the data maximum.

Minimum cutoff This value will set the lower range used in the colour assignment. Data values below this value are drawn using the same colour as the one used for the mimimum cutoff value. `Reset` will set this back to the default case of the data minimum.

Colour assignment Mode of how colours are assigned by dividing the data range in `Number of colours` intervals. The per colour interval size can be fixed (linear) or increasing from small to large (logarithmic).

Draw mode

`Fill` Show values as solid filled pixels.

`Contour` Draw as contours.

For an uncertain attribute, the folowing options are also available:

Colour assignment If a value is selected in the probability graph, an additional colour assignment mode is available: `Confidence interval`.

Confidence level (1 - alpha) If the `Confidence interval` colour assignment mode is selected, this option can be used to configure the confidence level.

Exceedance probabilities When this option is selected, exceedance probabilities will be shown, instead of cumulative probabilities.

CHAPTER

SIX

DATA

6.1 Data sets

Attributes are stored in data sets. Depending on the data set formats, data sets are stored in a single file in the local or in a remote filesystem, or in multiple files, or in tables in a database, or are part of a single file containing multiple data sets, for example. Whatever is needed to store the values of a *single* attribute comprises the data set.

Attribute values in a data set describe the variation of a single attribute (e.g.: height, concentration, temperature). Dimensions are used to structure data set values (see section *Dimensions*). Unique combinations of such dimensions give rise to different data set types (see section *Data set types*). And various data formats are used to store data set values (see section *Data formats*).

6.2 Dimensions

Dimensions structure data set values. Attribute values (potentially) vary along dimensions. For example, a raster contains spatially varying attribute values. Similar, a time series contains temporally varying attribute values.

A data set lacking attribute values for a certain dimension is taken to be constant along that dimension. For example, a single raster contains values that vary in space, but not in time. Its values are constant in time because the data set contains no information about the temporal variation.

The next sections describe the dimension that are supported by Aguila.

6.2.1 Space

Aguila supports visualisation of both raster and feature data. Spatial raster data supported by Aguila is formatted as a raster with cells of equal width and height.

Aguila offers 2D and 2.5D views for rasters and 2D views for features.

6.2.2 Time

Temporal data supported is data which is defined at discrete integer time steps. Sophisticated schemes to link these discrete time steps to real time are only possible by using the XML interface described in section *Xml Startup Configuration*.

Aguila offers time series views for temporal data. Additionally, it adds support for iteration through time (animation) to every view which is able to visualise temporal data, for example the 2D map view.

6.2.3 Uncertainty

If, instead of fixed variables, the available data are random variables (e.g. resulting from a Monte Carlo analysis or statistical modelling), Aguila can view them when they are encoded as (cumulative) probability distribution functions. The encoding is done by supplying maps that have for each location (and optionally for each time step) the attribute value corresponding to a fixed cumulative probability level, a value in `[0,1]`. The simplest example would be if all variables followed a uniform distribution, in which case we only need to specify the map with the minimum (distribution value `0`) and maximum (distribution value `1`); Aguila uses linear interpolation for values in between. More complex distributions functions can be specified with maps for a set of distribution values. E.g. for a normal distribution one could specify the maps corresponding to probability distribution values `0.01`, `0.05`, `0.1`, `0.25`, `0.5`, `0.75`, `0.9`, `0.95` and `0.99`. Aguila assumes then straight lines in between these values.

Distribution values do not need to be symmetric; if much emphasis is put on the upper tail, the lower tail may be discretized by very few values (like, say, `0.01`, `0.5`, `0.9`, `0.91`, `0.92`, `0.93`, ...).

The interface of Aguila assumes that all distribution values provided are part of a regular sequence. Both examples above are subsets of the sequence `0.01`, `0.02`, `0.03`, ..., `0.99`. Minimum, maximum, and step size need to be specified for a sequence. Aguila will read those values present, and will linearly interpolate inbetween them.

6.2.4 Scenarios

Different scenarios contain data values for the same attributes which should be visualised at the same time, and with identical topological constraints. When data is loaded from different scenarios, Aguila will create views for each scenario. Scenarios are explicitly supported by the *2D Multimap View*. This view shows spatial attributes from different scenarios in a single window.

