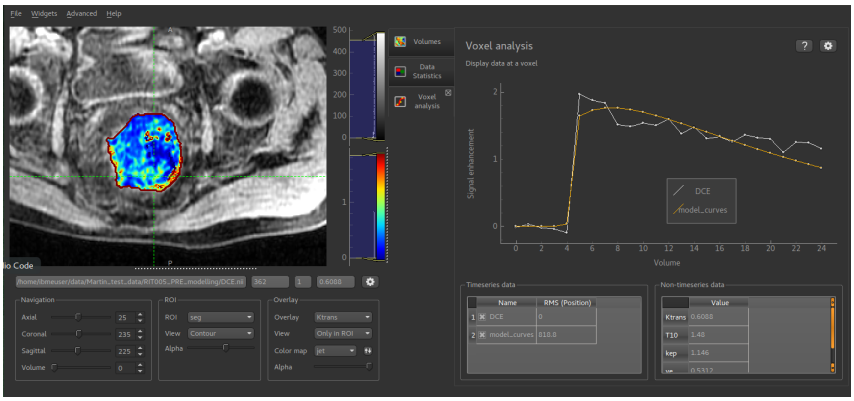
quantiphyse Documentation

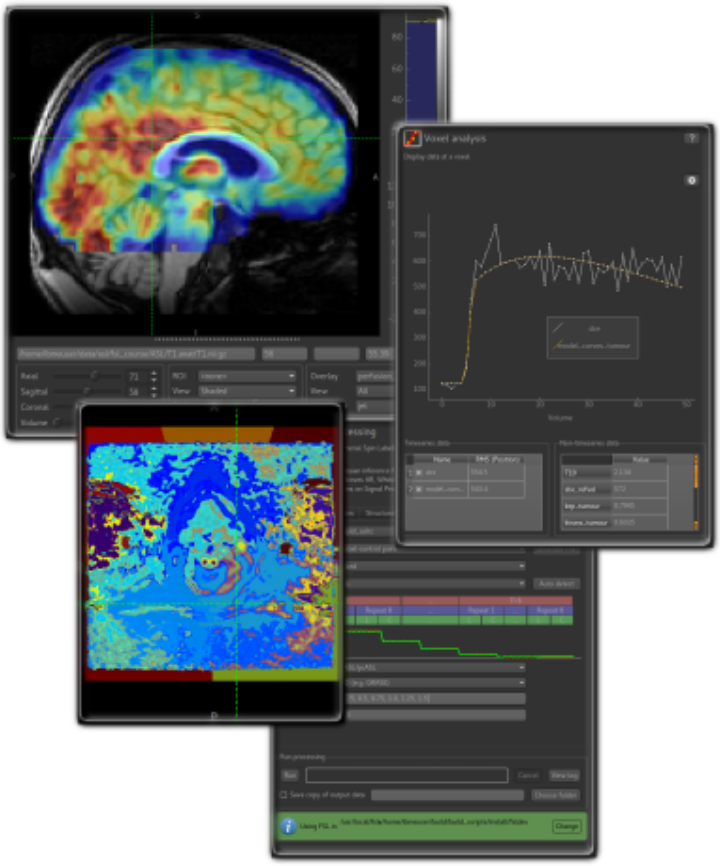
Martin Craig, Ben Irving

May 15, 2019

Contents



Quantiphyse is a visualisation and analysis tool for 3D and 4D biomedical data. It is particularly suited for physiological or functional imaging data comprised of multi volumes in a 4D (time-) series and/or multimodal imaging data. Quantiphyse is built around the concept of making spatially resolved measurements of physical or physiological processes from imaging data using either model-based or model-free methods, in a large part exploiting Bayesian inference techniques. Quantiphyse can analyse data both voxelwise or within regions of interest that may be manually or automatically created, e.g. supervoxel or clustering methods.



CHAPTER 1

Features

- 2D orthographic visualisation and navigation of data, regions of interest (ROIs) and overlays
- Universal analysis tools including clustering, supervoxel generation and curve comparison
- Tools for CEST, ASL, DCE and DSC-MRI analysis and modelling
- Integration with selected FSL tools
- ROI generation
- Registration and motion correction
- Extensible via plugins - see *Quantiphyse plugins*.

CHAPTER 2

License

© 2017-2019 University of Oxford

Quantiphyse is **free for non commercial** use. The license details are displayed on first use and the `LICENSE` file is included in the distribution. For further information contact the OUI Software Store. If you are interested in commercial licensing you shold contact OUI in the first instance.

CHAPTER 3

Tutorials

- CEST-MRI tutorial
- IMAGO ASL-MRI tutorial
- FSL ASL-MRI tutorial

CHAPTER 4

Getting Quantiphyse

Quantiphyse is available on PyPi - see *Installation of Quantiphyse*.

Major releases of Quantiphyse are also available via the Oxford University Innovation Software Store. The packages held by OUI have no external dependencies and can be installed on Windows, Mac and Linux. They may lag behind the current PyPi release in terms of functionality.

CHAPTER 5

User Guide

5.1 Getting Started

5.1.1 Installation of Quantiphyse

Quantiphyse is in PyPi and therefore *in principle* if you have Python, installation is as simple as:

```
pip install quantiphyse
```

In practice it is often *not* as simple as this. The main reason is `PySide` (the library we use for the user interface). This needs to be compiled against a rather old version of the QT GUI library which requires separate installation.

Alternatively a binary version of PySide can be installed but a suitable package isn't available for every version of Python.

Below are a number of 'recipes' for different platforms which have been verified to work. If you find a problem with one of these recipes, please report it using the Issue Tracker.

Note: To use some plugins you'll need to have a working `FSL` installation. For more information go to FSL installation.

Platforms

- *Ubuntu 16.04 / 18.04*
- *Centos 7*
- *Windows*
- *Mac OSX*
- *Homebrew*

- *Anaconda*

Ubuntu 16.04 / 18.04

From a terminal window:

```
sudo apt install libqt4-dev qt4-qmake cmake python-dev python-setuptools
```

To install pip on Ubuntu 16.04:

```
sudo easy_install pip
```

On Ubuntu 18.04:

```
sudo apt install python-pip
```

Now install the application:

pip install quantiphyse –user

The last step will take a while! The PySide GUI library is being built - the terminal will show:

```
Running setup.py install for PySide ... |
```

Go get a coffee and come back later.

The recipe above just installs the main application. To install plugins use:

```
pip install quantiphyse-cest quantiphyse-asl quantiphyse-cest quantiphyse-dce␣
↪quantiphyse-dsc quantiphyse-t1 quantiphyse-fsl quantiphyse-sv --user
pip install deprecation==1.2 --user
```

The last step corrects a startup problem caused by a dependency - see the *Frequently Asked Questions* for more information.

Alternatively, you can use *Anaconda* in Ubuntu.

You can also use the method above in a virtualenv or a Conda environment. To do this:

- Run the first `sudo apt install` command above
- Create and activate a Conda or virtual environment, e.g. as described in the *Anaconda* section
- Run the `pip install` commands above

This is a slightly better method as it keeps Quantiphyse and all it's dependencies in an isolated environment, however it does mean you will need to activate the environment in order to run Quantiphyse.

Centos 7

This recipe was tested in a Gnome Desktop installation. Open a terminal window and use the following:

```
sudo yum install qt-devel cmake python-devel gcc gcc-c++
sudo easy_install pip
pip install cython numpy six==1.10.0 setuptools --upgrade --user
pip install quantiphyse --user
```

The last step will take a while! The PySide GUI library is being built - the terminal will show:

```
Running setup.py install for PySide ... |
```

Go watch some cat videos and come back later.

The recipe above just installs the main application. To install plugins use:

```
pip install quantiphyse-cest quantiphyse-asl quantiphyse-cest quantiphyse-dce␣
↪quantiphyse-dsc quantiphyse-t1 quantiphyse-fsl --user
pip install deprecation==1.2 --user
```

The last step corrects a startup problem caused by a dependency - see the *Frequently Asked Questions* for more information.

Alternatively, you can use *Anaconda* in Ubuntu.

Windows

On Windows we strongly recommend using the Anaconda python distribution to install Python - see *Anaconda* below.

Mac OSX

On Mac we recommend either the Anaconda python distribution - see *Anaconda* or *Homebrew*. The system python has difficulties installing `PySide` due to the old version of Qt that is required.

Homebrew

To be completed…

Anaconda

Anaconda (https://www.anaconda.org) is an easy to install distribution of Python which also includes the `conda` tool for installing packages. We find `conda` generally better than `pip` for dependency management and binary packages such as `pyside`. Anaconda can be installed on Windows, Mac and Linux.

You will need to install the Anaconda environment before using any of these recipes. When selecting a Python version, `Python 2.7` is the version on which Quantiphyse has been most tested, however you can also use `python 3.x`. We intend to make Quantiphyse compatible with both version of Python for the foreseeable future although we are currently moving to Python 3 as the main development platform.

Once installed, use the following commands from a command prompt:

```
conda create -n qp
conda activate qp
conda config --add channels conda-forge
conda install cython funcsigs matplotlib nibabel numpy pillow pyqtgraph pyside pyyaml␣
↪requests scipy scikit-learn scikit-image setuptools six pandas deprecation
pip install quantiphyse --no-deps
```

This installs the basic Quantiphyse app. To install plugins use pip, for example this is to install all current plugins:

```
pip install quantiphyse-cest quantiphyse-asl quantiphyse-cest quantiphyse-dce␣
↪quantiphyse-dsc quantiphyse-t1 quantiphyse-fsl quantiphyse-sv
pip install deprecation==1.2
```

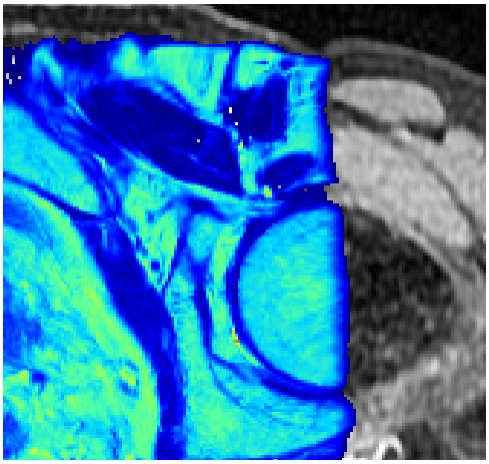
The last step corrects a startup problem caused by a dependency - see the *Frequently Asked Questions* for more information.

On Mac you will also need to do:

```
pip install pyobjc
```

In the future we hope to put Quantiphyse into conda itself so the whole process can consist of `conda install quantiphyse`.

5.1.2 Overview



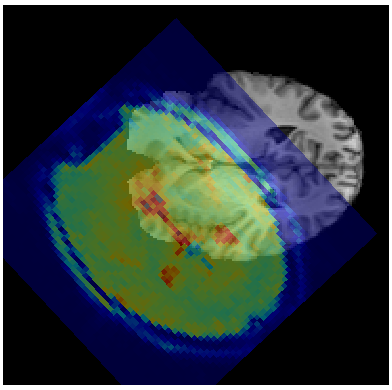
Quantiphyse is a visual tool for quantitative analysis of medical images. The aim is to bring advanced analysis tools to users via an easy-to-use interface rather than focusing on the visualisation features themselves.

The software is also designed to support advanced usage via non-GUI batch processing and direct interaction with the Python code.

Quantiphyse works with two types of data:

- 3D / 4D data sets.
- Regions of interest (ROIs). These must be 3D and contain integer data only. Voxels with the value zero are taken to be outside the region of interest, nonzero values are inside. ROIs with more than one nonzero value describe multi-level regions of interest which are handled by some of the tools.

One data set is used as the *main data*. This defaults to the first 4D data to be loaded, or the first data to be loaded, however you can set any data set to be the main volume. Data which should be treated as an ROI is normally identified when it is loaded (or created by a processing widget), however this can be changed after the data is loaded if it is incorrect.



Data orientation

Quantiphyse keeps all data loaded from files in its original order and orientation.

For display purposes, it takes the following steps to display data consistently:

- A *display grid* is derived from the grid on which the main data is defined. This is done by flipping and transposing axes only so the resulting grid is in approximate RAS orientation. This ensures that the right/left/anterior/posterior/superior/inferior labels are in a consistent location in the viewer. Note that the main data does *not* need resampling on to the display grid as only transpositions and flips have occured.
- Data which is defined on a different grid will be displayed relative to the display grid. If possible this is done by taking orthogonal slices through the data and applying rotations and translations for display. In this case no resampling is required for display.
- If the data cannot be displayed without taking a non-orthogonal slice, this is done by default using nearest neighbour interpolation. This is fast, and ensures that all displayed voxels represent 'real' raw data values. However, display artifacts may be visible where the nearest neighbour changes from one slice to another.
- To avoid this, the viewer options allow for the use of linear interpolation for slicing. This is slightly slower but produces a more natural effect when viewing data items which are not orthogonally oriented.

It is important to reiterate that these steps are done for *display* only and do not affect the raw data which is always retained.

Analysis processes often require the use of multiple data items all defined on the same grid. When this occurs, typically they will resample all the data onto a single grid (usually the grid on which the main data being analysed is defined).

For example if fitting a model to an ASL dataset using a T1 map defined on a different grid, the T1 would be resampled to the grid of the ASL dataset. Normally this would be done with linear interpolation however cubic resampling is also available. This is the decision of the analysis process. The output data would then typically be defined on the same grid, however again this is the choice of the analysis process.

Special cases

Quantiphyse will try to handle some special cases which would otherwise prevent data being loaded and processed properly.

Multi-volume 2D data

Some data files may be 3 dimensional, but must be interpreted as multiple 2D volumes (e.g. a time series) rather than a single static 3D volume. When a 3D data set is loaded, an option is available to interpret the data as 2D multi-volumes. To access this option, click the `Advanced` checkbox in the data choice window.

Note: In Nifti files the first 3 dimensions are required to be spatial, so where this occurs with a Nifti file it implies that the file is incorrectly formed and you should ideally correct the acquisition step which produced it.

5.1.3 Orientation

Loading and Saving Data

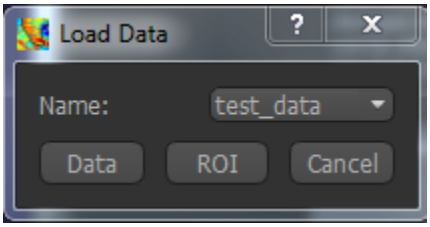
File Formats

This software package works with NIFTI volumes. Some builds may contain experimental support for folders of DICOM files, however this is not well tested.

Alternative packages which are able to convert DICOM files to NIFTI include the following:

- itk-snap
- dcm2nii
- Or the batch version which allows a number of volumes to be converted dcm2niibatch

Loading data using Drag and Drop



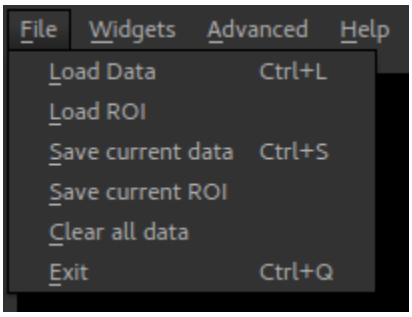
You can drag and drop single or multiple files onto the main window to load data. You will be prompted to choose the type of data:

The suggested name is derived from the file name but is modified to ensure that it is a valid name (data names must be valid Python variable names) and does not clash with any existing data.

If you choose a name which is the same as an existing data set, you will be asked if you wish to overwrite the existing data.

When dropping multiple files you will be asked to choose the type of each one. If you select *cancel* the data file will not be loaded.

Loading data using the menu



The `File -> Load Data` menu option can be used to load data files:

You will be prompted to choose the file type (data or ROI) and name in the same was as drag/drop.

Saving Data

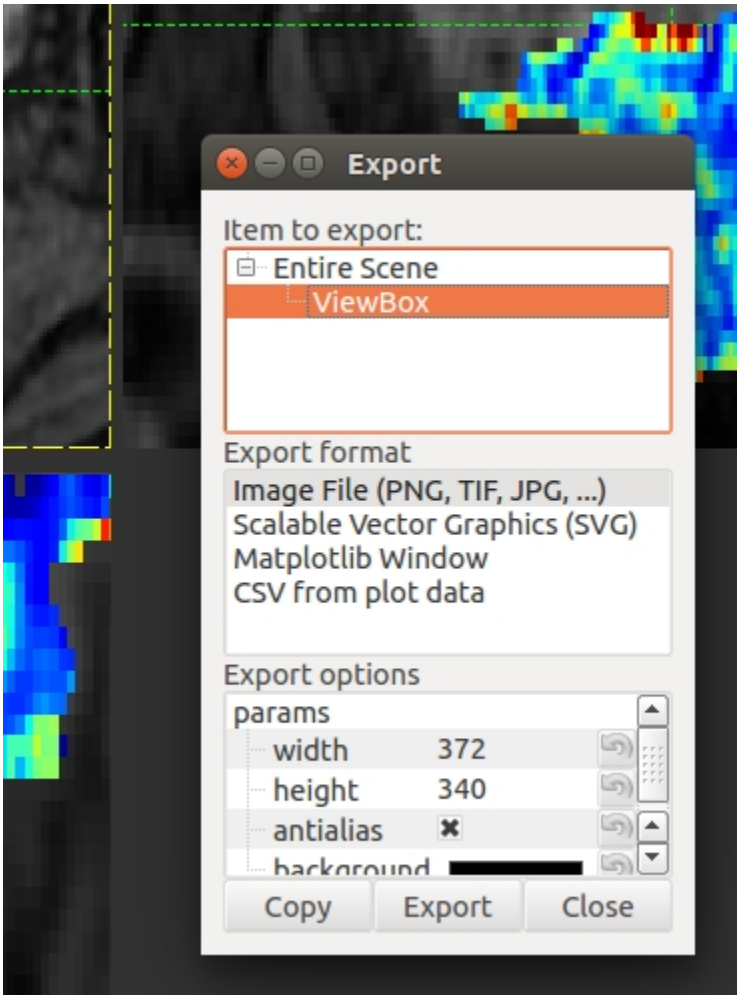
The following menu options are used for saving data:

- File -> Save current data
- File -> Save current ROI

So, to save a data set you need to make it the current data, using the Overlay menu or the Volumes widget. Similarly to save an ROI you need to make it the current ROI. Saving the main data can be done by selecting it as the current overlay.

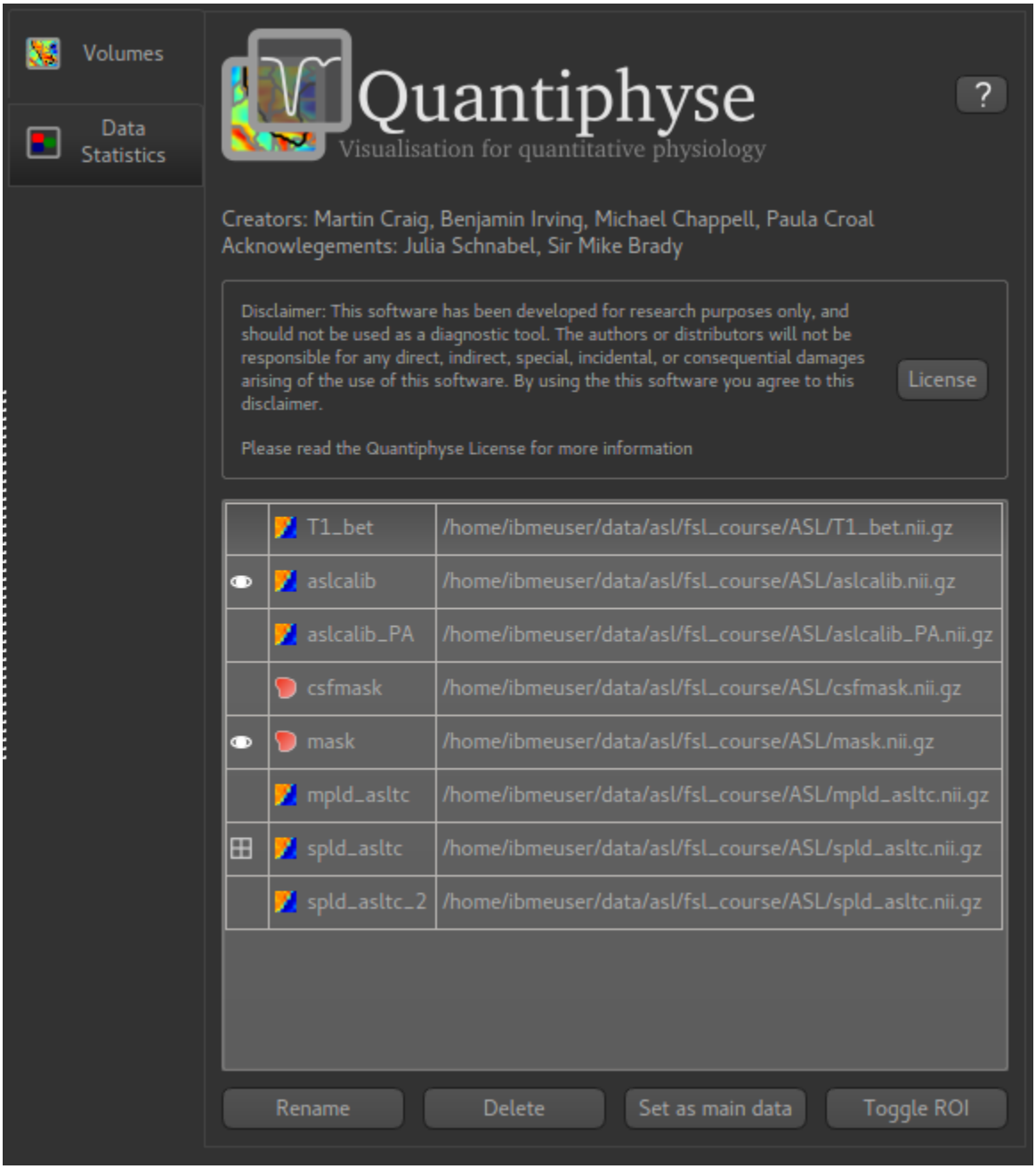
Save a screen shot or plot

- Right click on an image or plot
- Click *Export*
- A view box will appear with the various format options.
- *svg* format will allow editing of the layers and nodes in inkscape or another vector graphics viewer.



The Volumes List

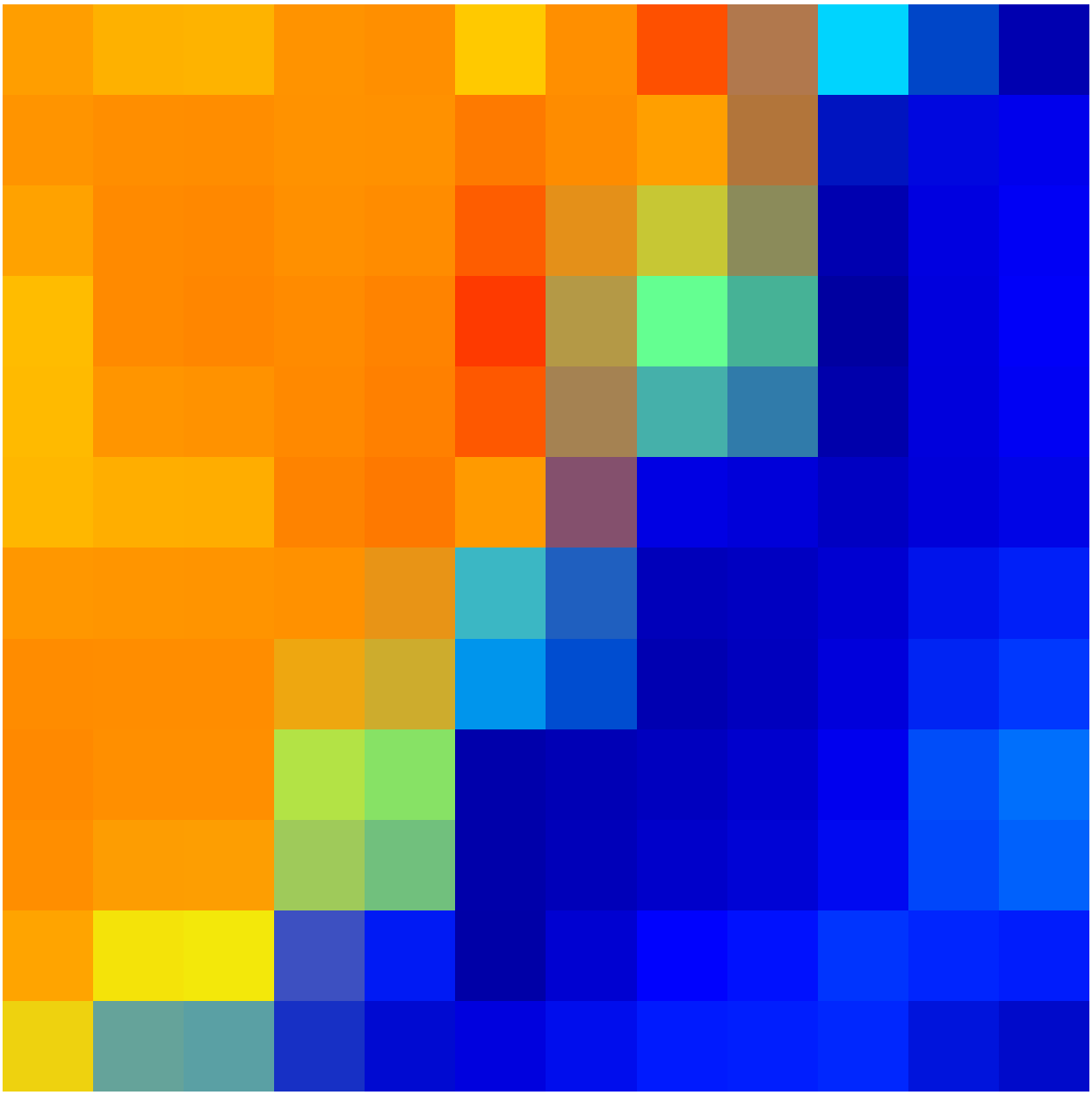
After loading data it will appear in a list on the `Volumes` widget, which is always visible by default:



The icon on the left indicates whether the data is visible or not:  indicates that this is the main data (and will appear as a greyscale background),  indicates that this data item is visible, either as an ROI or an overlay on top of the background.

