SciCraft User Manual

Version 0.18

2nd July 2006

2006 © SciCraft Development Team

Contents

List of Figures

Chapter 1

Installation

This chapter contains information on how to install SciCraft on both Windows and Linux systems. It also contains a section on programs that can be installed to enhance SciCraft's preformance, but which are not needed for a "barebones" SciCraft installation. The last section is where thirdparty software can be located.

1.1 Windows

On Windows all these packaged is included in the installation file with the exception of Octave, R and MiKTeX. To install SciCraft download the latest version and start the installation file.

1.2 Linux

On Linux the users have to install all third party software from scratch. A complete list of dependencies can be found in Appendix A. After installing all dependencies SciCraft can be installed in one of the following ways:

- With a Debian package: `dpkg -i <scicraft-package>.deb`
- With tar.gz package (python source): `tar zxf <scicraft-package>.tar.gz`
  ```
  cd scicraft-<version number>
  make install
  ```

On a debian system it is also possible to add the following line to `/etc/apt/sources.list` to install using apt. This repository also provides a debian package of python-qwt.

```
deb http://www.scicraft.org/debian/ unstable main contrib
```

The Debian package will install a system wide installation of SciCraft. It will also take care of all dependencies the Debian way.

1.3 Functionality Enhancements

Scicraft can offer more functionality by adding the following optional packages:

- PyQwt version 4.0.rc0
  This package allows scicraft to display interactive two dimensional qwt plots.

- Python Numarray version 0.9
  Needed by PyQwt.
- PyMol version 0.93
  Allows SciCraft to do work on molecules.
- PyX
  Allows SciCraft to export diagrams to Encapsulated PostScript format.
- SOAPpy
  Allows SciCraft to communicate with gene annotation database.
- pyXML version 0.8.3 or later
  Reading data from gene annotation database.
- fpconst version 0.6.0 or higher
  Required by SOAPpy.

1.4 Downloads

- SciCraft: `http://www.scicraft.org/?cat=3`
- PyQt: `http://www.riverbankcomputing.co.uk/pyqt`
- VTK: `http://public.kitware.com/VTK/`
- RPy: `http://rpy.sf.net`
- PyMol: `http://pymol.sf.net`
- Octave: `http://www.octave.org/`
- R: `http://www.r-project.org/`
- Numpy: `http://sourceforge.net/projects/numpy`
- Pyqwt: `http://pyqwt.sourceforge.net/download.html`
- Numpy: `http://pyqwt.sourceforge.net/download.html`
- Numarray: `http://pyqwt.sourceforge.net/download.html`
- PyX: `http://pyx.sourceforge.net`
- SOAPpy: `http://pywebsvcs.sourceforge.net`
- pyXML: `http://pyxml.sourceforge.net`
- fpconst: `http://research.warnes.net/projects/rzope/fpconst/`

Chapter 2

User guide

This chapter introduces the user to the most essential features of SciCraft. A walk through the graphical user interface will be given along with a description of the most common scenarios in SciCraft. We refer to chapter 3 where central SciCraft concepts are explained. It is highly recommended that the users get a clear understanding of these concepts before starting to explore SciCraft.

2.1 Definitions

These are the key terms that a user should have some knowledge about:

Module Diagram: A Module Diagram is a container, placeholder, for nodes that are doing the actual work in SciCraft.

Node: A Node is an component in the Module Diagram that is capable of doing various data processing.

Port: You have two types of ports; input ports and output ports. An input port is capable at receiving data while an output port will send data.

Connection: A connection is the term used for two connected ports. A connection is a one-to-one relation. Please note that the graphical user interface operates with a term called Link which is a representation of, possibly several, connections between two ports.

Parameter: A Parameter is a value/setting which defines a property within a Node. An example could be the number of iteration a for-node will do.

2.2 User interface

When starting SciCraft the user will be presented with something similar to the window seen in Figure 2.1. The items seen in the Figure are the following:

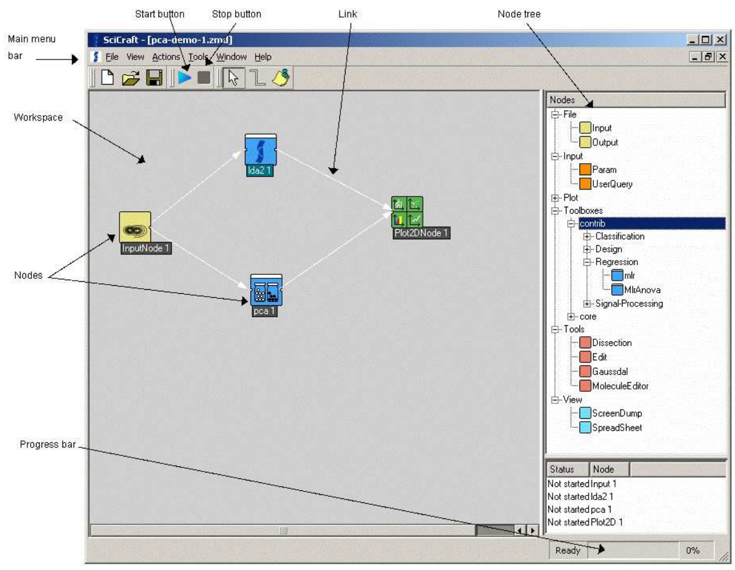
Workspace: The workspace is the working area where you are able to add and connect nodes to generate a data flow.

Nodes: These are the main building blocks in SciCraft. Nodes represent some kind of data methods or operators. E.g. the OctaveFileReader seen on the picture is responsible for reading data into the module diagram, the PCA node is an Octave function running a principal components analysis and so on.

Link: A link represent a connection between two nodes. Please note that the link seen on the workspace is only graphical. As explained later in this section, the nodes can contain several in and output ports. These can been seen and connected by double clicking on the link on the workspace.

NodeTree: The node tree shows all the nodes that can be added to the current diagram.

Progress bar: This bar shows the current status when running a module diagram. It will display the progress when running a diagram, when the diagram is done and also indicate if someone went wrong during execution.



Status	Node
Not started	Input 1
Not started	lda2 1
Not started	pca 1
Not started	Plot2D 1

Figure 2.1: The main window in SciCraft

2.2.1 How to add a node

Start by choosing the particular node you want to add from the node tree and then *double click* on the name. The chosen node will then appear in the leftmost corner of the current workspace. When adding several nodes, they will be placed next to each other horizontally. If a node in a row is moved, new nodes will be placed at the first free place. An alternative to double click on the node is *drag 'n' drop*. This is easily done by choosing a node from the nodetree and then drag it to the desired location on the the workspace while holding down the left mousebutton. The node is dropped where the left mousebutton is released.

2.2.2 How to connect nodes

Before you decide to connect nodes, you have to choose the linking tool from the main menu bar, as shown in Figure 2.2 where the link button is toggled (shortcut: Ctrl+L). This tool is used for connecting nodes ONLY. The button left to the linking button is used to move and mark collections of nodes, the pointing tool (shortcut: Ctrl+P). This tool is also used when editing nodes and links.

It is also possible to create links between nodes when using the pointingtool. By clicking on the *right mousebutton* on a node a small menu will appear, as illustrated in Figure 2.3. 'Create link' enables you to make a connection between the chosen node and any node in the workspace. I addition to this, it is possible to choose from the menu to *remove* and *view* the current node.

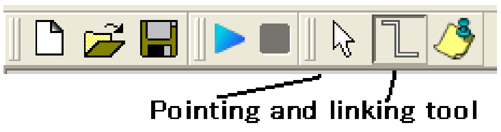


Figure 2.2: Pointing and linking tool

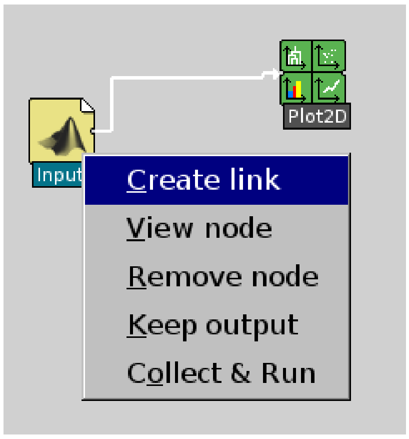
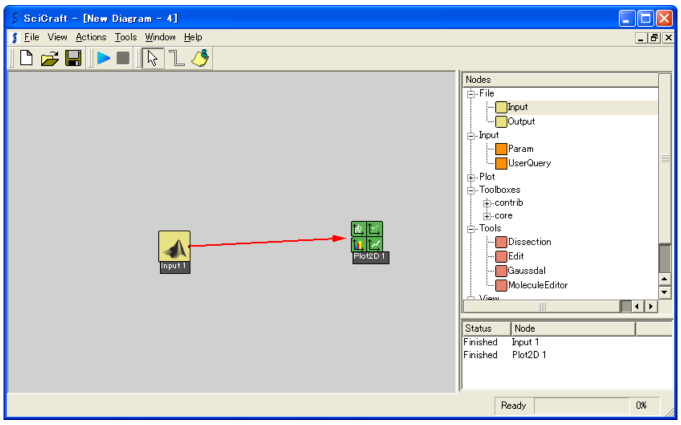


Figure 2.3: Creating links when using the pointing tool

When connecting nodes you start at the **FROM** node, left click on this node and then left click on the **TO** node. A line will now be drawn between these two nodes and they are linked together, as illustrated in Figure 2.4. If you look closer on the nodes you will notice that most node images has one or more *slots* on the edge of the node icons. When you left click on the nodes, the line will start (or end) at the slot closest to where you just clicked.



Status	Node
Finished	Input 1
Finished	Plot2D 1

Figure 2.4: Connecting nodes

After connecting two nodes in the GUI you are now ready to set the input and output to and from the nodes. This means that you have to switch to pointing tool. A node is capable of having several input and output ports. These ports have a name and often a type and description. When connecting two node ports you have to know which port in the **FROM** node that should be connected to a specific port in the **TO** node. If you double click on the link you just created you will see a window similar to Figure 2.5. Specify which ports you want to connect by clicking one output and one input port and then click on *Connect*. A similar operation is repeated when disconnecting ports, select ports and click *Disconnect*.

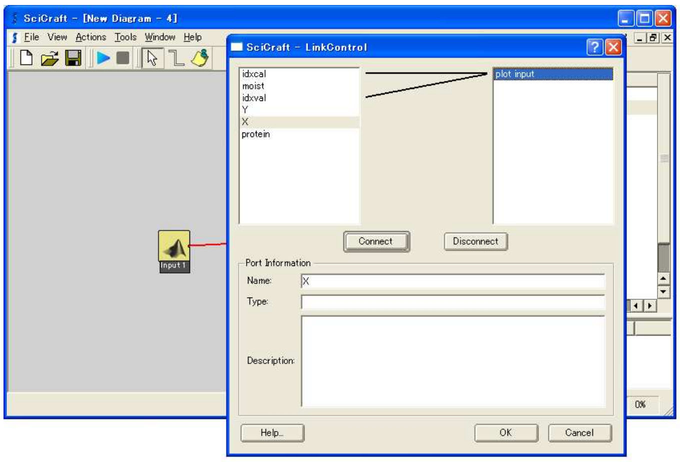


Figure 2.5: Connecting ports

So just using a link to connect two nodes does not specify any flow of data between the nodes, you have to double click on a newly created link and specify the port connections in order to define the flow of data between the nodes. An exception to this rule occurs when you connect two nodes with only one output and one input port. In these cases, the ports are automatically connected when the node link is created.

The difference between node connections (links) and port connections is illustrated in Figure 2.6. The node connections are specified by drawing links between nodes as described above. Port connections are defined by double clicking on links and connecting ports as in Figure 2.5.

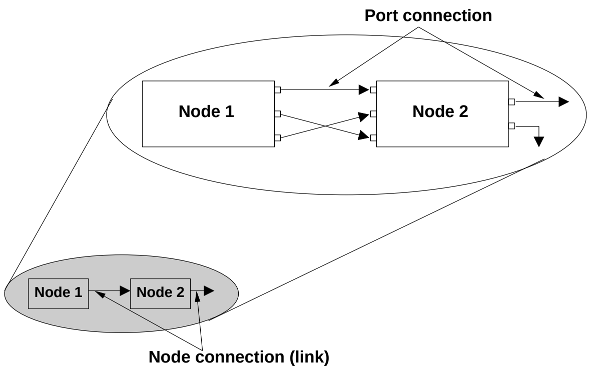


Figure 2.6: Node links and port connections

2.2.3 How to add a comment and modify comment settings

To add a comment, select the tool icon represented by a post-it note in the toolbar - shown in figure 2.7 - and then *click* the desired position in the workspace. A comment now appears in the workspace where the mouse was *clicked*. To change the settings of this comment, simply *doubleclick* it and a settings window will pop up, as shown in figure 2.8. This window gives you the opportunity to modify the following:

Comment Text: This may be modified by changing the text field just below the heading *Comment text*.

Comment Font type: This may be modified by clicking the leftmost button in the toolbar. This button has an icon with *Abc* displayed on it. See figure 2.9. If this button is *clicked* a standard window for modifying font will appear.

Comment Font colour: This may be modified by clicking the second button from the left in the toolbar. This button has an icon with *two different coloured T's* displayed on it. See figure 2.9. If this button is *clicked* a standard window for modifying colour will appear.

Comment Background colour: This may be modified by clicking the rightmost button in the toolbar. This button has an icon with *three different coloured folders* displayed on it. See figure 2.9. If this button is *clicked* a standard window for modifying colour will appear.

As you change the comment attributes a preview field below the heading *Comment preview* will be updated for convenience. However, changes are not applied to the actual comment until the *Ok*-button is *clicked*. The *Cancel*-button may also be *clicked* if it is desired to cancel all changes done to the comment's attributes.

Also comments may be selected and moved around by using the *pointing tool*. Selected comments may also be removed by using the *delete*-key on the keyboard.

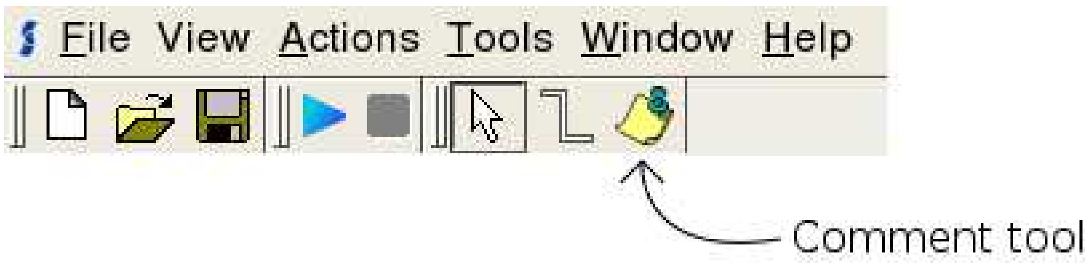


Figure 2.7: Comment tool

2.2.4 How to select and move collection of nodes

First of all, to be able to mark a collection of nodes, or even one node, you have to choose the pointing tool, as illustrated in Figure 2.2. Nodes placed in the workspace, linked or not, can be moved, one by one or as a collection of several nodes. Begin by setting the mousepointer on the desired starting point and hold down the *left mousebutton*. Now a dotted rectangle will appear on the workspace. You are now free to drag the dotted rectagle over the nodes you want to move. When the nodes you want to mark are covered by the rectangle, release the left mousebutton, see Figure 2.10 for illustration. The colour of the selected nodes labels will change, as an indication of the selection. Set the mousepointer on one of the selected nodes, hold down the *left mousebutton*, and move the selected nodes wherever you want in the workspace.

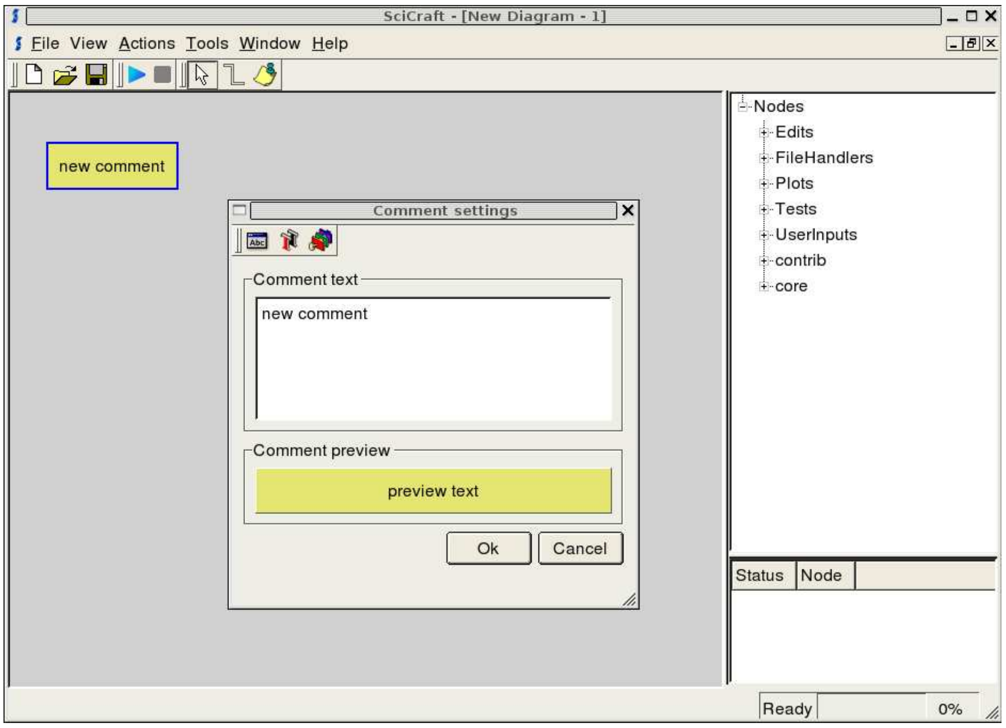


Figure 2.8: Comment settings

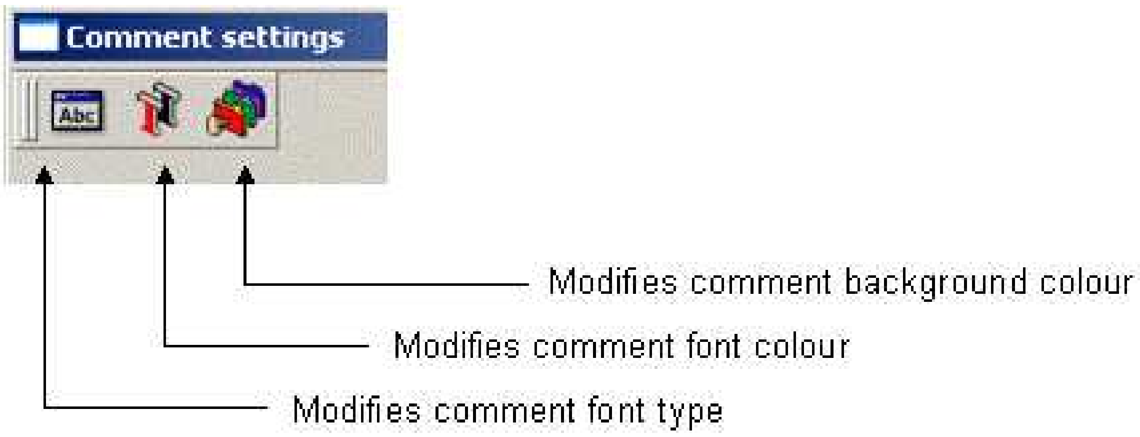

Comment settings
Abc
Modifies comment background colour
Modifies comment font colour
Modifies comment font type

Figure 2.9: Comment settings toolbar

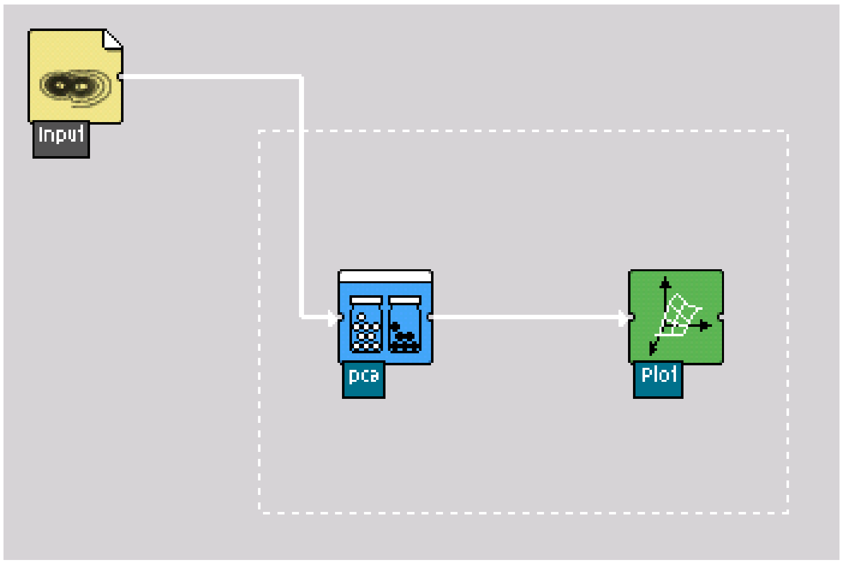
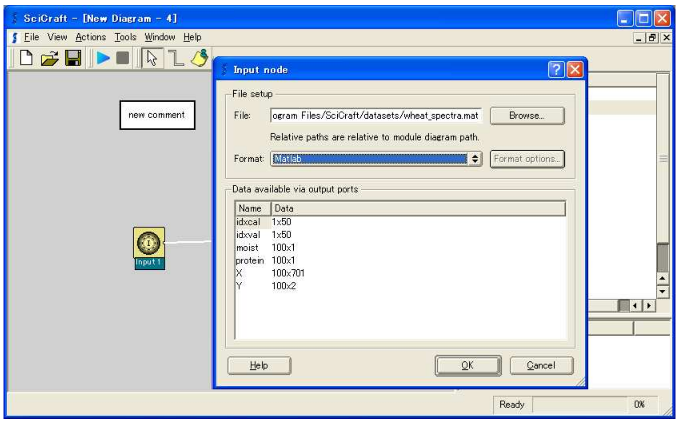


Figure 2.10: Selection of nodes

2.2.5 How to get input and output data

In most cases you need to import some kind of dataset into the module diagram. This could for instance be a matrix from a Matlab file or data from an R-object. To be able to import this data into the diagrams you need an input node. This node is a specialised reader node that is able to read different kinds of dataformats. Create an input node, which is a subnode of *filehandlers*, and double click on it to get the node dialog, as illustrated in Figure 2.11. Continue by choosing the desired input file. If the chosen file is supported by Scicraft, the corresponding file type and file type description will we shown. For output you repeat this process with a file writer node.