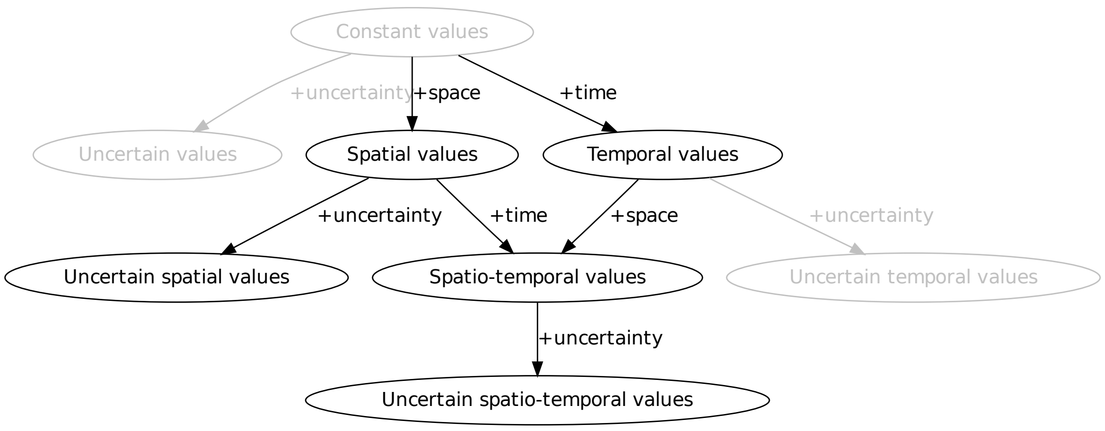
Note: When scenarios are used, we still speak of a single attribute.

6.3 Data set types

As described above, Aguila supports data sets with various kinds of dimensions. Every unique combination of dimensions gives rise to a data set type.

For some types of data sets, there are convenient data formats available, for example raster formats. For other data set types, we had to come up with some conventions to be able to store the values. There is, for example, no convenient data format availabe for storing uncertain spatio-temporal values.

This section describes the various types of data sets, and the next section describes the formats for storing the attribute values and some naming conventions used. The grey nodes in the figure below show data set types are not implemented. They are drawn and mentioned in the text for completeness.



6.3.1 Constant values

The simplest data set type is the constant. A constant is just a single value, its value does not vary along one of the data dimensions.

Examples:

- An empirical constant.
- A boundary value, not to be exceeded.

Note: Not implemented.

6.3.2 Uncertain values

Uncertainty adds information about the probability distribution of values that may represent the attribute. Note that there is no spatial or temporal variation in this case.

Note: Not implemented.

6.3.3 Temporal values

Temporal values vary in time, but they have no spatial variation or even spatial reference.

Examples:

- Rainfall for a relatively small area and assumed te be constant for the whole area

Aguila has support for reading temporal values from many table data formats.

6.3.4 Uncertain temporal values

TODO

Note: Not implemented.

6.3.5 Spatial values

Spatial values vary in space, but not in time.

Examples:

- Raster with elevation values.
- Feature layer with population density per administrative area.

Raster, feature and vector data sets contain spatial values. An exception to this rule is that Aguila can also visualise only the geometry of a feature layer.

Vector attributes are attributes with a direction and a magnitude. They are currently raster based, meaning that two rasters with magnitudes in x- and y-directins are used to store the values.

Aguila has support for reading spatial data from many raster data formats, feature layer formats and *vector data formats*.

6.3.6 Uncertain spatial values

Uncertain spatial values are spatial values for which the probability distribution is known.

Currently, Aguila supports reading quantiles of spatial values, for example a range of the `0.01`, `0.05`, `0.10`, `0.25`, `0.50`, `0.75`, `0.90`, `0.95`, `0.99` quantile levels.

6.3.7 Spatio-temporal values

Spatio-temporal values are spatial values which also vary in time.

Currently, Aguila supports reading time steps of spatial values, for example a range of the steps 1 through 1000.

6.3.8 Uncertain spatio-temporal values

Uncertain spatio-temporal values are spatio-temporal values for which the probability distribution is known.