Currently one overlay and one ROI are visible at a time. This limitation was intended to support the most common use case, but in the future the option to display multiple overlaid images will be added.

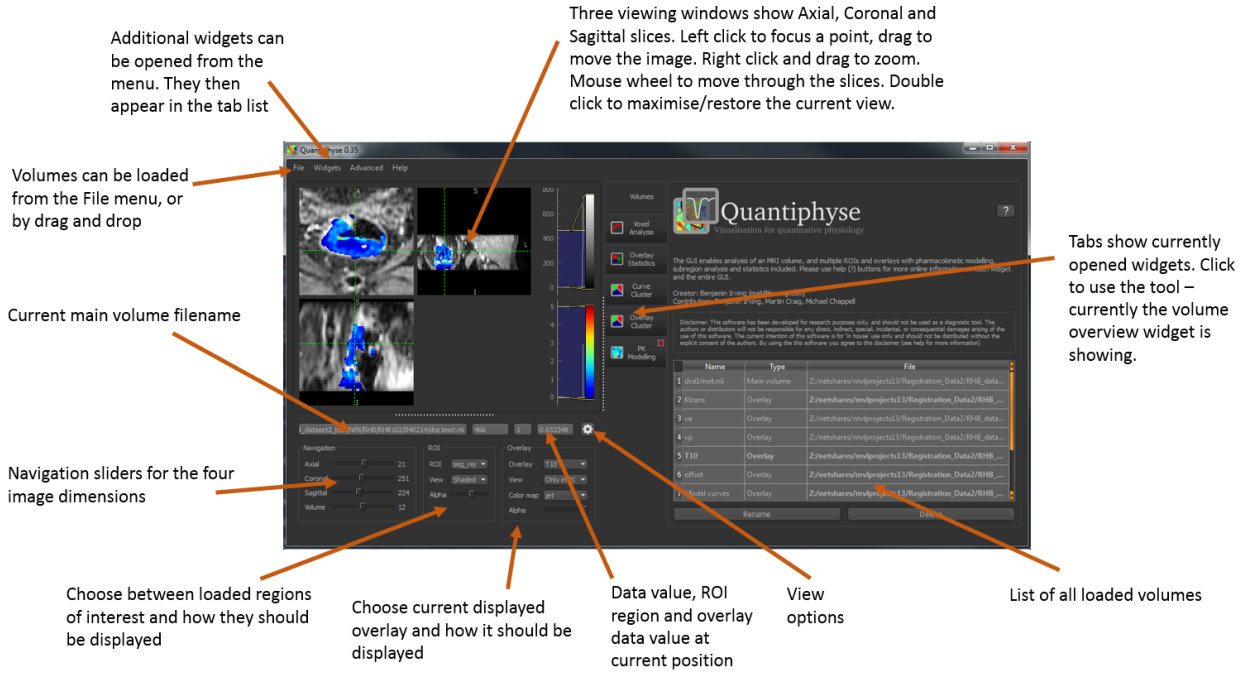
The icon next to the data name shows whether it is an  ROI or a



data set.

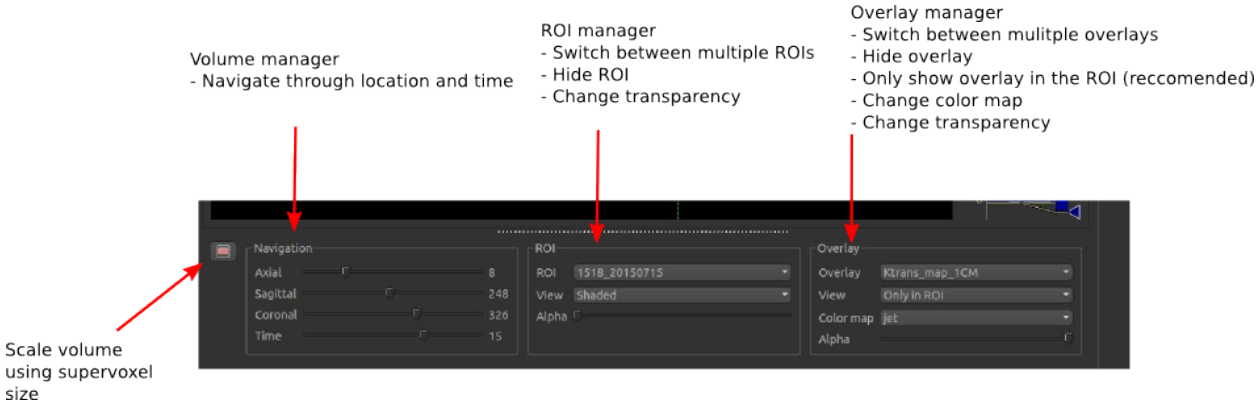
The Main Window

The main window is quite busy, below is an overview of the main functions:

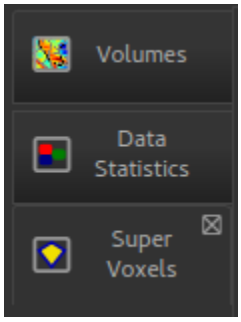


The Navigation Bar

The navigation bar is below the main image viewer and allows the current viewing position, current ROI and current data to be changed:



Using Widgets



Widgets appear to the right of the viewer window. Most widgets are accessed from the 'Widgets' menu above the viewer.

When selected, a widget will appear with a tab to the right of the viewer. You can switch between opened widgets by clicking on the tabs. A widget opened from the menu can be closed by clicking on the X in the top right of its tab.

Widgets may have very different user interfaces depending on what they do, however there are a number of common elements:

Help button

This opens the online documentation page relevant to the widget. Internet access is required.

Options button

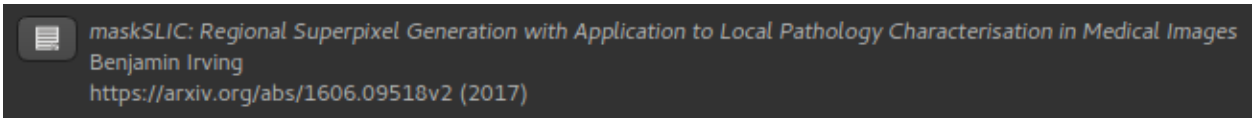
This shows any extended options the widget may have. It is typically used by widgets which display plots as that limits the space available for options.

Batch button

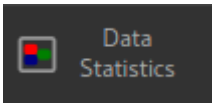
This displays the batch code required to perform the widget's processing, using the currently selected options. This can be useful when building batch files from interactive exploration. It is only supported by widgets which provide image processing functions.

Citation

Many widgets are based around novel data processing techniques. The citation provides a reference to a published paper which can be used to find out more information about the underlying method. If you publish work using a widget with a citation, you should at the very least reference the paper given.

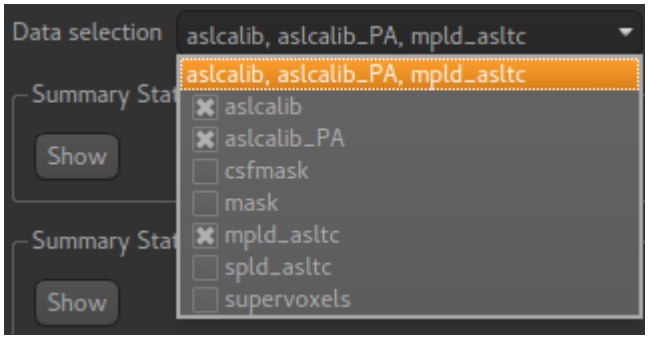


Clicking on the citation button performs an internet search for the paper.

5.1.4 

Data Statistics Widget

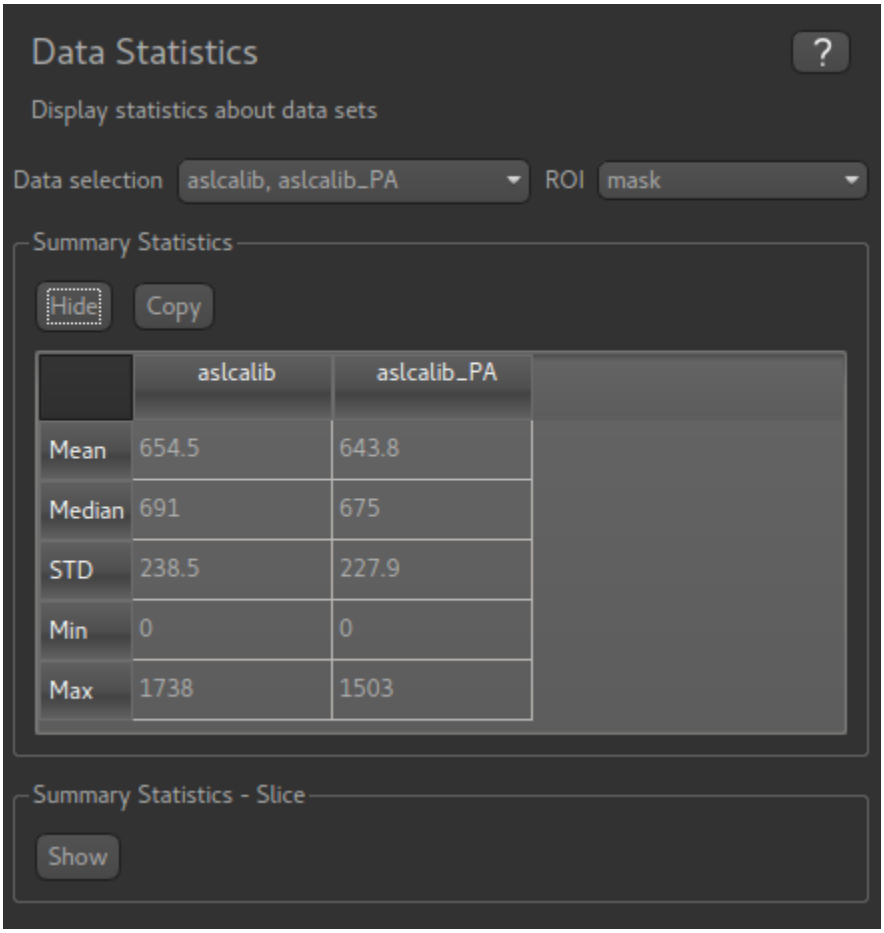
This widget displays summary statistics for selected data. The mean, median, standard deviation and range are presented.



You can select any number of data items and an optional ROI from the menus at the top. Clicking on the menu brings up a list of checkboxes to select the data items you want to include. Clicking outside the menu closes the list.

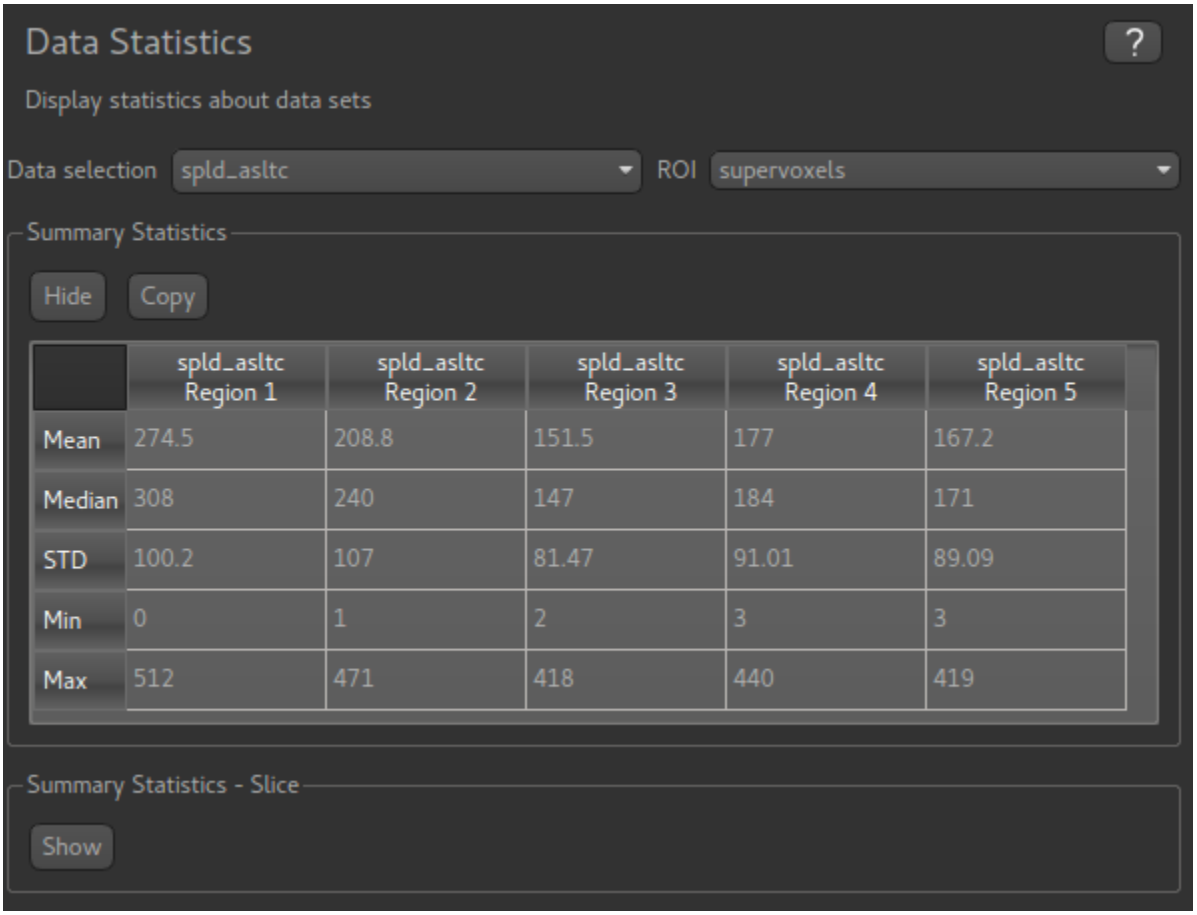
If an ROI is selected then the summary statistics are presented separately for every region within that ROI:

In this example statistics for two data sets are presented within a single-region ROI:

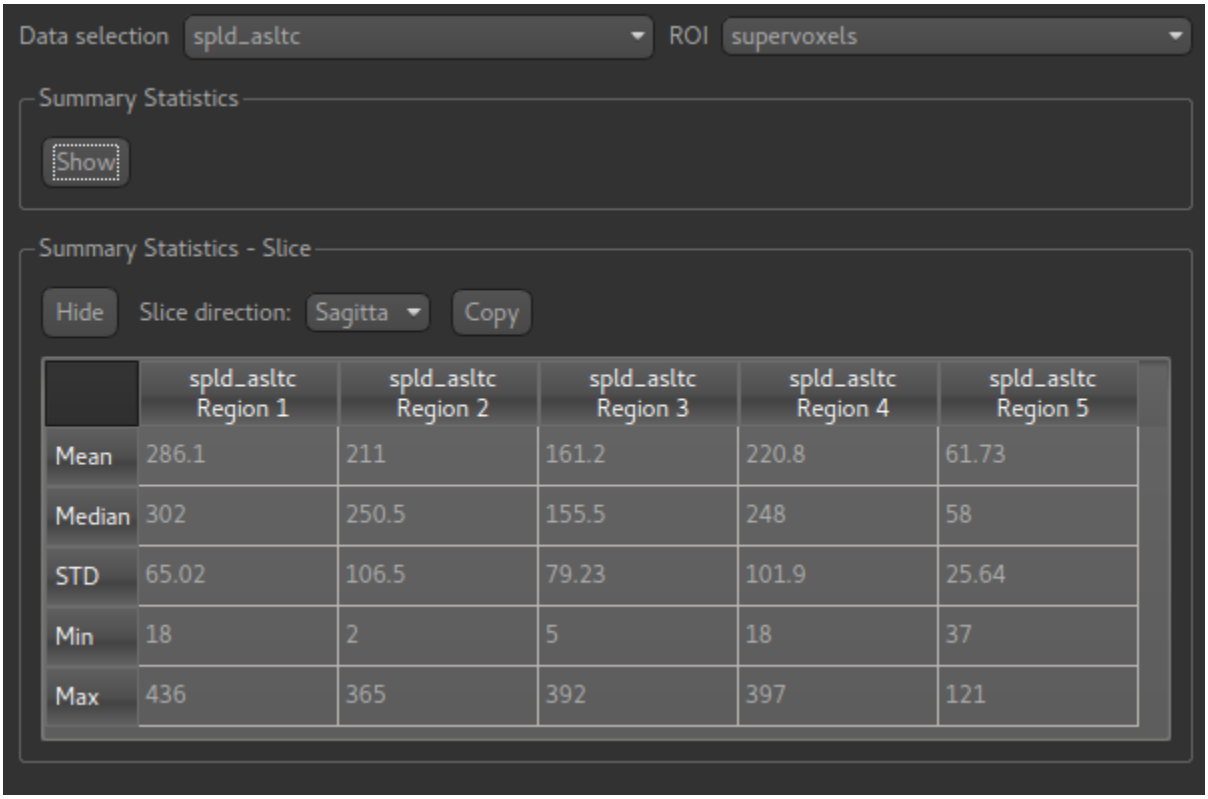


The `Copy` button for each table copies the data to the clipboard in a tab-separated form which should be suitable for pasting into spreadsheets such as Excel.

In this example we display statistics for a single data set in each region of a multi-region ROI (which was generated by the `Supervoxels` widget):



The `Summary Statistics - Slice` table can also be displayed - it presents essentially the same information but over the current slice shown in the viewer (either axial, coronal or sagittal):



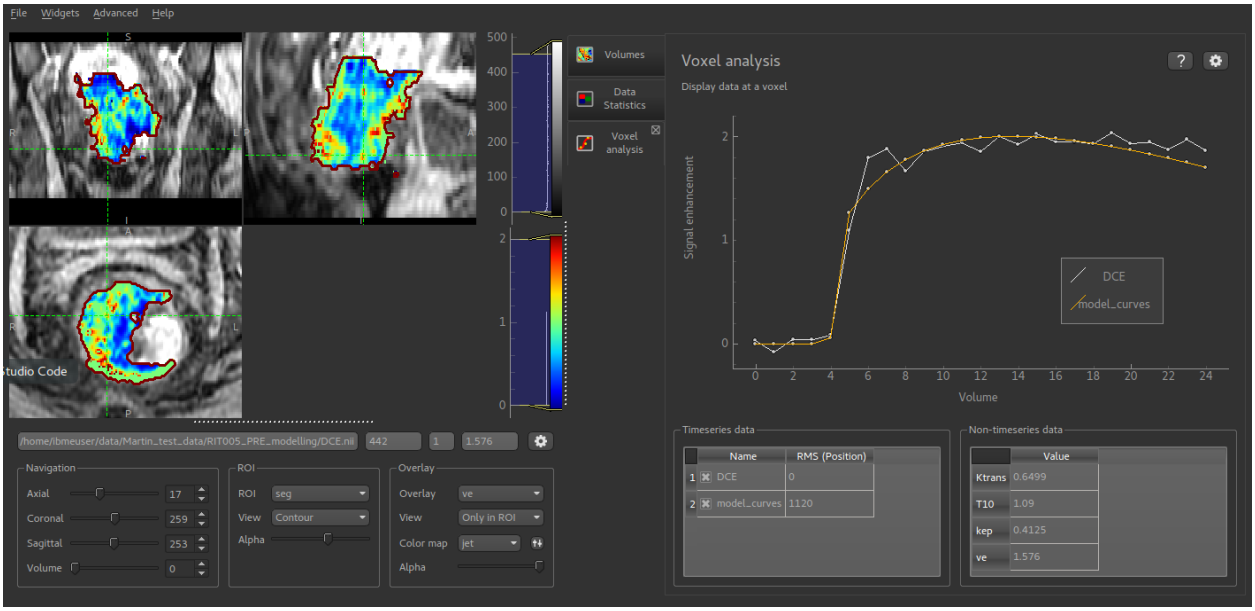
5.1.5 Voxel analysis

This widget shows data at the selected voxel and is visible by default.

The upper part of the widget shows a plot of selected time-series (4D) data. A list of 4D data sets is shown below the plot on the left hand side. Data can be included or removed from the plot by checking/unchecking the data set name in this list.

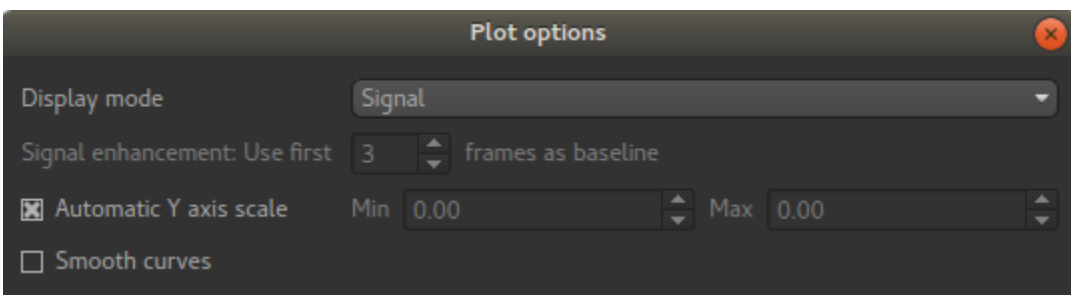
The table on the lower right of the widget shows the value of each 3D data set at the selected point.

Selecting voxels in the viewer window updates the displayed data to the current position.



One use of this widget is comparing the output of a modelling process with the input data. In this screenshot the output of a DCE PK modelling process is overlaid on the original data curve so the degree of fit can be assessed. The parameter outputs from this modelling process are 3D data sets so the value of these parameters (Ktrans, kep, etc) can be viewed in the lower right table.

The options button allows the behaviour of the plot to be changed:



You can choose to plot either the raw data or to transform the timeseries data to signal enhancement curves. This uses the selected number of volumes as 'baseline' and scales the remainder of the data such that the mean value of the baseline volumes is 1. The data is then plotted with 1 subtracted so the baseline has value 0 and a data value of 1 means a signal enhancement of 1, i.e. a doubling of the baseline signal.

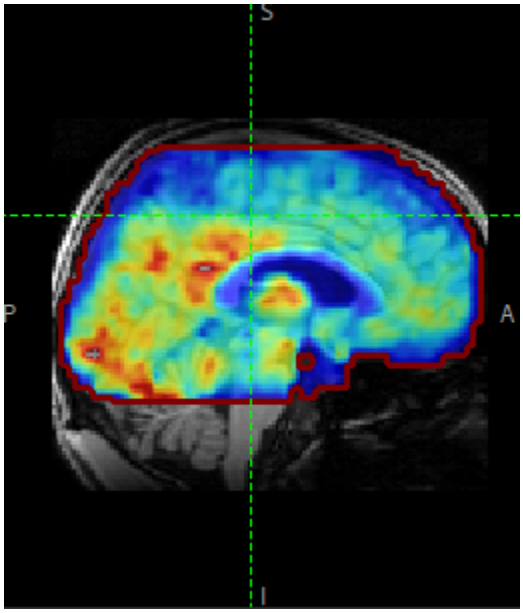
5.2 Arterial Spin Labelling (ASL) MRI

- *Widgets -> ASL -> ASL data processing*

This widget provides a complete pipeline for Arterial Spin Labelling MRI analysis using the Fabber Bayesian model fitting framework. The pipeline is designed for brain ASL MRI scans and some of the options assume this, however with care it could be used for other types of ASL scan.

5.2.1 Tutorials

Arterial Spin Labelling Tutorial



In this practical you will learn how to use the BASIL tools in FSL to analyse ASL data, specifically to obtain quantitative images of perfusion (in units of ml/100 g/min), as well as other haemodynamic parameters.

This tutorial describes the analysis using Quantiphyse - the same analysis can be performed using the command line tool or the FSL GUI. The main advantage of using Quantiphyse is that you can see your input and output data and take advantage of any of the other processing and analysis tools available within the application.

We will mention some of this additional functionality in Quantiphyse as we go, but do not be afraid to experiment with any of the built-in tools while you are following the tutorial.

This practical is based on the FSL course practical session on ASL. The practical is a shorter version of the examples that accompany the Primer: *Introduction to Neuroimaging using Arterial Spin Labelling*. On the website for the primer you can find more examples.

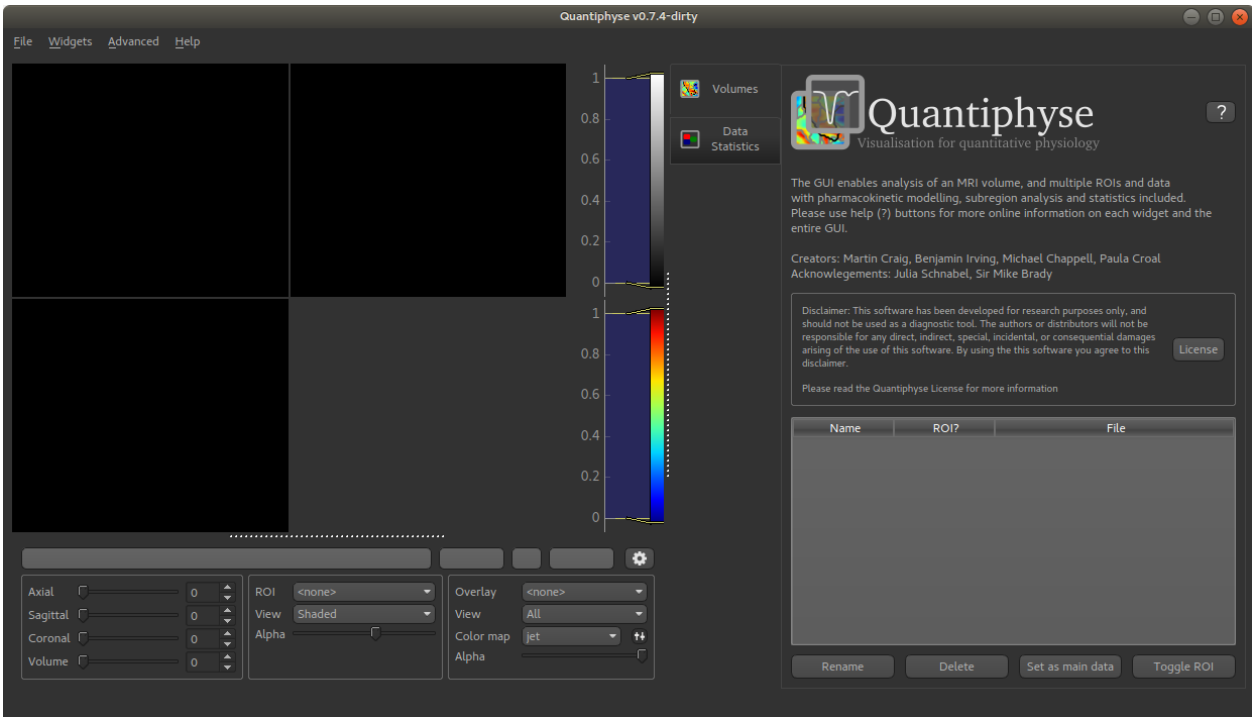
http://www.neuroimagingprimers.org/examples/introduction-primer-example-boxes/

Contents

Basic Orientation

Before we do any data modelling, this is a quick orientation guide to Quantiphyse if you've not used it before. You can skip this section if you already know how the program works.