File setup

File: ogram Files/SciCraft/datasets/wheat_spectra.mat

Relative paths are relative to module diagram path.

Format: Matlab

Data available via output ports

Name	Data
idxcal	1x50
idxval	1x50
moist	100x1
protein	100x1
X	100x701
Y	100x2

Figure 2.11: Getting input from files.

2.2.6 How to run a module diagram

Running a diagram is done by clicking the *Play* button on the toolbar, or by clicking *Run* via the *Actions* menu.

When running a diagram, SciCraft will first validate the diagram. If the diagram fails the validation, a pop-up with information about errors will appear. The most common error when validating is missing connections on an input port.

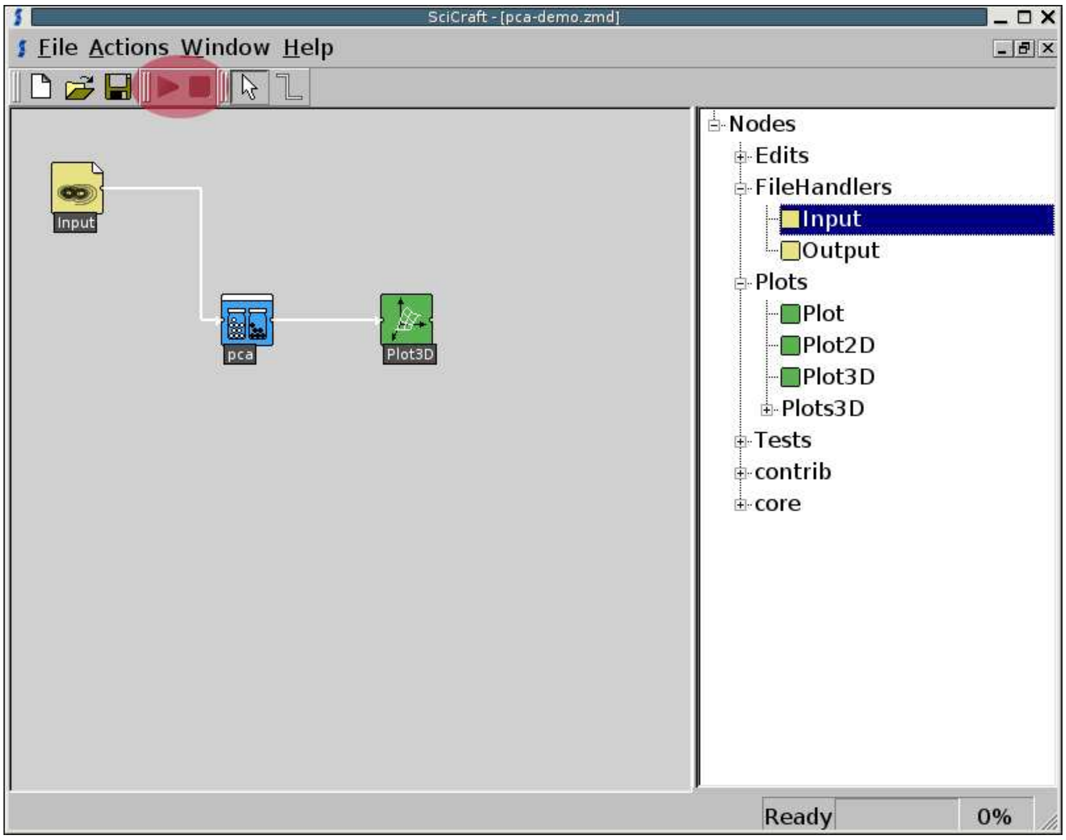


Figure 2.12: Running a module diagram.

2.2.7 How to run a single node

Running a single node is accomplished by right-clicking the node and selecting *Collect & Run*. As opposed to regular diagram running, where nodes send data to next connected node, this collects data from all nodes providing data and then runs the node.

The *Collect & Run* functionality depends on all nodes providing data keep the output from their last run. This setting must be enabled manually on all nodes providing data to the single node. To enable this setting, select *Keep output* after right-clicking the node.

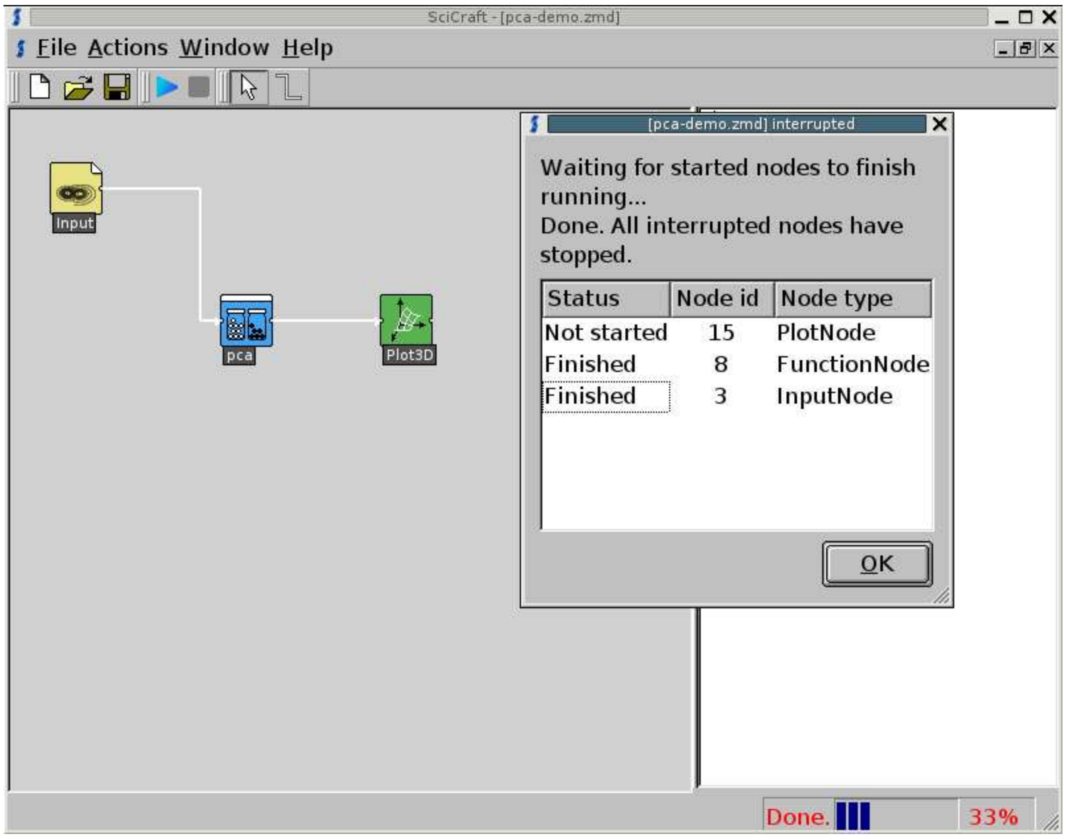
Using *Keep Output* and *Collect & Run* it is possible to build a single node, enable *Keep output*, then run it with *Collect & Run*, then create a second node, connect it to the first, enable *Keep output* and then run it with *Collect & Run*. By creating a series of nodes, and running them as you go along, you can inspect return values along links, change parameters, and interactively create an entire diagram without having to run all nodes several times.

This is particularely useful when dealing with nodes that use an extended amount of time for calculations.

The reason that the *Keep Output* setting of each node is not enabled by default is that keeping output data of all nodes readily available might require alot of memory or disk on a system, depending on your data sets. Keeping all data might deplete your systemś resources.

2.2.8 How to stop a running diagram

After you have started a diagram by clicking the *Play* button on the toolbar, or by clicking *Run* via the *Actions* menu, you may stop the diagram while it is running. This is for example quite useful if you by a mistake started a diagram that takes a long time to run, with, say, the wrong parameters set in a node. Stopping the diagram may be done either by pressing the *Stop* button on the toolbar, or by clicking *Stop* via the *Actions* menu. The *Play* and *Stop* buttons are shown in Figure 2.12.



Waiting for started nodes to finish running...
Done. All interrupted nodes have stopped.

Status	Node id	Node type
Not started	15	PlotNode
Finished	8	FunctionNode
Finished	3	InputNode

Figure 2.13: Stopping a module diagram and the status of the nodes after the stop button has been pressed.

When the *Stop* button has been pushed the dialog box in Figure 2.13 appears on the screen. At the same time the text in the progress bar at the bottom of the SciCraft window turns red and shows the text *Interrupted* and the *Stop* button is disabled. This is when the running nodes are stopping themselves. Some nodes may use longer time than others to stop. As the running nodes finish running, they change their *State* in the dialog box from *Running* to *Interrupted*. When all the running nodes have finished running, the text in the progress bar changes to *Done* and the text inside the dialog box says that it is done interrupting the nodes. Now the *Play* button is enabled again.

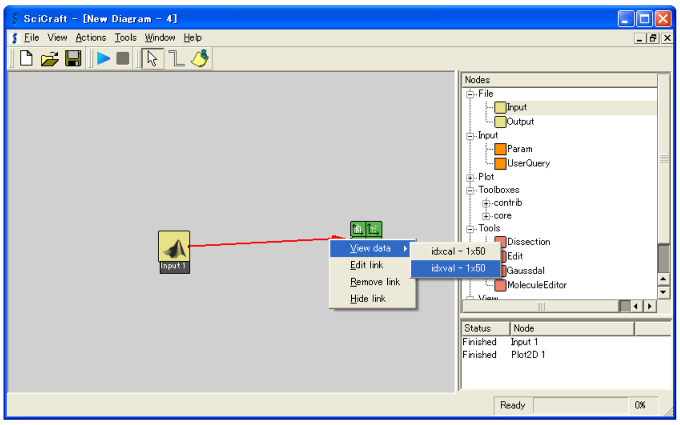
What happens when you press the stop button during a diagram run, is that the nodes that are not yet started at the time when the button was pressed are not started at all. Those nodes that are possible to stop are stopped in a nice way, those that are not are allowed to finish. The nodes that were finished before the button was pressed are finished and will contain valid data.

During a diagram run, it is possible to switch to another window within SciCraft and work on another diagram, or even run it, whatever state the first diagram is in. One could also press the *OK* button in the dialog at any time, but it will not be possible to run the diagram again until all running nodes are safely stopped.

2.2.9 How to inspect data flow in links

.

After the module diagram has been run it is possible to inspect the data flowing through the different links. This option is located in the *links context menu* (see figure 2.14). This menu is reached by *right clicking* on links.



Status	Node
Finished	Input 1
Finished	Plot2D 1

Figure 2.14: Link context menu

The *View data* sub-menu lists the variables that flows through the higlighted link. Each item in the list is on the format `variable-name - info` where info can be one of, or a combination of, the following (see figure 2.14 for an example):

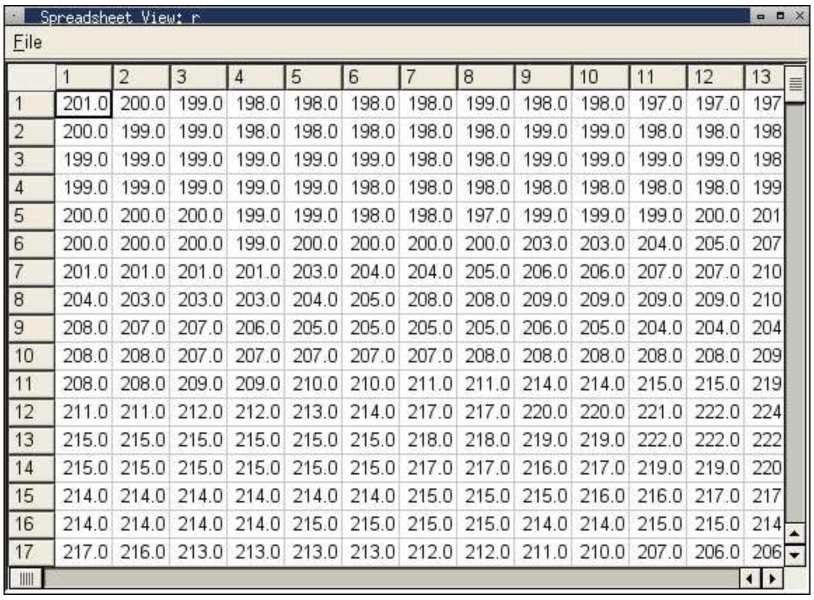
- String "String"
- Scalar "scalar"
- Matrix "rows × columns (× <any additional dimensions>)"
- Vector "1 × <number of items>"

By clicking on one of the variables in the *View data* sub-menu, you will find a dialog box with a spreadsheet popping up. The data in the spreadsheet is the data that flows through the link, in the variable you clicked on. An example of such a dialog box is shown in figure 2.15.

SciCraft enables you to save the data that is in a spreadsheet, such as the one shown in figure 2.15. By selecting the *File* menu's *Save data...* option, you may save the data in the spreadsheet, using all the formats that SciCraft supports for output (see table 3.1 for a list of the formats that SciCraft supports).

2.2.10 How to inspect data in a node

The data available in a node can be inspected in the node context menu, which is accessed by right-cliking the node. The menu item 'Inspect data' is located at the bottom of the node



	1	2	3	4	5	6	7	8	9	10	11	12	13
1	201.0	200.0	199.0	198.0	198.0	198.0	198.0	199.0	198.0	198.0	197.0	197.0	197
2	200.0	199.0	199.0	198.0	198.0	198.0	198.0	198.0	199.0	199.0	198.0	198.0	198
3	199.0	199.0	199.0	199.0	199.0	199.0	198.0	198.0	199.0	199.0	199.0	199.0	198
4	199.0	199.0	199.0	199.0	199.0	198.0	198.0	198.0	198.0	198.0	198.0	198.0	199
5	200.0	200.0	200.0	199.0	199.0	198.0	198.0	197.0	199.0	199.0	199.0	200.0	201
6	200.0	200.0	200.0	199.0	200.0	200.0	200.0	200.0	203.0	203.0	204.0	205.0	207
7	201.0	201.0	201.0	201.0	203.0	204.0	204.0	205.0	206.0	206.0	207.0	207.0	210
8	204.0	203.0	203.0	203.0	204.0	205.0	208.0	208.0	209.0	209.0	209.0	209.0	210
9	208.0	207.0	207.0	206.0	205.0	205.0	205.0	205.0	206.0	205.0	204.0	204.0	204
10	208.0	208.0	207.0	207.0	207.0	207.0	207.0	208.0	208.0	208.0	208.0	208.0	209
11	208.0	208.0	209.0	209.0	210.0	210.0	211.0	211.0	214.0	214.0	215.0	215.0	219
12	211.0	211.0	212.0	212.0	213.0	214.0	217.0	217.0	220.0	220.0	221.0	222.0	224
13	215.0	215.0	215.0	215.0	215.0	215.0	218.0	218.0	219.0	219.0	222.0	222.0	222
14	215.0	215.0	215.0	215.0	215.0	215.0	217.0	217.0	216.0	217.0	219.0	219.0	220
15	214.0	214.0	214.0	214.0	214.0	214.0	215.0	215.0	215.0	216.0	216.0	217.0	217
16	214.0	214.0	214.0	214.0	215.0	215.0	215.0	215.0	214.0	214.0	215.0	215.0	214
17	217.0	216.0	213.0	213.0	213.0	213.0	212.0	212.0	211.0	210.0	207.0	206.0	206

Figure 2.15: Inspection of link data

context menu. If there are no data available in the node, the menu item will be disabled (greyed out).

If there are data available in the node, a menu with the items 'Input ports' and 'Output ports' is displayed when you activate the 'Inspect data' item. If one of them is disabled, there are no ports of the specified type with data on them. By activating one of the items, a list of ports will be displayed. Beside the port name, the type and/or size of the data is shown. By clicking on a port item, a spreadsheet displaying the port data will be opened. It is possible to save the data in the spreadsheet to file, refer to section 2.2.9 for more info on saving data.

2.2.11 How to inspect output from a function node

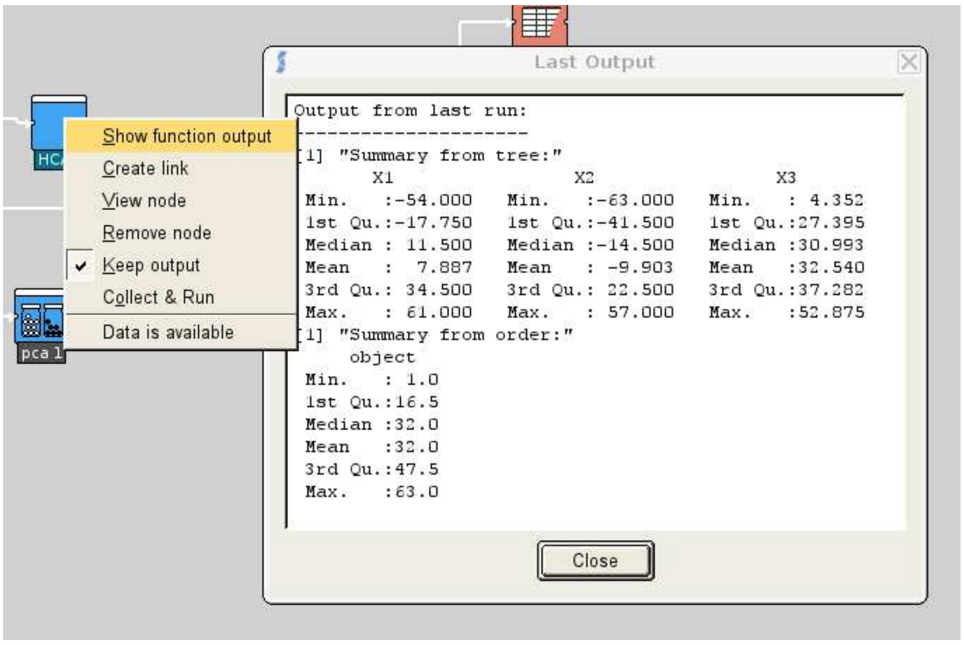
When a module diagram has been run, it is possible to inspect what the function has printed during the last execution of the diagram. This is useful when developing new functions for SciCraft (see section 2.4.1 for details on how to write functions for SciCraft). E.g., some find it useful to print summaries of objects to the screen, when developing and using functions written in R. This output will not be printed to the console when running it under SciCraft. Instead the output is available by right-clicking on a function node (as shown in figure 2.16) and selecting the *Show function output* option. Then the output that would normally have been printed to the console during the last execution will be shown in a dialog box, as shown in figure 2.16.

The output shown in the dialog box in figure 2.16, was generated by inserting the following into a R script (in this case the `HCA.R` file):

```
print(summary(an_object))
```

2.2.12 How to open a plot window

Plot windows are opened by double clicking on the node. If you have not executed the diagram, the plot will be empty. For more on plotting, refer to section **??**.



```
Output from last run:
--------------------
[1] "Summary from tree:"
       X1                 X2                 X3
 Min.   :-54.000   Min.   :-63.000   Min.   : 4.352
 1st Qu.:-17.750   1st Qu.:-41.500   1st Qu.:27.395
 Median : 11.500   Median :-14.500   Median :30.993
 Mean   :  7.887   Mean   : -9.903   Mean   :32.540
 3rd Qu.: 34.500   3rd Qu.: 22.500   3rd Qu.:37.282
 Max.   : 61.000   Max.   : 57.000   Max.   :52.875
[1] "Summary from order:"
     object
 Min.   : 1.0
 1st Qu.:16.5
 Median :32.0
 Mean   :32.0
 3rd Qu.:47.5
 Max.   :63.0
```

Figure 2.16: Showing the output of a function node.

2.2.13 How to export diagrams to Encapsulated PostScript format

The Python Pyx package must be installed for this option to work. On a Windows platform this may be bundeled with SciCraft, depending on which SciCraft package is chosen for download. Choose *File* in the main menu bar and then *Export diagram to EPS* from the file menu. From the save file dialog, you can choose to save it as an image in black and white or with colours. The current diagram will then be saved in the Encapsualted PostScript (*.eps) format.

2.3 Changing SciCraft Settings

Many important settings in SciCraft can be changed through the Preferences dialog box. To access the settings choose *File* in the main menu, then *Preferences*. The preferences dialog contains three different tabs with different settings the user can modify. Each tab is explained in the following subsections.