Currently, Aguila supports reading quantiles of spatial values, for example a range of the `0.01`, `0.05`, `0.10`, `0.25`, `0.50`, `0.75`, `0.90`, `0.95`, `0.99` quantile levels.

6.3.9 Scenario values

TODO

6.4 Data formats

Exactly which data formats Aguila supports depends on how Aguila is compiled. Potentially, Aguila can support many formats. The distributed Aguila supports about 50 raster formats (eg: PCRaster raster format, Hdf4, GeoTIFF, GML, ESRI's binary grid), a dozen feature formats (eg: ESRI's Shapefile) and a few table formats (eg: text, ODBC, Sqlite). Exactly which formats are supported can be deduced by looking at the script used to build the support libraries [1].

The data format used to store the data set determines the way the database must be configured. A data format might support data with some dimensions, like raster data with two spatial dimensions, but lack support for other dimensions, like the time dimension. Support for missing dimensions can be added to formats using naming conventions for file names, or by storing this information in an attribute table, for example. Most raster formats do not support more dimensions than the two space dimensions, but using naming conventions, information about

[1] Eventually Aguila will be able to list the formats built in.

the time dimension can be added easily. This results, for example, in a name like dem_1.map for the first time step. If a raster format supports the time dimension, such a naming convention is not needed and the application can read the information about this dimension from the data source.

6.4.1 Raster data formats

The raster formats supported are PCRasters CSF 2.0 raster file format and the formats supported by the Geospatial Data Abstraction Library (GDAL). Information about these formats, including free (conversion) software and manuals can be found at the PCRaster website and the GDAL website.

For formats with default file name extensions there is no need to supply an extension.

Aguila uses the value scale property to decide how to visualise an attribute. In the CSF raster file format this value scale is stored in the data set. In data sets read using GDAL, this information may or may not be available. If a meta data item named `PCRASTER_VALUESCALE` is available, it is used to determine the value scale. If it is not available, the color interpretation of the data set is used to determine the value scale. Lastly, the type of the attribute values is used to determine the value scale. Integral types result in a nominal value scale and floating point types result in a scalar value scale to be used.

The folowing values for the `PCRASTER_VALUESCALE` meta data item are recognized:

- `VS_BOOLEAN`: boolean attribute
- `VS_NOMINAL`: nominal attribute
- `VS_ORDINAL`: ordinal attribute
- `VS_SCALAR`: scalar attribute
- `VS_DIRECTION`: directional attribute
- `VS_LDD`: ldd attribute

The folowing color interpretation values are recognized:

- `GCI_GrayIndex`: scalar attribute

Naming conventions

Raster formats store spatial data in seperate files. Each file contains spatial attribute values for one map. To add information about other dimensions Aguila supports the folowing naming conventions:

Table 6.1: Raster file format conventions

Dimension	Convention
Scenarios	Data for different scenarios must be stored in sub directories. When a scenarios dimension is configured Aguila searches its data in directories named after the scenarios.
Cumulative probabilities	The different quantile levels are reflected in the filename. The filenaming rule is `name_<level>{.extension}`. The quantile level must be a floating point value between `0` and `1.0`. The file name extension is optional.
Time	Time steps are reflected in the filename. The filenaming rule is `name_<step>{.extension}`. The time step must be a positive integral value greater than `0`. The file name extension is optional.

This means that, for example, a raster data source with scenarios `simple` and `complex`, and temporal quantile levels has rasters named `simple/co2_1_0.001.map` and `complex/co2_100_0.5.map`.

The PCRaster convention for naming spatio-temporal raster data is also supported. PCRaster uses the 8.3 DOS convention where each member of a stack is named by its name, possibly 0's and a time step number. So, a PCRaster model might output `dem00000.001`, `dem00000.002`, etc. When the time dimension is set up right, the name (`dem` in this case) can be used to name such a stack. Also supported is the (depricated) convention of naming a stack by the first member, folowed by a +-sign and the last time step of the stack, `dem00000.001+1000` for example.

6.4.2 Feature layer formats

The feature layer formats supported are the formats supported by the OGR Simple Feature Library (OGR). Information about these formats, including free (conversion) software and manuals can be found at the OGR website.