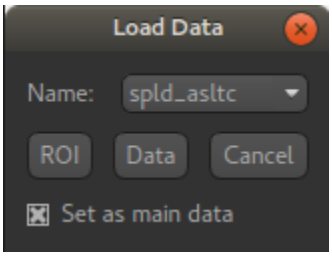
Start the program by typing `quantiphyse` at a command prompt, or clicking on the Quantiphyse icon in the menu or dock.



Loading some data

If you are taking part in an organized practical, the data required will be available in your home directory, in the `fsl_course/ASL` folder. If not, the data can be can be downloaded from the FSL course site: https://fsl.fmrib.ox.ac.uk/fslcourse/ (Scroll down to the section entitled `Data Files` and choose the ASL data from the list of download links).

Start by loading the ASL data into Quantiphyse - use `File->Load Data` or drag and drop to load the file `spld_asltc.nii.gz`. In the `Load Data` dialog select `Data`.



The data should look as follows:

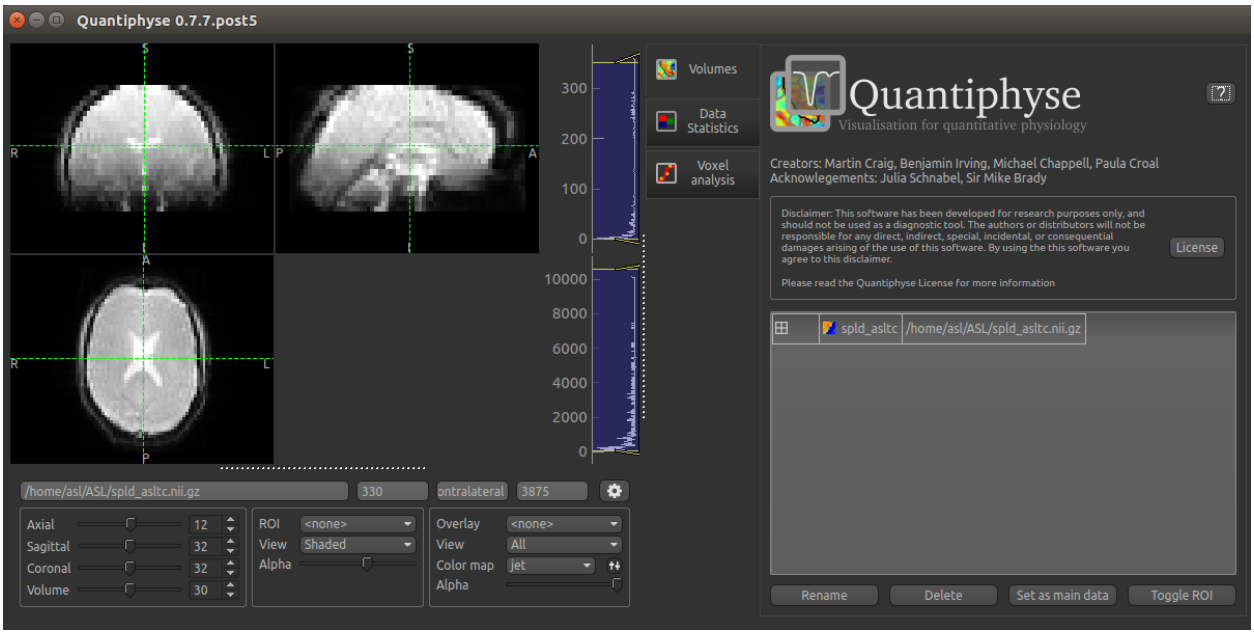


Image view

The left part of the window contains three orthogonal views of your data.

- Left mouse click to select a point of focus using the crosshairs
- Left mouse click and drag to pan the view
- Right mouse click and drag to zoom
- Mouse wheel to move through the slices
- Double click to 'maximise' a view, or to return to the triple view from the maximised view.

View and navigation controls

Just below the viewer these controls allow you to move the point of focus and also change the view parameters for the current ROI and overlay.

Widgets

The right hand side of the window contains 'widgets' - tools for analysing and processing data. Three are visible at startup:

- `Volumes` provides an overview of the data sets you have loaded
- `Data statistics` displays summary statistics for data set
- `Voxel analysis` displays timeseries and overlay data at the point of focus

Select a widget by clicking on its tab, just to the right of the image viewer.

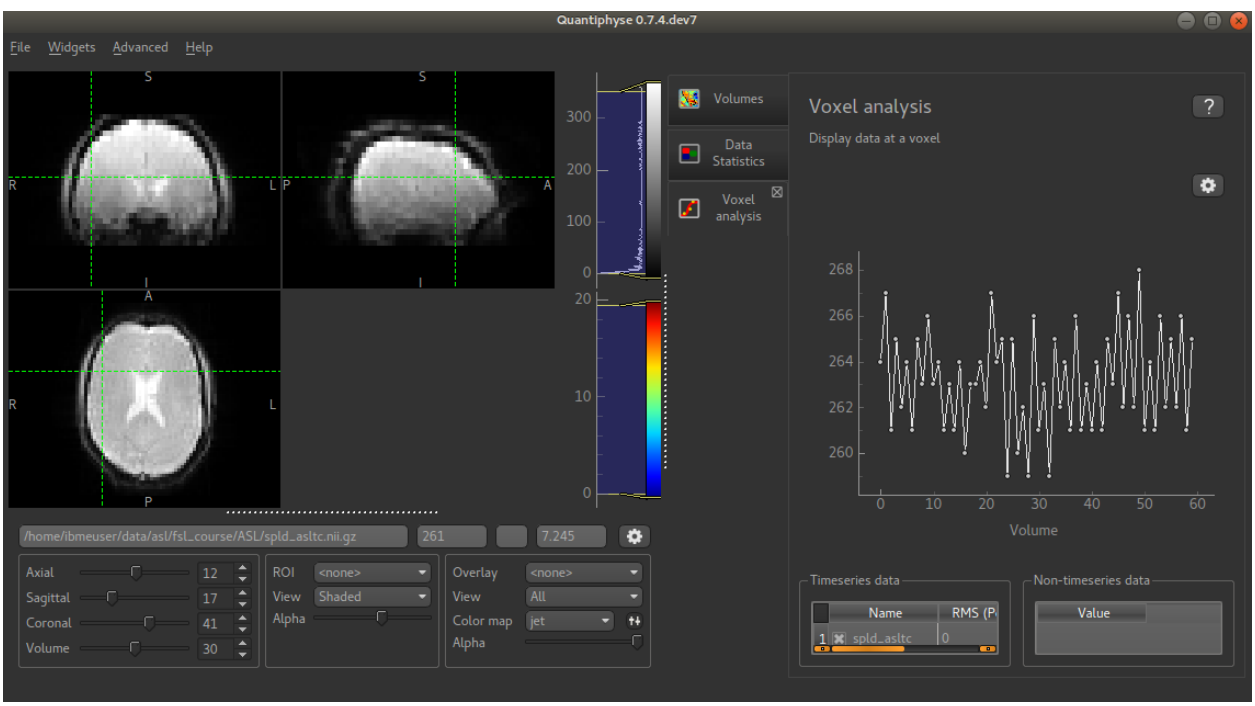
More widgets can be found in the `Widgets` menu at the top of the window. The tutorial will tell you when you need to open a new widget.

For a slightly more detailed introduction, see the *Getting Started* section of the User Guide.

Perfusion quantification using Single PLD pcASL

In this section we will generate a perfusion image using the simplest analysis possible on the simplest ASL data possible.

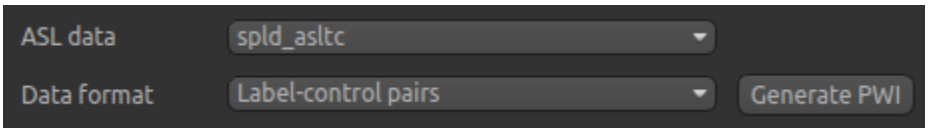
Click on the `Voxel Analysis` widget - it is visible by default to the right of the main image view, then click on part of the cortex. You should see something similar to this:



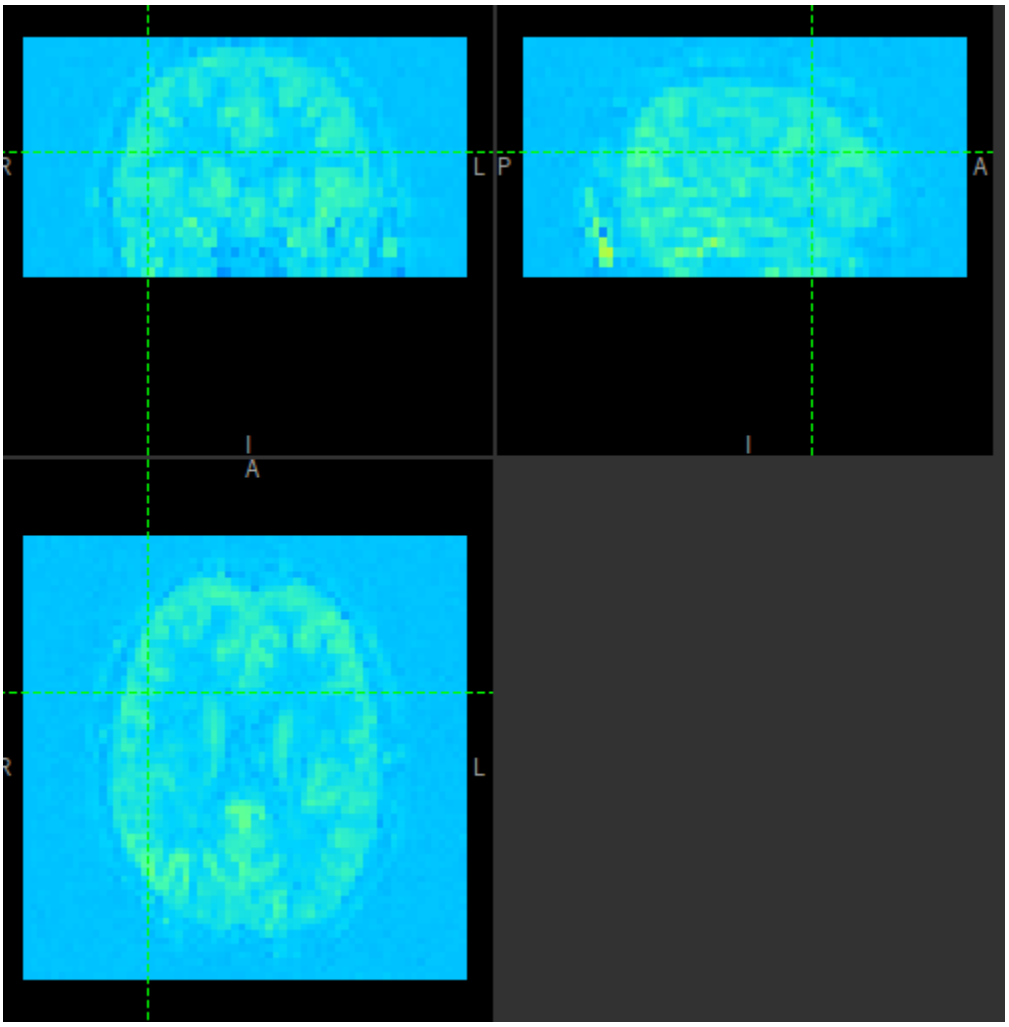
You can see that the data has a zig-zag low-high pattern - this reflects the label-control repeats in the data. Because the data was all obtained at a single PLD the signal is otherwise fairly constant.

A perfusion weighted image

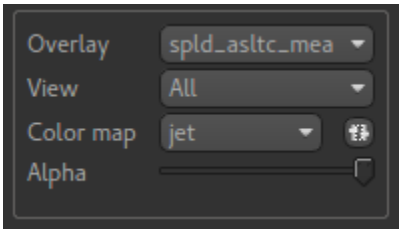
Open the `Widgets->ASL->ASL Data Processing` widget. We do not need to set all the details of the data set yet, however note that the data format is (correctly) set as `Label-control pairs`.



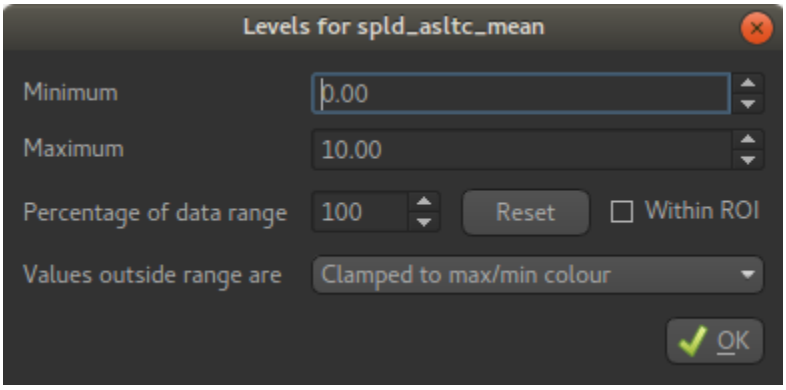
Click on the `Generate PWI` button. This performs label-control subtraction and averages the result over all repeats. The result is displayed as a colour overlay, which should look like a perfusion image:



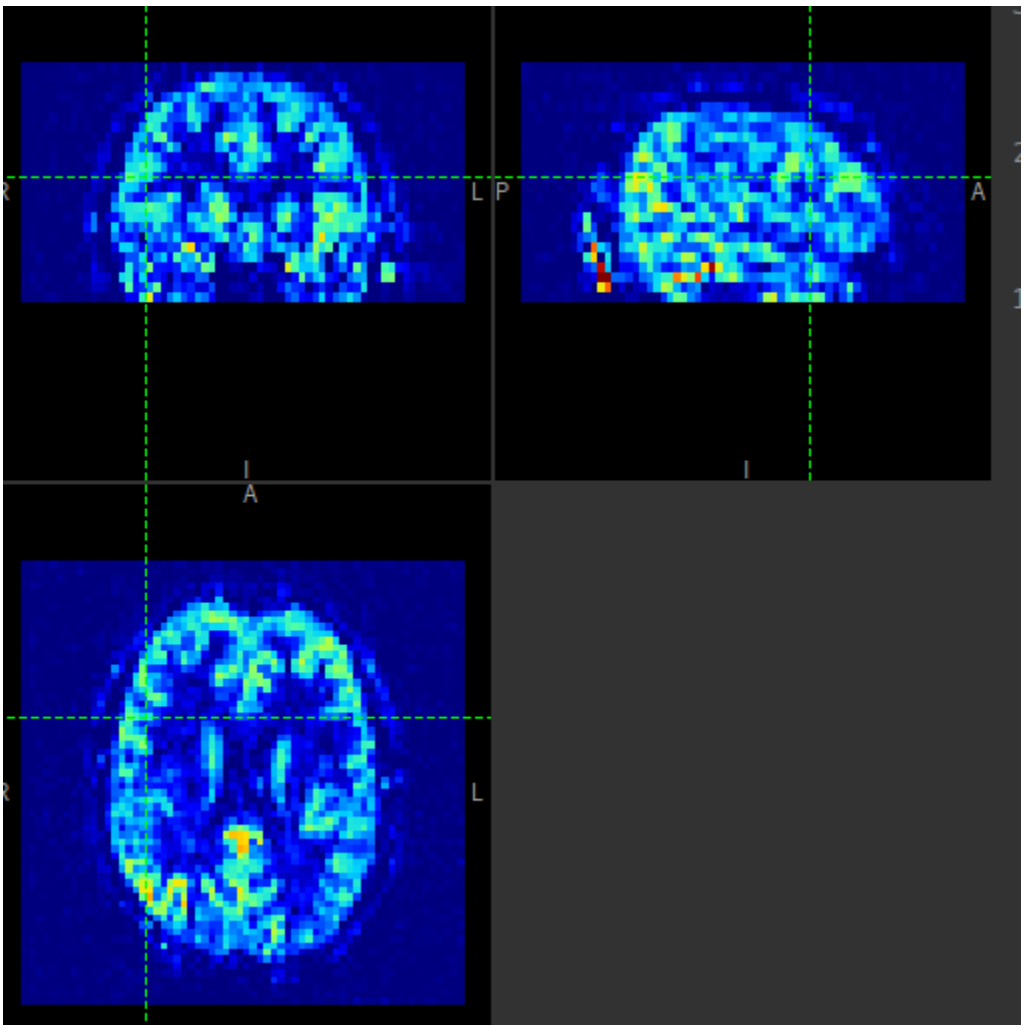
We can improve the display a little by adjusting the colour map. Find the overlay view options below the main image view:



Next to the `Color Map` option (which you can change if you like!) there is a levels button  which lets you change the min and max values of the colour map. Set the range from `0` to `10` and select `Values outside range` to `Clamped`.



Then click `Ok`. The perfusion weighted image should now be clearer:



You could also have modified the colour map limits by dragging the colourmap range widget directly - this is located to the right of the image view. You can drag the upper and lower limits with the left button, while dragging with the right button changes the displayed scale. You can also customize the colour map by clicking on the colour bar with

the right button.

> **Warning:** Dragging the colourmap is a little fiddly due to a GUI bug. Before trying to adjust the levels, drag down with the **right** mouse button briefly on the colour bar. This unlocks the automatic Y-axis and will make it easier to drag on the handles to adjust the colour map.

Model based analysis

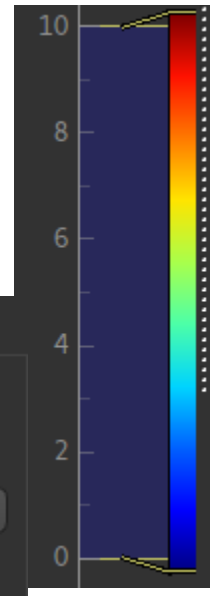
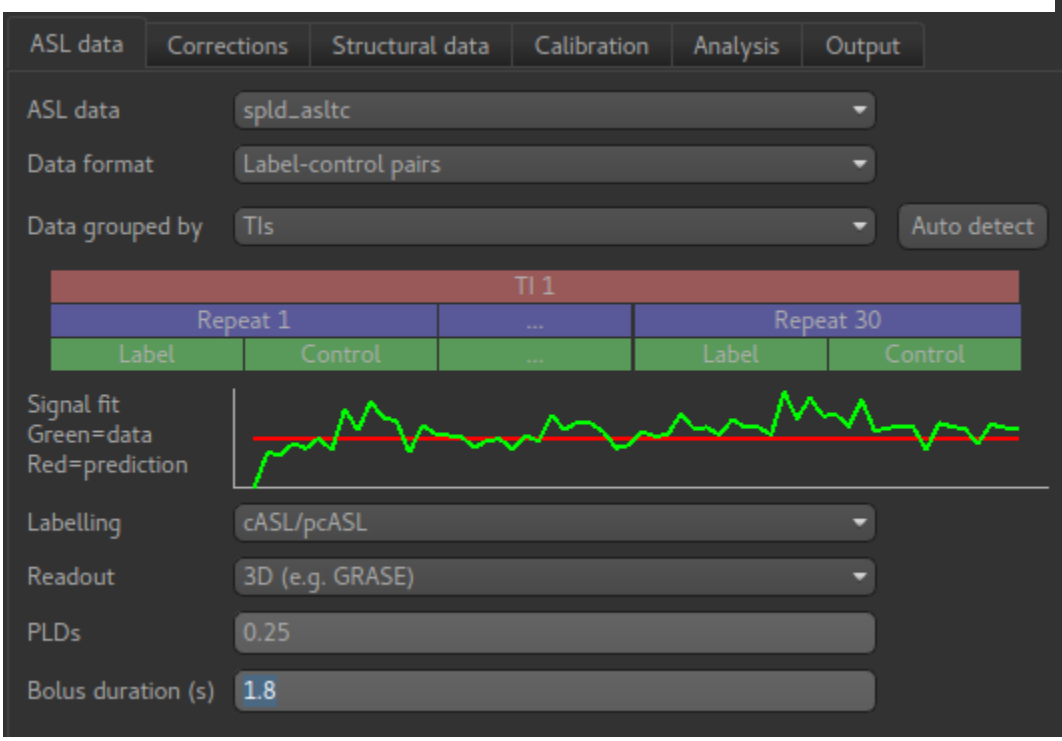


Fig. 1: Colour map widget

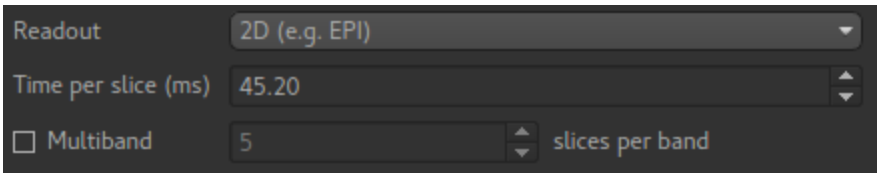
This dataset used pcASL labeling and we are going to start with an analysis which follows as closely as possible the recommendations of the ASL Consensus Paper[1] (commonly called the 'White Paper') on a good general purpose ASL acquisition, although we have chosen to use a 2D multi-slice readout rather than a full-volume 3D readout.

Looking at the `ASL data processing` widget we used to generate the PWI, you can see that this is a multi-page widget in which each tab describes a different aspect of the analysis pipeline. We start by reviewing the information on the first page which describes our ASL data acquisition:

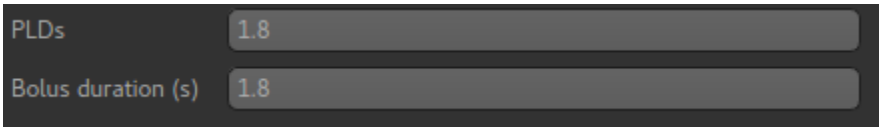


Most of this is already correct - we have label-control pairs and the data grouping does not matter for single PLD data (we will describe this part of the widget later in the multi-PLD analysis). The labelling method is correctly set as `cASL/pcASL`. However we have a 2D readout with 45.2ms between slices, so we need to change the `Readout` option to reflect this. When we select a 2D readout, the option to enter the slice time appears automatically.

[1] Alsop, D. C., Detre, J. A., Golay, X. , Günther, M. , Hendrikse, J. , Hernandez-Garcia, L. , Lu, H. , MacIntosh, B. J., Parkes, L. M., Smits, M. , Osch, M. J., Wang, D. J., Wong, E. C. and Zaharchuk, G. (2015), Recommended implementation of arterial spin-labeled perfusion MRI for clinical applications: A consensus of the ISMRM perfusion study group and the European consortium for ASL in dementia. Magn. Reson. Med., 73: 102-116. doi:10.1002/mrm.25197



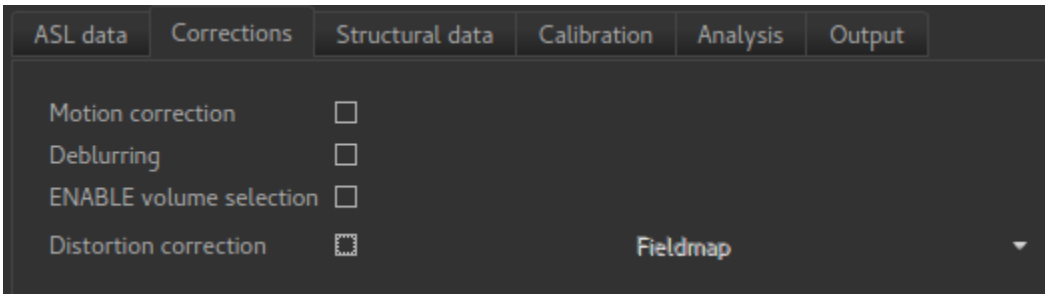
The bolus duration of 1.8s is correct, however we have used a post-labelling delay of 1.8s in this data, so enter `1.8` in the `PLDs` entry box.



(Simple) Perfusion Quantification

In this section we invert the kinetics of the ASL label delivery to fit a perfusion image, and use the calibration image to get perfusion values in the units of ml/100g/min.