2.3.1 Modifying general preferences

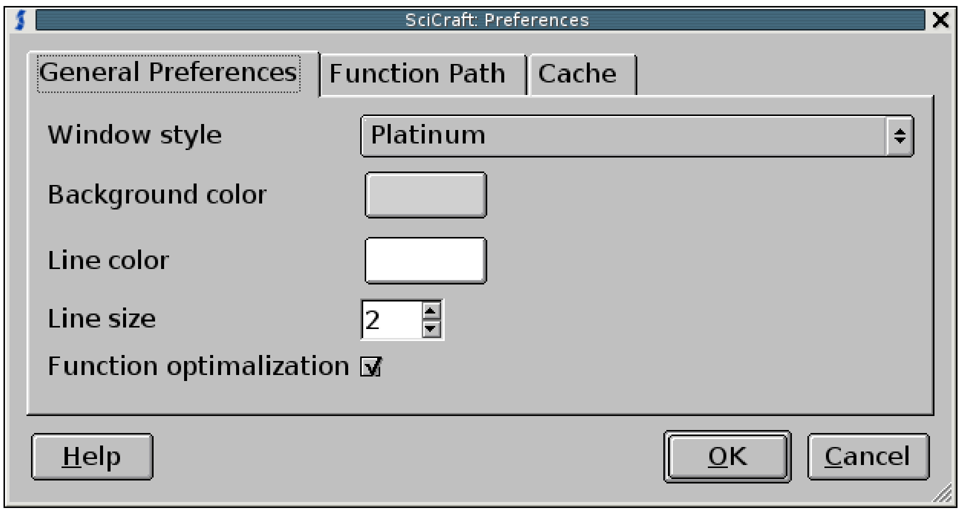
The *General Preferences* tab contains settings allowing the user to adjust the user interface, including settings width and color of lines and background color in the module diagram. It is also possible to change the current look of the application to match a specific operating system.

2.3.2 Function optimalization

In the *General Preferences* tab you have the option to enable function optimalization. This feature comes into play when you have several function nodes connected. These nodes will then be combined to a single function script that can be executed within a single process. So far this feature only affects Octave function nodes.

2.3.3 Modifying function paths

If you store your functions in other locations than the directory “Functions” under SciCraft’s home directory, you need to add the locations in SciCraft. To add new paths, access the *Function Path* tab (similar to Figure 2.17). A list of your current paths are displayed. To add a new path, write it in the editline, or click “Browse” to select it in a filebrowser, and use the add button to add it to the list. All functions within the directories should now be available as nodes in SciCraft.



Window style: Platinum
Background color
Line color
Line size: 2
Function optimalization ✓

Figure 2.17: SciCraft preferences, and changing paths.

2.3.4 Modifying cache settings

When using *Collect & Run*, as described in section 2.2.7, it is possible to store data to disk when exiting SciCraft. This makes the kept output data available when starting SciCraft next time, and makes it possible to use *Collect & Run* on nodes dependent on that data without having to run the previous nodes.

To enable storing such data between runs of SciCraft, the program cache has to be enabled. The cache is enabled by default and data is stored in a subdirectory within the users home directory. Since the amount of data can be quite large in some calculations, it is possible to change the cache path to another directory or clear all data in the currently set cache directory.

When setting the cache to a specific directory, all cache entries will be stored in a subdirectory called *cache*. When clearing the cache, only files in this subdirectory will be deleted.

Changing the warning limit will not affect how or when cache is stored, but it will report a warning to the user if the limit is exceeded, allowing the user to clear the cache, or turn it off. Setting the limit to 0 turns warnings off.

For more information on *Collect & Run* functionality, see section 2.2.7.

2.4 Using third party functions in SciCraft

This section deals with integrating your own functions or tool boxes to SciCraft. To be able to do so, SciCraft must support the programs associated with the language you write the function in. SciCraft currently supports the following: Octave, R and Python. To be able to integrate the functions in SciCraft, you sometimes have to follow certain guidelines when writing the functions. See section 2.4.1 for further information.

Making your functions accessable in SciCraft is done by creating a XML-file (refered to as zml files). This is usually done by using the built on ZMLCreator that you will find on the *File* menu, see section 2.4.2. If your files are located in another location than <path-to-scicraft>/Functions, you have to add the path to the path list that SciCraft searches through when adding all the functions. This is done by adding the path via the SciCraft loadpath form (section 2.3.3).

Your function scripts may print text to the screen when they are run. This output will not be visible in your terminal when you run the function under SciCraft. Instead the output will be collected when the function is executed, and the output is available by right-clicking the function node, and selecting *Show function output* (see section 2.2.11 for further details).

2.4.1 Writing functions for SciCraft

Currently, R, Octave and Python functions are accepted by SciCraft. Octave is very similar to Matlab^{TM}, so users who know Matlab^{TM} should easily be able to port their functions to Octave. In most cases, no changes are neccessary. As SciCraft is an open source project, we assume users will contribute by writing plugins for other appropriate systems as well.

R

SciCraft should be able to handle all R functions with some modifications. When writing functions in R, you have to have a return statement which creates names for each returned value. This can be done by using the following statement:

```
return(list(name1=val1, name2=val2, ... , nameN=valN))
```

Example:

```
mlr <- function(formula,rawdata)
{
c <- lm(as.formula(formula),rawdata)
return(list(mlrdata=c))
}
```

Python

When importing a Python function into SciCraft there are a few things to keep in mind.

1. The function node you create needs both an input and an output port.
2. The input port(s) that you create has to have the same name(s) as the parameters in your python function.
3. All import statements have to be done inside the python functions.
4. You can only return one variable from your function. If you need to return more than one variable, add these to a dictionary that you return as your result.

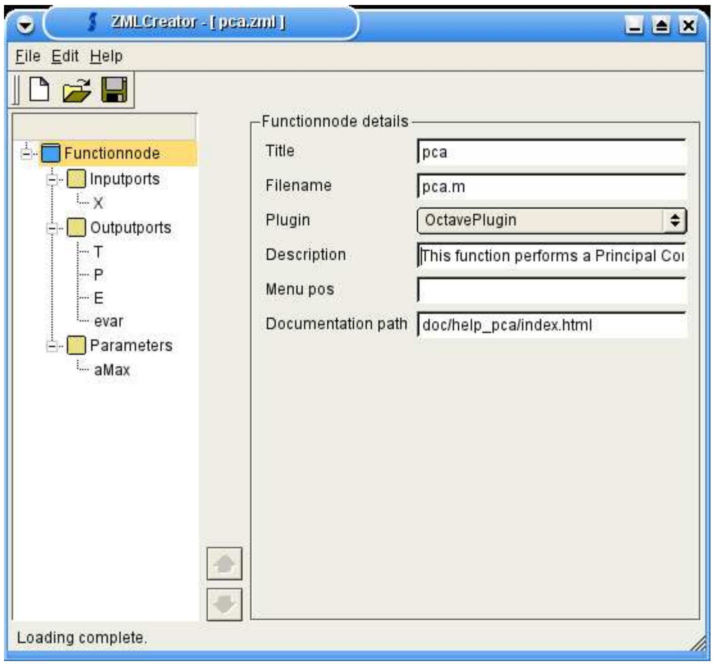
Below is a short example of a python function and a ZML description of this function:

```
def test(number1, number2):

    import math
    result = {}

    result1 = math.sin(number1)
    result2 = sunNumber * number2

    result[‘‘result1’’] = result1
    result[‘‘result2’’] = result2

    return result

<function>
    <title>test</title>
    <filename>test.py</filename>
    <description>Just a short test</description>
    <plugin>PythonPlugin</plugin>
    <input>
        <port>
            <name>input1</name>
            <type>Integer</type>
            <description>Math.sin calculation</description>
        </port>
        <port>
            <name>input2</name>
            <type>Integer</type>
            <description>Multiplier</description>
        </port>
    </input>
    <output>
        <port>
            <name>result1</name>
            <type>Float</type>
            <description>Partial result, multiplier</description>
        </port>
        <port>
            <name>result2</name>
            <type>Float</type>
            <description>Final result</description>
        </port>
     </output>
</function>
```

2.4.2 Importing functions into SciCraft

Creating ZML files

To use a function in SciCraft, you will need a zml file which describes the function. These can either be created manually, or by using the supplied ZMLCreator (Figure 2.18). Be warned that SciCraft may not start correctly if it encounters bad zml files. The ZMLCreator can be started through the SciCraft file menu. The text field at the bottom of the screen will show what the zml will look like when saved.



Functionnode details

Title	pca
Filename	pca.m
Plugin	OctavePlugin
Description	This function performs a Principal Coı
Menu pos	
Documentation path	doc/help_pca/index.html

Figure 2.18: The ZMLCreator.

One of the easiest ways to understand what the different variables mean, and how they work, may be to look at the already existing functions. If you want to view a zml file through the ZMLCreator, just select open through the file menu. The already existing functions will be available under <path-to-sicraft>/toolboxes/.

Functionnode settings

Under the starting screen (Functionnode), there are several values you can set:

- Title: This will be the name of the Function, which will be displayed in the GUI.
- Filename: This shall be the path to the file were the function you want to run is situated. The active directory will be set to the file the zml file resides in, so if the other file is in the same directory, just writing its name will be sufficient.
- Plugin: The exact name of the plugin you want to run, currently OctavePlugin, RPlugin and PythonPlugin. All available plugings will reside int he directory Library/Plugins. Even though a combobox is used for selection, you may write whatever you want here.

- Description: The text which will show up under the description field when you open a node dialog.
- Menu Pos.: Normally, a node's position in the node-tree will be decided by the directory it's in. If this value is set however, it will be placed there instead. To indicate several directories, please use ',' as a seperator (this may be changed later).
- Documentation path: Path to documentation concerning this functionnode.

Inputport settings

To add an Inputport please select add inputport under the edit menu. Values that may be set are:

- Name: The name of the inputport as well as the name under which the data will be made available to the function.
- Description: Whatever you write here, will show up on under the description fields in SciCraft.
- Named: Determines whether the port name should be send as a named argument to the target function.

Outputport settings

To add an Outputport please select add outputport under the edit menu. Values that may be set are:

- Name: The name of the outputport as well as the variablename SciCraft will check for data.
- Description: Whatever you write here, will show up on under the description fields in SciCraft.
- Subobject: Associate a part of the output from a R function with this port, e.g. the string *out$version* will extract the attribute *version* from the return value *out*. *(Only available when plugin type for the functionnode is set to RPlugin).*

Parameter settings

To add a Parameter please select add parameter under the edit menu. All parameters may have restrictions placed upon them. The values you can set means the following:

- Name: The name of the parameter as well as the name under which the data will be made available to the function.
- Description: Whatever you write here, will show up on under the description fields in SciCraft.
- Type: The valid types are: Integer, Float, Text, Fileopen, Filesave. There are no maximum limit for Integer(or rather Long objects which large values will be stored as) or Text objects. Python uses double-precision Floats. If type is Fileopen or Filesave, the user will be prompted with a Filedialog in the GUI, and the parametervalue will be set to the path.

- Min/Max: For integers and floats, these will be restrict the values the user may set the parameter to. Please use '.' and not ',' as seperator.
- Min/Max: For strings, these will be the max/min number of chars the value can have.
- Step: If set for integers and floats, a step value may be set.
- Default: Available for all types; this value will simply be the default value which is showned when no value has been set yet.
- Restrictions: If set, this options will override all other restrictions and the user will be prompted with a ComboBox in which he may select one of the values from this list.

The following example shows a function that will use a PCA script with two input and three output ports:

```
<function>
    <title>pca</title>
    <filename>pca.m</filename>
    <description>This function performs a Principal Components Analysis</description>
    <plugin>OctavePlugin</plugin>
    <input>
        <port>
            <name>X</name>
            <type>Text</type>
            <description>Input data matrix</description>
        </port>
        <port>
            <name>aMax</name>
            <type>Integer</type>
            <description>Number of components</description>
        </port>
    </input>
    <output>
        <port>
            <name>t</name>
            <type>Integer</type>
            <description>Scores</description>
        </port>
        <port>
            <name>P</name>
            <type>Integer</type>
            <description>Loadings</description>
        </port>
        <port>
            <name>E</name>
            <type>Float</type>
            <description>Residuals</description>
        </port>
     </output>
</function>
```

2.5 Plotting in 2D

This section explains the 2D plotting functionality in SciCraft.

Plot window: is the whole window. This window will appear whenever you double click on a plot node in the module diagram.

(Plot 2D node: is a node located under the "Plots" part of the node list in the main SciCraft window. When you double click a Plot2D node in the node list, a green node will appear in your workspace. Double clicking this green node again will result in the plot window appearing.)

Render window: or Plot renderer is the area in which a plot is drawn and displayed. As you can see, a plot window can contain several render windows and a render window can contain several plots.

Plot: is a graphical representation and display of some data. In SciCraft a plot must reside inside a render window to be displayed.

Plot window control: is the part of the plot window that controls what kind of plots that should be displayed, and how they should be displayed. We will come back with a detailed description of the plot window control later.

2.5.1 Plot 2D Window

The plot 2D window is the whole node window. It contains other elements like render windows and window controllers. The render windows are located at the top of the plot 2D window, and the window controller at the bottom. Figure 2.19 shows an example of a plot 2D window.

From the menubar at the top left corner of the plot 2D window, various features of the plot 2D window can be controlled. Table 2.1 shows the different options available.

The render window

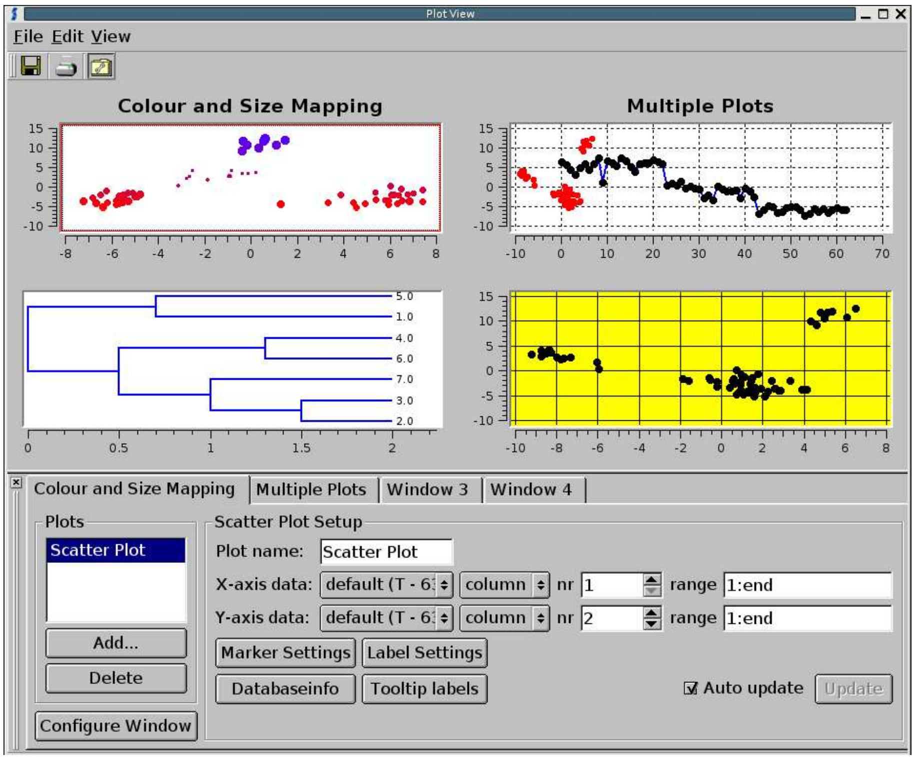
The render window contains the 2D plots. Each render window has a pair of x- and y-axis. A plot window can contain several render windows. The number of render windows can be controlled from the settings option in the pull down menu at the top left corner of the plot 2D window. See 2.5.4 for a more detailed description. Figure 2.20 show the render windows for the previous example.

A red border around the render window indicates that this window is active. To activate a render window, you can either click on the render window or select the tab corresponding to the correct render window in the window controller. When a render window is active it means that you can add and remove plots to the window and change settings for the render window and the plots it contains. Only one render windows can be in the active state simultaneously.

Window controller

The window controller is central to the control of the different render windows and plots in a plot window. The window controller consists of several tabs, one for each render window in the plot window. Figure 2.21 show the window controller for the previous example.

A window controller tab consists of two main groups of controllers. The control options for the entire render window is placed to the left, and the control options for each plot in the render window is placed to the right in the window controller.



Plot name: Scatter Plot
X-axis data: default (T - 6: column nr 1 range 1:end
Y-axis data: default (T - 6: column nr 2 range 1:end
☑ Auto update

Figure 2.19: Plot 2D Window

The left side of the window controller contains a list of all plots in the current render window.

Menu	Menu item	Shortcut key	Description
File	Save Custom Plot As	None	Saves a custom plot
File	Print Selected	Ctrl + P	Prints the plot in the plot window that is selected
File	Print All	None	Prints all plots in the plot window on the same sheet
File	Export	None	Submenu with options to export the current plot, or all plots, to the EPS format
File	Close Window	Ctrl + W	Closes the plot window
Edit	Settings	Ctrl + S	Open the plot setup dialog 2.5.4
View	Toggle Administrative Toolbar	None	Shows or hides the toolbar
View	Toggle Configuration Window	Ctrl + T	Shows or hides the window controller 2.5.1
View	Refresh	F5	Updates plot data, i.e. after running the modulediagram
Tools	Gene Ontology Browser		Connection to Gene Annotation Database
Tools	Data Composer		Inspect data restricted by selection

Table 2.1: Plot 2D menubar

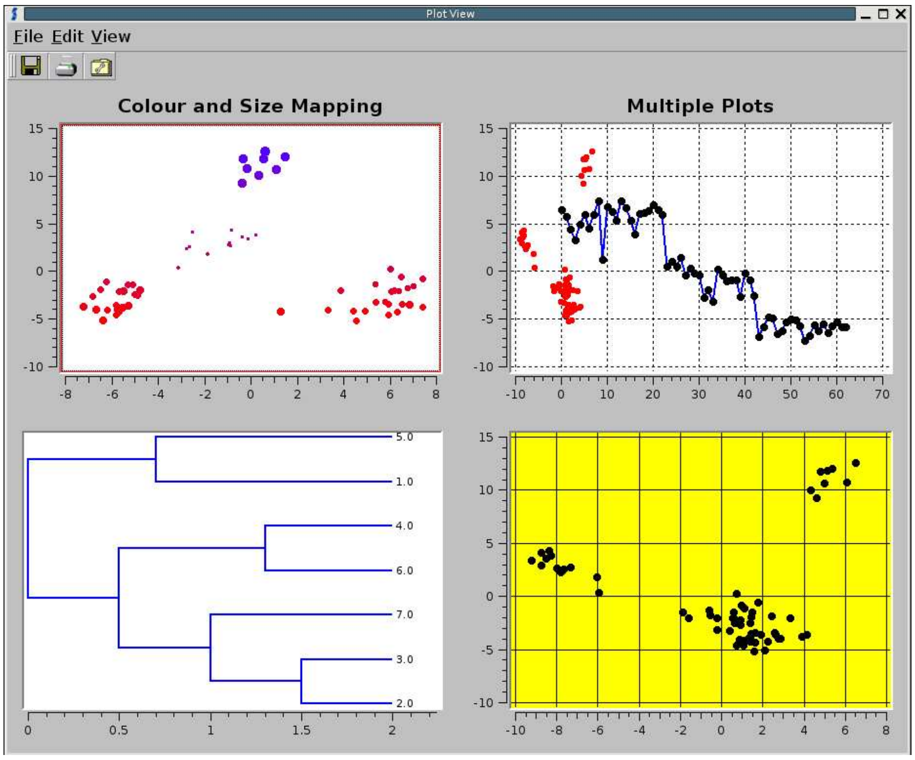
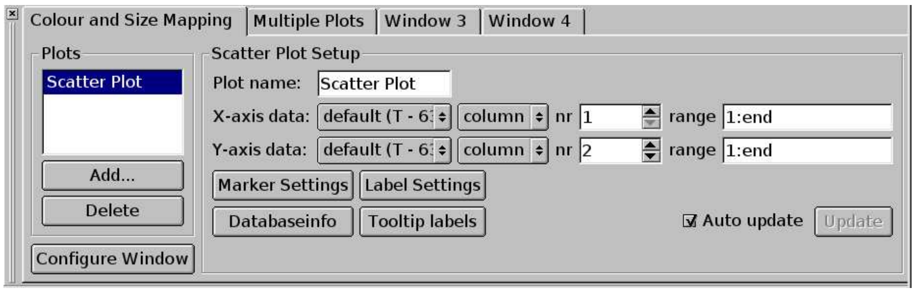


Figure 2.20: Render Window



Scatter Plot Setup

Plot name: Scatter Plot

X-axis data: default (T - 6 column nr 1 range 1:end

Y-axis data: default (T - 6 column nr 2 range 1:end

☑ Auto update

Figure 2.21: Window controller

It also contains options for adding and deleting plots from the render window. Push the “Add” button to add a new plot to the window. From the “Add plot” dialog box, select which plot type to add to the render window. For a detailed description of the different plot types, see 2.5.2. The new plot will appear in the plot list. To delete a plot from the plot window, highlight the correct plot in the plot list and push the “Delete” button.

You can highlight one of the plots in the plot-list, and the plot setup dialog will appear to the right in the window controller. The plot setup dialog makes it possible to set the name of the selected plot. Default name is the name of the plot type. To specify which port x- and y-axis data comes from, select the correct ports from the “X-axis data” and “Y-axis data” drop down lists. Which column or row in the dataset to plot, is specified in a similar way. Default is to plot column 1 at both x- and y- axis. For a more detailed description of plot settings, see 2.5.5.

Picking

Picking is a feature that allows a user to “pick” one or more objects from a plot and perform operations on them. When a object is “picked”, it will change colour.

There are two ways to perform picking: Picking a single element, and picking a group of elements. To pick a single element, place the cursor over the object and push the left mouse button. To pick a group of elements, push the left mouse button and drag the mouse. A rectangle will now appear in the render window. When the left mouse button is released, all objects within the rectangle will be marked.

To unmark the marked objects simply place the cursor in the render window and push the left mouse button. If a new point or area is selected, the old selection will be unmarked. To make a new selection without unmark the old selection, push the “Shift” button. When the “Shift” button is used, marked objects will be unmarked if the user tries to mark them again.

Filter

It is possible to add one or more outlier filter to picked elements. With this filter on, you can run the diagrams again, and the marked outliers will be removed from the graphs.

Zooming

Scicraft allows users to zoom in and inspect details of a renderwindow. To zoom in, push the right mouse button and drag the mouse. As with picking, a rectangle will appear in the render window. The area within the rectangle will be zoomed in. It is possible to zoom in several times. To zoom out, push “Shift” + right mouse button to zoom one level out, or push the middle mouse button to zoom out to the original plotimage.