Naming conventions

Aguila visualises attributes, not whole data sources. Feature data sources contain one or more feature layers and each layer contains zero or more attributes. The folowing naming rule must be used to tell Aguila which attribute to visualize:

```
datasource/layer{/attribute}
```

When the attribute is not provided, Aguila will show the geometry of the layer. For example, given a ESRI Shapefile with information about countries, the folowing command will draw the country borders:

```
aguila countries.shp/countries
```

And the folowing command will draw each country's population:

```
aguila countries.shp/countries/population
```

Known file name extensions are optional and can be left out.

Each feature layer attribute contains information about the spatial variation of the attribute. To add information about other dimensions Aguila supports the use of an external attribute table in a database. The attribute table must be named after the feature layer and contain a field called `fid` (which is short for feature id). Aguila will join the internal attribute table of the feature layer data source to the external table using the feature id fields present in both tables.

See *Table data formats* for more information about setting up the external attribute table.

Note: Currently, only SQLite databases have been used for storing external attribute tables for feature layers. In principle, all other table formats should also be supported. Also, since the name of the database and table are the same as the datasource and feature layer, it is not possible to pass server name and/or user credentials.

6.4.3 Vector data formats

Two raster datasets are used to hold the magnitude values for the x- and y-direction of a vector attribute. Because of this, all formats described in section *Raster data formats* can be used to store vector magnitude values.

Naming conventions

There is a simple naming scheme for vector datasets:

```
name_(x|y){.extension}
```

So, the values for a vector attribute called `flow` are stored in two raster datasets called `flow_x` and `flow_y`. Default extensions are optional.

Naming conventions for vector datasets with more dimensions are the same as those described in section *Naming conventions*.

6.4.4 Table data formats

Table file formats supported are the text and geoEAS formatted files and formats supported by the QtSql module.

Naming conventions

Naming a table data set depends on the format used to store the table in. For file formats the name of the data set is the name of the file it is stored in. For tables served by database management systems different conventions apply: `myname(mypasswd)@myserver:mydatabase/mytable`. Some of the elements of this naming conventions might be optional, depending on the configuration of your database server. The server might, for example, grant read access to everybody, which means the account information in the name is redundant: `myserver:mydatabase/mytable`. The database server might be your production machine, which means the server name is redundant: `mydatabase/mytable`.

The table must contain a field named after the attribute. The values of this field will be visualized.

Attribute variation for one or more dimensions must be stored using a primary key consisting of one or more of the folowing field names: `scenario`, `date`, `quantile`. For each unique combination of these, an attribute value can be stored in the table.

Note: In case the table is used as an external attribute table for a feature layer, the primary key must also include the feature id field (`fid`).

CHAPTER

SEVEN

XML STARTUP CONFIGURATION

While configuration of Aguila is possible with the use of command line options and a configuration file, a more sophisticated scheme is possible by configuring Aguila with an XML start up file. The XML interface is unique in the following ways:

- Specify draw properties per data item.
- Specify real dates (e.g. 2005-02-10T18:15:00) and real time intervals (e.g. 24 hours).
- Map data with different time dimensions to a single time line.

The XML interface is intended as an intermediate interface for other applications. For example, from a Python script it is relatively easy to generate the XML file. To use the XML interface a basic understanding of XML is required.

The XML Schema (`Aguila.xsd`) and samples of XML script files are distributed with a data set in the `demos/xml` distribution directory. References to `.xml` files in this chapter can be found in that directory. Note that the Aguila application does not require the file `Aguila.xsd` to be present. The correct version of `Aguila.xsd` is compiled into the Aguila application. For ease of reference, the schema is copied verbatim in section *Aguila.xsd*

7.1 Basic structure

In this section we only discuss the overall structure of the XML file. The detailed XML structure is documented in element specification within `Aguila.xsd`. Each Aguila XML file should have the following structure:

```
<?xml version="1.0" encoding="UTF-8"?>
<aguila
   xmlns="http://www.pcraster.nl/pcrxml"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://www.pcraster.nl/pcrxml Aguila.xsd">
  <visualisationGroup>
   <!-- contents consisting of elements:
        - cursorValueMonitorFile  0 or 1
        - searchSpace             0 or 1
        - data                    0 or more
        - view                    0 or more
     -->
  </visualisationGroup>
</aguila>
```

The elements of the basic structure will be discussed in the reverse order of appearance.

view A number of views can be specified. All view contents is specified by the dataset name as an `item` element. That same name may optionally appear as a data element. `Example1.xml` illustrates a number of points on the view specification:

- The first appearance yields the position in the cursor and values matrix. Use a `valueOnly` element to obtain the preferred order (`dem.map`).