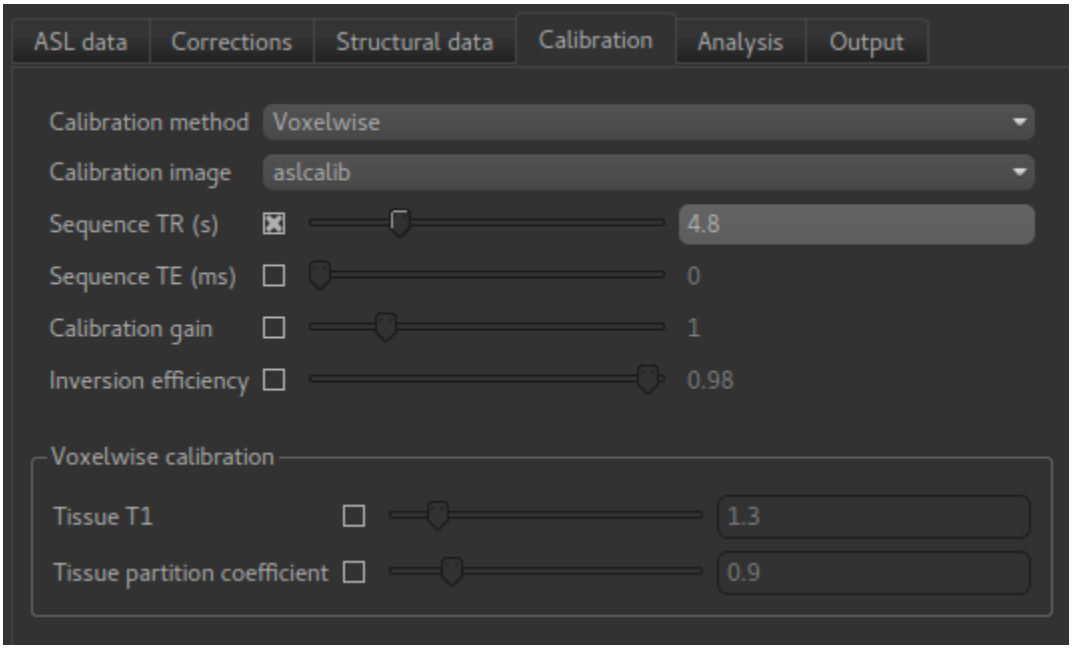
Firstly, on the `Corrections` tab, we will uncheck `Motion Correction` which is enabled by default:



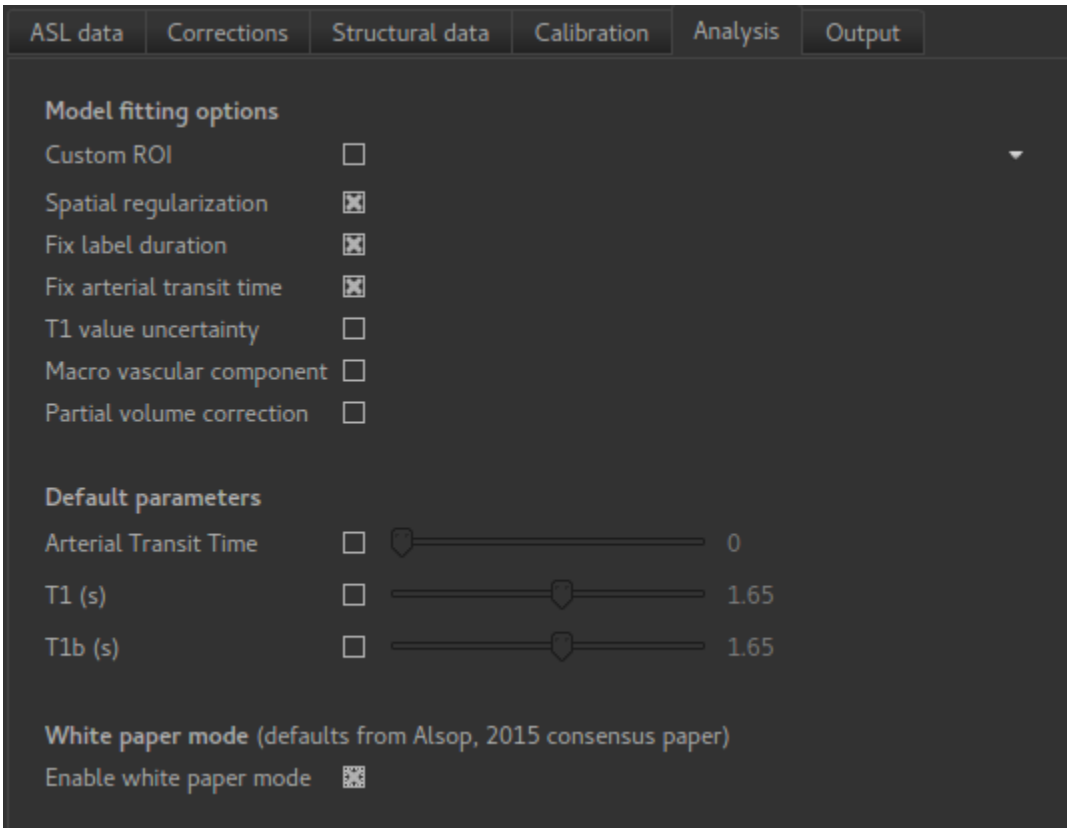
For this run we will skip the `Structural data` tab, and instead move on to `Calibration`. To use calibration we first need to load the calibration image data file from the same folder containing the ASL data - again we can use drag/drop or the `File->Load Data` menu option to load the following file:

- `aslcalib.nii.gz` - Calibration (M0) image

On the `Calibration` tab we set the calibration method as `Voxelwise` which is recommended in the white paper. We also need to select the calibration image we have just loaded: `aslcalib`. The TR for this image was 4.8s, so click on the `Sequence TR` checkbox and set the value to 4.8. Other values can remain at their defaults.



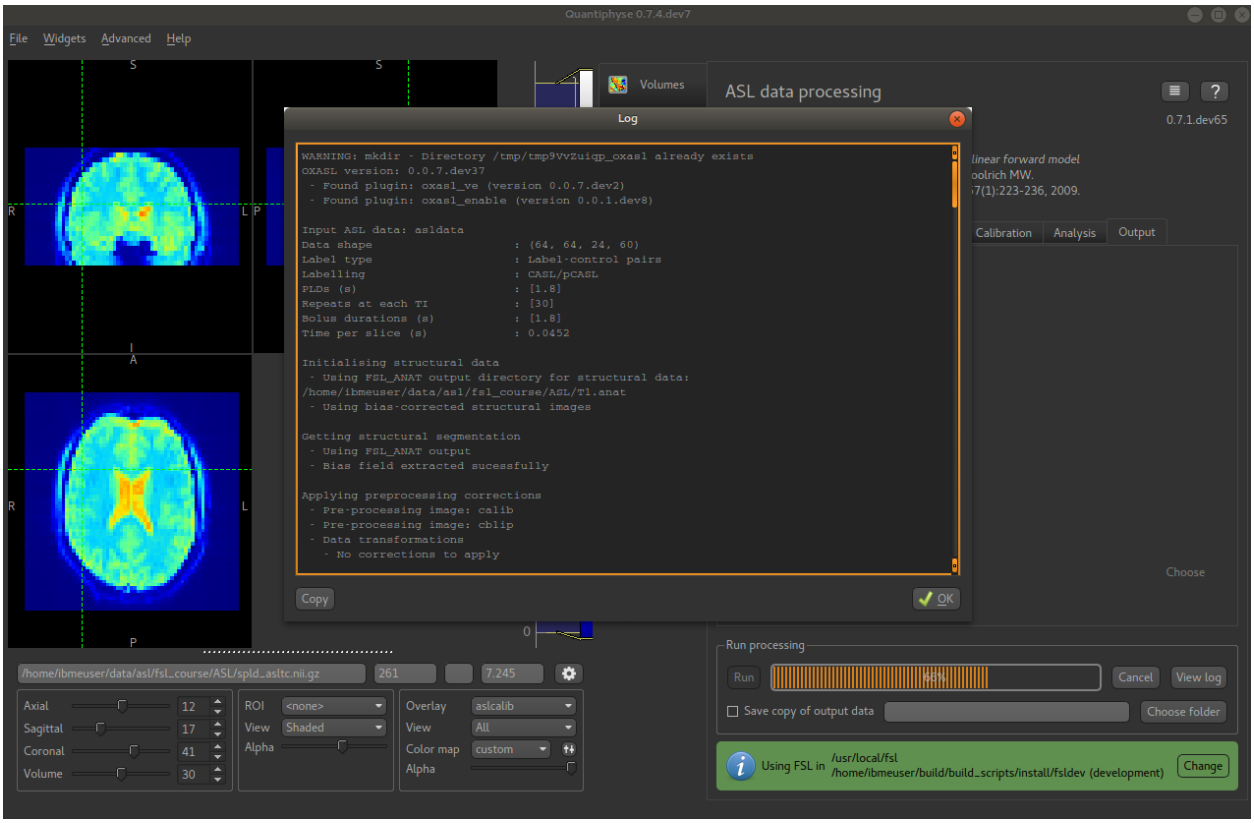
On the `Analysis` we select `Enable white paper mode` at the bottom which sets some default values to those recommended in the White paper.



We will not change the defaults on the `Output` tab yet, but feel free to view the options available.

We are now set up to run the analysis - but before you do, check the green box at the bottom of the widget which reports where it thinks FSL is to be found. If the information does not seem to be correct, click the `Change` button and select the correct location of your FSL installation.

Finally click `Run` at the bottom to run the analysis. You can click the `View Log` button to view the progress of the analysis which should only take a few minutes.

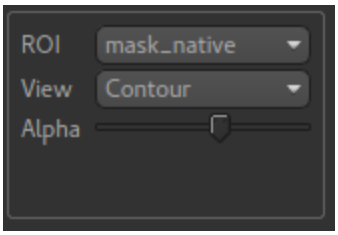


Note: While you are waiting you can read ahead and even start changing the options in the GUI ready for the next analysis that we want to run.

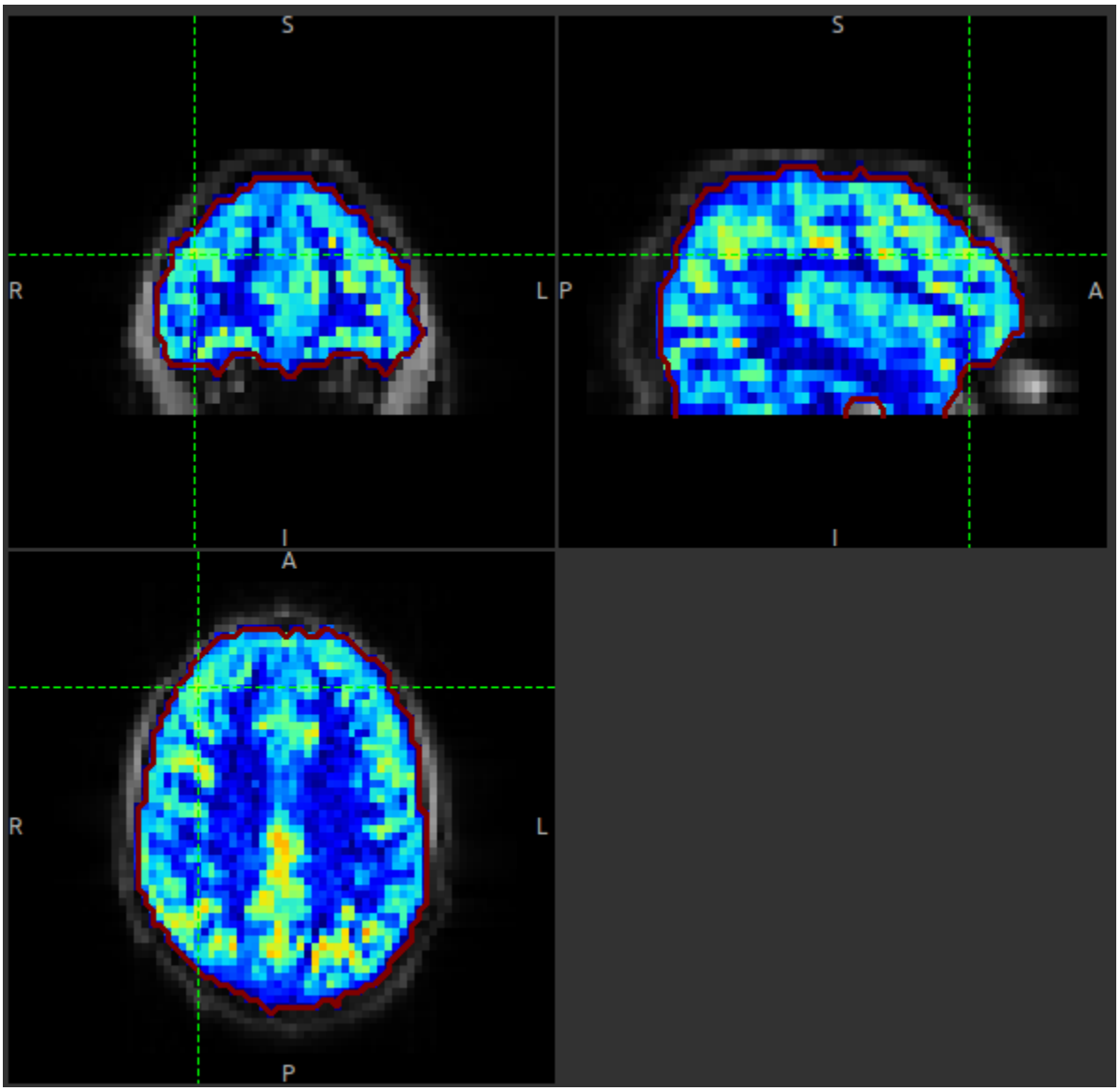
Once the analysis had completed, some new data items will be available. You can display them either by selecting them from the `Overlay` menu below the image display, or by clicking on the `Volumes` widget and selecting them from the list. The new data items are:

- `perfusion_native` - Raw (uncalibrated) perfusion map
- `perfusion_calib_native` - Calibrated perfusion data in ml/100g/min
- `mask_native` - An ROI (which appears in the ROI selector under the image view) which represents the region in which the analysis was performed.

The images may be clearer if we modify the view style for the ROI from `Shaded` to `Contour` (in the ROI options box underneath the image view). This replaces the translucent red mask with an outline:



The `perfusion_calib_native` image should look similar to the perfusion weighted image we created initially, however the data range reflects the fact that it is in physical units in which average GM perfusion is usually in the 30-50 range. To get a clear visualisation set the color map range to 0-150 using the Levels button  as before. You can also select `Only in ROI` as the `View` option just above this so we only see the perfusion map within the selected ROI. The result should look something like this:



Improving the Perfusion Images from single PLD pcASL

The purpose of this practical is essentially to do a better job of the analysis we did above, exploring more of the features of the GUI including things like motion and distortion correction.

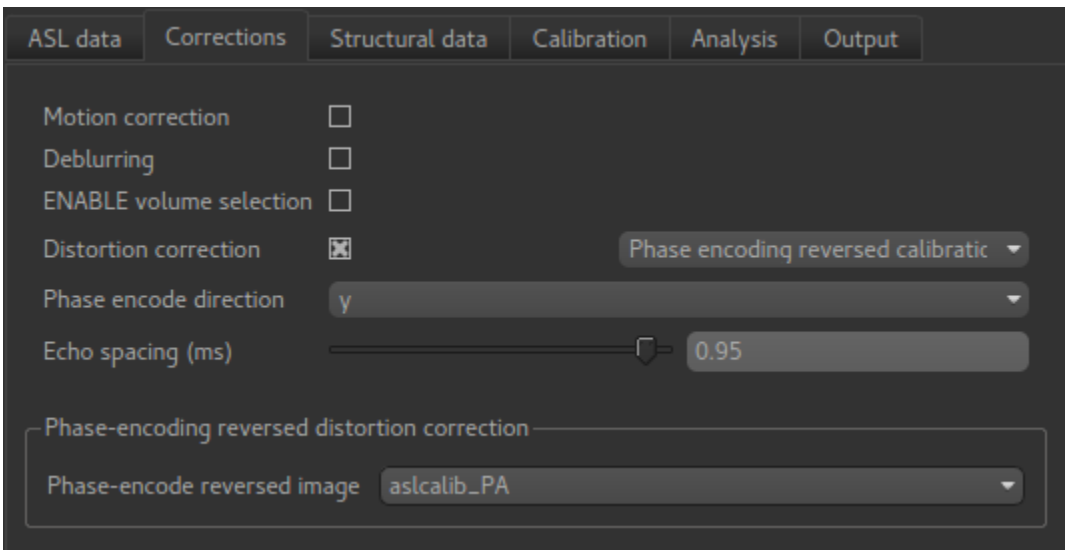
Motion and Distortion correction

First we need to load an additional data file:

- `aslcalib_PA.nii.gz` - this is a 'blipped' calibration image - identical to `aslcalib` apart from the use of posterior-anterior phase encoding (anterior-posterior was used in the rest of the ASL data). This is provided for distortion correction.

Go back to the GUI which should still be setup from the last analysis you did.

On the `Corrections` tab, we will check `Motion Correction` to enable it, and and click on the `Distortion Correction` checkbox to show distortion correction options. We select the distortion correction method as `Phase-encoding reversed calibration`, select `y` as the phase encoding direction, and `0.95` as the echo spacing in ms (also known as the dwell time). Finally we need to select the phase-encode reversed image as `aslcalib_PA` which we have just loaded:



On the `Analysis` tab, make sure you have `Spatial regularization` selected (it is by default). This will reduce the appearance of noise in the final perfusion image using the minimum amount of smoothing appropriate for the data.

In order to compare with the previous analysis we might want the output to have a different name. To do this, on the `Output` tab, select the `Prefix for output data names` checkbox and provide a short prefix in the text box, e.g. `new_`.

Note: As an alternative to using a prefix, you can also rename data items from the `Volumes` widget which is visible by default. Click on a data set name in the list and click `Rename` to give it a new name.

Now click `Run` again.

For this analysis we are still in 'White Paper' mode. Specifically this means we are using the simplest kinetic model, which assumes that all delivered blood-water has the same T1 as that of

the blood and that the Arterial Transit Time should be treated as 0 seconds.

As before, the analysis should only take a few minutes, slightly longer this time due to the distortion and motion correction. Like the last exercise you might want to skip ahead and start setting up the next analysis.

The output will not be very different, but if you switch between the old and new versions of the `perfusion_calib_native` data set you should be able to see slight stretching in the anterior portion of the brain which is the outcome of distortion correction.

To do this select the `Volumes` widget and in the data list click on the left hand box next to the data item you want to see. An 'eye' icon will appear here  indicating that this data set is now visible. Switch between `new_perfusion_calib_native` and `perfusion_calib_native` to see the different - it helps if you set the colour map range the same for both data sets.

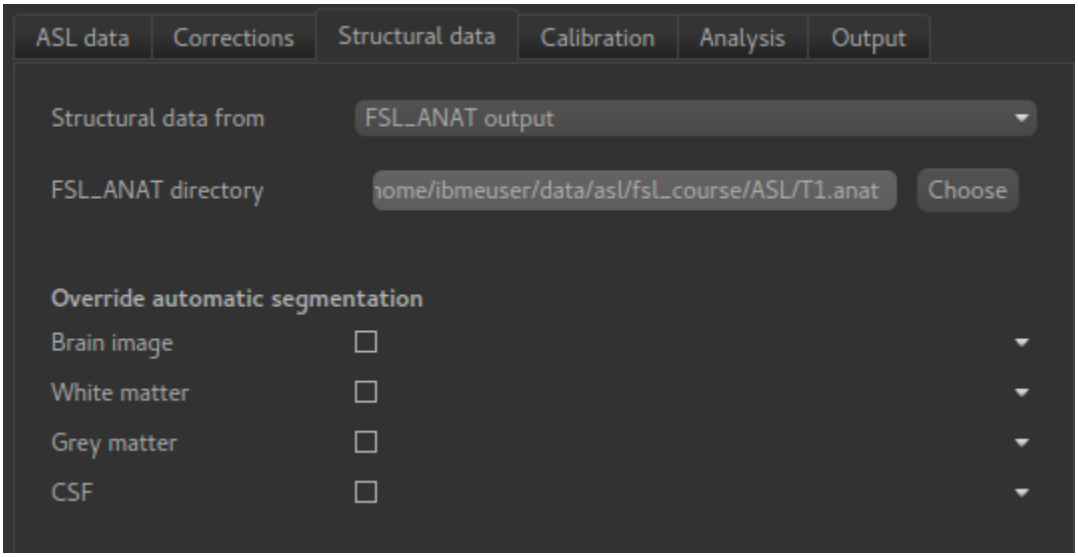
This data does not have a lot of motion in it so the motion correction is difficult to identify.

Making use of Structural Images

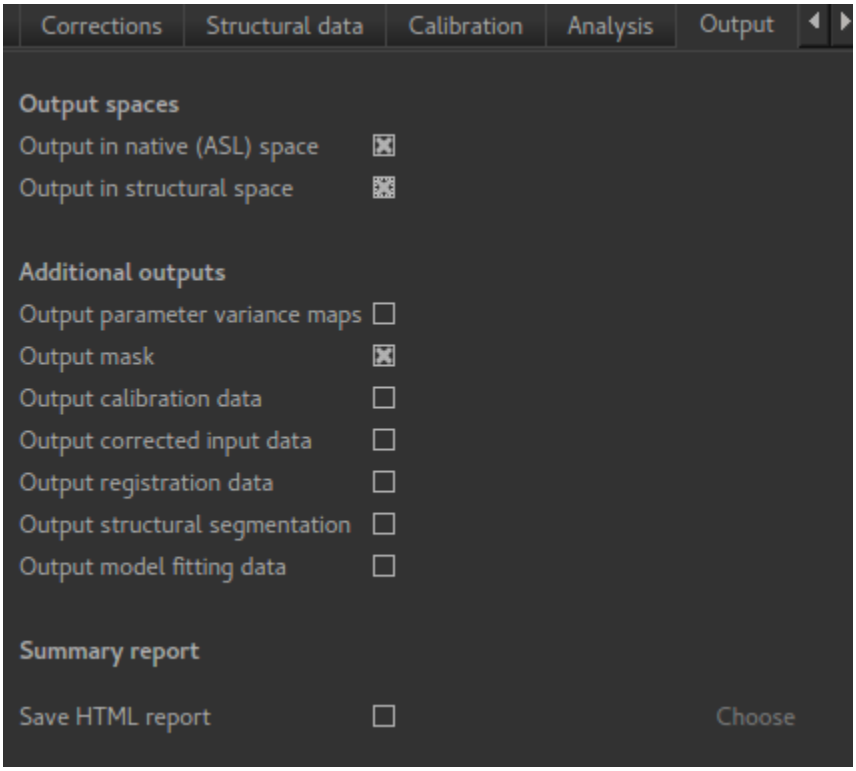
Thus far, all of the analyses have relied purely on the ASL data alone. However, often you will have a (higher resolution) structural image in the same subject and would like to use this as well, at the very least as part of the process to transform the perfusion images into some template space. We can provide this information on the `Structural Data` tab.

You can either load a structural (T1 weighted) image into Quantiphyse and select `Structural Image` as the source of structural data, or if you have already processed your structural data with `FSL_ANAT` you can point the analysis at the output directory. We will use the second method as it enables the analysis to run faster. On the `Structural Data` tab, we select `FSL_ANAT` output and chooses the location of the FSL_ANAT output directory (`T1.anat`):

Note: If a simple structural image was provided instead of an `FSL_ANAT` output folder, the `FAST` segmentation tool is automatically run to obtain partial volume estimates. This adds considerably to the run-time so it's generally recommended to run `FSL_ANAT` separately first.



If we want to output our data in structural space (so it can be easily overlaid onto the structural image), click on the `Output` tab and check the option `Output in structural space`:



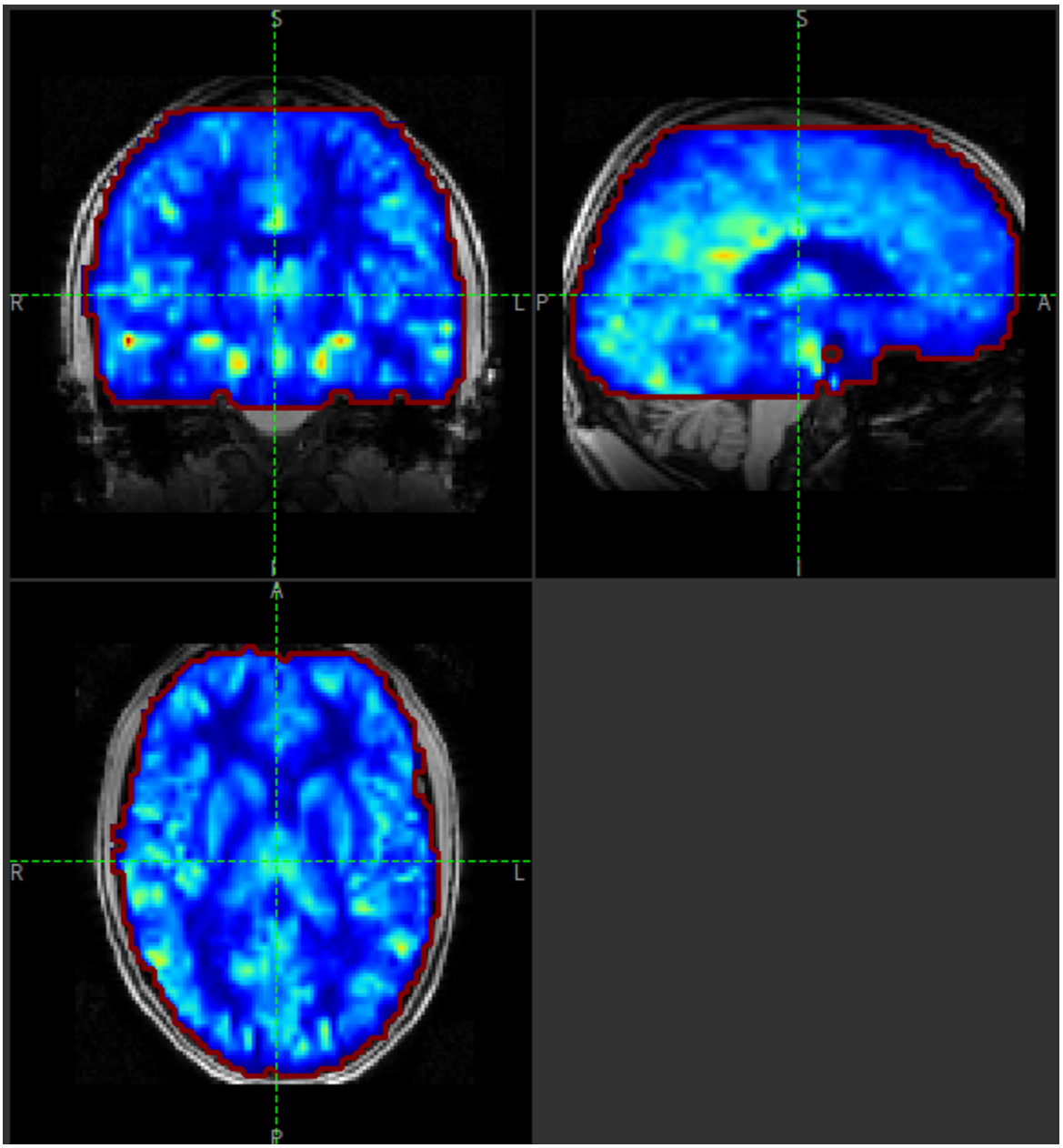
This analysis will take somewhat longer overall (potentially 15-20 mins), the extra time is taken up doing careful registration between ASL and structural images. Thus, this is a good point to keep reading on and leave the analysis running.