Printing and exporting plots

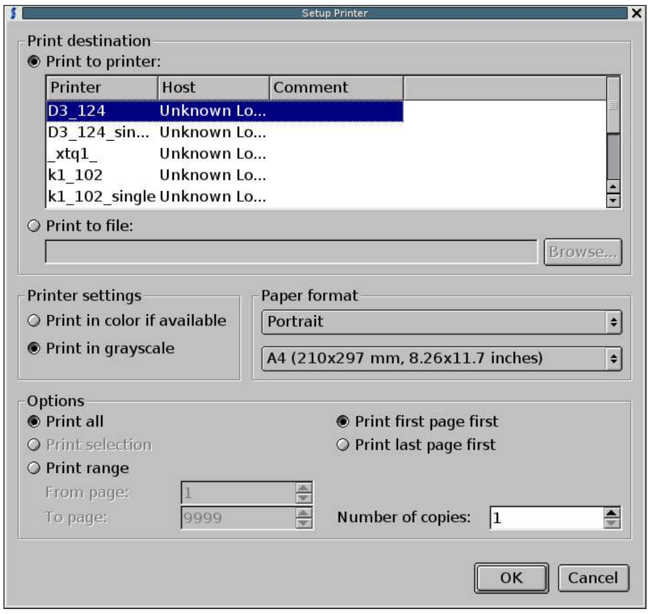
When a plot window has been set up, SciCraft allows its users to print the plots in the plot window, or export them as an Encapsulated PostScript (EPS) file.

In the *File* menu there are four options that allow the user to export or print the plots:

Print Selected... When this option is selected, the plot that is selected (i.e. the plot with the red line around it, and the plot with the corresponding tab in the configuration dock window selected) gets printed through the *Print Plot* dialog. The *Print Plot* dialog is shown in Figure 2.22, in which the user may choose between various standard options for printing documents. The selected plot will be printed so that it fills out the sheet of paper in the best way, without loosing its plot’s aspect ratio.

Print All... When this option is selected, the plots in the Plot Window gets printed through the *Print Plot* dialog. The *Print Plot* dialog is shown in Figure 2.22, in which the user may choose between various standard options for printing documents. All plots will be printed so that it fills out the sheet of paper in the best way, without loosing its plots’ aspect ratio.

Export → Selected Plot to EPS... When this option is selected, the plot that is selected (i.e. the plot with the red line around it, and the plot with the corresponding tab in the configuration dock window selected) gets exported to an EPS file. The file name is chosen through a simple file browser dialog. The selected plot will be exported without loosing its plot’s aspect ratio.


Setup Printer

Print destination

(●) Print to printer:

Printer	Host	Comment
D3_124	Unknown Lo...	
D3_124_sin...	Unknown Lo...	
xtq1	Unknown Lo...	
k1_102	Unknown Lo...	
k1_102_single	Unknown Lo...	

( ) Print to file:

Browse...

Printer settings

( ) Print in color if available

(●) Print in grayscale

Paper format

Portrait

A4 (210x297 mm, 8.26x11.7 inches)

Options

(●) Print all

( ) Print selection

( ) Print range

From page: 1

To page: 9999

(●) Print first page first

( ) Print last page first

Number of copies: 1

OK Cancel

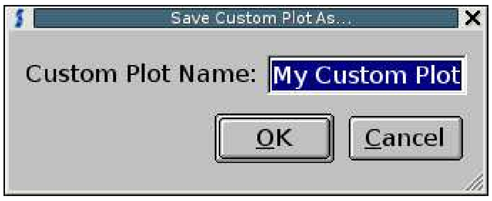
Figure 2.22: The Figure shows the Print dialog in which the user may choose which printer to print to, with or without colours, print to file and so on.

Export → All Plots to EPS. . . When this option is selected, the plots in the Plot Window gets exported to an EPS file. The file name is chosen through a simple file browser dialog. All plots will be exported without loosing its plots' aspect ratio.

Saving and using custom plots

Table 2.1 says that it is possible to save a custom plot to a file. This saves a fully set up plot window, with all its settings including input ports, the plots and all the properties of the different plots. The one thing that is not saved is the data on the input ports, so that it is possible to reuse the configured plot windows with different data on the input ports.

When the *Save Custom Plot As...* option from the *File* menu is chosen the dialog of Figure 2.23 shows up.


Save Custom Plot As...

Custom Plot Name: My Custom Plot

OK Cancel

Figure 2.23: Saving Custom Plots. Type in the name of the new Custom Plot here.

In this dialog the name of the new Custom Plot is written and will be saved to disk as this name. When *Ok* has been pressed the Custom Plot is saved to disk and shows up in the tree view in the SciCraft main window as shown in Figure 2.24. Now the Custom Plot may be used in a Module Diagram as any other plot node.

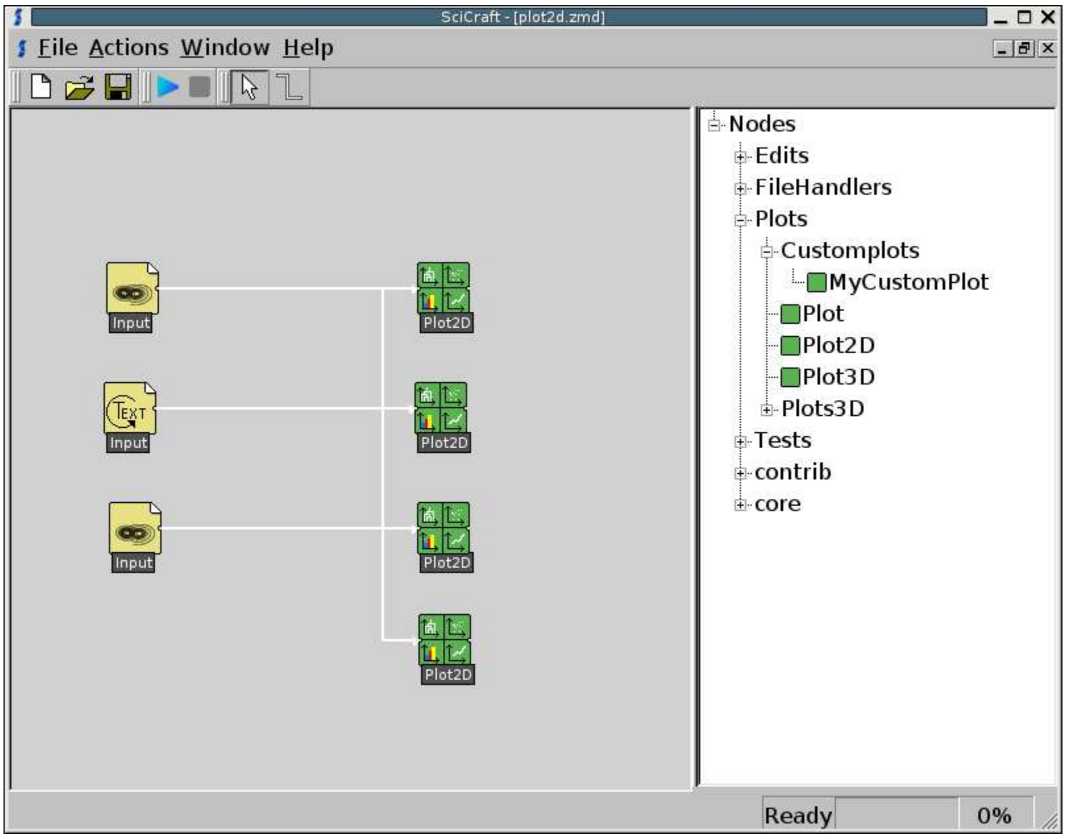


Figure 2.24: When a Custom Plot has been saved, the name of the new Custom Plot shows up in the Tree View in SciCraft.

2.5.2 Plot types

This section describes on a general level the different plot types that is included in SciCraft. For a more thorough description of how to set the settings on the different plots, see section 2.5.5.

Scatter Plot

Figure 2.25 shows a sample Scatter Plot. A Scatter Plot takes a set of coordinates and sets markers on each coordinate's location in a coordinate system. The coordinate sets can be configured by selecting rows and/or columns from one of the input data matrices. The appearance of the markers are highly configurable, as it is possible to change the colour, size, and shape of all markers.

Line Plot

Figure 2.26 shows a sample Line Plot. Just as a Scatter Plot, a Line Plot takes a set of coordinates and sets markers on each coordinate's location in a coordinate system. In addition, lines are drawn between the markers in the same order as the coordinates are ordered. An additional option on the Line Plot is that it is possible to use the order in which the Y-coordinates are ordered as the X-coordinate of each coordinate pair. The appearance of the markers and lines between the markers are highly configurable, as it is possible to change the colour, size, and shape of all markers, and the colour, width, and style of the lines.

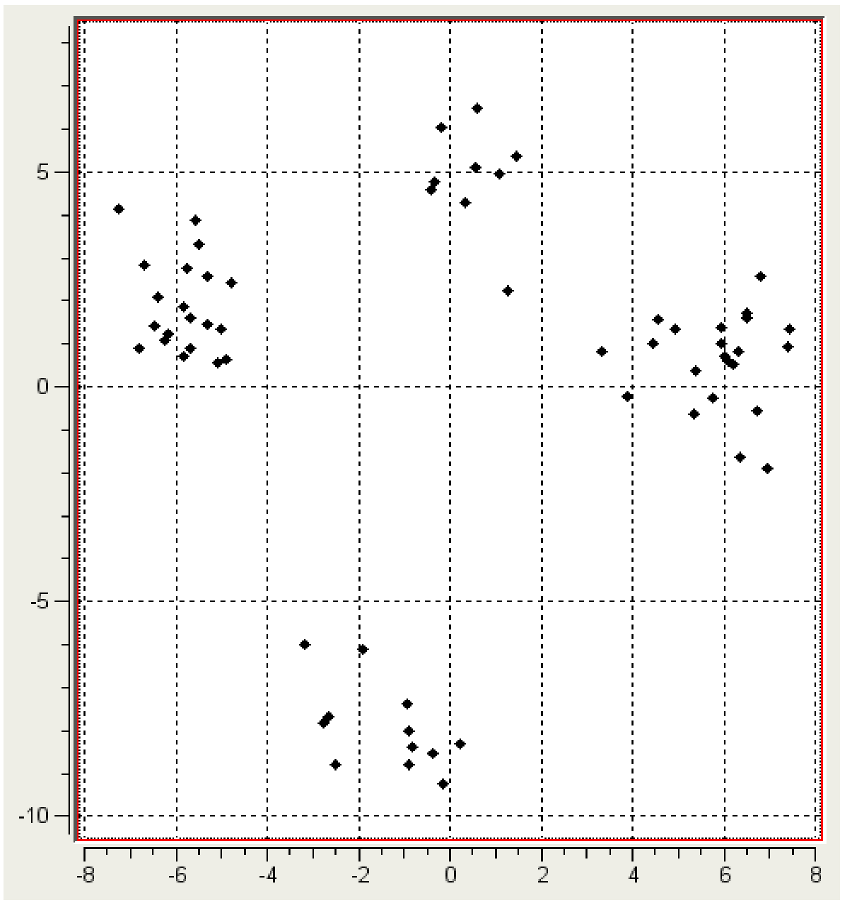


Figure 2.25: Scatter Plot

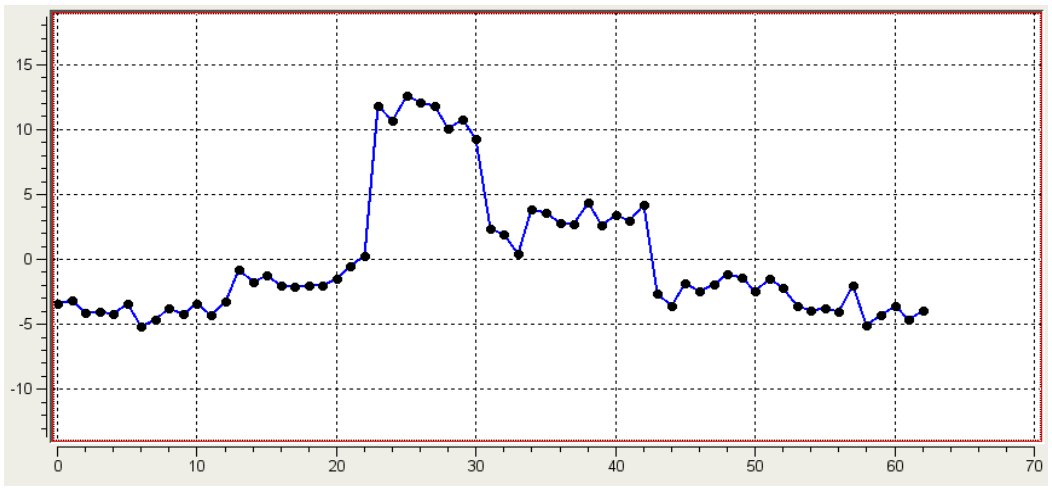


Figure 2.26: Line Plot

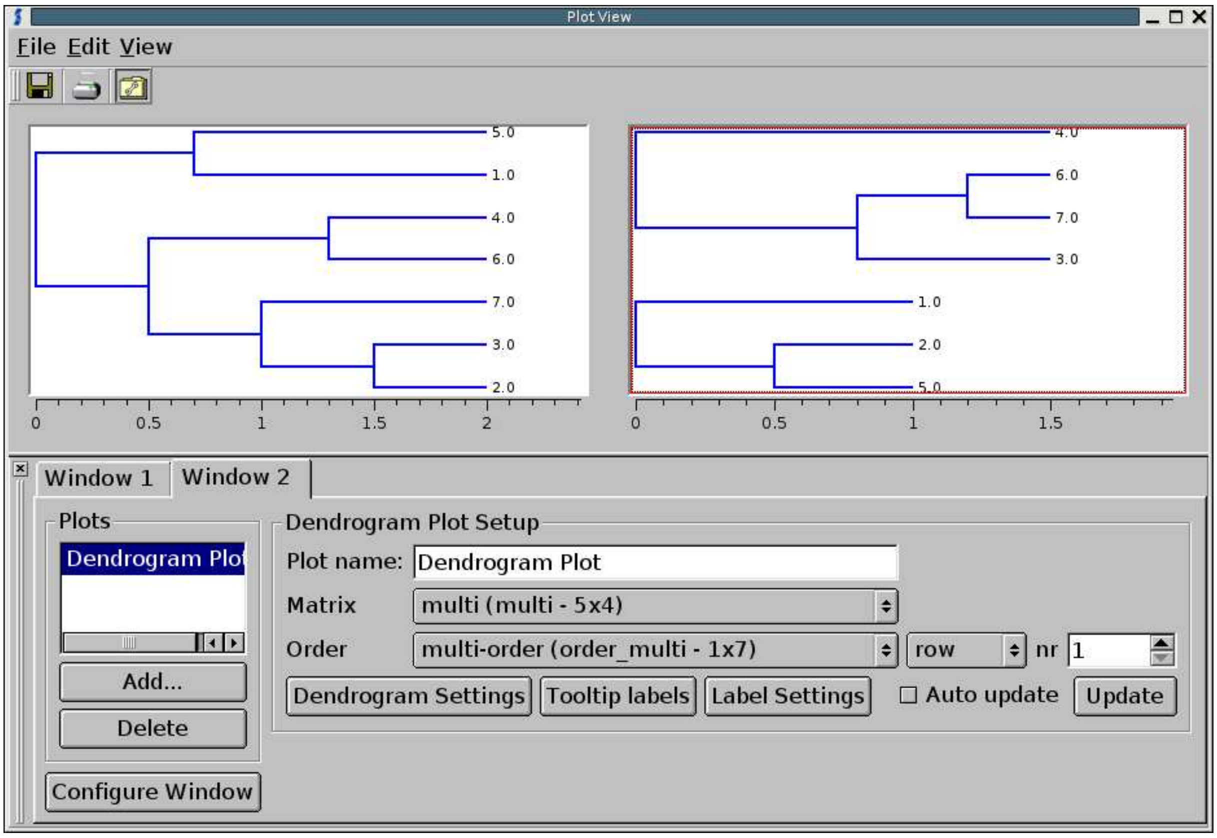
Dendrogram Plot

Dendrogram plot is a hierarchical, binary cluster tree plot. This tree is generated with the help of a matrix that describes the clusters and an order list which states in what order the objects appear. The current data format for drawing a dendrogram plot in SciCraft is Agnes (Agglomerative nesting). This format looks like this:

```
matrix =
   1.00000  -2.00000  -3.00000   0.50000
   2.00000  -6.00000  -4.00000   0.70000
   3.00000   1.00000  -7.00000   1.00000
   4.00000  -1.00000  -5.00000   1.30000
   5.00000   3.00000   2.00000   1.50000
   6.00000   5.00000   4.00000   2.00000
order =
  2  3  7  6  4  1  5
```

The *matrix* data set tell us how the clusters relates to each other and the *order* set tells us in which order the objects appear in. In later edition we hope to support other data formats as well.

An example of a dendrogram plot can be seen in Figure **??**.



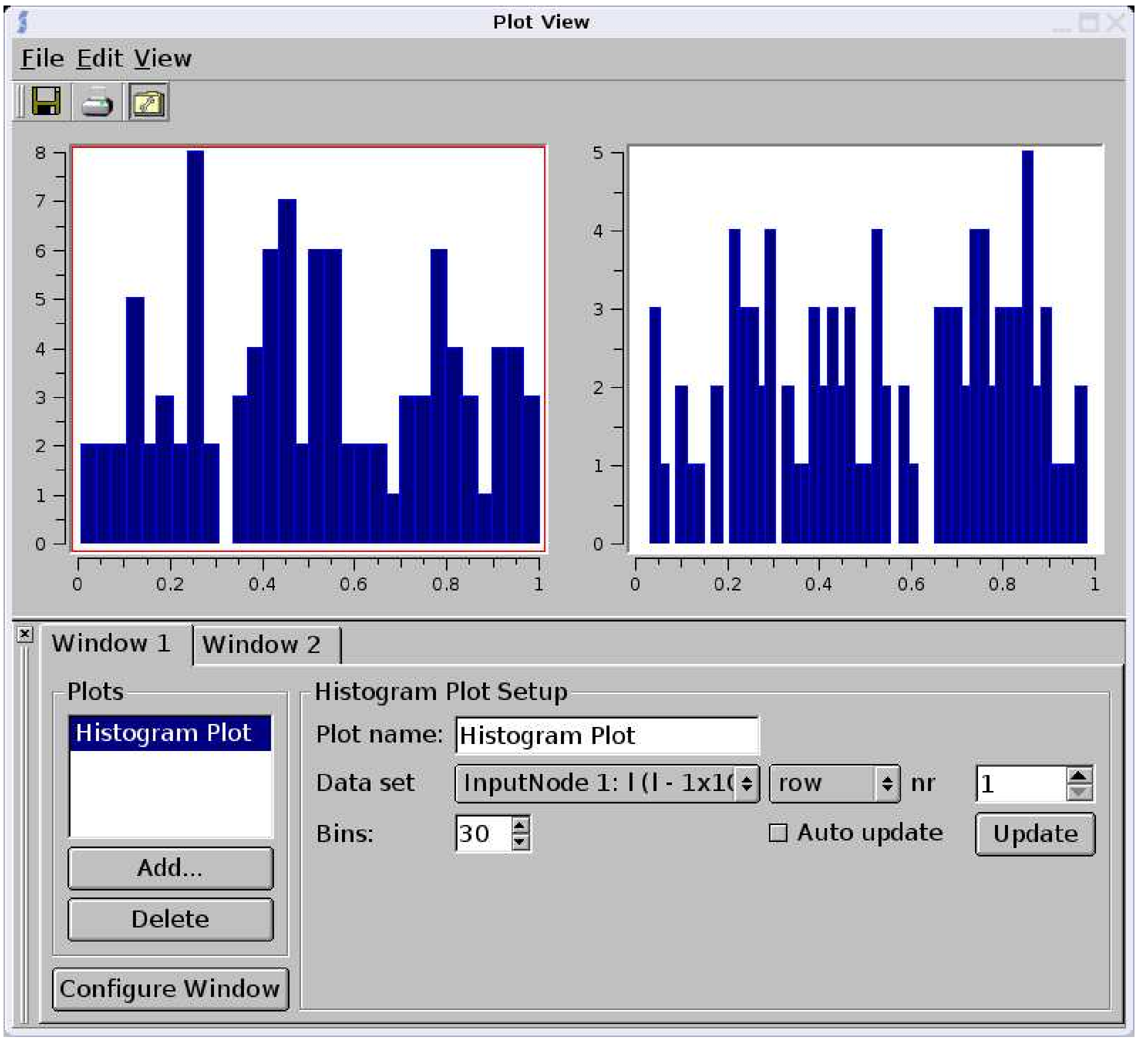
Dendrogram Plot Setup
Plot name: Dendrogram Plot
Matrix: multi (multi - 5x4)
Order: multi-order (order_multi - 1x7) row nr 1

Figure 2.27: Dendrogram plot

Histogram Plot

Histogram plot is a statistical plot which is used to graphically summarise and display the distribution of a data set. The histogram plot needs two inputs, a vector containing the data that will be counted and segmented and a scalar that defines how many bins/buckets will be used.

An example can be seen in Figure 2.28.



Histogram Plot Setup

Plot name: Histogram Plot

Data set: InputNode 1: I (I - 1x1(row nr 1

Bins: 30 ☐ Auto update Update

Figure 2.28: Histogram plot

Image Plot

The image plot displays a two dimensional matrix of values as a color mapped image. The values are mapped linearly from lowest to highest value against one of the color maps.

Multi Line Plot

The multi line plot is a plot capable of showing several line plots, based on rows on columns in a matrix. Which rows or columns you want to see can be decided by entering a range specification in the *Matrix range* field. The resulting rows/columns can also be manipulated through a range in the *Vector range* field.