- Time series file and cursor value graphs can be combined. See `runoff` and `runoff.tss` combined in a view in `example1.xml`.

`data` A `data` element is used to tag a dataset name used in a `view` element with additional specifications: its draw properties and `dataSpace`. For example, in `example1.xml` the colour assignment of `upstreamArea.map` is set to `shifted logarithmic`.

`searchSpace` With the `searchSpace` element one can describe the dimensions in which all data should be found if the data itself has no `dataSpace` sub element. In `example1.xml` the 28 model steps are set in the `searchSpace` and the timesteps are mapped to a real calendar date with a 6 hour time increment.

`cursorValueMonitorFile` See the section called *Program options*.

`fileToGetCursorValue` This element defines what file is read to set the applications cursor at each time `Get` is pressed in the `Cursor Value Window`. Currently only the x and y (space dimensions) are read. The file should be an XML file with the `aguilaCursor` element as root. See `get.xml` set in `example1.xml`.

7.2 Aguila.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    Style Guide:
    See PCRaster.xsd
-->

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:pcr="http://www.pcraster.nl/pcrxml"
           elementFormDefault="qualified"
           targetNamespace="http://www.pcraster.nl/pcrxml">

<xs:include schemaLocation="commonTypes.xsd"/>

<!-- document (root) elements -->
<xs:element name="aguila"              type="pcr:Aguila">
 <xs:annotation>
   <xs:documentation>Aguila configuration/startup file</xs:documentation>
 </xs:annotation>
</xs:element>
<xs:element name="aguilaCursorValues"  type="pcr:AguilaCursorValues">
 <xs:annotation>
   <xs:documentation>contents of what is created when starting Aguila
                     with the cursorValueMonitorFile option
   </xs:documentation>
 </xs:annotation>
</xs:element>
<xs:element name="aguilaCursor"        type="pcr:Cursor">
 <xs:annotation>
   <xs:documentation>contents expected for fileToGetCursorValue,
                     connected to Get-button in Cursor Value Window
   </xs:documentation>
 </xs:annotation>
</xs:element>

<!--type definitions-->
<xs:complexType name="Aguila">
 <xs:annotation>
   <xs:documentation>Aguila configuration/startup file</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="multiView"     type="pcr:NrRowsNrCols"
                                  minOccurs="0" maxOccurs="1">
    <xs:annotation>
     <xs:documentation>Only used to translate a command line option.
     </xs:documentation>
    </xs:annotation>
  </xs:element>
  <xs:element name="visualisationGroup"     type="pcr:VisualisationGroup"
                                   minOccurs="1" maxOccurs="1"/>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="VisualisationGroup">
 <xs:sequence>
  <xs:element name="cursorValueMonitorFile" type="xs:string"
                                    minOccurs="0" maxOccurs="1"/>
  <xs:element name="fileToGetCursorValue"   type="xs:string"
                                    minOccurs="0" maxOccurs="1"/>
  <xs:element name="searchSpace"   type="pcr:DataSpace"
                                    minOccurs="0" maxOccurs="1"/>
  <xs:element name="data"          type="pcr:AguilaData"
                                    minOccurs="0" maxOccurs="unbounded">
   <xs:annotation>
     <xs:documentation>Explictly defined data sources
     </xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="view"   type="pcr:AguilaView"
                             minOccurs="0" maxOccurs="unbounded">
    <xs:annotation>
      <xs:documentation>
       Order of elements will define the order in the Cursor Value Dialog.
      </xs:documentation>
    </xs:annotation>
  </xs:element>
  <!--
  <xs:element name="initialCursor" type="pcr:Cursor"
                                    minOccurs="0" maxOccurs="1">
   <xs:annotation>
     <xs:documentation>Define initial cursor</xs:documentation>
   </xs:annotation>
  </xs:element>
  <xs:element name="namedCursors" type="pcr:NamedCursors"
                                    minOccurs="0" maxOccurs="unbounded"/>
   -->
 </xs:sequence>
</xs:complexType>