You will find some new data sets in the overlay list, in particular:

- `perfusion_calib_struc` - Calibrated perfusion in structural space

  This is the calibrated perfusion image in high-resolution structural space. It is nice to view it in conjunction with the structural image itself. To do this, load the `T1.anat/T1.nii.gz` data file

and select `Set as main data` when loading it. Then select `perfusion_calib_struc` from the `Overlay` menu and select `View` as `Only in ROI`:



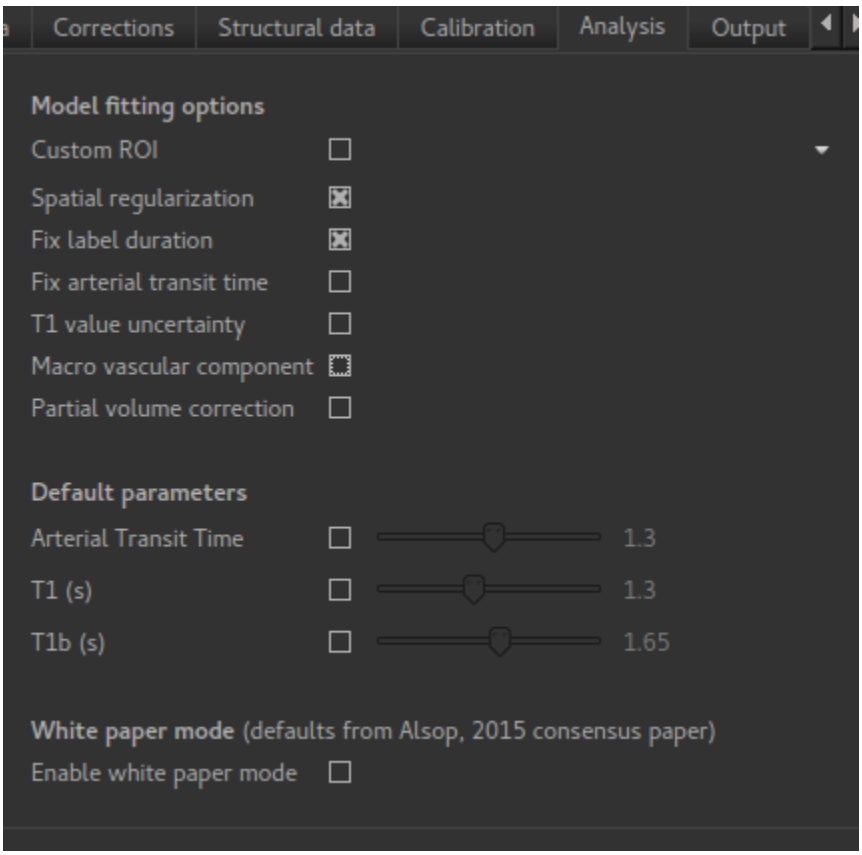
You can move the `Alpha` slider under the overlay selector to make the perfusion map more or less transparent and verify that the perfusion map lines up with the structural data.

Different model and calibration choices

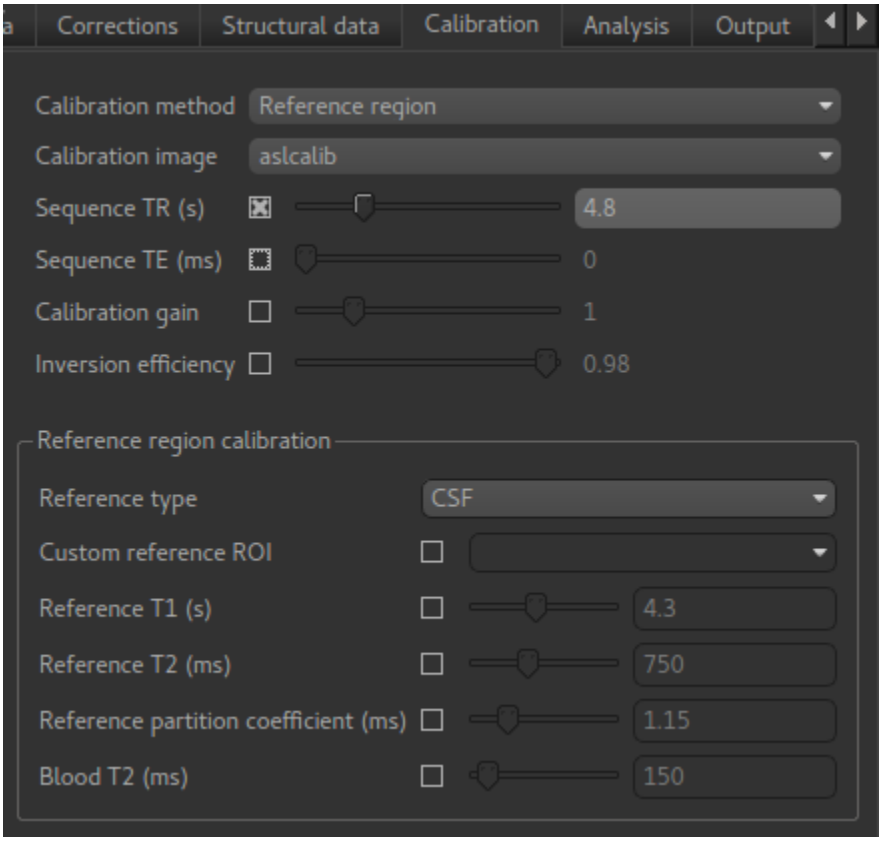
So far to get perfusion in units of ml/100g/min we have used a voxelwise division of the relative perfusion image by the (suitably corrected) calibration image - so called 'voxelwise' calibration. This is in keeping with the recommendations of the ASL White Paper for a simple to implement quantitative analysis. However, we could also choose to use a reference tissue to derive a single value for the equilibrium magnetization of arterial blood and use that in the calibration process

instead - the so-called 'reference region' method.

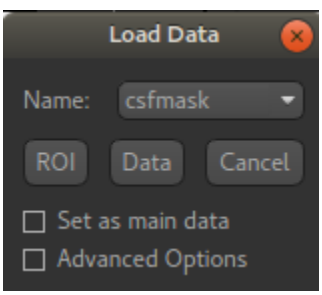
Go back to the analysis you have already set up. We are now going to turn off 'White Paper' mode, this will provide us with more options to get a potentially more accurate analysis. To do this return to the 'Analysis' tab and deselect the 'White Paper' option. You will see that the 'Arterial Transit Time' goes from 0 seconds to 1.3 seconds (the default value for pcASL in BASIL based on our experience with pcASL labeling plane placement) and the 'T1' value (for tissue) is different to 'T1b' (for arterial blood), since the Standard (aka Buxton) model for ASL kinetics considers labeled blood both in the vasculature and the tissue.



Now that we are not in 'White Paper' mode we can also change the calibration method. On the `Calibration` tab, change the `Calibration method` to `Reference Region`.

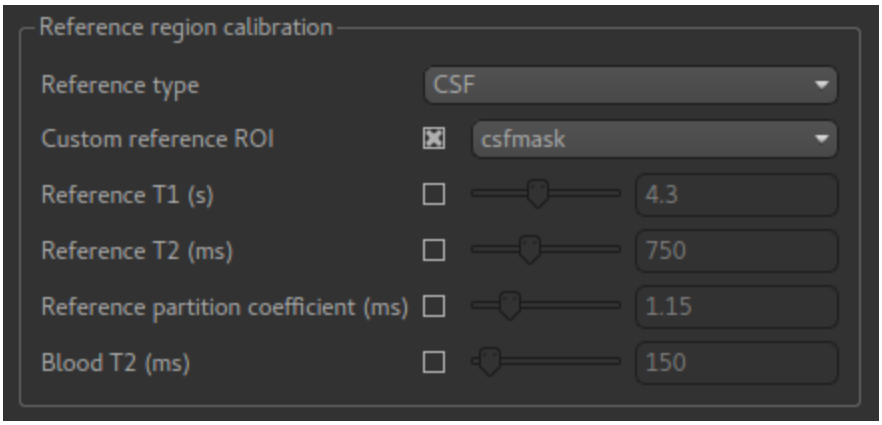


The default values will automatically identify CSF in the brain ventricles and use it to derive a single calibration M0 value with which to scale the perfusion data. However this is quite time consuming, so we will save ourselves the bother and provide a ready-made mask which identifies pure CSF voxels. To do this, first load the dataset `csfmask.nii.gz` and be sure to identify it as an ROI (*not* Data).



Note: If you incorrectly load an ROI as a data set you can switch it to an ROI on the `Volumes` widget which is visible by default. Select the data from the list and click `Toggle ROI`.

Then select `Custom reference ROI` and choose `csfmask` from the list:



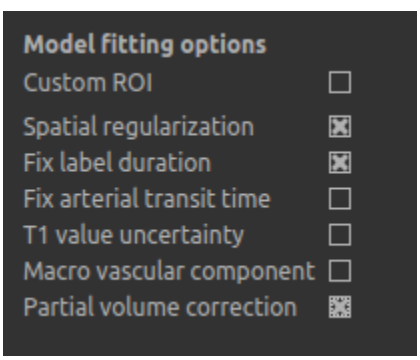
As before you may want to add an output name prefix so you can compare the results. Then click `Run` once more.

The resulting perfusion images should look very similar to those produced using the voxelwise calibration, and the absolute values should be similar too. For this, and many datasets, the two methods are broadly equivalent.

Partial Volume Correction

Having dealt with structural image, and in the process obtained partial volume estimates, we are now in a position to do partial volume correction. This does more than simply attempt to estimate the mean perfusion within the grey matter, but attempts to derive and image of gray matter perfusion directly (along with a separate image for white matter).

This is very simple to do. First ensure that you have provided structural data (i.e. the FSL_ANAT output) on the `Structure` tab. The partial volume estimates produced by fsl_anat (in fact they are done using fast) are needed for the correction. On the `Analysis` tab, select `Partial Volume Correction`.



To run the analysis you would simply click `Run` again, however this will take **a lot longer to run**. If you'd prefer not to wait, you can find the results of this analysis already completed in the directory `ASL/oxasl_spld_pvout`.

In this results directory you will still find an analysis performed without partial volume correction in `native_space` as before. The results of partial volume correction can be found in `native_space/pvcorr`. In this directory the output perfusion data `perfusion_calib.nii.gz` is now an estimate of perfusion **only in gray matter**. It has been joined by a new set of images for the estimation of white matter perfusion, e.g., `perfusion_wm_calib.nii.gz`.

It may be more helpful to look at `perfusion_calib_masked.nii.gz` (and the equivalent

`perfusion_wm_calib_masked.nii.gz`) since this has been masked to include only voxels with more than 10% gray matter (or white matter), i.e., voxels in which it is reasonable to interpret the gray matter (white matter) perfusion values - shown below.

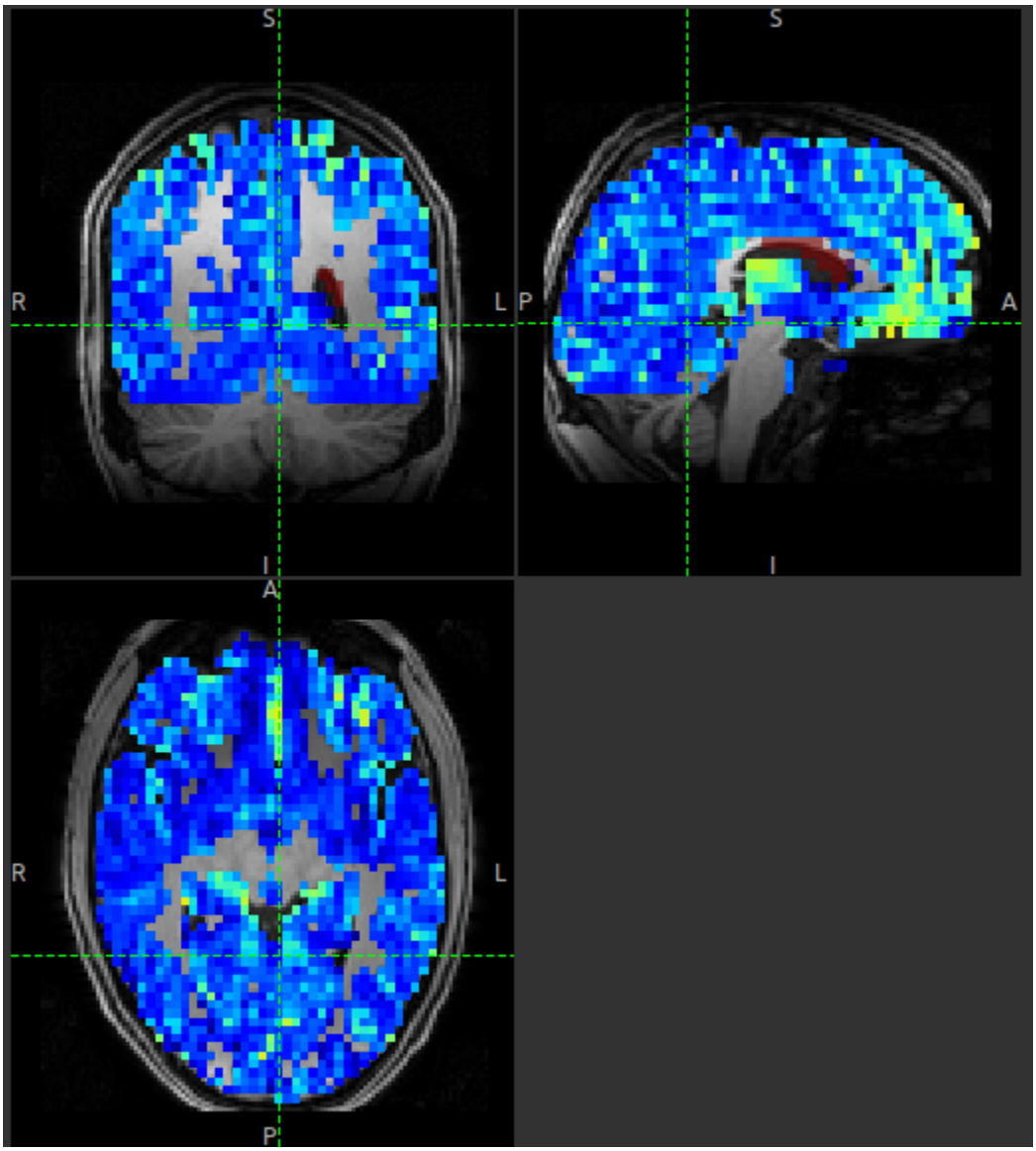


Fig. 2: GM perfusion (masked to include only voxels with >= 10% GM)

Perfusion Quantification (and more) using Multi-PLD pcASL

The purpose of this exercise is to look at some multi-PLD pcASL. As with the single PLD data we can obtain perfusion images, but now we can account for any differences in the arrival of labeled blood-water (the arterial transit time, ATT) in different parts of the brain. As we will also see we can extract other interesting parameters, such as the ATT in its own right, as well as arterial blood volumes.

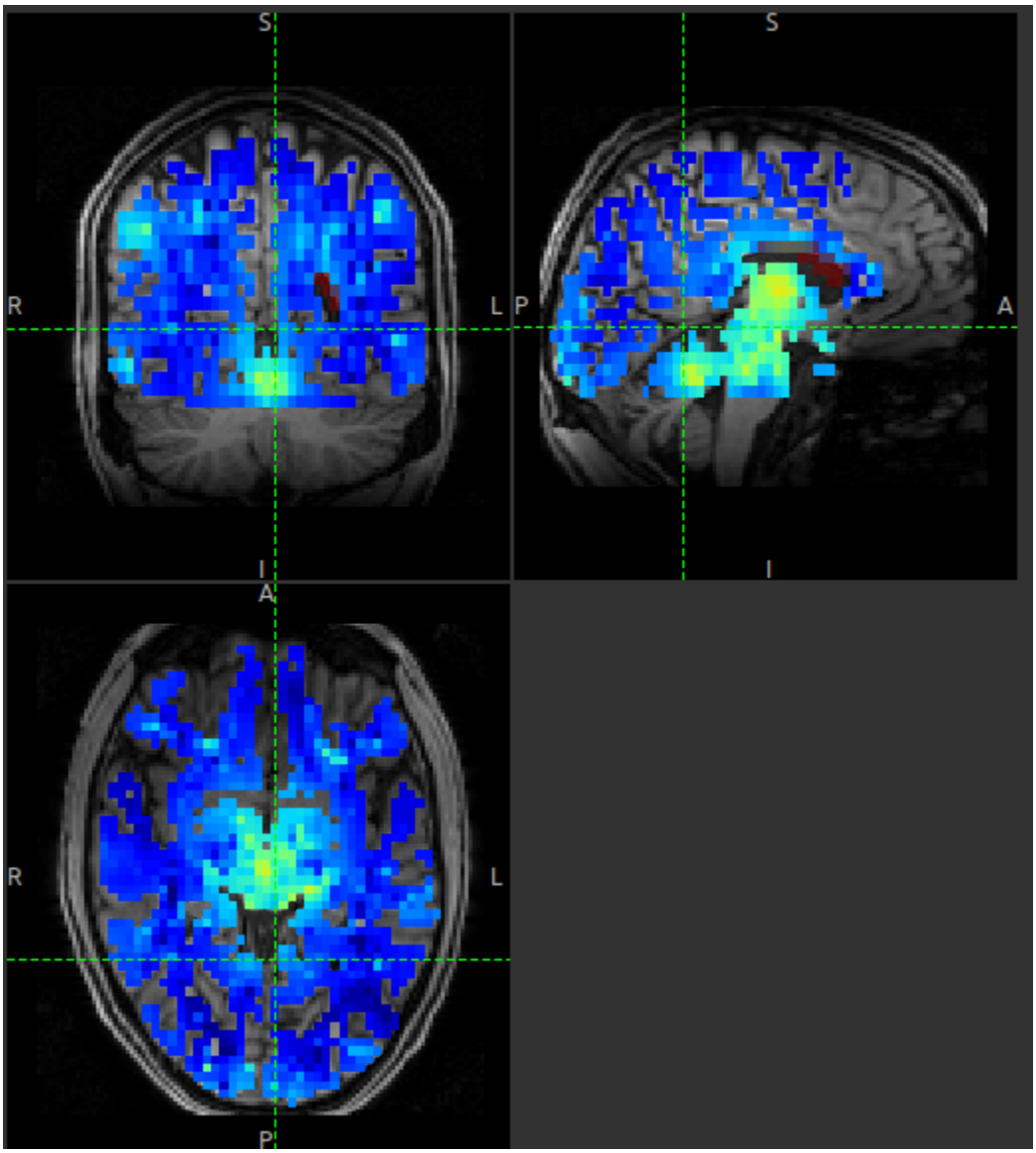


Fig. 3: WM perfusion (masked to include only voxels with >= 10% WM)

The data

Note: If you have accumulated a lot of data sets you might want to choose `File->Clear all data` from the menu and start from scratch again. Note that you will need to re-load the calibration and other input data. You can also delete data sets from the `Volumes` widget.

The data we will use in this section supplements the single PLD pcASL data above, adding multi-PLD ASL in the same subject (collected in the same session). This dataset used the same pcASL labelling, but with a label duration of 1.4 seconds and 6 post-labelling delays of 0.25, 0.5, 0.75, 1.0, 1.25 and 1.5 seconds.

The ASL data file you will need to load is:

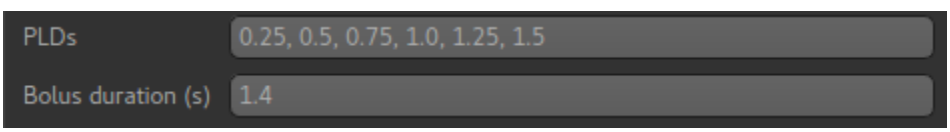
- `mpld_asltc.nii.gz`

  The label-control ASL series containing 96 volumes. Each PLD was repeated 8 times, thus there are 16 volumes (label and control paired) for each PLD. The data has been re-ordered from the way it was acquired, such that all of the measurements from each PLD have been grouped together - it is important to know this data ordering when doing the analysis.

Perfusion Quantification

Going back to the ASL data processing widget, we first go back to the *Asl Data* tab page and select our new ASL data from the choice at the top:

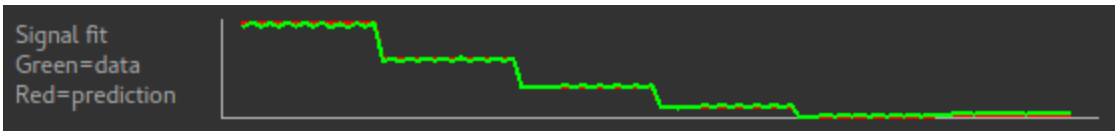
We need to enter the 6 PLDs in the `PLDs` entry box - these can be separated by spaces or commas. We also change the label duration to 1.4s:



As we noted earlier, in this data all of the measurements at the same PLD are grouped together. This is indicated by the `Data grouped by` option which defaults (correctly in this case) to `TIs/PLDs`. Below this selection there is a graphical illustration of the structure of the data set:

The data set volumes go from left to right. Starting with the top line (red) we see that the data set consists of 6 TIs/PLDs, and within each PLD are 8 repeats (blue), and within each repeat there is a label and a control image.

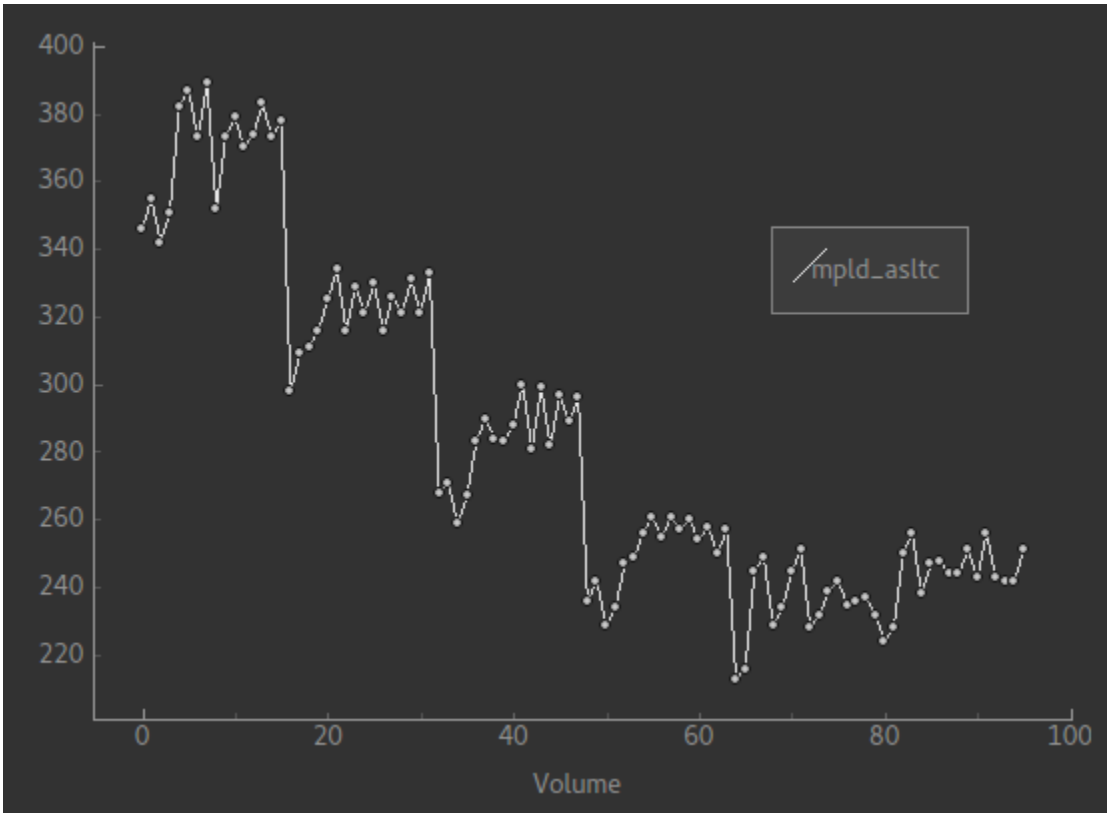
Below the grouping diagram, there is a visual preview of how well the *actual* data signal matches what would be expected from this grouping. The actual data signal is shown in green, the expected signal from the grouping is in red, and here they match nicely, showing that we have chosen the correct grouping option.



If we change the `Data Grouped by` option to `Repeats` (incorrect) we see that the actual and expected signal do not match up:

We can get back to the correct selection by clicking `Auto detect` which chooses the grouping which gives the best match to the signal.

Another way to determine the data ordering is to open the `Widget->Analysis->Voxel Analysis` widget and select a GM voxel, which should clearly shows 6 groups of PLDs (rather than 8 groups of repeats):



Each of the six roughly horizontal section of the signal represents the repeats at a given PLD and again the zig-zag pattern of the label-control images within each PLD are visible.

The remaining options are the same as for the single-PLD example:

- Labelling - `cASL/pcASL`
- Readout - `2D multi-slice` with `Time per slice` of 45.2ms

We can use the same structural and calibration data as for the previous example because they are the same subject. The analysis pipeline will correct for any misalignment between the calibration image and the ASL data. We can also keep the distortion correction setup from before.

This analysis shouldn't take a lot longer than the equivalent single PLD analysis, but feel free to skip ahead to the next section whilst you are waiting.

The results from this analysis should look similar to that obtained for the single PLD pcASL. That is reassuring as it is the same subject. The main difference is the a data set named `arrival`. If you examine this image you should find a pattern of values that tells you the time it takes for blood to transit between the labeling and imaging regions. You might notice that the `arrival` image was present even in the single-PLD results, but if you looked at it contained a single value - the one set in the Analysis tab - which meant that it appeared blank in that case.