Density Plot

Density plot is a plot that illustrates the distribution of data inside a region. The plot will count the number of data objects found and show the population as a colour that grows more and more white as the density increases. The density plot has an option for blurring the plot—averaging the density values using a mask with customisable size. In a blurred density plot, the colour of one density is influenced by the densities of all its neighbours. The number of neighbours that influences a density colour, is specified by the size of the averaging mask.

2.5.3 Plot Main Settings

This section describes how to set settings that applies to the entire plot node, eg. setting the input ports and choosing how many plot windows to use.

To display the settings dialog choose 'Settings' under the 'Edit' menu, or simply press 'Ctrl+S'.

Please notice that the new settings will not be valid until after you press the 'Apply' or the 'Ok' button. Pressing the 'Cancel' button at any time will discard the new settings and keep the old ones.

Input Ports

The 'Input ports' section of the main settings window is shown in figure 2.29.

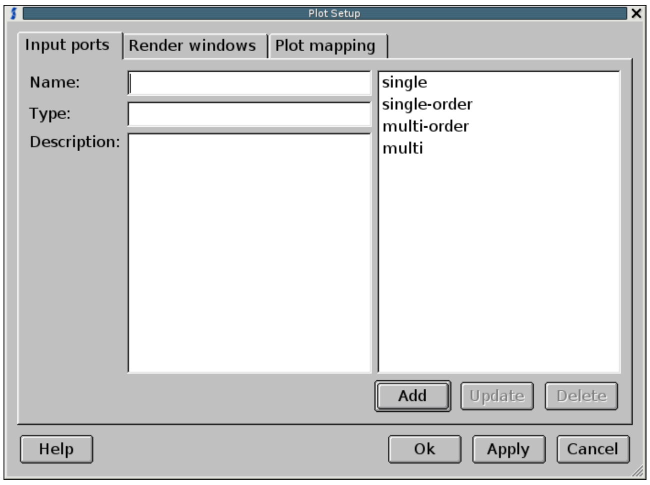


Figure 2.29: Plot main settings section

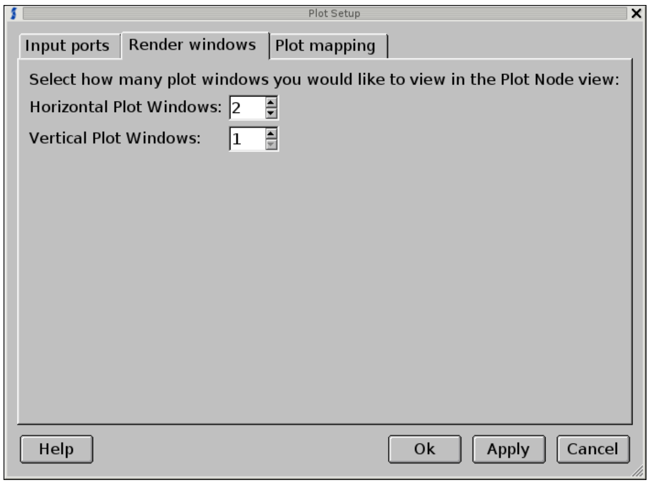
Adding an input port: Type a name and optionally, a type and description, and click the 'Add' button. A new input port has now been added to the plot node.

Updating an input port: Select a port from the port list to alter the settings of this port. Type the new information in the 'Name', 'Type' and/or 'Description' field and click 'Update' to update the port settings.

Delete an input port: Select a port from the port list and click the 'Delete' button to remove this port.

Render Windows

The 'Render windows' section is shown in figure 2.30.



Select how many plot windows you would like to view in the Plot Node view:

Horizontal Plot Windows: 2

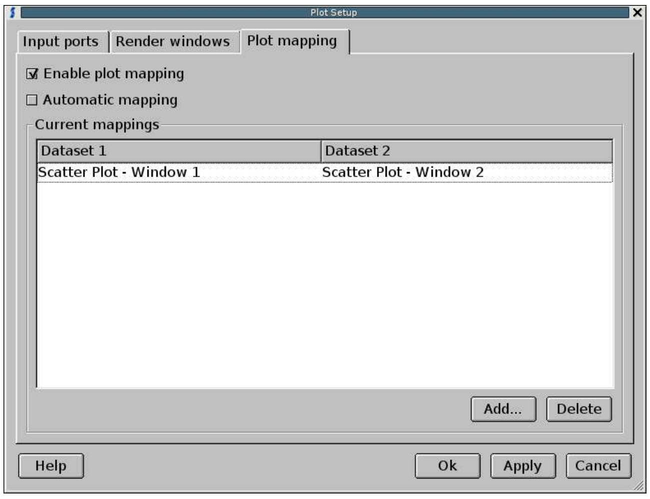
Vertical Plot Windows: 1

Figure 2.30: Render windows section

Select how many plot windows you would like to view in the Plot Node view.

Plot Mapping

The 'Plot Mappings' section is shown in figure 2.31.



[✓] Enable plot mapping

☐ Automatic mapping

Current mappings

Dataset 1	Dataset 2
Scatter Plot - Window 1	Scatter Plot - Window 2

Figure 2.31: Plot mapping section

Plot mapping works as follows: If two plots are mapped, selection in one of the two plots will also lead to a selection in the other plot. The mapping can be done automatically based on the dimension of the data set, or manually by simply selecting which two plots that should be mapped. Automatic mapping is switched off by default.

Enable plot mapping: The plot mapping feature can be switched on and off by clicking the 'Enable plot mapping' check box.

Automatic mapping: When this feature is enabled, any pair of plots with the same dimension will be mapped to each other. A dialog box will ask if you would like to map all your existing plots. Selecting 'Yes' will map all existing plots as well as any new plots when they are added. Selecting 'No' will only map new plots.

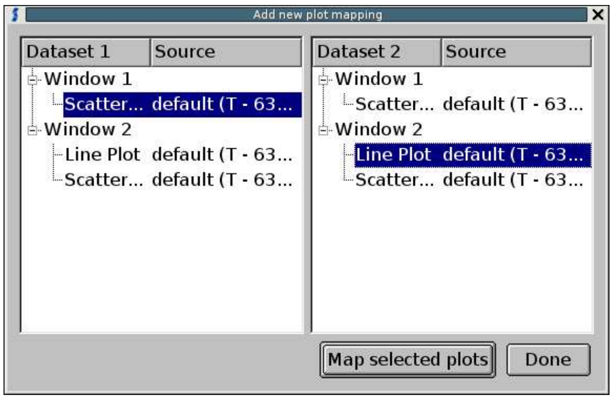

Add new plot mapping
Dataset 1
Source
Window 1
Scatter... default (T - 63...
Window 2
Line Plot default (T - 63...
Scatter... default (T - 63...
Dataset 2
Source
Window 1
Scatter... default (T - 63...
Window 2
Line Plot default (T - 63...
Scatter... default (T - 63...
Map selected plots
Done

Figure 2.32: Add plot map dialogue

Manually add a mapping: By clicking 'Add...' the dialog shown in figure 2.32 will appear. You will now be able to manually map any two plots to each other regardless of their dimension. Select two different plots and click 'Map selected plots' to map these plots. Click 'Done' to return to the 'Plot Mappings' section of the plot settings window.

Manually delete a mapping: Selecting a plot mapping and then 'Delete' will remove this mapping. Please notice that the 'Automatic mapping' feature will be disabled when deleting a mapping.

2.5.4 Render Window Settings

This section describes how to set settings that applies to one render window at a time.

General settings

The 'General' section of the render window settings window is shown in figure 2.33.

Background colour: Allows you to set the background colour of the render window.

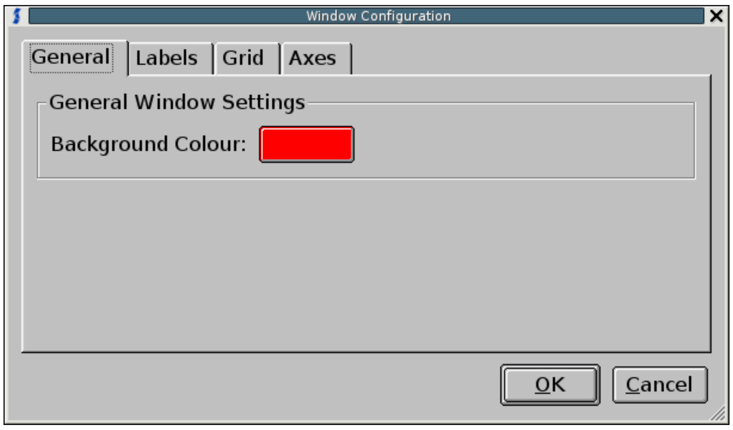
Label settings

The 'Label' section of the render window settings window is shown in figure 2.34.

Window title: Sets this text as the title of the render window.

X-axis label: Sets this text as the title of the X-axis.

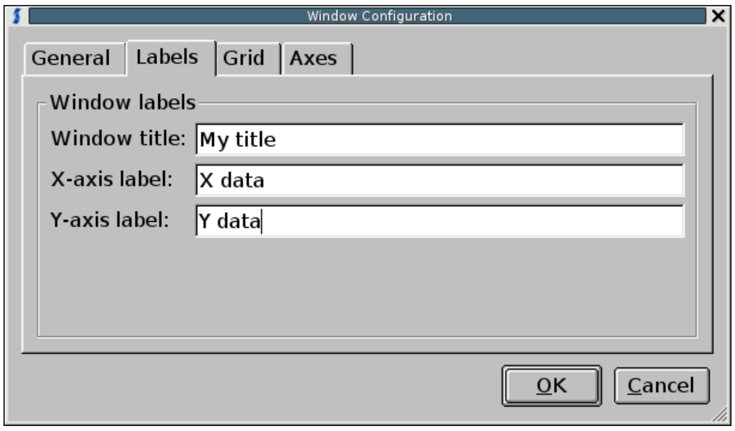
Y-axis label: Sets this text as the title of the Y-axis.



General Window Settings

Background Colour: 🟥

Figure 2.33: General section



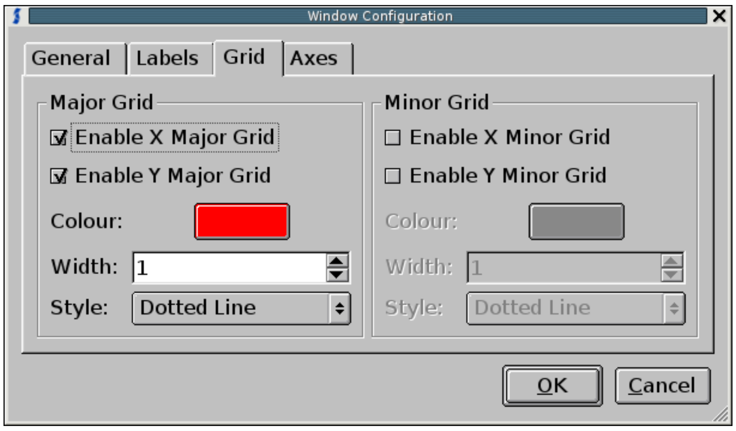
Window labels

Window title: My title

X-axis label: X data

Y-axis label: Y data

Figure 2.34: Label section



Major Grid

☑ Enable X Major Grid

☑ Enable Y Major Grid

Colour: 🟥

Width: 1

Style: Dotted Line

Minor Grid

☐ Enable X Minor Grid

☐ Enable Y Minor Grid

Colour:

Width: 1

Style: Dotted Line

Figure 2.35: Grid section

Grid settings

The ’Grid’ section of the render window settings window is shown in figure 2.35.

Enabling/disabling major/minor grid: With major grid enabled, a line will be drawn for each axis value. With minor grid enabled, a line will be drawn for each axis submarker.

Enabling/disabling X/Y grid: The major and minor grid can be enabled and disabled for the x and/or y axes.

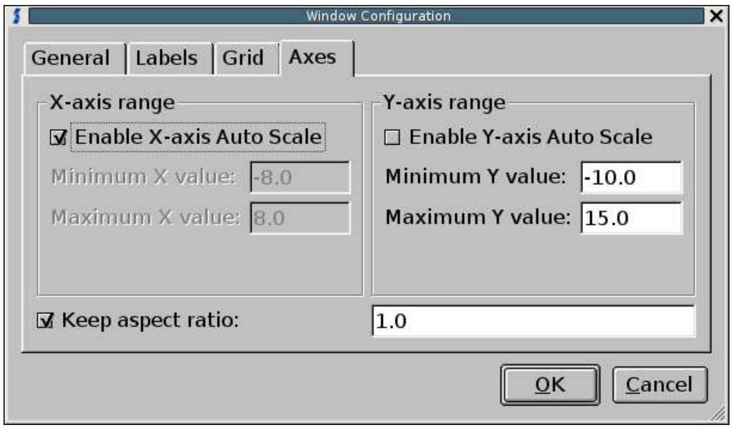
Colour: Sets the grid line colour.

Width: Sets grid line width.

Style: Sets grid line style.

Axis settings

The 'Axis' section of the render window settings window is shown in figure 2.36.



☑ Enable X-axis Auto Scale
Minimum X value: -8.0
Maximum X value: 8.0

☐ Enable Y-axis Auto Scale
Minimum Y value: -10.0
Maximum Y value: 15.0

☑ Keep aspect ratio: 1.0

Figure 2.36: Axis section

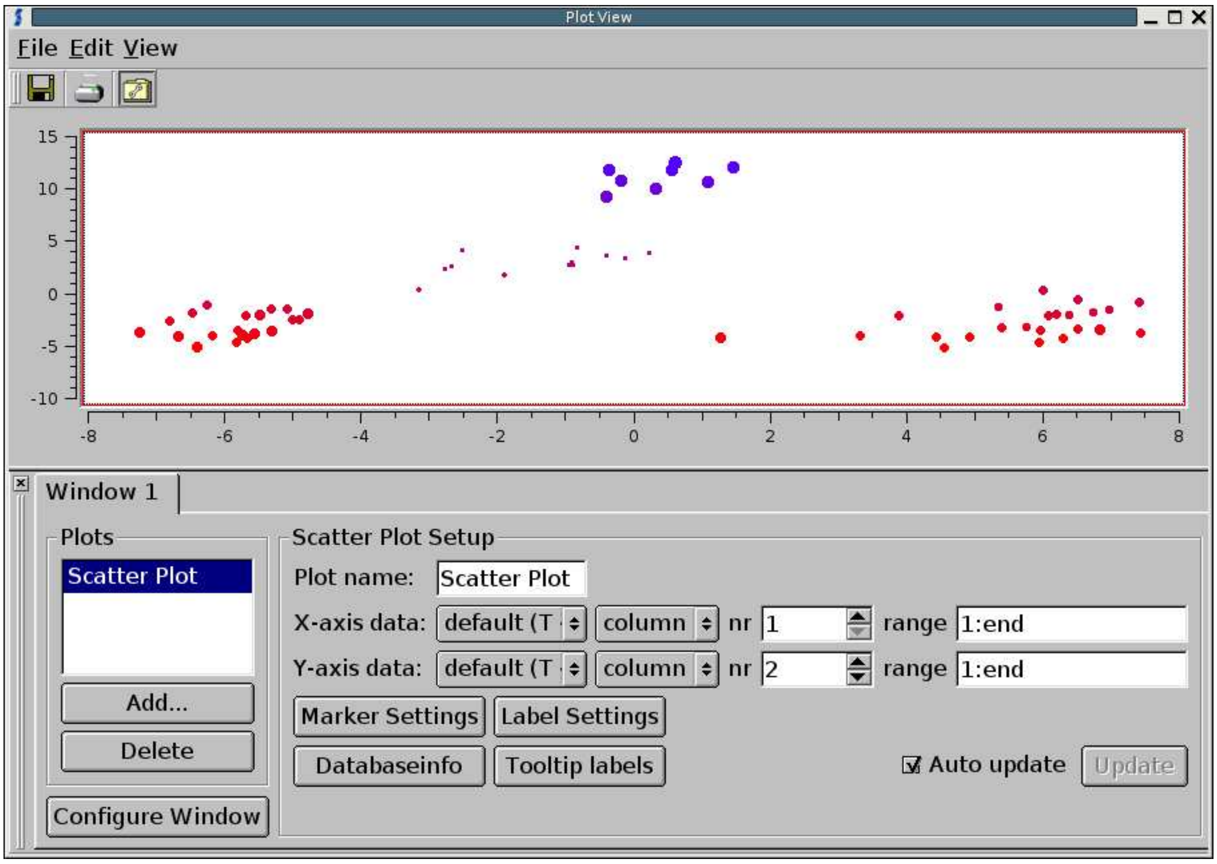
X-axis auto scale: If enabled, plots in the current window will be resized to fit inside the x-axis. If disabled, the min and max value of the x-axis will be set according to the values in the textboxes.

Y-axis auto scale: If enabled, plots in the current window will be resized to fit inside the y-axis. If disabled, the min and max value of the y-axis will be set according to the values in the textboxes.

Keep aspect ratio: If enabled, the aspect ratio between the x and y-axis will be kept constant. Notice that this option is not available if both the x-axis and y-axis auto scale option are diabled.

2.5.5 Plot settings

This section describes how to setup a 2d plot in a plot window. It applies to scatter, line and dendrogram plots.



Scatter Plot Setup

Plots: Scatter Plot

Plot name: Scatter Plot

X-axis data: default (T | column | nr 1 | range 1:end

Y-axis data: default (T | column | nr 2 | range 1:end

☑ Auto update

Figure 2.37: Scatter Plot Settings

Scatter Plot

Figure 2.37 shows a Scatter Plot which is configured properly to plot column 1 and column 3 of the *default* input port against each other.

There are four main fields in the configuration dialog:

Plot name: The name of this Scatter Plot is written here. This name is used to identify this Scatter Plot from any other plots that could be put in the same render window. If the *Plot name* is changed, the name of the selected plot in the list of plots on the left hand side gets updated.

X-axis data: In the leftmost combobox, you choose which input port, data is to be retrieved from. The port name is displayed with the variable name from the input and the dimensions of the data in parantheses behind the port name. **Note:** The variable name and dimensions of it is not available until the diagram is run. In the other combobox you select whether the data is to be retrieved from a *column* or a *row*. The spin box on the right hand side selects which column or row (depending on what option is made in the combobox) the data is to be retrieved from.

Y-axis data: The Y-axis data is set exactly the same way as the X-axis data.

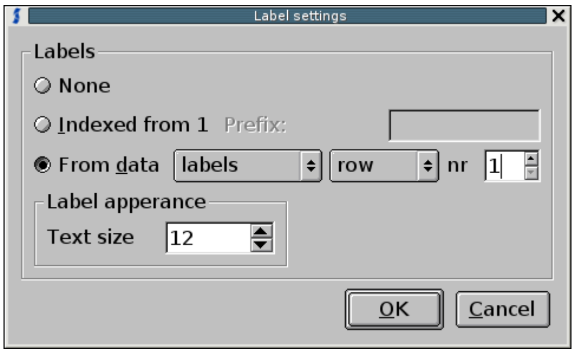
Range selection: Using a Matlab-like notation, subparts of the data from a column or row can be plotted. The default value is 1 : *end* (plot entire data set), a detailed list over notations can be seen in table 2.2.

Notation	Translates to
$m : n$	Index from m to n
$m : end$	Index from m to last index
$m : end - 5$	Index from m to last index - 5
$m : s : n$	Index from m to n with step s
$m_1 : n_1, m_2 : n_2$	Index from m_1 to n_1 and from m_2 to n_2

Table 2.2: Desctiption of range notations

How to *set marker labels* and *changing the layout of plot markers* are described in the following two paragraphs.

Setting marker labels Figure 2.38 shows the setup dialog, in which the marker label settings are set. Figure 2.38 will set labels on the plot markers from the data that is on the input port *labels*. It will retrieve the labels from *row* number *1* in the matrix in the input port data.


Label settings

Labels

○ None

○ Indexed from 1 Prefix:

● From data labels row nr 1

Label apperance

Text size 12

Figure 2.38: Setting marker labels

There are three main options for how to set marker labels:

None: No labels will be set on the plot markers when the *Ok* button is pressed. This is also the way to remove markers that has already been set.

Indexed from 1: Labels will be set on the plot markers when the *Ok* button is pressed. The labels that will be set will be a sequence, $1, 2, 3, \ldots$, if no prefix is set. The prefix is set by typing it into the textbox right of the *Indexed from 1* option. E.g. if the text string "SciCraft" is typed into the textbox, the sequence will be $SciCraft1, SciCraft2, SciCraft3, \ldots$.

From data: Labels will be set on the plot markers when the *Ok* button is pressed. The labels that will be set will be retrieved from the given *column* or *row* in the given data input port. The input port and *column* or *row* index is set in the exact same way as the data is set on the Scatter Plot described in the previous paragraph.

Changing layout of plot markers First of all, start by *clicking* on the *Marker Settings* button in the *Scatter Plot Setup* window. A new window will now appear on the screen, as illustrated in Figure 2.39.

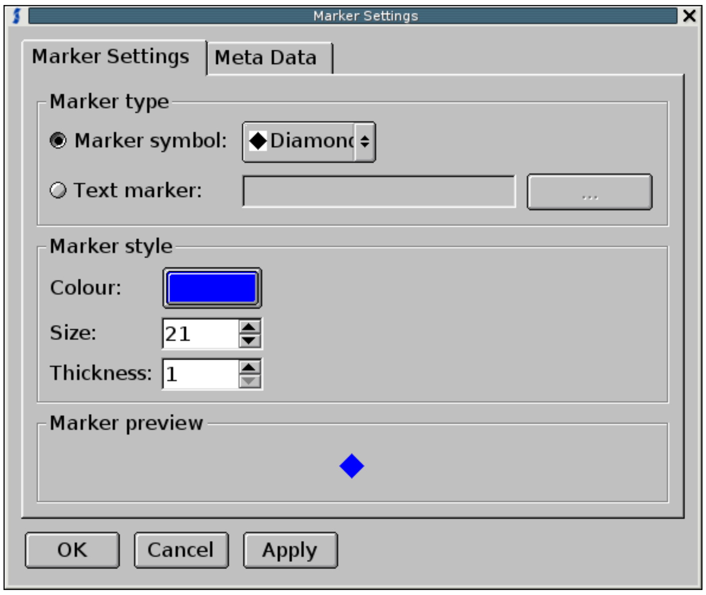


Figure 2.39: Marker Settings window

If you want to change the marker symbol, toggle the *Marker symbol* radiobutton and continue by choosing the desired symbol from the combobox. It is also possible to change the style of the marker. Set the colour by *clicking* on the colour button. A new window, where you can choose a colour from a colour map will appear on the screen. When you have picked a new colour and closed the window, the new colour will be set on the button. Size and thickness of the marker is changed by *clicking* on the vertical arrows or by typing the value directly.