<xs:complexType name="DataSpace">
   <xs:annotation>
     <xs:documentation>
      Note that in practice only the scenarios element can
      occur more then once. All other maxOccurs value different
      from 1 are only for internal Aguila reasons (converting command
          line options to XML internally).
     </xs:documentation>
   </xs:annotation>
 <xs:sequence>
   <xs:element name="scenarios" type="pcr:StringSet"
                        minOccurs="0" maxOccurs="unbounded"/>
   <xs:element name="quantiles" type="pcr:FloatRangeOrSet"
                        minOccurs="0" maxOccurs="unbounded"/>
  <!-- Not yet
   <xs:element name="samples" type="pcr:OneBasedIntegerRange"
                        minOccurs="0" maxOccurs="unbounded"/>
    -->
   <xs:element name="timesteps" type="pcr:Timesteps"
                          minOccurs="0" maxOccurs="unbounded"/>
 </xs:sequence>
</xs:complexType>


<xs:complexType name="Timesteps">
 <xs:complexContent>
  <xs:extension base="pcr:OneBasedIntegerRangeOrSet">
   <xs:sequence>
    <xs:element name="dateMapper" type="pcr:DateMapper"
                          minOccurs="0" maxOccurs="1"/>
    <!-- TODO allow dal::StepMapper like here to,
              choice between dateMapper and stepMapper
     -->
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:complexType name="DateMapper">
 <xs:sequence>
    <xs:element name="index" type="xs:unsignedInt">
     <xs:annotation>
       <xs:documentation>"epoch" of the index range
                         typically 1 or 0.
       </xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="timeOfIndex" type="xs:dateTime">
     <xs:annotation>
       <xs:documentation>full calendar date of the index sibling element
       </xs:documentation>
     </xs:annotation>
    </xs:element>
    <xs:element name="duration" type="pcr:TimeDuration">
     <xs:annotation>
       <xs:documentation>duration of a timestep
       </xs:documentation>
     </xs:annotation>
    </xs:element>
 </xs:sequence>
</xs:complexType>


<xs:complexType name="AguilaView">
 <xs:choice maxOccurs="1" minOccurs="1">
   <xs:element name="map"       type="pcr:StringSet"/>
   <xs:element name="drape"     type="pcr:StringSet"/>
   <xs:element name="timeGraph" type="pcr:StringSet"/>
   <xs:element name="probabilityGraph" type="pcr:StringSet"/>
   <xs:element name="valueOnly" type="pcr:StringSet"/>
   <xs:element name="default"   type="pcr:StringSet"/>
   <xs:element name="test"      type="pcr:StringSet"/>
 </xs:choice>
</xs:complexType>