Arterial/Macrovascular Signal Correction

In the analysis above we didn't attempt to model the presence of arterial (macrovascular) signal. This is fairly reasonable for pcASL in general, since we can only start sampling some time after the first arrival of labeled blood-water in the imaging region. However, given we are using shorter PLD in our multi-PLD sampling to improve the SNR there is a much greater likelihood of arterial signal being present. Thus, we might like to repeat the analysis with this component included in the model.

Return to your analysis from before. On the `Analysis` tab select `Macro vascular component`. Click `Run` again.

The results should be almost identical to the previous run, but now we also gain some new data: `aCBV_native` and `aCBV_calib_native`.

Following the convention for the perfusion images, these are the relative and absolute arterial (cerebral) blood volumes respectively. If you examine one of these and focus on the more inferior slices you should see a pattern of higher values that map out the structure of the major arterial vasculature, including the Circle of Willis. A colour map range of 0-100 helps with this, as well as clamping the colour map for out of range data:

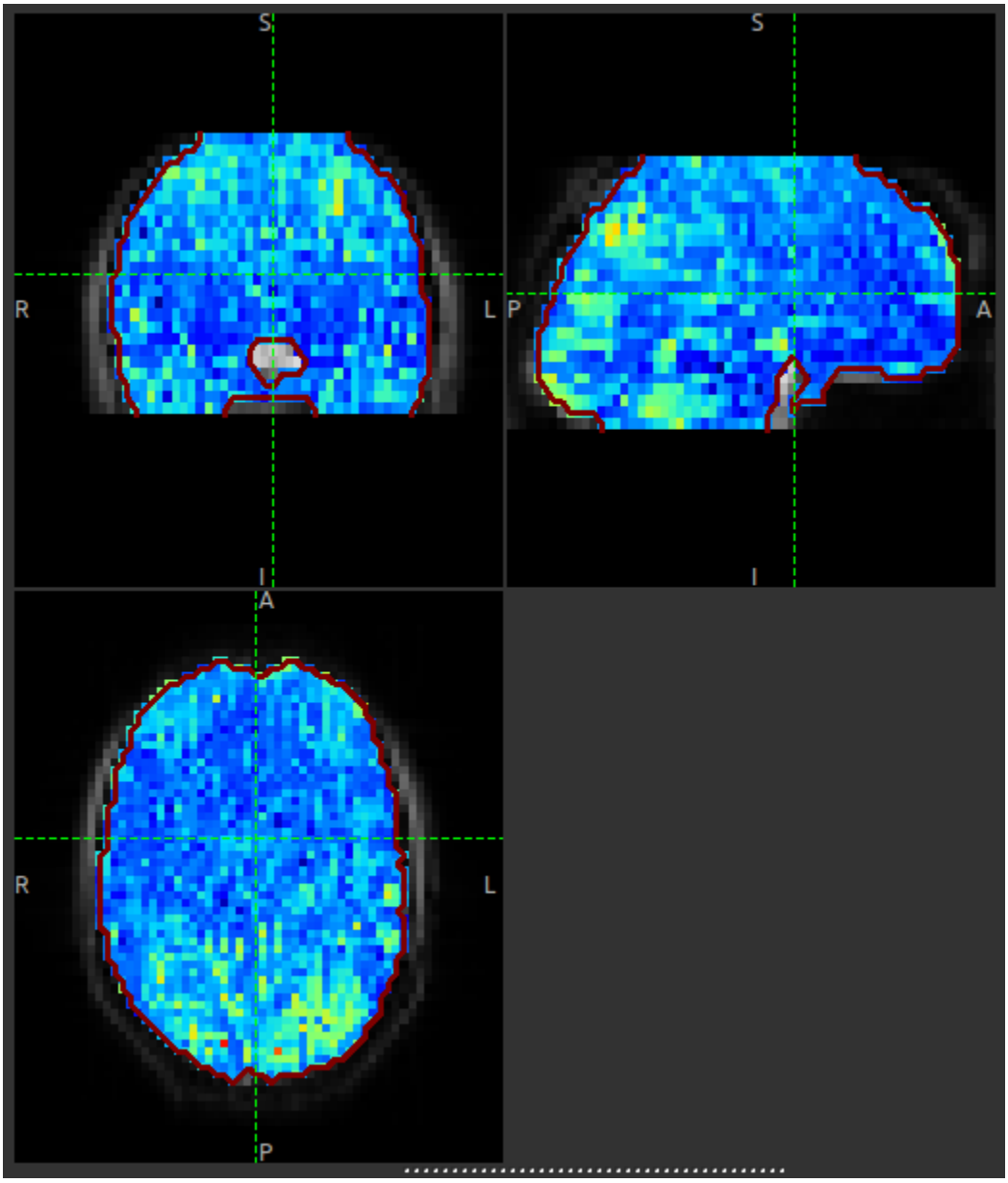
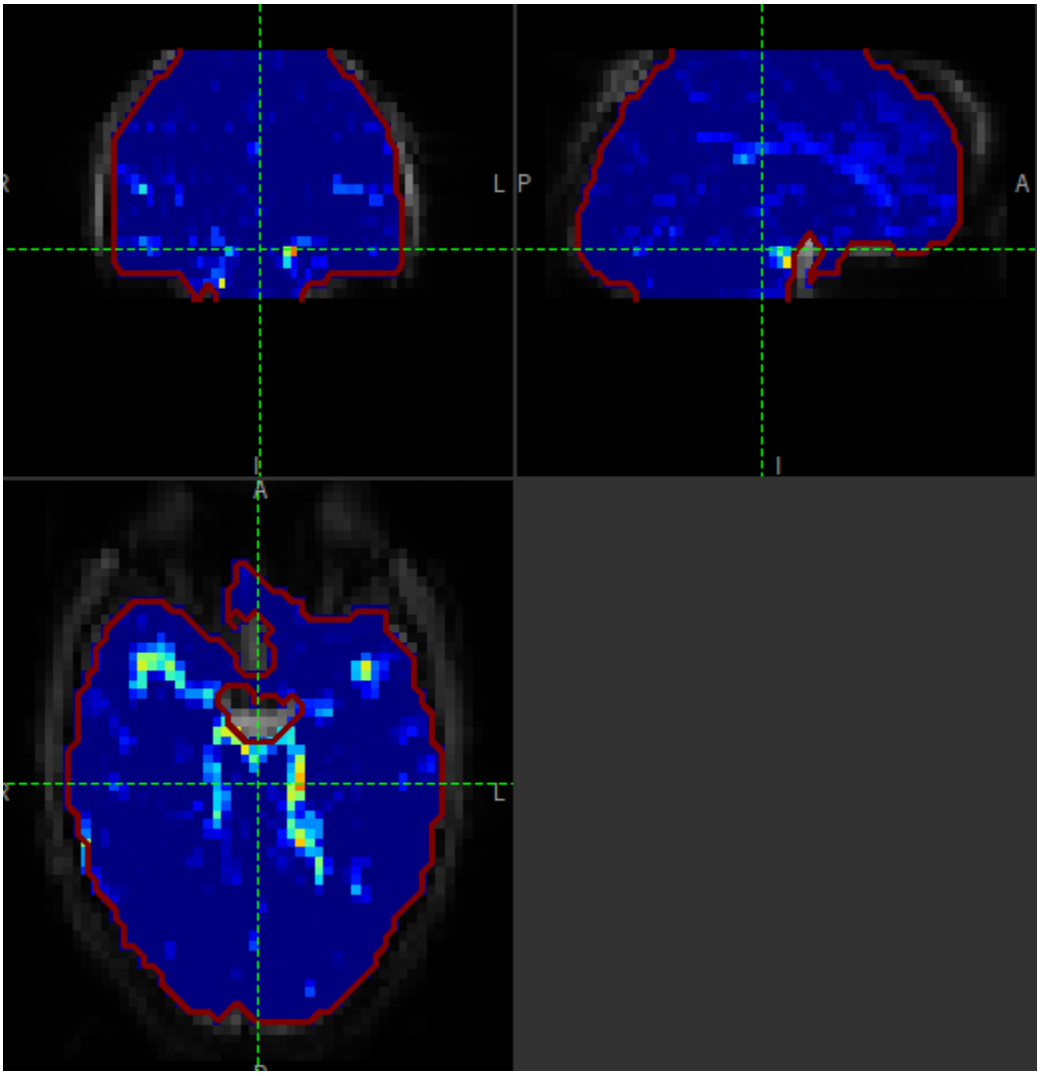


Fig. 4: Arrival time of the labelled blood showing delayed arrival to the posterior regions of the brain.

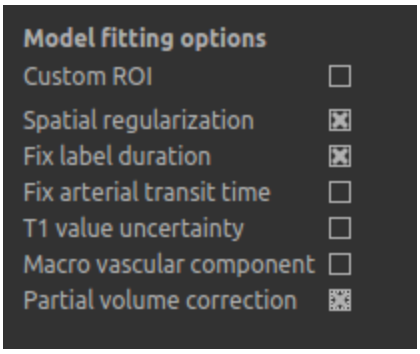


This finding of an arterial contribution in some voxels results in a correction to the perfusion image - you may now be able to spot that in the same slices where there was some evidence for arterial contamination of the perfusion image before that has now been removed.

Partial Volume Correction

In the same way that we could do partial volume correction for single PLD pcASL, we can do this for multi-PLD. If anything partial volume correction should be even better for multi-PLD ASL, as there is more information in the data to separate grey and white matter perfusion.

Just like the single PLD case we will require structural information, entered on the `Structure` tab. On the `Analysis` tab, select `Partial Volume Correction`.



Again, this analysis will not be very quick and so you might not wish to click `Run` right now.

You will find the results of this analysis already completed for you in the directory `~/fsl_course_data/ASL/oxasl_mpld_pvout`. This results directory contains, as a further subdirectory, `pvcorr`, within the `native_space` subdirectory, the partial volume corrected results: gray matter (`perfusion_calib.nii.gz etc`) and white matter perfusion (`perfusion_wm_calib.nii.gz etc`) maps.

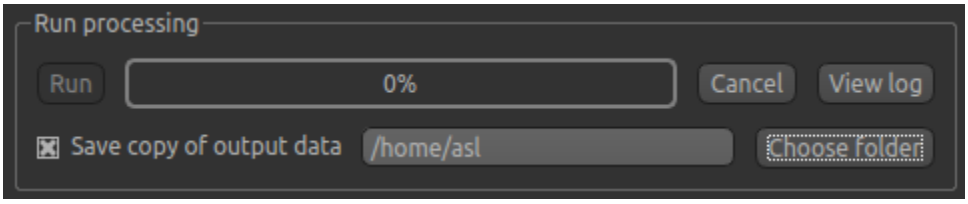
Alongside these there are also gray and white matter ATT maps (`arrival` and `arrival_wm` respectively). The estimated maps for the arterial component (`aCBV_calib.nii.gz` etc) are still present in the `pvcorr` directory. Since this is not tissue specific there are not separate gray and white matter versions of this parameter.

Additional useful options

A full description of the options available in the ASL processing widget are given in the reference documentation, however, here are a few in particular that you may wish to make use of:

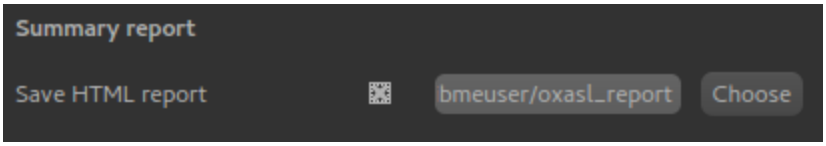
Save copy of output data

You can of course save the output data from your analysis using `File->Save Current Data` however it's often useful to have all the output saved automatically for you. By clicking on this option (underneath the `Run` button) and choosing an output folder, this will be done.



Generate HTML report

This option is available on the `Output` tab and will generate a summary report of the whole pipeline in the directory that you specify. To get this you will need to select the checkbox and enter or choose a directory to store the report in.



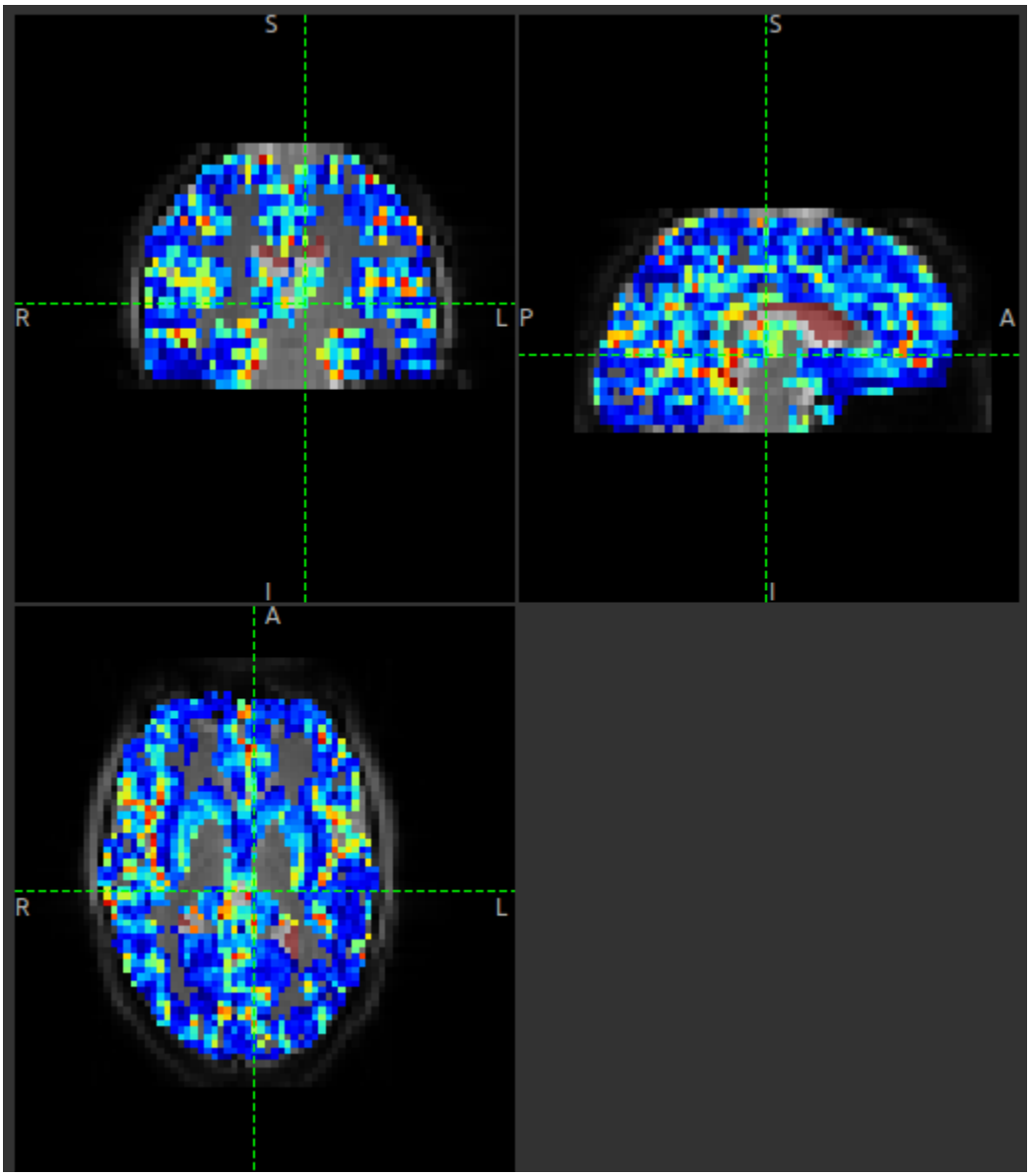


Fig. 5: GM perfusion (masked to include only voxels with >= 10% GM)

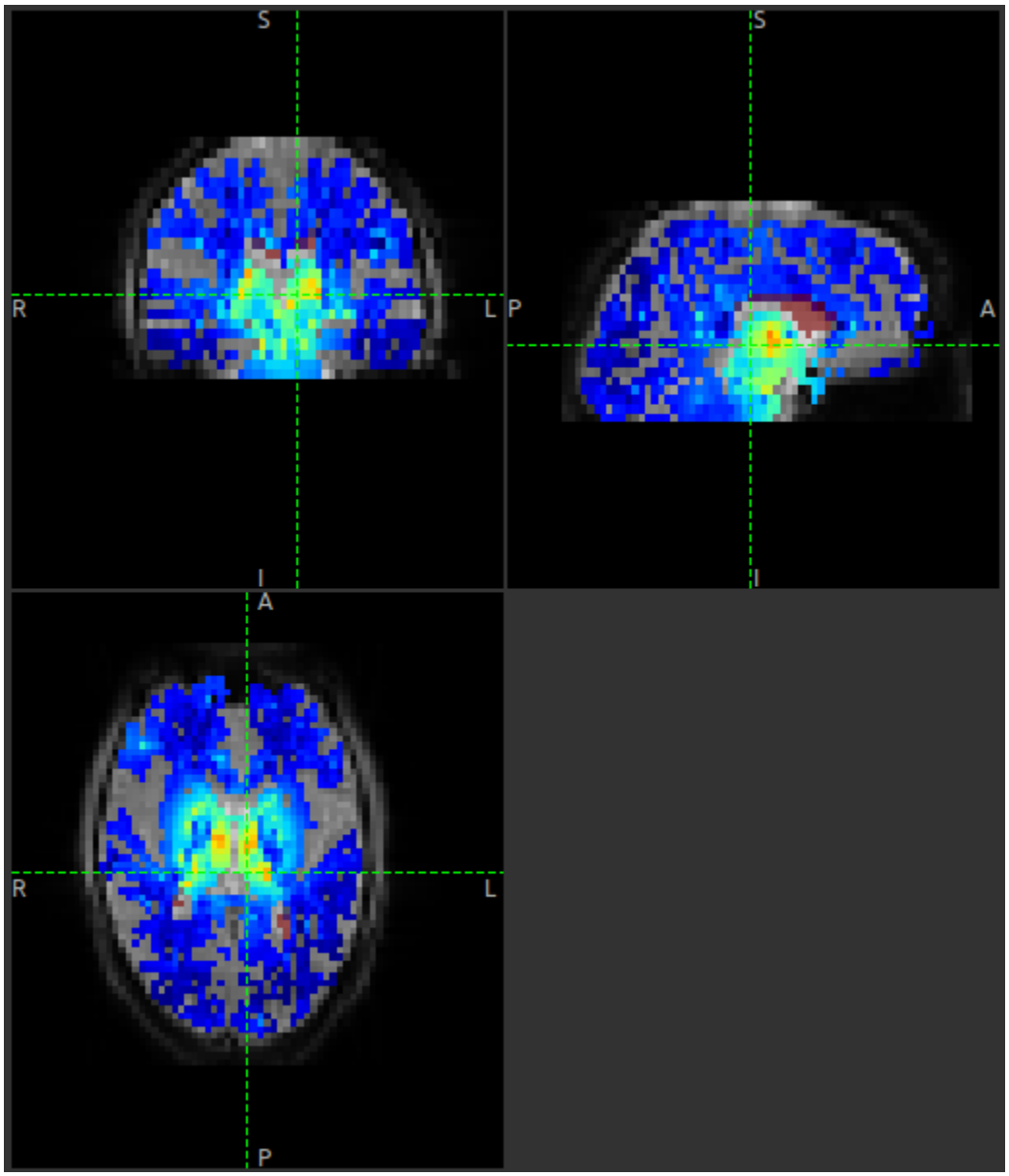
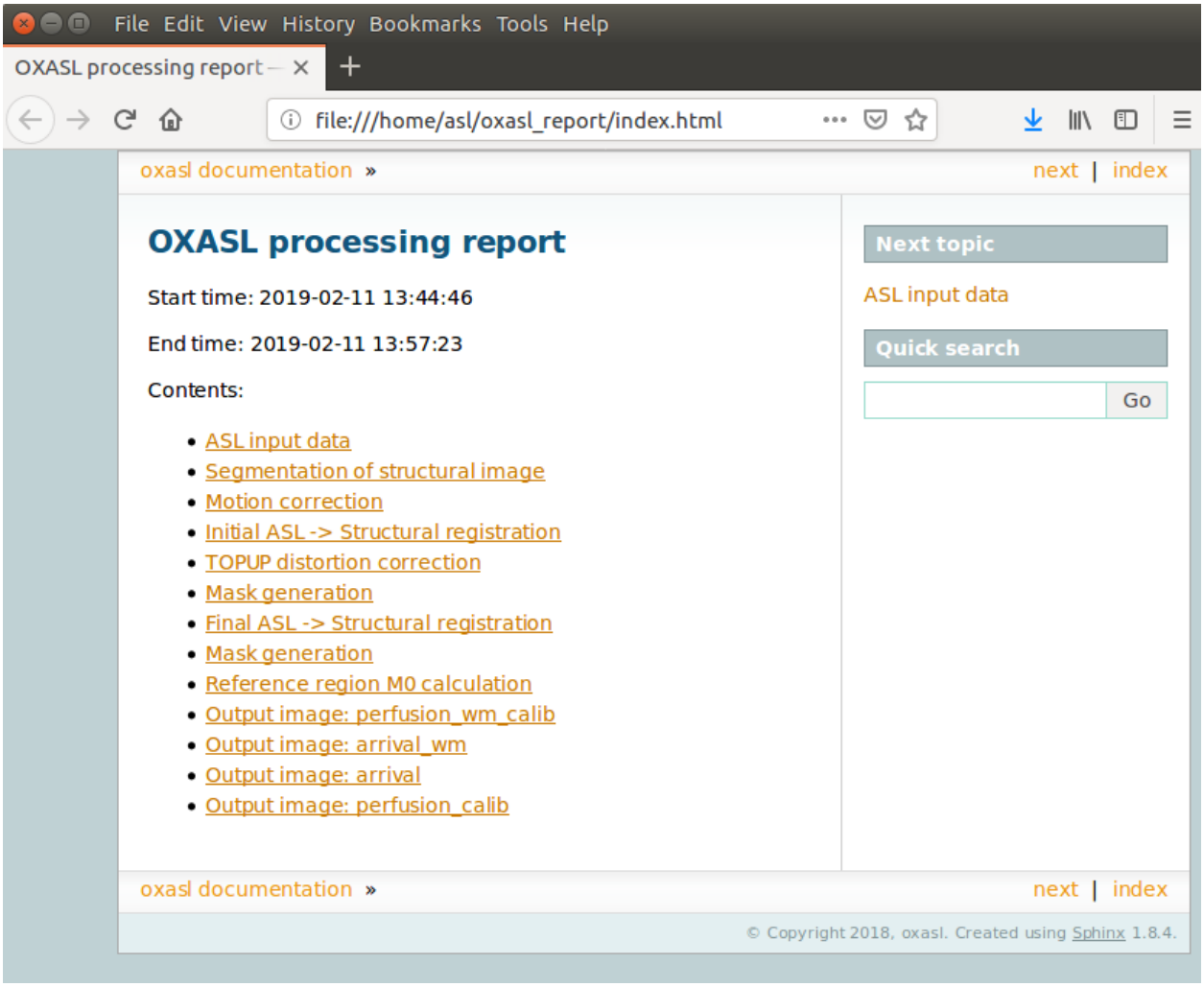


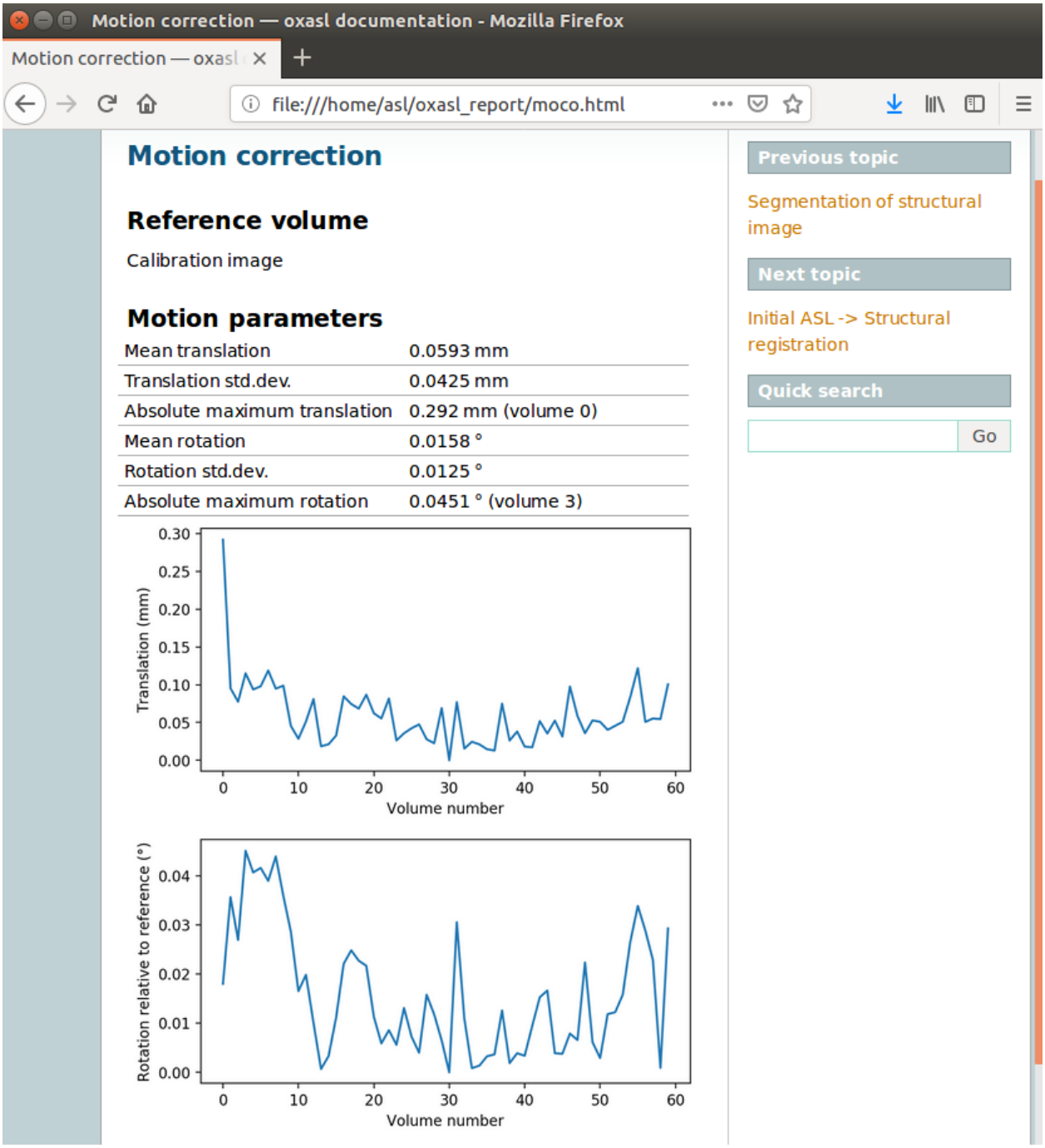
Fig. 6: WM perfusion (masked to include only voxels with >= 10% WM)

Quantiphyse will attempt to open the report in your default web browser when the pipeline has completed, but if this does not happen you can navigate to the directory yourself and open the `index.html` file.