Instead of using a marker symbol you have the opportunity to replace the marker symbol by a 'text' symbol. Start by *toggling* the radiobutton named *Text marker*. Continue by typing in your text in the line edit field. If you want to change the layout of the font, *click* on the button next to the line edit field. A new window will now appear on the screen where you can change the settings of the font. It is also possible to change the colour of the text. This is done in the same way as for marker symbols.

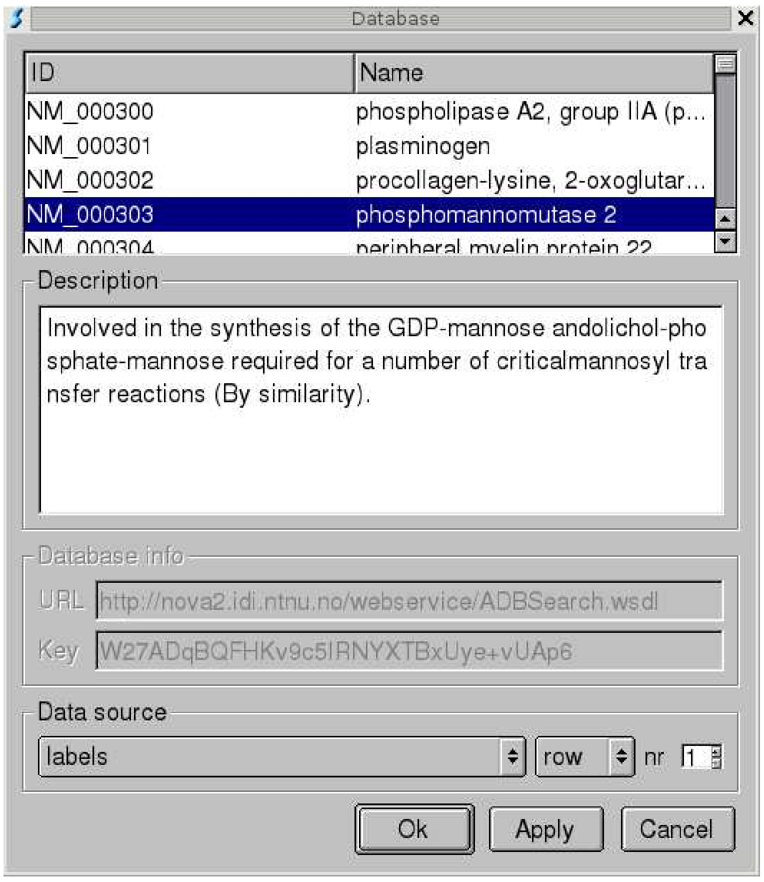
When all changes are done, *click* on the *Apply* button, and the new marker settings will be applied for the current plot. By *clicking* on the *Ok* button the new settings will be applied and the *Marker settings* window will be closed. A preview of the current marker will be shown at the buttom of the window.

(Databaseinfo) Look up data from Gene Annotation Database SciCraft enables the user to look up gene annotation data. Simply select one or more points in the scatter plot and click the *Databaseinfo* button, as shown in Figure 2.37. This will cause the Database information window, as shown in Figure 2.40. Select a data source and click *Apply* to look up the gene annotation information. A description of a specific gene is shown by clicking on it.

Brief descriptions of the variuos fields in the window are shown below:

ID: *GenBank Accession Number*, used to look up database information.

Name and Description: The name and description of the gene IDs are looked up in the database and displayed.



ID	Name
NM_000300	phospholipase A2, group IIA (p...
NM_000301	plasminogen
NM_000302	procollagen-lysine, 2-oxoglutar...
NM_000303	phosphomannomutase 2
NM_000304	peripheral myelin protein 22

Description

Involved in the synthesis of the GDP-mannose andolichol-pho sphate-mannose required for a number of criticalmannosyl tra nsfer reactions (By similarity).

Database info

URL http://nova2.idi.ntnu.no/webservice/ADBSearch.wsdl

Key W27ADqBQFHKv9c5IRNYXTBxUye+vUAp6

Data source

labels row nr 1

Figure 2.40: Database Information window

Database URL and Key: URL and Key to the Gene Annotation Database to be used. Current this is locked to *The Norwegian MicroArray Consortium Annotation Database*. More information regarding this database can be found here: http://nova2.idi.ntnu.no/annotdb/index.php?section=Home.

Data source: An example of a data source would be a vector containing a list of gene IDs.

Setting tooltip labels on markers SciCraft enables the user to use yet another dimension in plotting through the use of tooltip labels. Figure 2.41 shows a tooltip on a marker in a scatter plot. The label pops up like a tooltip does, when the mouse cursor is positioned over the marker.

The tooltips can be assigned to the different markers in the same way as one assign labels to the different markers. By pressing the *Tooltip labels* button that is located below the combo boxes in which one assigns the data to the markers, a dialog similar to the one in Figure 2.38 pops up. In the dialog it is possible to choose from the following three options:

None: No labels will be set as tooltips when the *Ok* button is pressed. This is also the way to remove tooltip labels that has already been set.

Indexed from 1: Labels will be set as tooltips when the *Ok* button is pressed. The tooltip labels that will be set will be a sequence, $1, 2, 3, \ldots$, if no prefix is set. The prefix is set by typing it into the textbox right of the *Indexed from 1* option. E.g. if the text string "SciCraft" is typed into the textbox, the sequence will be $SciCraft1, SciCraft2, SciCraft3, \ldots$.

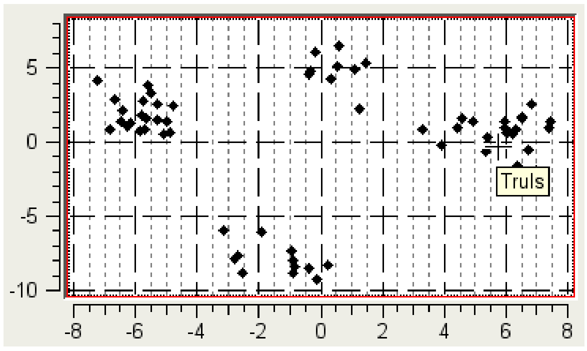


Figure 2.41: An example of a tooltip label in a Scatter Plot

From data: Labels will be set as tooltips when the *Ok* button is pressed. The tooltip labels that will be set will be retrieved from the given *column* or *row* in the given data input port. The input port and *column* or *row* index is set in the exact same way as the data is set on the Scatter Plot described previously in this section.

When one of the last two options is chosen, the labels will pop up as tooltips when the mouse is over a marker.

Line Plot

Figure 2.42 shows a Line Plot which is configured properly to plot column 2 of the *default* input port with increasing indexes.

There are three main fields in the configuration dialog:

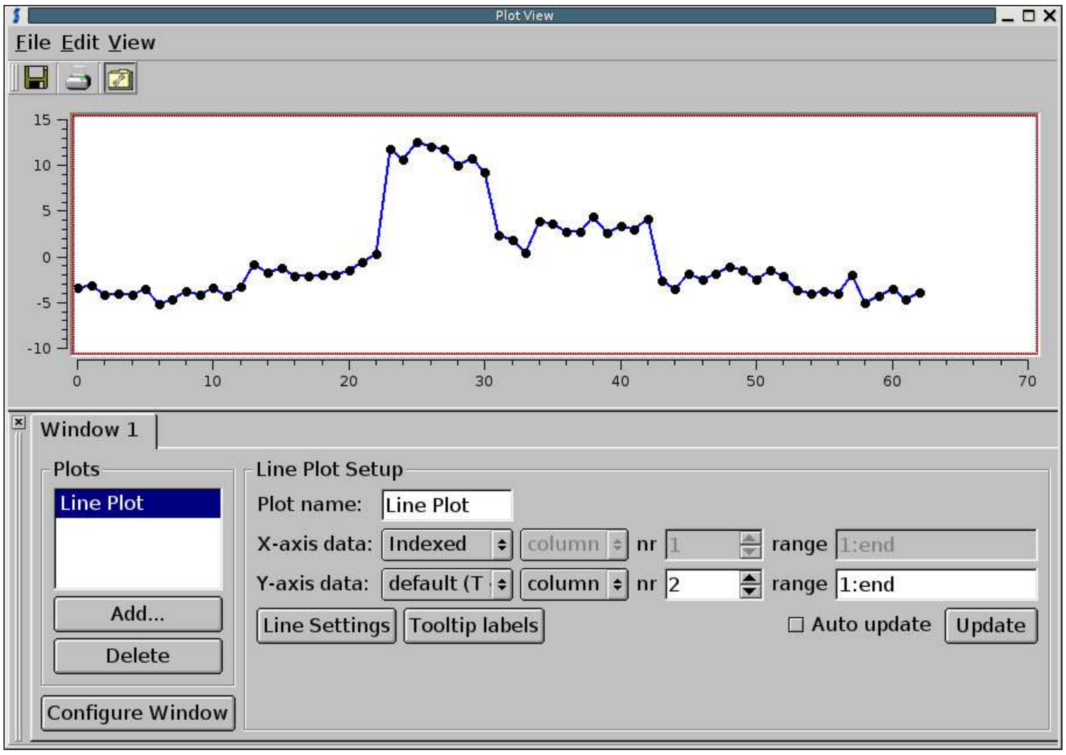
Plot name: The name of this Line Plot is written here. This name is used to identify this Line Plot from any other plots that could be put in the same render window. If the *Plot name* is changed, the name of the selected plot in the list of plots on the left hand side gets updated.

X-axis data: In the leftmost combobox, you choose which input port, data is to be retrieved from. The port name is displayed with the variable name from the input and the dimensions of the data in parantheses behind the port name. **Note:** The variable name and dimensions of it is not available until the diagram is run. In addition, there is the *Indexed* option, which makes the first value in the Y-data show up on position 1, the second on position 2 and so on. In the other combobox you select whether the data is to be retrieved from a *column* or a *row*. The spin box on the right hand side selects which column or row (depending on what option is made in the combobox) the data is to be retrieved from.

Y-axis data: The Y-axis data is set exactly the same way as the X-axis data, except that there is no option *Indexed*.

Range selection: Using a Matlab-like notation, subparts of the data from a column or row can be plotted. The default value is 1 : *end* (plot entire data set), a detailed list over notations can be seen in table 2.2.

How to *change the layout of plot markers* is described in the following paragraph.



Plot name: Line Plot
X-axis data: Indexed | column | nr 1 | range 1:end
Y-axis data: default (T | column | nr 2 | range 1:end

Figure 2.42: Line Plot settings

Changing layout of a line plot Start by *clicking* on the *Line Settings* button in *Line Plot Setup* window. A new window will then appear on the screen, as illustrated in figure 2.43.

If you want to change the current marker symbol a line plot, choose the *Marker Settings* tab and then follow the procedure as described in 2.5.5, ScatterPlot.

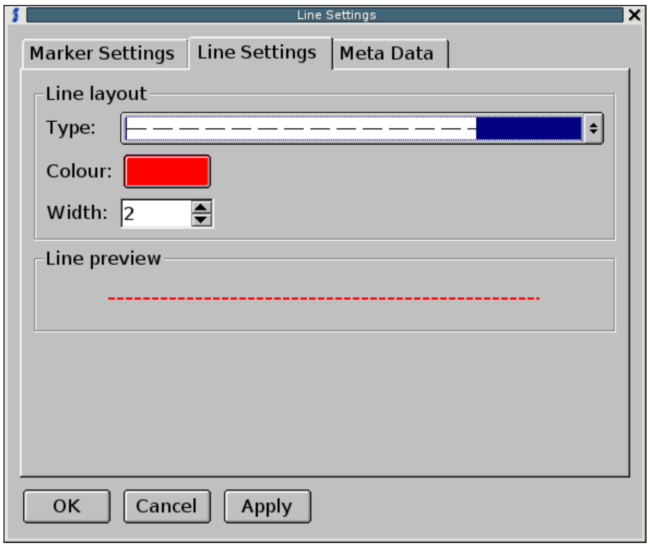
Otherwise, continue by *clicking* on the *Line Settings* tab. Change the the line type by choosing the desired one from the combobox. The line colour is set by *clicking* on the colour button. A new window will then appear, where you have the possibility to choose the desired colour. Pick the new colour from the map, and close the window. The new colour will then be set on the colour button. The width of the line is set by *clicking* on the vertical arrows or by typing the value directly. A preview of the line will be shown at the buttom of the window.

When all changes to the line is done, continue by *clicking* on the *Apply* button and the new settings for the line will be applied. If you *click* the *Ok* button the changes will be applied and the window will be closed.

Setting tooltip labels on a line plot SciCraft enables its users to set tooltip labels on a line plot. The tooltip labels will be set on each of the markers of the line in a plot. The tooltip labels are set in the same way as they are set on a scatter plot (see the corresponding paragraph, page 43, in the previous section for details).

Dendrogram Plot

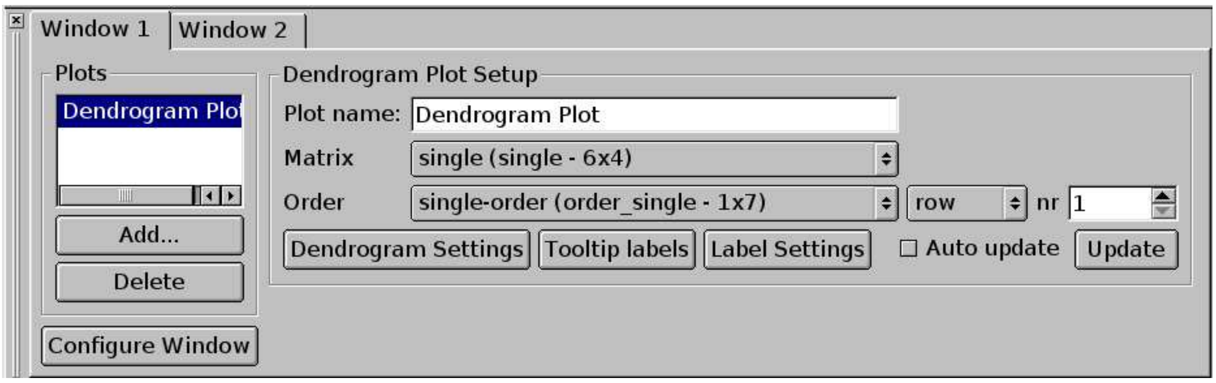
Setting up a dendrogram plot is really straight forward. You need to have two data sources (input ports), one with the cluster matrix and one with the order list. You select which port that contains the cluster matrix and which port that has the order data. On the order port you



Type: - - - - - - - - - - -
Colour: 🟥
Width: 2

Figure 2.43: Line setup window

have to select if the data is row or column and also which row or column to use. In Figure ?? the port *single* is used for the cluster matrix and row 1 from the *single-order* is used as order data.



Plot name: Dendrogram Plot
Matrix: single (single - 6x4)
Order: single-order (order_single - 1x7) row nr 1
☐ Auto update

Figure 2.44: Dendrogram Settings

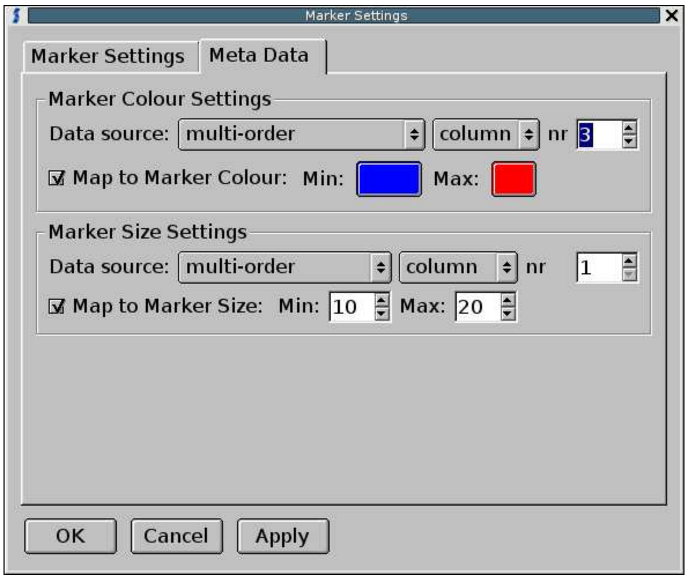
Setting tooltip labels on leaf nodes SciCraft enables its users to set tooltip labels on a dendrogram plot. The tooltip labels will be set on the leaf nodes of the dendrogram plot, in the order according to the order list on the input port. The tooltip labels are set in the same way as they are set on a scatter plot (see the corresponding paragraph, page 43, in the section before the previous section for details).

Setting labels on leaf nodes Labels can be set on the leaf nodes in the same way as tooltip labels by pressing the *Label Settings* button and choosing the desired settings. When choosing *Use order indices*, the leaf's order is printed, while these indices are used for lookup in the specified datasource when the *From data* radio button is set.

Colour and size mapping

It's possible to read meta data from any port defined in the plot node and map these data to colours and sizes of plot markers. This mapping possibility applies to both line and scatter plots. One may use either colour or size mapping, or both.

In both the Marker and Line Settings dialogs there is a *Meta Data* tab which is used to configure both the colour and size mapping from an input port. Figure 2.45 shows the settings window when the *Meta Data* tab has been chosen.



Marker Colour Settings
Data source: multi-order column nr 3
☑ Map to Marker Colour: Min: 🟦 Max: 🟥

Marker Size Settings
Data source: multi-order column nr 1
☑ Map to Marker Size: Min: 10 Max: 20

Figure 2.45: Meta Data Settings

Choosing meta data The data to use for mapping is chosen by selecting an input port on the desired type(s) of mapping and then choosing how to read the data, i.e. selecting column or row. The number of the column or row to use is then chosen by *clicking* the up/down arrows on the *spin box* or by typing the value directly. Note: If the module diagram has not yet been run, no data is available on any input ports, and the index or column/row number chosen may be out of bounds because no check can be made on the data. In addition, if the data set read from the selected port has a different size than that of the plot, the largest data set is stripped (end elements are removed) before mapping to achieve equal size on the two data sets.

Mapping data to marker colour If you want to map the meta data read from the chosen input port to marker colours, just check the checkbox named *Map to Marker Colour*. Then choose a min and a max colour. The min colour is the colour associated with the minimum value in the data set read from the input port, and similarly the max colour is the colour associated with the maximum value. This mapping is linear. To choose these colours, just *click* the corresponding colour buttons. A new dialog will appear. Select the desired colour and then press *Ok*. The new colour will then be set on the colour button.

Mapping data to marker sizes Mapping meta data to marker sizes is quite similar to that of mapping data to colours. Just check the checkbox named *Map to Marker Size*. The min size is the marker size associated with the minimum value in the data set, and similarly the max size corresponds to the maximum value. This mapping is also linear. Use the up/down arrows on the *spin boxes* or type the values directly.

Chapter 3

The nodes

This chapter will describe the system as a whole seen from the users point of view. It will describe the terminology used in SciCraft and describe the different nodes properties.

3.1 Overview

SciCraft have 76 different nodes which are divided into 6 categories; file, input , plot, toolboxes, tools and view. Each of the function nodes in toolboxes, which are linked up with the plug-ins in contrib and core makes use of an external program when they run.

3.1.1 File

These nodes (table 3.2 and 3.3) are capable of reading and writing plug-in specific files.

File type support

Input node	**Output node**
Matlab	R
BASE	Molecule(PDB)
Octave	Octave
R	
Molecule(PDB)	
Textfile	
GaussDal	
Experiment	

Table 3.1: File type support

Name	InputNode
Category	FileHandler
Description	Node that reads data from a file.
Usage	This node is used for retrieving data from a file into a Module Diagram. A file is read and the content in this file is parsed into data which can be used by SciCraft.
Parameters	Filename
Input ports	None
Output ports	One port for each variable in the current file will be added.

Table 3.2: Summary of the InputNode.

Name	OutputNode
Category	Filehandlers
Description	Node that writes data to a file.
Usage	This node is used for writing results calculated by SciCraft to a file. Only file formats supported by SciCraft can be written.
Parameters	Filename
Input ports	One port for each variable that will be written to the file.
Output ports	None

Table 3.3: Summary of the OutputNode.

3.1.2 Input

These nodes (table 3.4 and 3.5) are capable of reading input commands for SciCraft.

Name	Param
Category	Input
Description	Node that reads commands from the user.
Usage	This node is used for sending commands to Sci-Craft.
Parameters	Command
Input ports	None
Output ports	Data

Table 3.4: Summary of the ParamNode.

Name	UserQuery
Category	Input
Description	Node that reads queries from the user.
Usage	Use this node to query the system.
Parameters	Command
Input ports	None.
Output ports	Data

Table 3.5: Summary of the UserQuery.

3.1.3 Plot nodes

Nodes capable of showing plots based on data in SciCraft. These nodes are summed up in (table 3.6 and 3.7) and a detailed coverage of plotting is given in section **??**.

Name	PlotNode
Category	Plot
Description	Node that is used to make plot based on calculated data in the Module Diagram.
Usage	This node's task is to show plots bases on data coming in on the plotnode. In later edition of SciCraft it will be possible to select elements in the plot and then send the data and a command to an edit node to remove these elements from the dataset.
Parameters	None
Input ports	Arbitrarily many ports with input matrices
Output ports	In later editions there will be one exit ports where a command string is set. These ports will then be connected with an edit node which will process the current dataset.