<xs:complexType name="AguilaData">
     <xs:annotation>
      <xs:documentation>
            TODO name is unique, addressing different datasets
            with same "name" but yielding different datasets by
            having a different DataSpace is done by setting another
            path.
       </xs:documentation>
      </xs:annotation>
 <xs:sequence>
   <xs:element name="name"           type="xs:string"/>
   <xs:element name="dataSpace"      type="pcr:DataSpace"
                        minOccurs="0" maxOccurs="1"/>
   <xs:element name="drawProperties" type="pcr:DrawProperties"
                        minOccurs="0" maxOccurs="1"/>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="DrawProperties">
 <xs:sequence>
   <xs:element name="legendBorderValuesType" type="pcr:LegendBorderValuesType"
                        minOccurs="0" maxOccurs="1"/>
   <xs:element name="minimumCutOff" type="xs:double"
                        minOccurs="0" maxOccurs="1"/>
   <xs:element name="maximumCutOff" type="xs:double"
                        minOccurs="0" maxOccurs="1"/>
   <xs:element name="numberOfColours" type="pcr:Non0UnsignedInt"
                        minOccurs="0" maxOccurs="1"/>
   <xs:element name="colourAssignment" type="pcr:ColourAssignment"
                        minOccurs="0" maxOccurs="1"/>
   <xs:element name="drawMode" type="pcr:DrawMode"
                        minOccurs="0" maxOccurs="1"/>
   <xs:element name="palette" type="pcr:Palette"
                        minOccurs="0" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="Palette">
  <xs:sequence>
 <!-- choice
   <xs:element name="default" type="pcr:DefaultPaletteNames"/>
  -->
   <xs:element name="rgb" type="pcr:Rgb" minOccurs="1" maxOccurs="unbounded">
   <xs:annotation>
    <xs:documentation>
     * For nominal,ordinal,boolean data the element number (0-based) is mapped to the
       value. Values outside of [0,nr-of-rgb-elements> are mapped to element number:
       abs(value modulo nr-of-rgb-elements).
     * for scalar and directional data a colour ramp is created.
    </xs:documentation>
   </xs:annotation>
  </xs:element>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="Rgb">
  <xs:sequence>
   <xs:element name="r" type="xs:unsignedByte" minOccurs="1" maxOccurs="1"/>
   <xs:element name="g" type="xs:unsignedByte" minOccurs="1" maxOccurs="1"/>
   <xs:element name="b" type="xs:unsignedByte" minOccurs="1" maxOccurs="1"/>
  </xs:sequence>
</xs:complexType>

<!-- TODO the knows palettes
<xs:simpleType name="DefaultPaletteNames">
  <xs:restriction base="xs:string">
   <xs:enumeration value="netscape"/>
   <xs:enumeration value="greydirectional"/>
   <xs:enumeration value="etc"/>
  </xs:restriction>
</xs:simpleType>
-->

<xs:complexType name="LegendBorderValuesType">
 <xs:choice>
  <xs:element name="rounded" type="pcr:EmptyElement"/>
  <xs:element name="exact"   type="pcr:EmptyElement"/>
 </xs:choice>
</xs:complexType>

<xs:complexType name="ColourAssignment">
 <xs:choice>
  <xs:element name="linear"              type="pcr:EmptyElement"/>
  <xs:element name="trueLogarithmic"     type="pcr:EmptyElement"/>
  <xs:element name="shiftedLogarithmic"  type="pcr:EmptyElement"/>
  <xs:element name="confidenceLevel"     type="pcr:ConfidenceLevel"/>
 </xs:choice>
</xs:complexType>

<xs:complexType name="ConfidenceLevel">
 <xs:sequence>
  <xs:element name="value" type="xs:double">
   <xs:annotation>
    <xs:documentation>
        (1 - alpha)
        TODO: range [0,1]
   </xs:documentation>
  </xs:annotation>
  </xs:element>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="DrawMode">
 <xs:choice>
  <xs:element name="fill"        type="pcr:EmptyElement"/>
  <xs:element name="contour"     type="pcr:EmptyElement"/>
 </xs:choice>
</xs:complexType>

<xs:complexType name="AguilaCursorValues">
 <xs:annotation>
   <xs:documentation>Contents Aguila cursor dump file</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="aguilaCursorValue"  type="pcr:AguilaCursorValue"
                          minOccurs="0" maxOccurs="unbounded"/>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="AguilaCursorValue">
 <xs:annotation>
   <xs:documentation>Aguila current cursor dump</xs:documentation>
 </xs:annotation>
 <xs:sequence>
  <xs:element name="cursor"     type="pcr:Cursor"
                          minOccurs="0" maxOccurs="1"/>
  <xs:element name="dataValue"  type="pcr:DataValue"
                          minOccurs="0" maxOccurs="unbounded">
     <xs:annotation>
       <xs:documentation>y coordinate
       </xs:documentation>
```

```
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="Cursor">
  <xs:annotation>
    <xs:documentation>Current cursor position in the dimensions
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="time" type="xs:unsignedInt"
                            minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation>integer timestep
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="date" type="xs:dateTime"
                            minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation>real calendar date
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="x"    type="xs:double"
                            minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation>x coordinate
        </xs:documentation>
      </xs:annotation>
    </xs:element>
    <xs:element name="y"    type="xs:double"
                            minOccurs="0" maxOccurs="1">
      <xs:annotation>
        <xs:documentation>y coordinate
        </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="DataValue">
  <xs:sequence>
    <xs:element name="name"  type="xs:string"/>
    <xs:element name="value" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