Below is an example of the information included in the report:

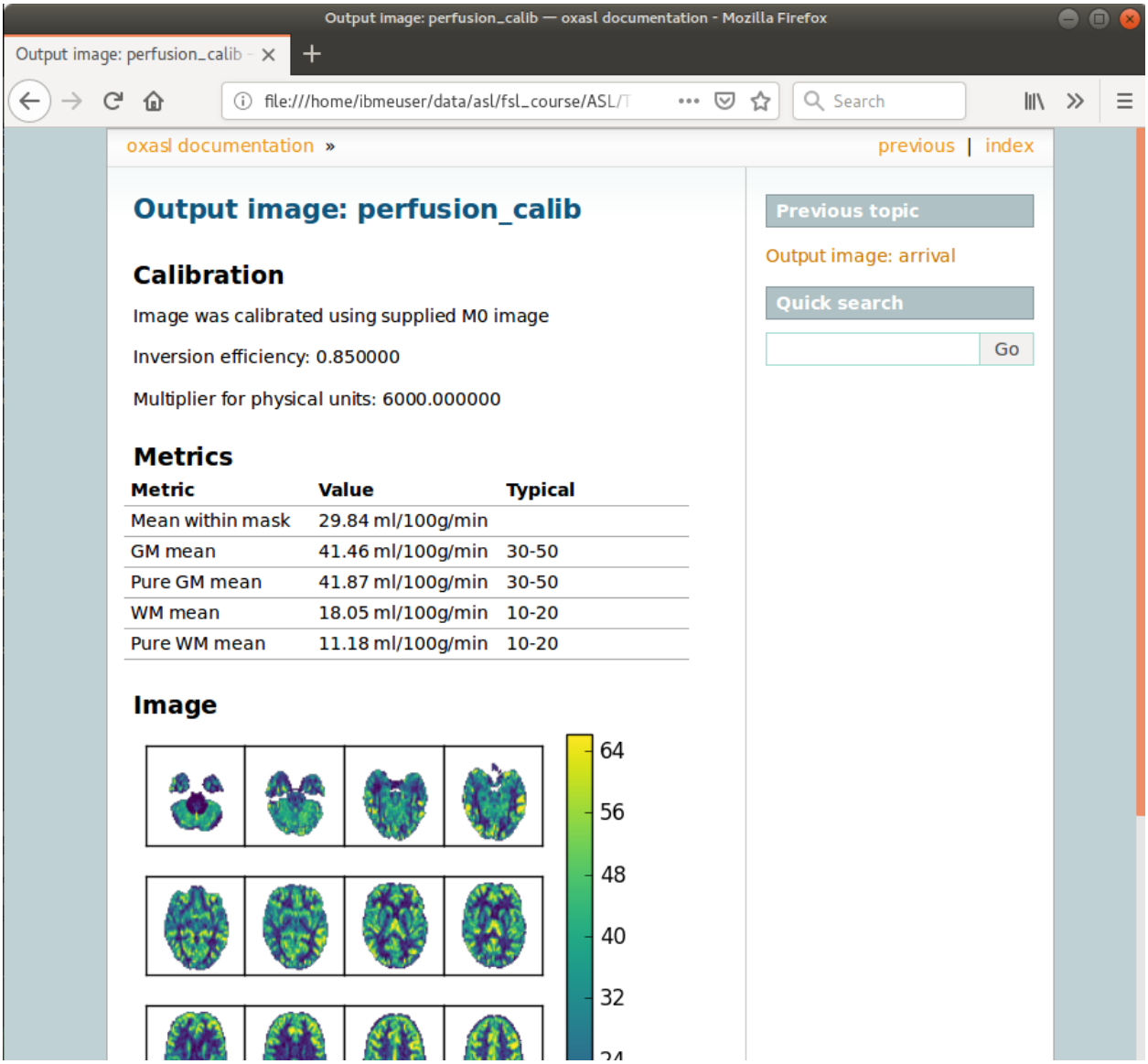


The links are arranged in the order of the processing steps and each link leads to a page giving more detail on this part of the pipeline. For example here's it's summary of the motion correction step for the single-PLD data:



This shows that there's not much motion generally and no particularly *bad* volumes.

If we click on the perfusion image link we get a sample image and some averages in GM and WM. This is useful to check that the analysis seems to have worked and the numbers are in the right range:



References

A walkthrough tutorial based on the FSL course practical session on ASL

Perfusion quantification in Tumours using Multi-PLD pCASL

The purpose of this exercise is to look at some multi-PLD pcASL in a clinical example of glioblastoma multiforme[1][2] to assess how perfusion changes within the tumour.

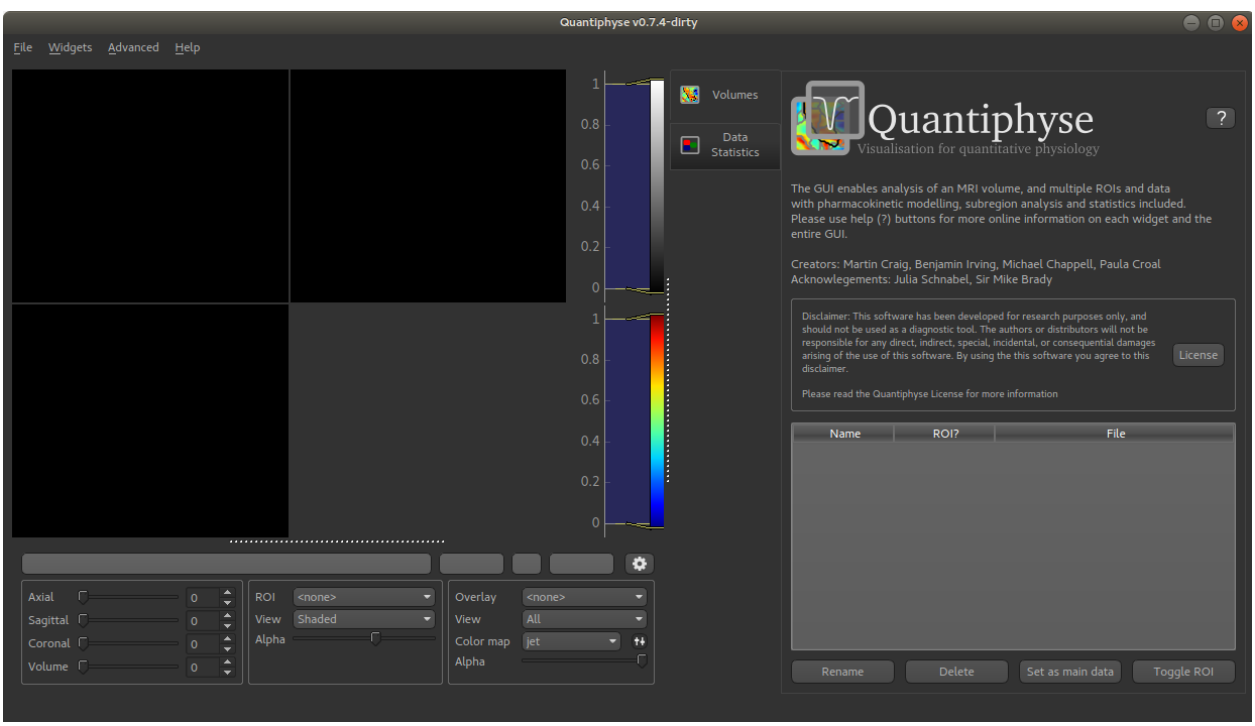
Contents

[1] Croal et al., Proc. ISMRM, 2019
[2] https://www.oncology.ox.ac.uk/trial/imago

Basic Orientation

Before we do any data processing, this is a quick orientation guide to Quantiphyse if you've not used it before. You can skip this section if you already know how the program works.

Start the program by typing `quantiphyse` at a command prompt, or clicking on the Quantiphyse icon in the menu or dock.



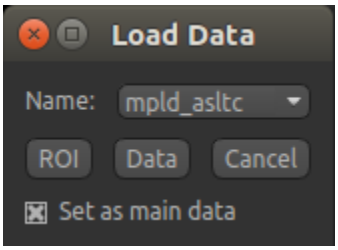
Loading the data

If you are taking part in an organized practical workshop, the data required may be available in your home directory, in the `fsl_course/IMAGO` folder. If not, an encrypted zipfile containing the data can be downloaded below - you will be given the password by the course organizers:

- Self extracting Windows archive
- Encrypted 7zip archive for Unix

Note: To extract the 7zip archive on Linux, download and then use the command `7z x IMAGOASL_Michigan.7z`

Start by loading the ASL data into Quantiphyse - use `File->Load Data` or drag and drop to load the file `mpld_asltc.nii.gz`. In the `Load Data` dialog select `Data`.



The data should look as follows:

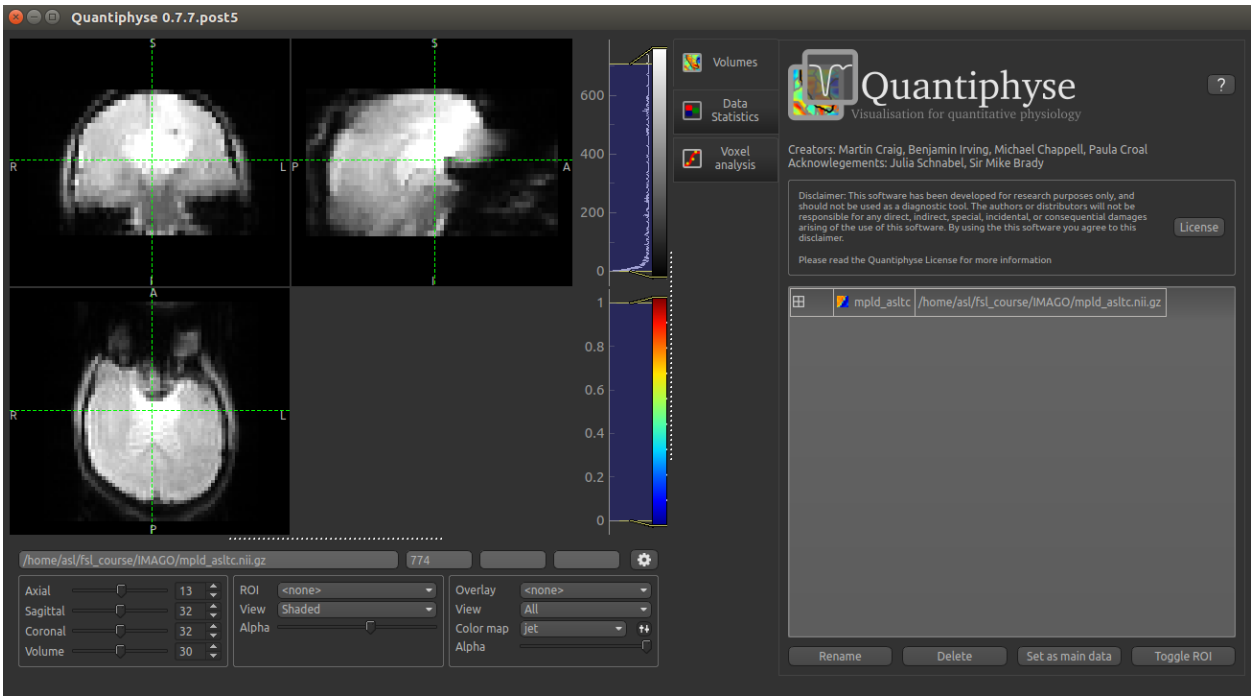


Image view

The left part of the window contains three orthogonal views of your data.

- Left mouse click to select a point of focus using the crosshairs

- Left mouse click and drag to pan the view
- Right mouse click and drag to zoom
- Mouse wheel to move through the slices
- Double click to 'maximise' a view, or to return to the triple view from the maximised view.

View and navigation controls

Just below the viewer these controls allow you to move the point of focus and also change the view parameters for the current ROI and overlay.

Widgets

The right hand side of the window contains 'widgets' - tools for analysing and processing data. Three are visible at startup:

- `Volumes` provides an overview of the data sets you have loaded
- `Data statistics` displays summary statistics for data set
- `Voxel analysis` displays timeseries and overlay data at the point of focus

Select a widget by clicking on its tab, just to the right of the image viewer.

More widgets can be found in the `Widgets` menu at the top of the window. The tutorial will tell you when you need to open a new widget.

For a slightly more detailed introduction, see the *Getting Started* section of the User Guide.

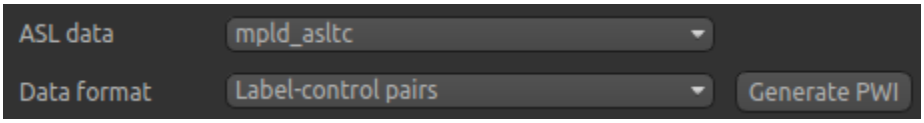
Perfusion quantification

In this section we will quantify perfusion for the dataset just loaded.

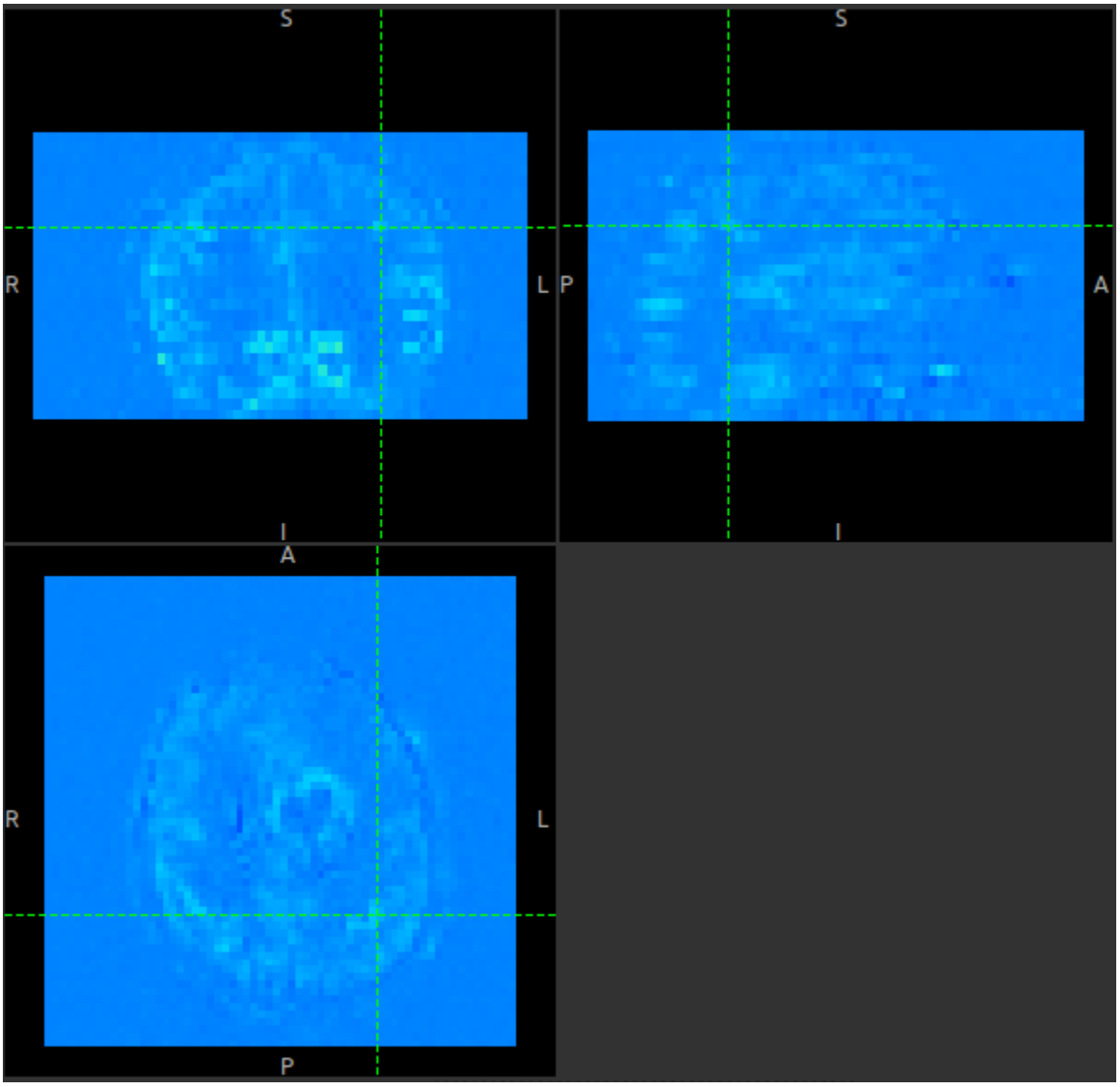
This dataset used pCASL labelling, with a duration of 1.8 seconds, and 5 post-labeling delays of 0.4, 0.8, 1.2, 1.6 and 2.0 seconds. The label-control ASL series contains 60 volumes, with each PLD repeated 6 times, thus there are 12 volumes (label and control paired) each PLD. The data is in the order that it was acquired, which will be important for setting up the analysis.

A perfusion weighted image

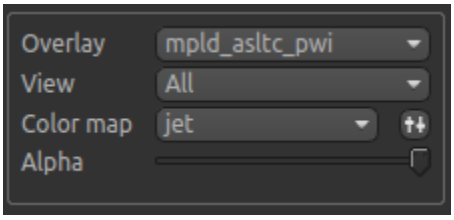
Open the `Widgets->ASL->ASL Data Processing` widget. We do not need to set all the details of the data set yet, however note that the data format is (correctly) set as `Label-control pairs`.



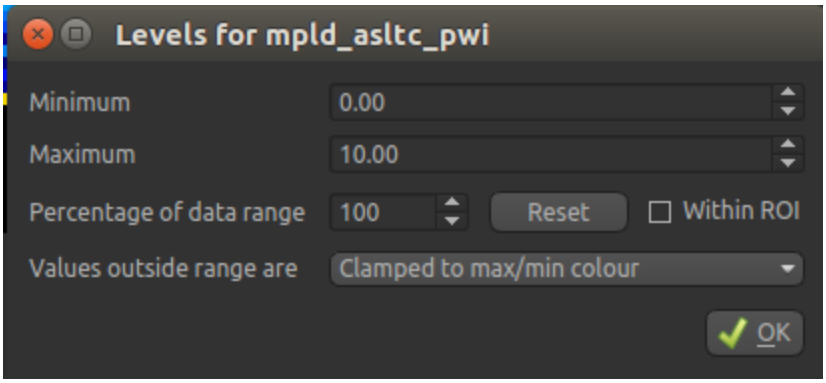
Click on the `Generate PWI` button. This performs label-control subtraction and averages the result over all repeats. The result is displayed as a colour overlay, which should look like a perfusion image:



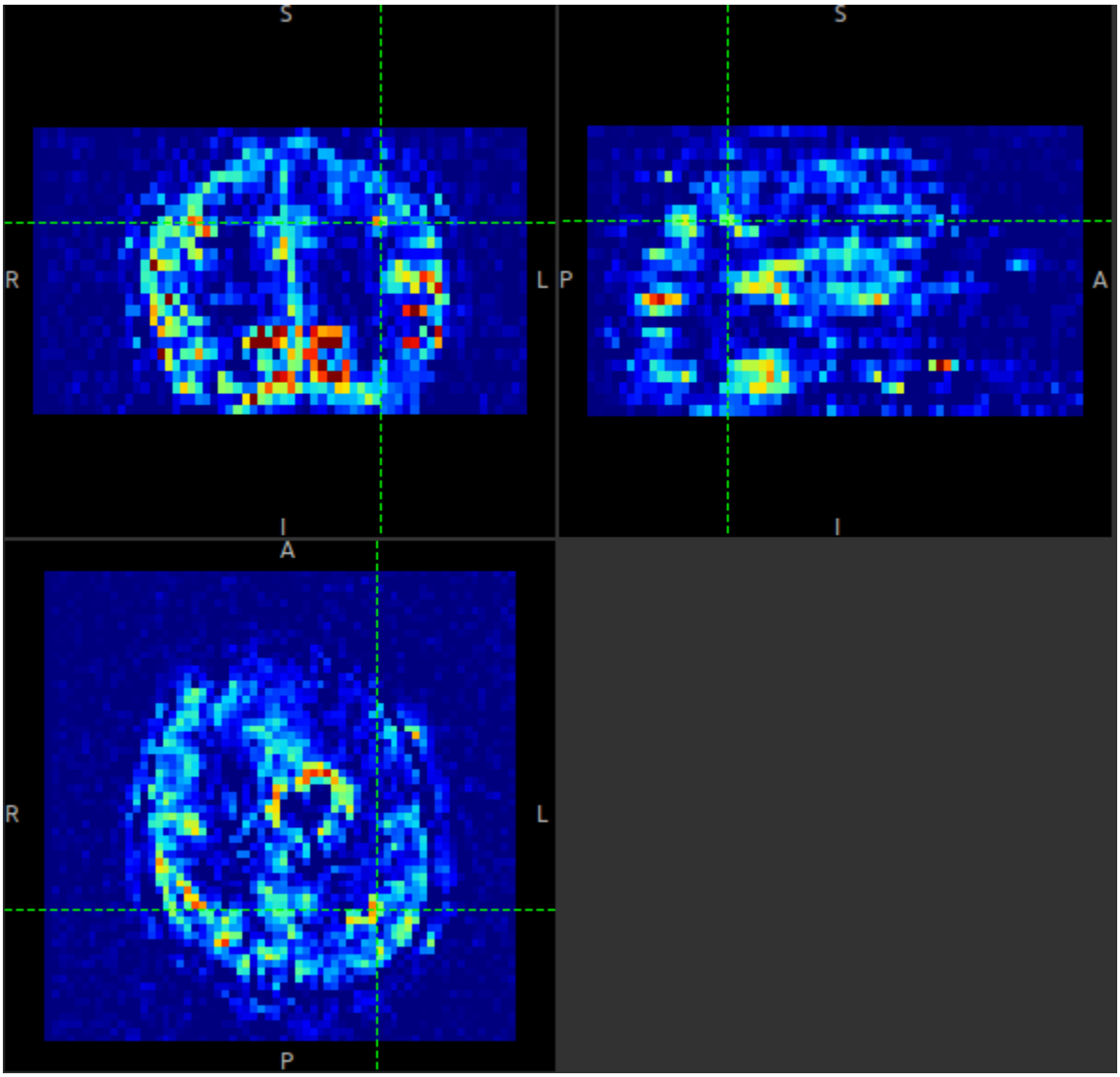
We can improve the display a little by adjusting the colour map. Find the overlay view options below the main image view:



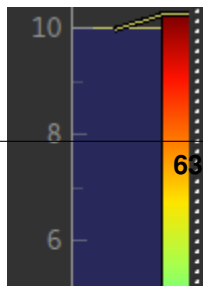
Next to the `Color Map` option (which you can change if you like!) there is a levels button  which lets you change the min and max values of the colour map. Set the range from `0` to `10` and select `Values outside range` to `Clamped`.



Then click `Ok`. The perfusion weighted image should now be clearer:



You could also have modified the colour map limits by dragging the colourmap range widget directly - this is located to the right of the image view. You can drag the upper and lower limits with the left button, while dragging with the right button changes the displayed scale. You can also customize the colour map by clicking on the colour bar with the right button.

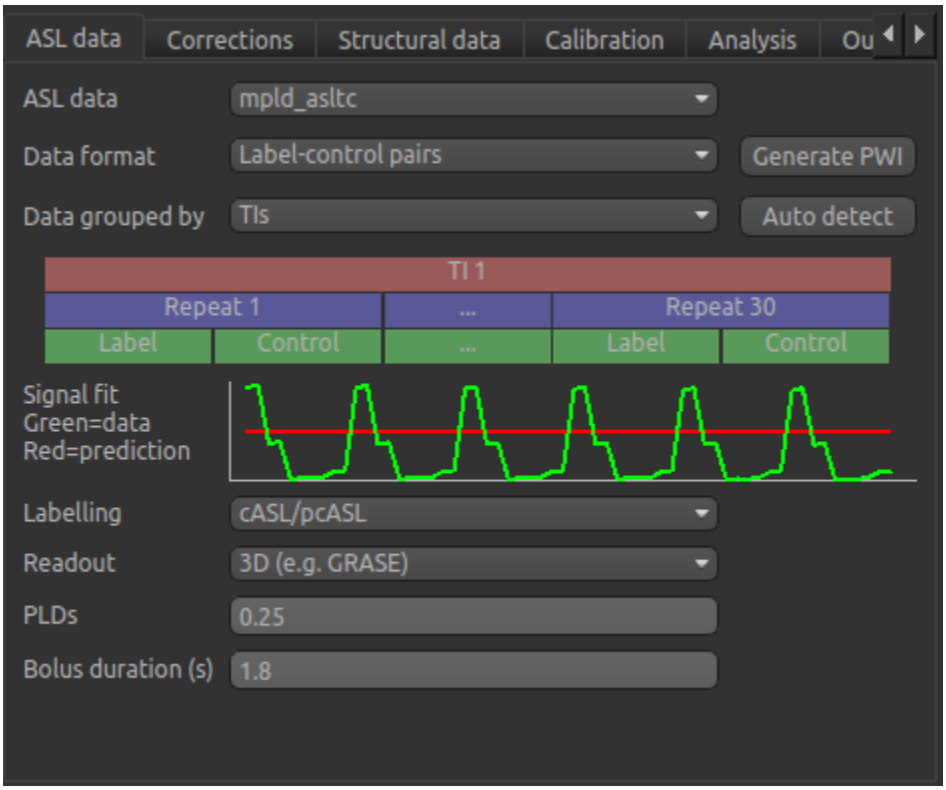


Warning: Dragging the colourmap is a little fiddly due to a GUI bug. Before trying to adjust the levels, drag down with the **right** mouse button briefly on the colour bar. This unlocks the automatic Y-axis and will make it easier to drag on the handles to adjust the colour map.

