ParaView Python Crib Sheet

ParaView is an open-source modular visualization environment that is popular for the analysis and presentation of the results of computational simulations. This document does not cover plugins because 1,) a plugin is different as it is compiled with the ParaView source code (this means a plugin should be more computationally efficient than using Python scripts within a ParaView session) and 2,) it can have a graphical user interface (GUI) that is integrated into the ParaView GUI. A plugin is more sophisticated than a Python script but a script can be the basis of a plugin.

This crib sheet was created for ParaView version 5.7.0 in 2020 and revised for ParaView version 5.10 in 2022.

This crib sheet is a test or pilot version.

The ParaView Python Interface

There are a variety of ways to use Python with a ParaView Session.

Tracing

You can record your actions during a ParaView session in a Python code, which can be can be re-run as a script. You can start a recording through the menu system, at the bottom of the `Tools` menu there is a `Trace` option i.e., `Tools → Start Trace`. The recording is switched off at the same menu place but the option is now `Tools → Stop Trace`.

The Python script can be watched as it is recorded in a separate window to ParaView thus allowing you to observe what functions a GUI action relates to. These scripts can be copied and pasted into a text file and saved with a `.py` extension. They can then be edited and run through the Internal Python Shell.

The Internal Python Shell

You can open an internal Python Shell that is attached to ParaView through the menu system. This is made visible through the option `View → Python Shell`. (In older versions of ParaView this was available in `Tools → Python Shell`). You can type in Python commands here or there is a button at the bottom of the shell to browse for and then run a Python script.

The External Python Shell

If you run the `pvpython` binary from a shell you can type Python commands into the shell and it will execute Python commands in a ParaView session that has no Window and is not visible. In Linux you would run the command `./pvpython` to run the Python enabled ParaView binary.

You will need to import the Python module `paraview.simple` to be able to access the paraview functions (this is automatically import in the internal Python shell) and the syntax is currently `from paraview.simple import *` but can reliably taken from there.

Batch Mode – Not Interactive

There are many reasons why you may not want to run ParaView interactively for example if you want to run it in batch mode through a scheduling system. This is achieved through `pvbatch` which is different to `pvpython` in only 2 ways: it only accepts commands from input scripts, and it will run in parallel if it was built using MPI.

Information on finding the binary is given in the following sections.

ParaView Home Directory

While some of the ParaView Python functionality can be accessed through the user interface not all can. There are several binary versions of ParaView that provide different access and functionality. These binary files are kept in the /bin directory where the software is installed.

The binaries are:

- hydra_pmi_proxy
- mpiexec
- paraview
- paraview-mesa
- pvbatch
- pvdataserver
- pvpython
- pvrenderserver
- pvserver

Other files are:

- qt.conf

We are only interested in 2 of the binaries if we wish to run a normal session of ParaView and use Python in that session. The binary `paraview` is executed for a normal ParaView session and `pvpython` to run an external Python shell which executes ParaView unseen, with no GUI.

Finding the Binary Files

The easiest way to execute a binary file is to find it in the system's menu and click on the menu option there. It may be that the `pvpython` binary is not obvious in the menu system. If that is the case you need to find the directory where ParaView is installed and then navigate to the `bin` directory which stores all the binaries.

On Windows

Find the ParaView menu option for the system or on the Desktop then right click on it and open the Properties menu. The path (location) of the binary is given there. Be careful not to alter this path.

On Linux

If ParaView is installed directly for all user to access all the time then type the command `which paraview` to find the path. If paraview is available through an environment module then you first need to load the paraview module which a command something similar to `module load paraview` (the exact syntax of this will depend on how it was set up by the system administrator) and then type the `which paraview` command.

Work Around with a Local ParaView Binary Installation

If you do not have access to the installation directories then you can either use a local binary version or compile a version in your user area. To use basic Python functionality, you do not need to compile your own version so you can download a local binary version.

Go to the paraview website and download the version of paraview that is correct for your operating system. At the time of writing the url for the download was https://www.paraview.org/download/

On Windows

Create a `paraview` directory in your user area that is easy for you to access. Move the downloaded file to the `paraview` directory that you have created. The file will be a zip file (with the file extension `.zip`) and extract the contents of this zip file to this directory. You can normally start the file extraction function by clicking on a zip file with your right mouse button and selecting `Extract` as an option from the menu that appears. You will find the binary file in the first level of directories in the extracted files.

On Linux

Create a `paraview` directory in your user area that is easy for you to access. Move the downloaded file to the `paraview` directory that you have created. The file will be a tar gzipped file (with the extension `.tar.gz`) and extract the contents of this file into newly created directory. You can extract the contents of this file with the `tar -xvf` command.

Executing a Binary

Using a file browser (Linux and Windows) navigate to the directory where ParaView is installed and then move into the `bin` directory. Then double click on the binary you wish to run.

On Windows

In Windows you can also execute a binary by searching the menu and clicking on the right ParaView option in the menu or you can save the option to the DeskTop and click on it from there.

On Linux

From the command line type `./paraview &`

Web Links

Python Specific

ParaView/Python Scripting – Quick Start tutorial for ParaView 3.6 or higher:
https://www.paraview.org/Wiki/ParaView/Python_Scripting

Video of using the internal Python shell to read and write CFD data:

https://www.youtube.com/watch?v=7lqYJxmp4_4

ParaView and Python - overview notes:
https://www.paraview.org/Wiki/ParaView_and_Python

ParaView Guide 5.7.0 – the paraview guide has Python snippets for many functions also links to information on paraview.simple Python module and the API:
https://www.paraview.org/paraview-guide/
https://kitware.github.io/paraview-docs/latest/python/

ParaView and Batch – this is out of data pvpython can no longer take arguments, explains the old differences between pvpython and pvbatch where both could run Python replaced by the Python Scripting page.
https://www.paraview.org/Wiki/ParaView_and_Batch

util.vtkAlgorithm Module - a convenient Python-friendly mechanism to add new algorithms
https://kitware.github.io/paraview-docs/latest/python/paraview.util.vtkAlgorithm.html

simple example python filters and data generators:
https://www.paraview.org/Wiki/Python_Programmable_Filter
https://www.paraview.org/Wiki/ParaView/Simple_ParaView_3_Python_Filters

ParaView/Plugin How To – shows how plugins are more complex than just using Python with ParaView:
https://www.paraview.org/Wiki/ParaView/Plugin_HowTo

General

Web site:
http://www.paraview.org/

Wiki:
http://www.paraview.org/Wiki/ParaView

Mailing list:
http://www.paraview.org/mailing-lists/

The ParaView Tutorial (and Data):
http://www.paraview.org/Wiki/The_ParaView_Tutorial