</xs:schema>
```

CHAPTER

EIGHT

FAQ

8.1 Why can't my dataset be opened?

Did you set PCRASTER_DAL_FORMATS (see section *Environment variables*) and is the dataset formatted in a different format than the ones listed by this variable? Solution is to either add the name of this new format to the variable or to unset the variable.

Another possibility is that Aguila currently does not have support for reading the dataset. Aguila contains support for reading many data formats, but not for all data formats. See section *Support* for information about how to request a new feature.

CHAPTER

NINE

SUPPORT

9.1 Comments, questions, ideas, etc

The `pcraster-info` mailing list is a discussion platform for users of PCRaster and related software. The PCRaster R&D team also uses the list to announce new software releases. We recommend that every PCRaster user becomes a member of the list. More information can be found at the pcraster-info mailing list page. The archives are on the pcraster-info archives page.

Once you are subscribed, you can send your comments, questions, ideas and discussion issues to `pcraster-info@geo.uu.nl`. English is the preferred language. Imperfect English is no problem, it is a mailing list, not a classroom English.

9.2 Report a bug

If you think Aguila did something incorrect please file a bug report. Any error messages containing the phrases: “assertion failed” or “programming error” are always errors in the software we should fix. There is a Bugs and feature requests tracker you can use to file a bug report.

Here are some guide lines when submitting a bug report:

- Describe the bug in as much detail as possible.
- Provide (a link to) a zip or archive file with all data and scripts so we can reproduce the bug. If you have a large model (long script and/or much data) try to make a simpler model that still gives the same error.
- Try sending an attachment of your screen showing possible error messages:
  1. Press `PrntScrn` (PrintScreen) on your Keyboard.
  2. Open Paint: `Start->All Programs->Accessories->Paint`.
  3. Copy Screen in untitled: `Paint Menu->Edit->Paste`.
  4. Save graphics File: `File->Save` (type `.PNG` is preferred).
  5. Attach the saved file to your email.

9.3 Request a feature

You can use the same Bugs and feature requests tracker as for reporting bugs.

CHAPTER

TEN

HISTORY

10.1 1.2 (in development)

- X-axis of the probability graph view now scales when the cutoff values of the attribute values are adjusted. (#2907978)
- Time graph view now correctly shows probability values when appropriate. (#2865535 and #2865536)
- Added support for starting Aguila with the probability graph view. Also added support for configuring the probability graph view in the Xml startup configuration file. (#2796370)
- Added support for visualizing *vector data* (attributes with a direction and a magnitude).
- Added support for directories when naming attributes that are located somewhere else than the current directory:

  ```
  $ aguila --timesteps [1,100] MyProject/MyData/Dem
  ```

- Improved panning when a 2D map is zoomed.
- Vector and Ldd attributes are not drawn when there are only a few pixels available per cell (large number of cells and/or zoomed out a lot). In that case a transparant cell is drawn to show the area of non-missing value cells. The vector and ldd directions are shown again when more pixels become available (when the map is zoomed into). This is done to keep Aguila responsive when large data sets are loaded. In such cases vector and ldd directions where not visible anyway.
- Re-added 'zoom by rectangle', by popular request. See section *Map Controls*.
- For all data read using GDAL, the color interpretation is taken into account when determining the value scale of a raster attribute. For example, satellite imagery values stored in a byte geoTiff raster are treated as scalar data when the color interpretation stored in the raster is reported by GDal as being Gray. (#2873963)
- For all data read using GDAL, the PCRASTER_VALUESCALE meta data item is taken into account. If it is set, it overrides the color interpretation and type id that are otherwise used to determine the value scale, see section *Raster data formats*. (#2873989)

10.2 1.1

- Added support for visualizing feature data.

10.3 1.0

First release in the 1-series. Since we are about to add some new features we first made an official release of the current version.

CHAPTER

ELEVEN

LICENSE

Aguila is licensed according to the GNU General Public License, version 2 and is hosted at SourceForge.net. This means it is open source software and all resources needed to build Aguila yourself can be found at the SourceForge project page.