Table 3.6: Summary of the Plot nodes.

Name	DevelPlot
Category	Plot
Description	PlotNode under development. Better plots, but less functionallity.
Usage	This node's task is to show plots bases on data coming in on the userplot node.
Parameters	None
Input ports	Arbitrarily many ports with input matrices
Output ports	In later editions there will be one exit ports where a command string is set. These ports will then be connected with an edit node which will process the current dataset.

Table 3.7: Summary of the DevelPlot nodes.

3.1.4 Toolboxes/function nodes

Plugin nodes which executes different functions in an external program, e.g. Octave or R.

Name	FunctionNode
Category	None
Description	Node responsible for communication with external programs.
Usage	This node is used for communication with external programs. SciCraft contains scripts which are used to generate different function nodes. These nodes will then use these scripts to process data through the external program.
Parameters	Depending on the type of script the node contains. None of these are meant to be edited by the user.
Input ports	One port for each variable needed by the script this node will execute.
Output ports	One port for each data element generated by the external program.

Table 3.8: Summary of the Function nodes.

3.1.5 Contrib

Classification

Kmeansc node The node performs k-means cluster analysis. Table 3.9 gives a summary of the Kmeansc node.

Name	Kmeansc
Category	Contrib/Classification
Description	Node performs k-means cluster analysis.
Usage	Add input matrix, number of clusters to search for, and maximum number of iterations.
Parameters	Command
Input ports	Input matrix, numbers of clusters and max iterations.
Output ports	An exit port with CNT/R object with cluster centers.

Table 3.9: Summary of the Kmeansc node.

Knn2 node The node performs k-nearest neighbour classification. Table 3.10 gives a summary of the Knn2 node.

Name	Knn2
Category	Contrib/Classification
Description	Node performs k-Nearest neighbour classification.
Usage	Add data matrix as the independent variables for the calibration samples. Add a vector containing the classes for the samples given in the matrix. A value for the number of neighbours considered. An optinal matrix with a set of test cases can be added. .
Parameters	Command
Input ports	Input matrix, A vector containing classes, number of neighbours, test matrix.
Output ports	Percentwise calibration error, prediction error and predicted classes.

Table 3.10: Summary of the Knn2 node.

lda2 node The node performs linear discriminant analysis. Table 3.11 gives a summary of the lda2 node.

Pcalda node The node performs a linear discriminant analysis using the scores from a principal component analysis. The optimal number of components is found using crossvalidation. Table 3.12 gives a summary of the pcalda node.

Name	lda2
Category	Contrib/Classification
Description	Node performs linear discriminant analysis.
Usage	Add a matrix with values for the calibration and a vector containing the classes for the samples given in the matrix.
Parameters	Command
Input ports	Input matrix, vector containing the different classes.
Output ports	R object with the estimated classification rule. The percentwise calibration error and prediction error.

Table 3.11: Summary of the lda2 node.

Name	Pcalda
Category	Contrib/Classification
Description	Node performs lda on pca data.
Usage	Add a matrix containing values for the calibration and a vector containing different classes.
Parameters	Command
Input ports	Input matrix, input vector.
Output ports	R object with estimated classification rule, percentwise calibration error and prediction error, loadings from the pca.

Table 3.12: Summary of the pcalda node.

Pcapred node The node performs prediction of classes using principal analysis as preprocessing to linear or quadratic discrimination analysis. Table 3.13 gives a summary of the pcapred node.

Pcaqda node The node performs quadratic discriminant analysis using the scores from a principal component analysis. The optimal number of components is found using crossvalidation.. Table 3.14 gives a summary of the pcaqda node.

Pred2 node The node performs prediction of classes using linear or quadratic discrimination analysis. Table 3.15 gives a summary of the pred2 node.

Qda2 node The node performs quadratic discriminant analysis. Table 3.16 gives a summary of the qda2 node.

Name	Pcapred
Category	Contrib/Classification
Description	Node performs prediction of classes based on pca before linear or quadratic discrimination analysis.
Usage	An Pcapred node makes use of a rule from either the pcalda or pcaqda node, the loadings from either pcalda orpcaqda and matrix with the independent variables.
Parameters	Command
Input ports	Classification rule, pcalda or pcaqda loadings, matrix.
Output ports	Predicted classes.

Table 3.13: Summary of the Pcapred node.

Name	pcaqda
Category	Contrib/Classification
Description	Node performs quadratic discriminant analysis using scores from a pca.
Usage	The pcaqda operates on a matrix with indepentdent variables for the calibration, with a vector containing the classes for the samples given in the matrix.
Parameters	Command
Input ports	Input matrix, and vector containign the classes.
Output ports	R object, precentwise clibration error, prediction error and loadings from the pca.

Table 3.14: Summary of the Pcaqda node.

Name	Pred2
Category	Contrib/Classification
Description	Node performs prediction of classes using linear or quadratic discrimination analysis.
Usage	Add classification rule from either lda2 or qda2 node, and a matrix with independent variables as the sample with unknown classes.
Parameters	Command
Input ports	lda2 or qda2 rule, input matrix.
Output ports	The predicted classes for the unknown samples.

Table 3.15: Summary of the Pred2 node.

Name	qda2
Category	Contrib/Classification
Description	Node performs quadratic discriminant analysis.
Usage	Add a matrix with independent variables for classification, a vector containing classs for the sample and a test set.
Parameters	Command
Input ports	Input matrix, vector with classes, test set.
Output ports	R object with est. class. rule, percentwise calibration error, prediction error and list of predicted classes from the test data.

Table 3.16: Summary of the qda2 node.

Other nodes under contrib (Design, Regression, Signal-Processing)

- AddCenter
- Anova
- CCD
- CompEffects
- FacDesign
- FracDesign
- MlrAnova
- mlr
- fssir
- MSC
- emsc
- Normalize
- OSC
- SIS
- LSPsmooth

3.1.6 Nodes under core (chemometrics and microarray)

- HCA
- Regression predict
- SciCraftPCA
- SciCraftPLS
- oldpca
- workPLS
- AddNoise
- FFT1Drows
- KNearestNeighbour
- LinearDiscrimPrediction
- LinearDiscriminantAnalysis
- PLS
- interpretpls

- predictPLS
- HierarchicalClusterAalysis
- KMeansClustering
- PCA
- AutoScale
- Correlation
- AgilentFilterRG
- AgilentLoad
- ArrayImage
- DiffExp
- FilterMA
- FilterRG
- NormBetweenArrays
- NormWithinArray
- SpikeCheck
- affytools
- bgcorrect
- genelist
- gprload

3.1.7 Edits

Dissection

The edit node is capable of editing and transforming data in SciCraft. Table 3.18 gives a summary of the EditNode. A detailed description of this node is provided in section 3.3.

Name	Dissection
Category	Tools
Description	Node that can remove parts of a dataset.
Usage	Can alter the data you use for calculations.
Parameters	None
Input ports	Matrix
Output ports	Matrix

Table 3.17: Summary of the dissection.

Edit node

The edit node is capable of editing and transforming data in SciCraft. Table 3.18 gives a summary of the EditNode. A detailed description of this node is provided in section 3.3.

Name	EditNode
Category	Edit
Description	Node that can remove one or several rows from a dataset.
Usage	An edit node makes use of a command string to edit the data it receives. Its intention in later editions of SciCraft is to make it possible to select data in a plot node and then tell the edit node to remove these elements.
Parameters	Command
Input ports	commandInput – the enter point for data.
Output ports	An exit port for the edited data.

Table 3.18: Summary of the EditNode.

Gaussdal node

This node is yet highly experimental. Use it at your own risk, and do not expect it to behave as intended. It has basic functionality, but lacks stability and extended functionality. The Gaussdal node takes as input molecule data from a GaussDal database query, and maps the molecules relative to a reference molecule. The node is connected to Octave's Procrustes routine, running this routine on the mapped molecules. A summary of the node is found in table 3.19. A more detailed description can be found in section 3.4.

Name	Gaussdal node
Category	Edit
Description	Molecule mapper for Gaussdal database data.
Parameters	Mapping parameters
Input ports	Data from Input-node reading Gaussdal files, column names, optional molecule mapping matrix.
Output ports	Molecule data, mapping (not implemented yet)

Table 3.19: Summary of the Gaussdal node.

MoleculeEditor

The MoleculeEditor node is capable of showing and editing mulecules in a 3D window. Table 3.20 gives a summary of the MoleculeEditor node.

Name	MoleculeEditor
Category	Tools
Description	Node that can show and edit molecules in 3D.
Usage	Node that can show and edit molecules in 3D.
Parameters	None
Input ports	Molecule data.
Output ports	Molecule data.

Table 3.20: Summary of the MoleculeEditor.

3.1.8 View

Nodes capable of inspecting data currently in the nodes after running the diagrams. These nodes are summed up in (table 3.21 and 3.22).

Name	ScreenDump
Category	View
Description	Node that is used to make plot based on calculated data in the Module Diagram.
Usage	This node's task is to save screendumps to file.
Parameters	None
Input ports	None
Output ports	Imagefile

Table 3.21: Summary of the ScreenDump node.

SpreadSheet Node

The spreadsheet node may take several matrices, vectors and scalars on its input port and display them in a spreadsheet. Table 3.22 gives a summary of the SpreadSheet Node, while a more detailed description is presented in section 3.5.

Name	SpreadSheet Node
Category	View
Description	Node that displays the selected data in a familiar spreadsheet.
Usage	View matrices, vectors and scalars in a spreadsheet representation. It is possible to save all the data that is in the spreadsheet to file, and to make selections which may be copied to the clipboard, for pasting it into another application.
Parameters	None
Input ports	matrices – the entry point for data.
Output ports	None

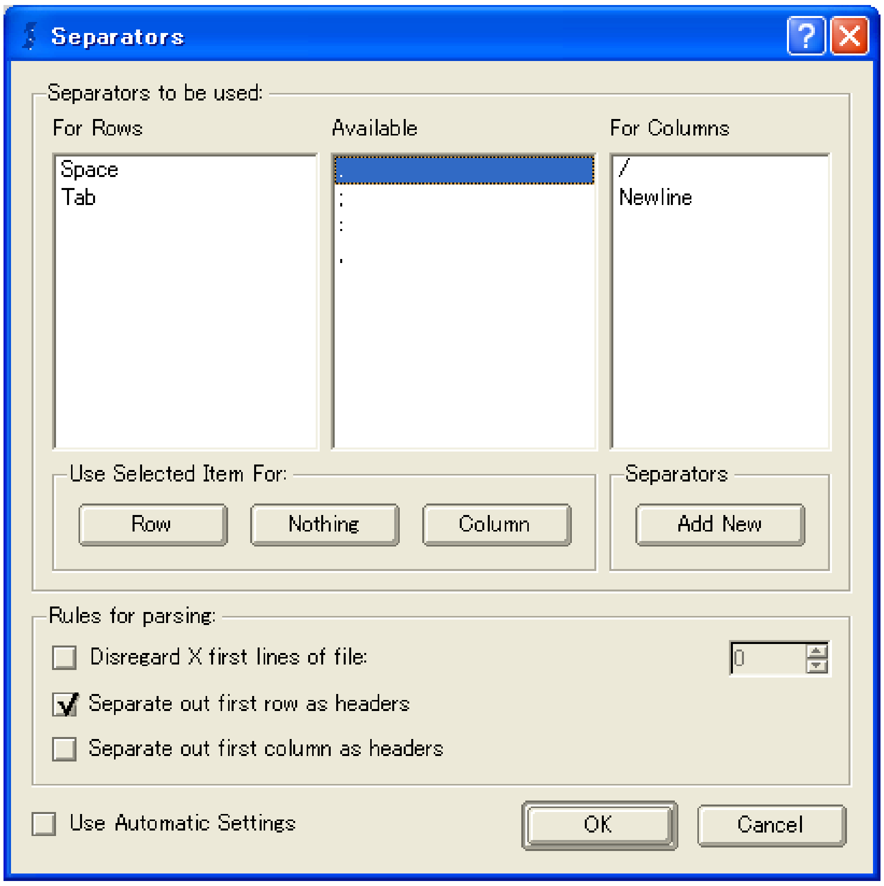
Table 3.22: Summary of the SpreadSheet Node.

3.2 Input Node

The input node is capable of reading all kinds of dataformats supported by SciCraft. After typing or selecting a file, the filetype and a discription of the filtype will apear. The different variables contained in the file will be readily available in the bottom half of the dialog. The input node funktions the same for all kinds of datafiles, except one; the textfiles. The reading of textfiles is described in 3.2.1, below.

3.2.1 Reading Textfiles

Since textfiles are so general, and can be used for so much in SciCraft, they are not limited to a specific format. When filetype *textfile* is selected, the format option button "Set separators" is unlocked. Pressing this button gives you the dialog in figure 3.1.



Separators to be used:

For Rows	Available	For Columns
Space	.	/
Tab	;	Newline
	:	
	,	

Rules for parsing:

- [ ] Disregard X first lines of file: 0
- [✓] Separate out first row as headers
- [ ] Separate out first column as headers

- [ ] Use Automatic Settings

Figure 3.1: Setting separators for reading a text file.

This dialog allows you to choose how the file should be represented in SciCraft. Whether you want it to be a "list of lines", a matrix where each line is a row and each word placed in a columb, or just a list of words. The Module Diagram "textreader.zmd" is an example of this. In addition to choosing between commenly used separators the dialog in figure 3.1 allows you to define your own by selecting *custom* and typing into the, now activated, field. You can also select the first row or column to be used as headers. If there are other information in the beginning that must be skipped, there is also an option to skip the X first lines.

3.3 EditNode

The edit node can be used to manipulate data before sending it on into the next node(s) in the module diagram. As of today, it is only possible to execute one command on a single EditNode. See section 3.3.1 for more details.

WARNING: With the current implementation, all the commands below can only handle matrices with numbers or single chars. Text in a matrix will make the operations fail. TECHNICAL: The reason for this, is that the package Numpy/Numeric is used to perform (parts) of the computations. The way we are using the package, does not support such matrices. How ever, the advantage is that the package is very fast.

3.3.1 Available commands

Currently, there are seven commands available (including the *passThrough* command that does nothing) in the EditNode. As mentioned before, it is as of today only possible to execute one command on a single EditNode. To perform several operations, you must use multiple EditNodes in your module diagram. We will make it possible to execute multiple commands on a single EditNode in a later version of SciCraft. For help on how to execute a command, see section 3.3.2.

In this section, commands are written in *italics*, arguments to commands are written with enclosing single quotes and mandatory arguments are presented in **bold face**.

joinMatrices

joinMatrices takes a number of matrices and joins them into one. The matrices given as input must have the same dimensions, except along the join axis. The join axis is specified with the argument 'axis'.

- Number of mandatory arguments: 1
- Total number of arguments: 1
- Valid input: $[2 \ldots N]$ inputs – matrices.
- Output: 1 output port with a single matrix. If operation fails, nothing is put on the output port.
- Arguments:
  1. **'axis'** – Must be an integer of value 0 or 1. If 'axis' is 0, the matrices are joined vertically, and if it is 1 the matrices are joined horizontally.

You can join a 10×50 matrix with a 10×100 matrix horizontally ('axis' $= 1$) and get a 10×150 matrix, but you cannot join them vertically, as they do not have the same number of columns.

passThrough

passThrough simply passes the data it reveices on unchanged. You want to use this if you want to be able to do picking in a plot and save the new dataset to a file or make use of it in another way in the module diagram.

- Number of mandatory arguments: 0
- Total number of arguments: 0

- Valid input: 1 input – anything
- Output: 1 output port, equals the data received on the input port.
- Arguments: None

removeColumns

removeColumns removes the specified columns from a dataset. The first column in the dataset is column number 1.

- Number of mandatory arguments: 1
- Total number of arguments: 2
- Valid input: 1 input – list, matrix
- Output: 1 output port with a list or matrix, depending on input. If all columns are removed, a empty list or matrix is returned.
- Arguments:
  1. **'columns'** – A list with the columns to remove, specified by column index. The first column is column 1.
  2. 'dimension' – The number of columns in the dataset to operate on. If there is a mismatch between the dataset on the input port and this argument, the command will not be executed. Must be specified as an integer.

removeRows

removeRows removes the specified rows from a dataset. The first row in the dataset is row number 1.

- Number of mandatory arguments: 1
- Total number of arguments: 2
- Valid input: 1 input – list, matrix
- Output: 1 output port with a list or matrix, depending on input. If all rows are removed, a empty list or matrix is returned.
- Arguments:
  1. **'rows'** – A list with the rows to remove, specified by row index. The first row is row 1.
  2. 'dimension' – The number of rows in the dataset to operate on. If there is a mismatch between the dataset on the input port and this argument, the command will not be executed. Must be specified as an integer.

subset

subset takes a list or a matrix as input, and returns a subset as specified by the arguments ‘upperLeft’ and ‘lowerRight’. To understand which values are extracted, think of a rectangle with corners as specified by the coordinates. The coordinates are inclusive; if you specify the element $(2, 5)$ as the upper left coordinate, this element will be included in the subset.

NOTE! With lists, it is possible to wrap around. For example, if you have list with 10 elements, you can specify ‘upperLeft’ as 8 and ‘lowerRight’ as 5. You will then get a list with the elements 8, 9, 10, 1, 2, 3, 4 and 5. This will not work with matrices.

- Number of mandatory arguments: 2
- Total number of arguments: 3
- Valid input: 1 input – list, matrix
- Output: 1 output port with a subset of the input as specified.
- Arguments:
  1. **‘upperLeft’** – The upper left coordinate for the subset. Must be specified as a tuple.
  2. **‘lowerRight’** – The lower right coordinate for the subset. Must be specified as a tuple.
  3. ‘dimension’ – The dimensions of the dataset to reduce. If the dimensions do not match, the command is not executed.

transpose

transpose transposes a dataset, i.e. swaps the axis specified. If the argument ‘axis’ is ommitted, the commands flips the order of all axis. If a matrix (rank 2) is supplied, the rows will become columns (and vice versa). If the rank of the dataset is 3 or higher, ie. a cube, you can use the ‘axis’ argument to specify the new axis order. For instance, (1,0,2) would swap axis 0 and 1 of a cubic, but leave axis 2 as it was.

NOTE! *transpose* is currently only supported for datasets of rank 2 or lower.

- Number of mandatory arguments: 0
- Total number of arguments: 1
- Valid input: 1 input – list, matrix
- Output: 1 output port with the transposed dataset.
- Arguments:
  1. ‘axis’ – A tuple specifying the new axis order. If ommitted, the order of all axis is flipped. This argument is of use only when the rank of the dataset is 3 or higher.

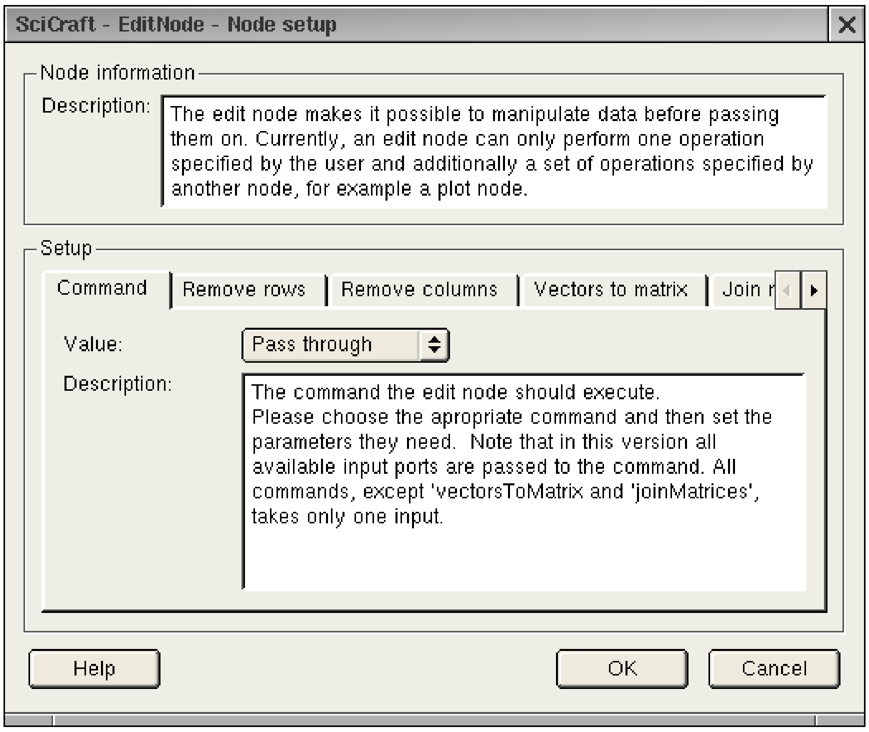
vectorsToMatrix

vectorsToMatrix takes a number of vectors and transforms them into a matrix. The vectors must be of equal length, and you must specify which axis the vectors are to be joined along.

- Number of mandatory arguments: 1
- Total number of arguments: 1
- Valid input: $[2 \ldots N]$ inputs – list/vector
- Output: 1 output port with the input vectors transformed into a matrix.
- Arguments:
  1. **'axis'** – An integer specifying which axis to join along. 0 makes the vectors rows, 1 makes them columns.

3.3.2 Executing commands

To execute a command in the edit node, first double click on the edit node in the module diagram to open the configuration window. A window like the one in figure 3.2 will appear. In the window shown, there are two main areas you need to interact with to configure the edit



Description: The edit node makes it possible to manipulate data before passing them on. Currently, an edit node can only perform one operation specified by the user and additionally a set of operations specified by another node, for example a plot node.

Value: Pass through

Description: The command the edit node should execute.
Please choose the apropriate command and then set the parameters they need. Note that in this version all available input ports are passed to the command. All commands, except 'vectorsToMatrix and 'joinMatrices', takes only one input.

Figure 3.2: Editnode configuration.

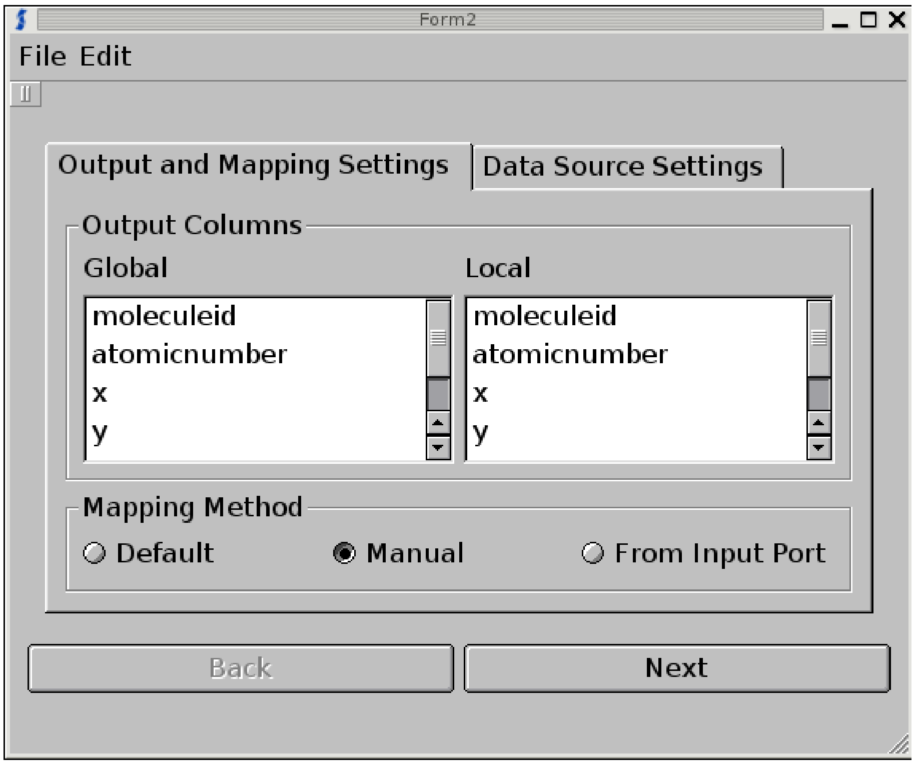
node. The first one is the list of tabs and the other one is the content of the tab.

The first thing to do, is to choose what command you want to execute. This is done by selecting the command in the spinbox under the tab "Commands". After this is done, you set the parameters that the command need in order to run, unless it does not need any. To do this you select the tab(s) that correspond with the command you selected. For instance, if you want to remove rows from a dataset, you first choose the command “Remove rows” in the spinbox under the tab “Commands”. Then you select the tab “Remove rows” and input the row number(s) for the row(s) you want to remove. Note that some commands do not need any parameters, and there will not be any tabs for these.

Please note that if selecting a parameter that needs two input data sets, like “joinMatrices”, you need to add a second input port, see **??**.

3.4 Gaussdal Node

The Gaussdal node is for treating molecule data retrieved from a SQL query in a Gaussdal database. The node is yet highly experimental, so use it at your own risk. Also, the documentation for the node will be improved during the next releases. To use this node one must connect the input to the Gaussdal node and then run the diagram. The input should be Gaussdal files with molecule data consisting of x- y- and z-coordinates, molecule id's and atomic numbers. Other attributes may also be provided, but is not mandatory. Also, a mapping matrix can be set on the "molecule_mapping" input port, giving information on which atom in which molecule is to be mapped to each other.



Output Columns

Global	Local
moleculeid	moleculeid
atomicnumber	atomicnumber
x	x
y	y

Mapping Method

Default [✓ Manual] From Input Port

Figure 3.3: Gaussdal molecule mapper dialog

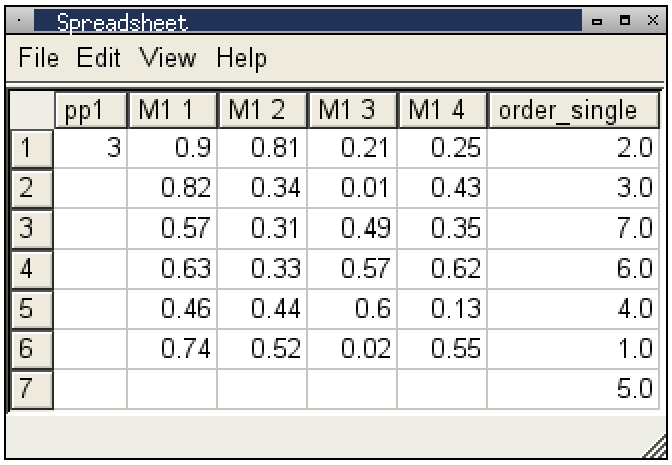
In the dialog of figure 3.3, the user can set global and local variables, and choose mapping type. The different mapping options are:

- *Default mapping:* generates a mapping based on atom order in the Gaussdal file, often giving the wanted mapping. This is set default when creating a new Gaussdal node, and by connecting a spreadsheet node to the output port on the Gaussdal node, the result can be viewed without opening the Gaussdal node.
- *Manual mapping:* the user creates a mapping by using the molecule wizard. Press next and follow instructions on screen. When the mapping process is finished, the user can choose whether she wants to run the Procrustes routine. If Procrustes is run, results are illustrated in a window. Select the wanted molecules from the list and press "Update".
- *From input port:* The mapping is performed according to a provided matrix on inputport "mapping_matrix" in the Gaussdal node. The matrix might e.g. have Octave format.

The options in the ”File” and ”Edit” menu are not working yet, but will be fixed as soon as possible.

3.5 SpreadSheet Node

The SpreadSheet node is capable of displaying matrices and vectors of data, as well as scalars. The vectors or matrices does not have to be of equal length. All the data that is set on the input port, "matrices" will be displayed in the spreadsheet. Figure 3.4 shows a typical setup.



	pp1	M1 1	M1 2	M1 3	M1 4	order_single
1	3	0.9	0.81	0.21	0.25	2.0
2		0.82	0.34	0.01	0.43	3.0
3		0.57	0.31	0.49	0.35	7.0
4		0.63	0.33	0.57	0.62	6.0
5		0.46	0.44	0.6	0.13	4.0
6		0.74	0.52	0.02	0.55	1.0
7						5.0

Figure 3.4: Spreadsheet

As can be seen in figure 3.4, there is three sets of data on the input port. The names of the data objects on the input port is shown in the top row. The data objects in the example are:

- A scalar, with the name "pp1", whose value is "3".
- A matrix, with the name "M1", with four columns and six rows. As can be seen in figure 3.4, the data object's name along with the column number is shown in the top row.
- A vector, with the name "order_single", with seven elements.

Figure 3.4 shows an example of the column-wise view-mode. The other view-mode is row-wise, and in that case the names of the data objects would be displayed on the left-most column of the spreadsheet. The "order_single" vector from the example would have been displayed as one row in the spreadsheet, with seven elements. The "M1" matrix from the example would have been displayed as four rows with six columns. Switching between the view-modes is accomplished through the *View* menu, or by using the shortcuts **"Ctrl + O"** for switching to the column-wise view-mode, or **"Ctrl + R"** for switching to the row-wise view-mode.

3.5.1 Copying data

It is possible to make single selections in the spreadsheet. There are three different ways of making selections:

- By clicking a single cell in the spreadsheet the single cell will be selected. By pressing the left mouse button on a cell and dragging the mouse to another, a range of cells are selected.
- By clicking a row or column header, that row or column is selected. By pressing the left mouse button on a row or column and dragging the mouse to another row or column header, a range of rows or columns are selected.

- By selecting the *Edit*-menu's option *Select all*, all the cells in the spreadsheet are selected. The shortcut for the *Select all*-action is **"Ctrl + A"**.

Once you have made a selection in any of the ways mentioned above, it is possible to select the *Edit*-menu's option *Copy* (the same result is achieved by using the shortcut **"Ctrl + C"**). Then all the selected data is copied to your platform's clipboard system. Then it is possible to paste the selection of data into, e.g., a spreadsheet system like Gnumeric or OpenOffice.org and do further data analysis there, or any other system that accepts pasting textual data. Please note that for large sets of data, the *Copy* action will take some time.

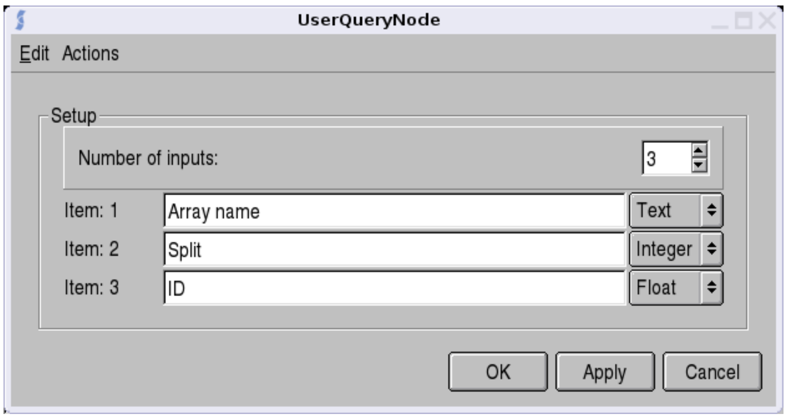
3.5.2 Saving data

It is possible to save all the data that is contained within a spreadsheet. By using the *File*-menu's option *Save data...* or pressing the shortcut **"Ctrl + S"**, you will be asked to enter a file name to save the data to. The filename's extension decides what kind of format the data contained within the spreadsheet will be saved in. The spreadsheet supports the same formats for saving as the Output nodes does. If the extension is not recognized, an error message will be displayed, and no data will be saved.

3.6 UserQueryNode

The *UserQueryNode* is a node that can be used to enter information about a dataset. The node takes a list as input and the output will be data that the user enters for each of the input rows. An example usage for this node would be a situation where the user needs to add some extra information about one of the datasets computed in a function node. The user query node can then be set up to require two data fields from the user about each row in the input. When the user has entered the information requested, the data will be sent as a matrix to the next node.

The UserQueryNode has two modes it can be in, *setup* and *use* mode. If there is no input on the node it will be in the mode, see Figure 3.5. Here the user can select how many input fields there will be for each input row. He can also add a column label for these fields and set a data type that the user data will be converted to.

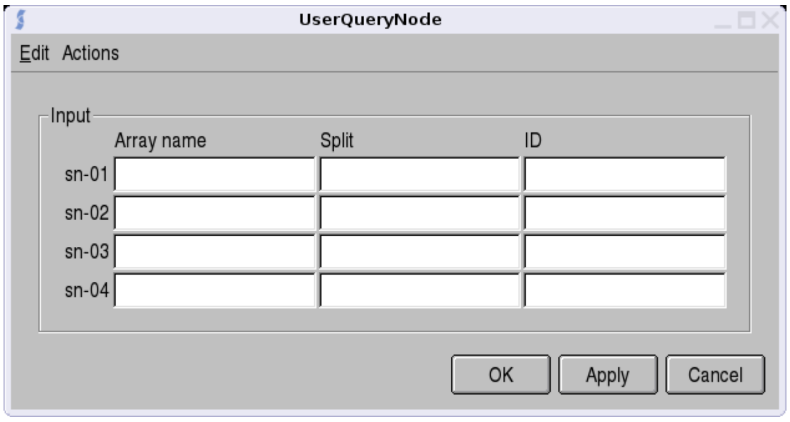


Number of inputs: 3

Item: 1	Array name	Text
Item: 2	Split	Integer
Item: 3	ID	Float

Figure 3.5: The UserQueryNode in setup mode.

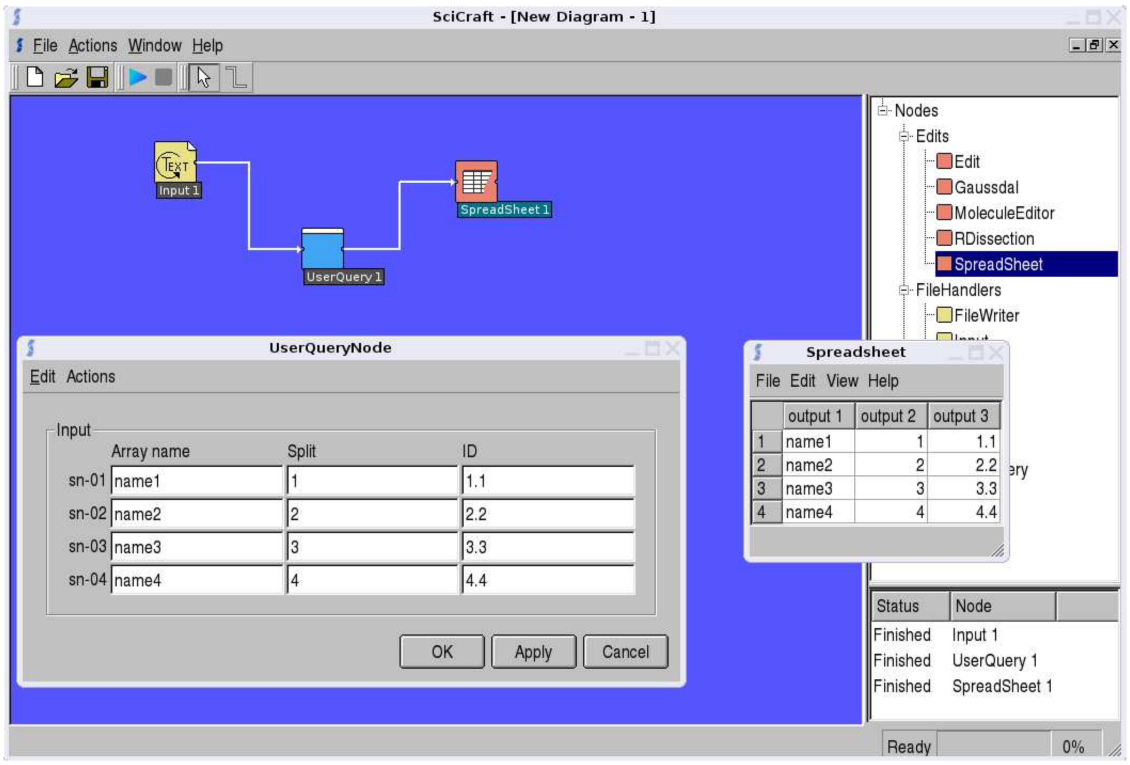
Figure 3.6 shows the node after is had been executed and has input it can use as row labels.



	Array name	Split	ID
sn-01			
sn-02			
sn-03			
sn-04			

Figure 3.6: The UserQueryNode in use mode.

After the node has be filled with data from the user, it can be executed again and the data will be available as output.



Input

	Array name	Split	ID
sn-01	name1	1	1.1
sn-02	name2	2	2.2
sn-03	name3	3	3.3
sn-04	name4	4	4.4

	output 1	output 2	output 3
1	name1	1	1.1
2	name2	2	2.2
3	name3	3	3.3
4	name4	4	4.4

Status	Node
Finished	Input 1
Finished	UserQuery 1
Finished	SpreadSheet 1

Figure 3.7: The UserQueryNode in use.

Chapter 4

Troubleshooting

This chapter aims at providing help and workarounds for problems in SciCraft. If your problem is not found in this chapter, please email us at scicraft@scicraft.org or file a bugreport at https://dev.scicraft.org/trac/.

4.1 R problems

4.1.1 Impossible to run R function nodes

If you are running SciCraft on Windows without a properly installed R you may get the following error message when executing an R function node: *This feature is unavailable. R is either not installed or not registered in the registry. Please consult the SciCraft documentation on how to install R correctly.* To fix this problem you need to ensure that R is installed and has registered its install path in the Windows registry. You have two options for solving this problem: either reinstall R and during the install, make sure you enable the option *Register R path for use by (D)COM server.* The second option is to create the registry key manually. This is done by placing the following in a file called e.g. register-r.reg:

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\R-core]

[HKEY_LOCAL_MACHINE\SOFTWARE\R-core\R]
"InstallPath"="C:\\Program Files\\R\\rw2001"
"Current Version"="2.0.1"
```

You can then execute the file. Make sure that you update this text with the correct path to your R installation.

4.1.2 Not enough results returned from R functions

When something is wrong with an R script you might get the following error message when you run it inside SciCraft: *HCA: Expected results for 2 ports, but plugin returned 0.* To see the specific error message give by R you need to enable R debugging. This is done on the preferences widget, click on the *File* menu and then click *Preferences.* It is recommended to disable this option when you are not looking for an R error.

Chapter 5

Third party copyright

The following listings provide information and documentation for software which is bundled with SciCraft.

5.1 ElementTree

ElementTree.py, Light-weight XML support for Python 1.5.2 and later. For documentation, visit http://effbot.org/zone/element-index.htm.

Copyright (c) 1999-2005 by Fredrik Lundh.

By obtaining, using, and/or copying this software and/or its associated documentation, you agree that you have read, understood, and will comply with the following terms and conditions:

Permission to use, copy, modify, and distribute this software and its associated documentation for any purpose and without fee is hereby granted, provided that the above copyright notice appears in all copies, and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of Secret Labs AB or the author not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

SECRET LABS AB AND THE AUTHOR DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANT- ABILITY AND FITNESS. IN NO EVENT SHALL SECRET LABS AB OR THE AUTHOR BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

5.2 Pyro

Pyro is Copyright (c) by Irmen de Jong.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Appendix A

Manual installation on Linux

If you have an unsupported Linux distribution or Unix-like operating system installed, but still want to use SciCraft, you may have to install one or more of the following packages manually. But be aware of that one or more of these packages might already be available as precompiled packages or install scripts for the OS that you are running. Also, some of these programs may have dependencies not covered here.

A.1 Dependencies

Python

- Version 2.3.3 or later
- Download link: `http://www.python.org/2.3.3/`
- Install guide: In the file distribution

Qt for X11

- Version: 3.3.3
- Download link: `ftp://ftp.trolltech.com/qt/source/qt-x11-free-3.3.3.tar.gz`
- Install guide: In the file distribution

PyQt

- Version: 3.13 or later
- Download link: `http://www.riverbankcomputing.co.uk/pyqt/download.php`
- Install guide: In the file distribution

VTK

- Version: 4.2.4 or later
- Download link: `http://www.vtk.org/files/release/4.2/VTK-4.2.5-tar.gz` and `http://www.vtk.org/files/release/4.2/VTKData-4.2.tar.gz`

- Install guide: In the file distribution. Here are some tips to make the process a little less painful:
  - When running `cmake -i` or `ccmake` make sure to enable
    - python_wrap
    - hybrid
    - shared libraries
  - Also make a note of `CMAKE_INSTALL_PREFIX`'s value, you will need it later.
  - After configuring and generating (in `ccmake`) compile and install:

    ```
    > make
    > make install
    ```

  - Install the python wrappers:

    ```
    > cd Wrapping/Python
    > python setup.py install
    ```

  - Add `$CMAKE_INSTALL_PREFIX/lib/vtk` to the system variable `LD_LIBRARY_PATH`. For example, if you use bash, and the value of `CMAKE_INSTALL_PREFIX` is `/usr/local`, run the following command:

    ```
    > export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib/vtk
    ```

    It may be a good idea to put this line in a system startup script like `/etc/profile` so users don't have to type the line every time they log in.

Octave

- Version: 2.1.X
- Download link: http://www.octave.org/download.html
- Install guide: http://www.octave.org/doc/octave_33.html

R

- Version: 1.8.1 or later
- Download link: http://cran.r-project.org/mirrors.html
- Install guide: http://cran.r-project.org/doc/manuals/R-admin.pdf

RPy

- Version: 0.3.5 or later
- Download link: http://rpy.sourceforge.net/download.html
- Install guide: See the `README` file in the file distribution

Numeric

- Version 23.3
- Download link: `http://sourceforge.net/projects/numpy/`
- Install guide: in the file distribution

A.2 Optional

These programs will enhance SciCraft's functionality significantly.

PyMOL

- Version: 0.93
- Download link: `http://sourceforge.net/project/showfiles.php?group_id=4546`
- Install guide: `http://pymol.sourceforge.net/newman/user/S0120install.html#4`

PyQwt

- Version: 0.97
- Download link: `http://sourceforge.net/project/showfiles.php?group_id=4546`
- Install guide: `http://pymol.sourceforge.net/newman/user/S0120install.html#4`

Numarray

- Version: 1.0
- Download link: `http://sourceforge.net/projects/numpy/`
- Install guide: `http://stsdas.stsci.edu/numarray/numarray-1.0.html/node9.html`

PyX

- Version 0.6.3
- Download link: `http://pyx.sourceforge.net/`
- Install guide: in the file distribution

SOAPpy

- Version 0.11.6
- Download link: `http://pywebsvcs.sourceforge.net/`
- Install guide: in the file distribution

A.3 Installation commands

To install SciCraft manually you run the following commands:

- Get latest source release from http://www.scicraft.org and place in a temporary working directory.
- Extract files from source distribution.

```
> tar xzvf scicraft_<version number>.tar.gz
```

- Become root, build and install files.

```
> cd scicraft_<version number>.tar.gz
> su -
> make install
```

This will install needed files in /usr/lib/scicraft, /usr/share/scicraft, /usr/share/doc/scicraft and a startupscript, scicraft, in /usr/bin.

Bibliography

[1] Bjørn Kåre Alsberg, Lars Kirkhus, Reidar S. Hagen, Øyvind Knudsen, Truls Tangstad, and Endre Anderssen. Zherlock: An open source data analysis software. *SAR and QSAR in Environmental Research*, 14:349–361, 2003.

[2] Alsberg B.K., Kirkhus L., Tangstad T., and Anderssen E. Data analysis of microarrays using scicraft. *Lecture notes in artificial intelligence 3303*, pages 58–68, 2004.