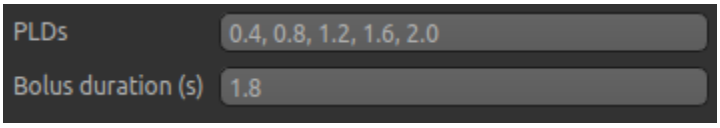
Model based analysis - Data set up

Looking at the `ASL data processing` widget we used to generate the PWI, you can see that this is a multi-page widget in which each tab describes a different aspect of the analysis pipeline.

We start by inputing the information on the first page which describes the ASL data. The defaults are shown below but we will need to change some of them to correctly describe our ASL acquisition.



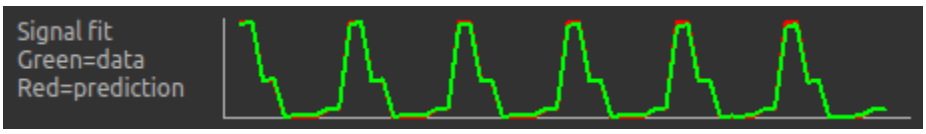
Firstly, we need to enter the 5 PLDs in the PLDs entry box – these can be separate by spaces or commas. We also make sure the label duration is set to 1.8s:



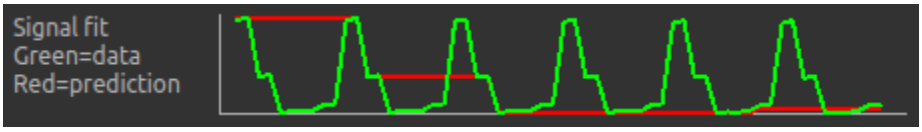
The data was acquired in label-control pairs (the default setting), and grouped by repeats. We need to change the `Data Grouped By` option to `Repeats` to reflect this. Below this selection there is a graphical illustration of the structure of the data set:

The data set volumes go from left to right. Starting with the top line (blue) we see that the data set consists of 6 repeats, and within each repeat there are 5 TIs (red), each with a label and control image (green).

Below the grouping diagram, there is a visual preview of how well the actual data signal matches what would be expected from this grouping. The actual data signal is shown in green, the expected signal from the grouping is in red, and here they match nicely, showing that we have chosen the correct grouping option.

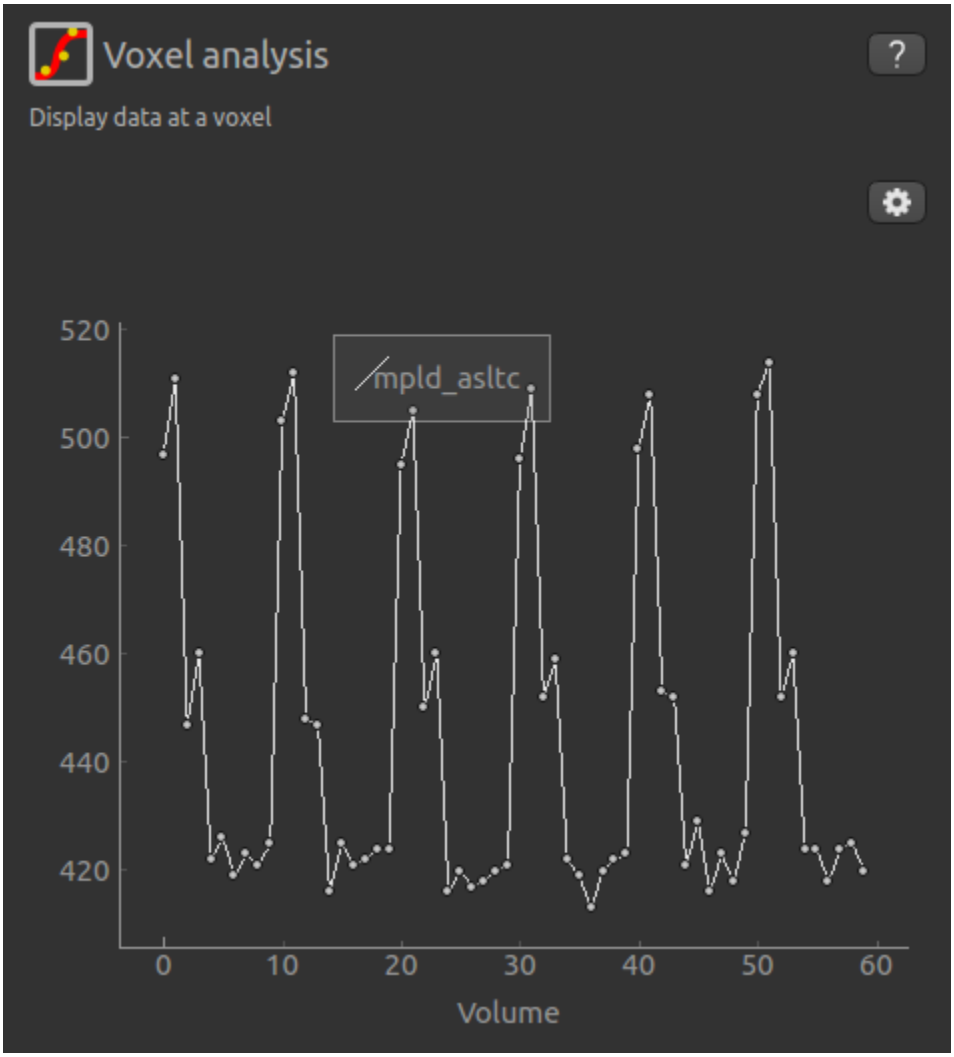


If we change the Data Grouped By option to TIs (incorrect) we see that the actual and expected signal do not match up:

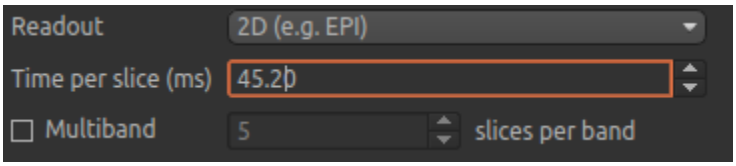


We can get back to the correct selection by clicking `Auto Detect` which chooses the grouping which gives the best match to the signal.

Another way to determine the data ordering is to select the `Voxel Analysis` widget and click on a GM voxel, which should clearly show 6 groups of repeats. Each of the 6 peaks represents a single repeat across all 5 PLDs, the zig-zag pattern of the label-control images are visible for each PLD.



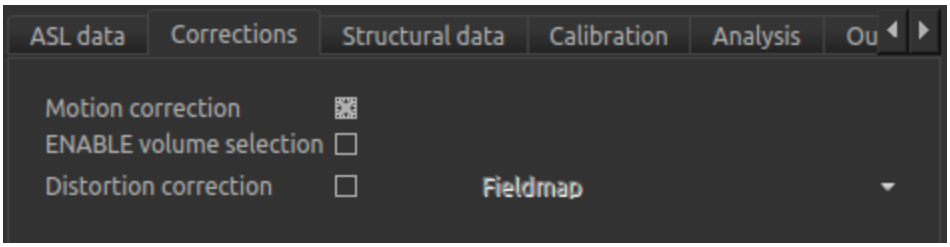
Returning to the ASL data processing page, we need to finalise our acquisition details. The labelling method is correctly set to cASL/pCASL, however we have a 2D readout with 45.2 ms between slices, so we need to change the Readout option to reflect this. When we select a 2D readout, the option to enter the slice time appears automatically.



Model based analysis - Analysis set up

In this section we invert the kinetics of the ASL label delivery to fit a perfusion image, and use the calibration image to get perfusion values in the units of ml/100g/min.

Firstly, on the `Corrections` tab, we will ensure that `Motion Correction` is checked (this should be enabled by default):

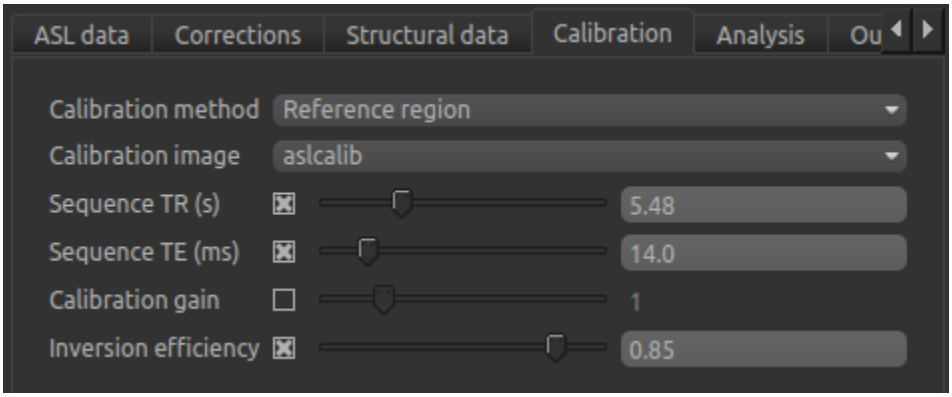


Due to potential challenges with MNI registration in the presence of tumour, we will work in the subject's native space, thus skip the `Structural` data tab, and instead move on to `Calibration`. To use calibration we first need to load the calibration image data file from the same folder containing the ASL data - again we can use drag/drop or the `File->Load Data` menu option to load the following files:

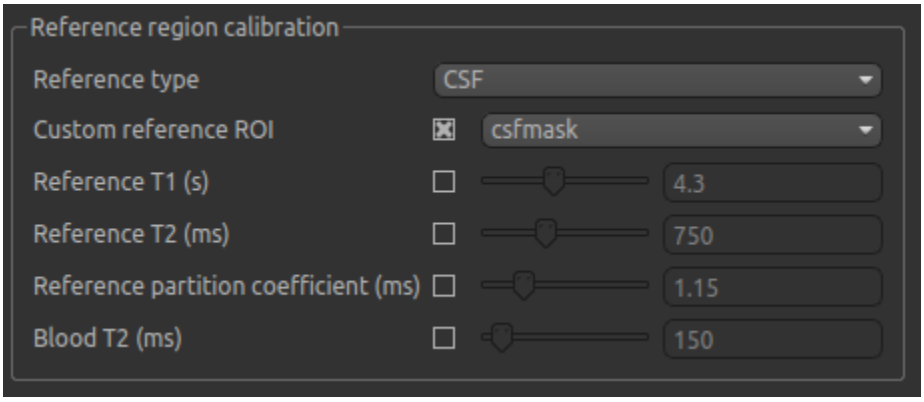
- `aslcalib.nii.gz` - Calibration (M0) image
- `csfmask.nii.gz` – CSF mask in subject's native space

Note: For the `csfmask` data ensure that this is loaded as an `ROI` in the data type selection box. If you forget to do this, you can modify it from the `Volumes` widget - click on the data set in the list and click the `Toggle ROI` button.

On the `Calibration` tab we will set the calibration method as `Reference region`, and will need to select the calibration image we have just loaded: `aslcalib`. The TR for this image was 5.48s, so click on the `Sequence TR` checkbox and set the value to `5.48`. Similarly, click on the `Sequence TE` checkbox and set the value to `14.0`. Finally, change the `Inversion efficiency` to `0.85` as we are using pCASL (the GUI is set to the PASL default of 0.98).

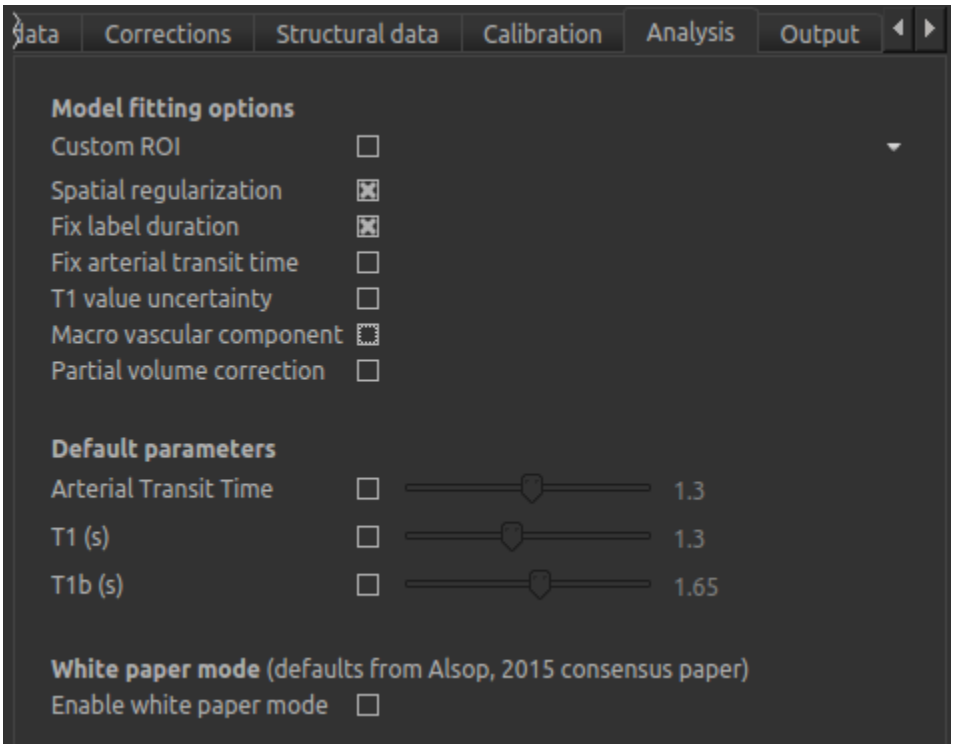


In the `Reference region` calibration box we will select the CSF option, and set the `Custom reference ROI` to the `csfmask` ROI which we have just loaded.

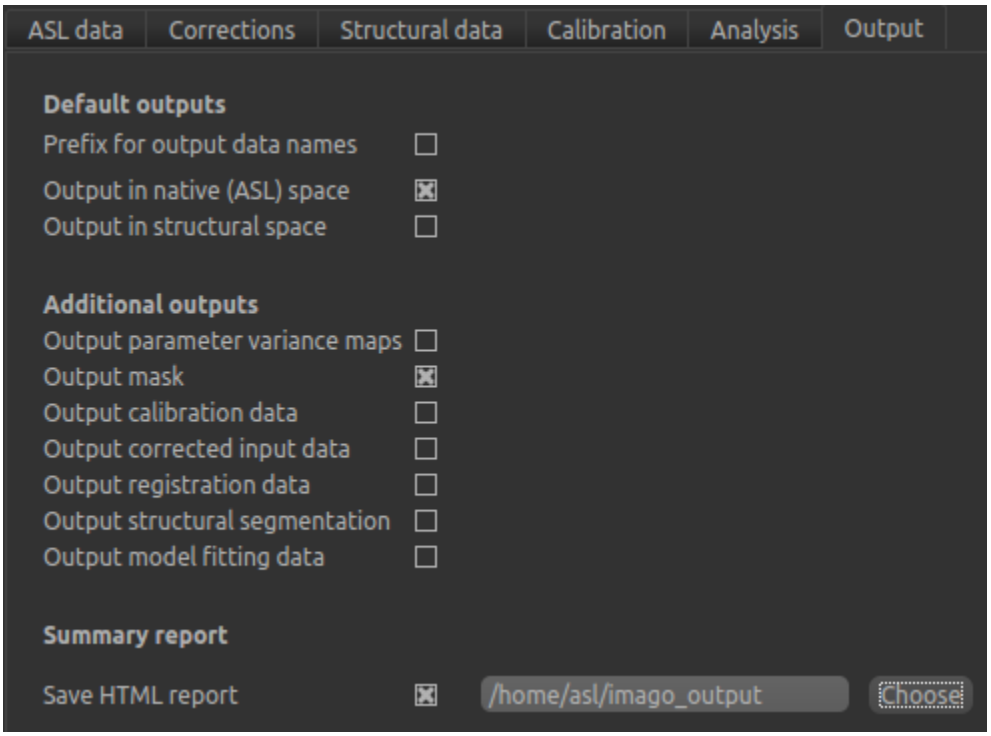


In the interest of time, this CSF mask has been made manually ahead of time, and provides a conservative mask within the ventricles.

On the `Analysis` tab the defaults do not need altering in this instance, except to turn the macrovascular component off.

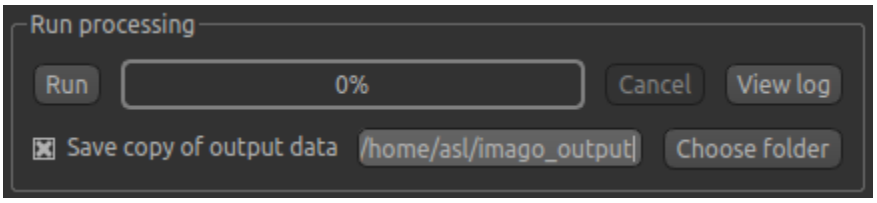


We will not change the defaults on the `Output` tab yet, but will select `Save HTML report`. Click `Choose` to set the output folder.

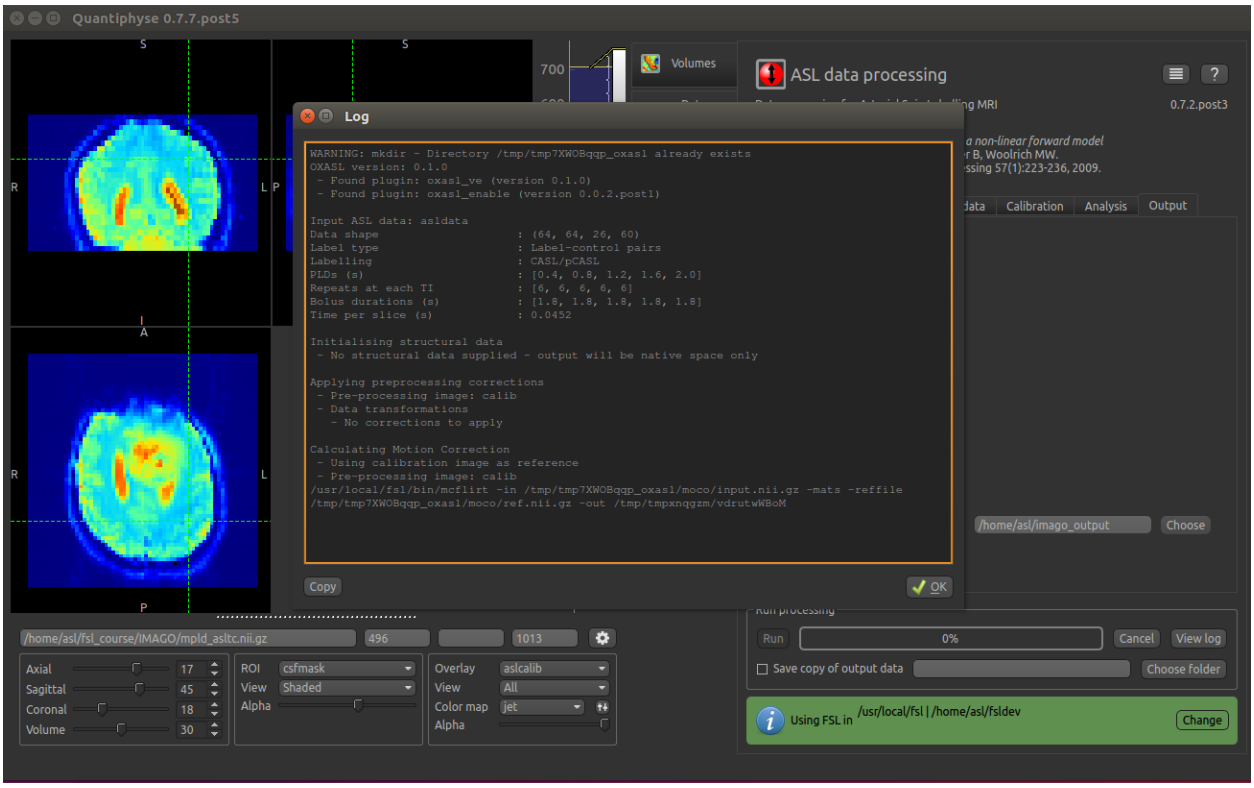


We are now set up to run the analysis - but before you do, check the green box at the bottom of the widget which reports where it thinks FSL is to be found. If the information does not seem to be correct, click the `Change` button and select the correct location of your FSL installation (if you are in an organized practical this should be correct).

As an additional step, you may want to save your output data. You can of course save the output data from your analysis after it has run using `File->Save Current Data`, however it's often useful to have all the output saved automatically for you. By selecting `Save copy of output data` (underneath the Run button) and choosing an output folder, this will be done.



Finally click `Run` at the bottom to run the analysis. You can click the `View Log` button to view the progress of the analysis which should only take a few minutes.

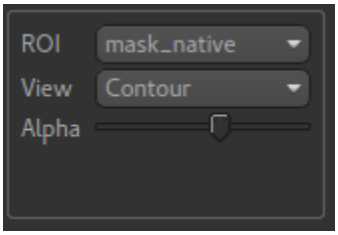


Once the analysis had completed (~5 mins), some new data items will be available. You can display them either by selecting them from the `Overlay` menu below the image display, or by clicking on the `Volumes` widget and selecting them from the list. The new data items are:

- `perfusion_native` - Raw (uncalibrated) perfusion map
- `perfusion_calib_native` - Calibrated perfusion data in ml/100g/min

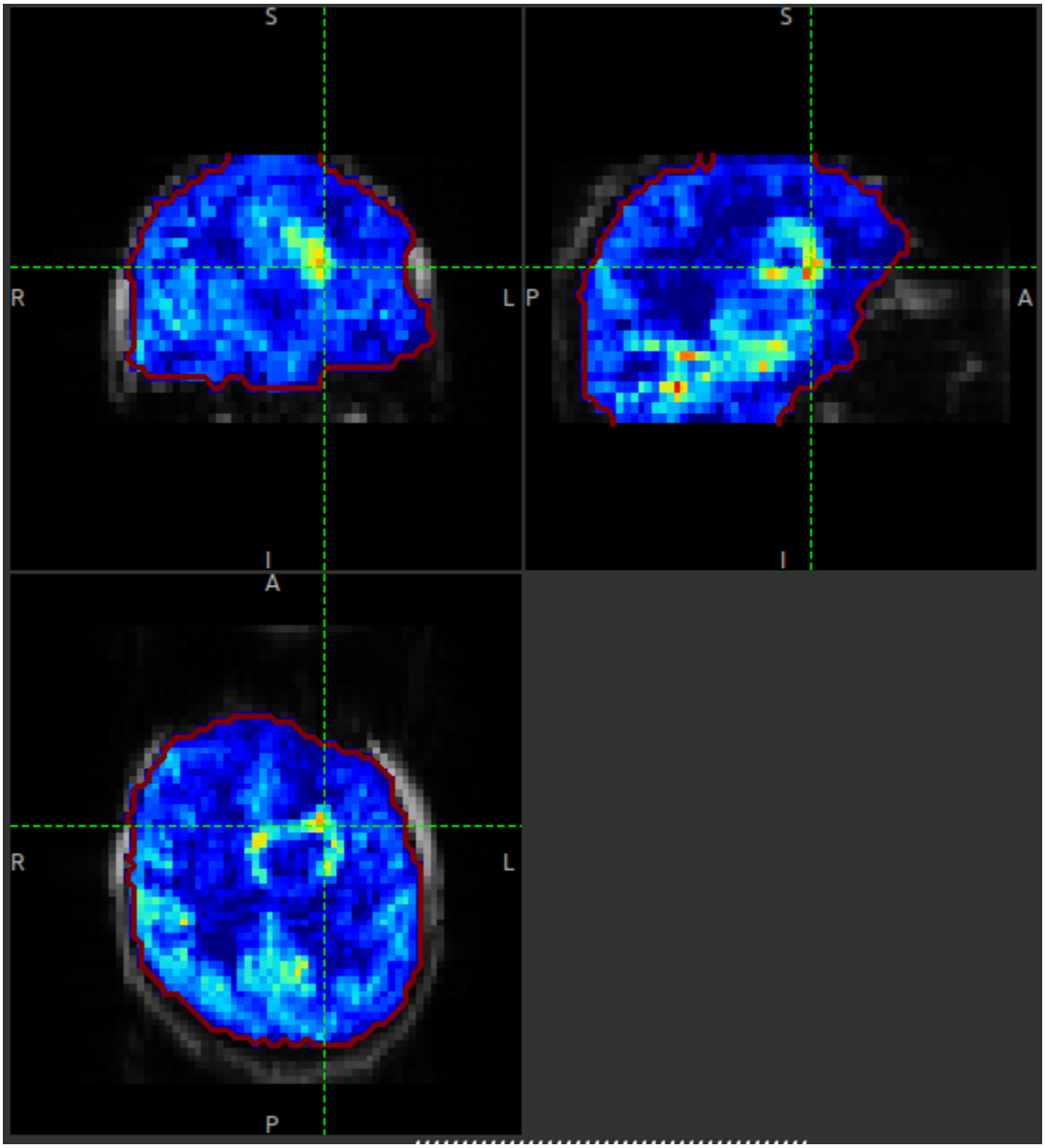
- `arrival_native` - time it takes for blood to transit between the labeling and imaging regions.
- `mask_native` - An ROI (which appears in the ROI selector under the image view) which represents the region in which the analysis was performed.

We can view these outputs within the brain mask only, by selecting `mask_native` from the ROI dropdown. The images may be clearer if we modify the view style for the ROI from `Shaded` to `Contour` (in the ROI options box underneath the image view). This replaces the translucent red mask with an outline:



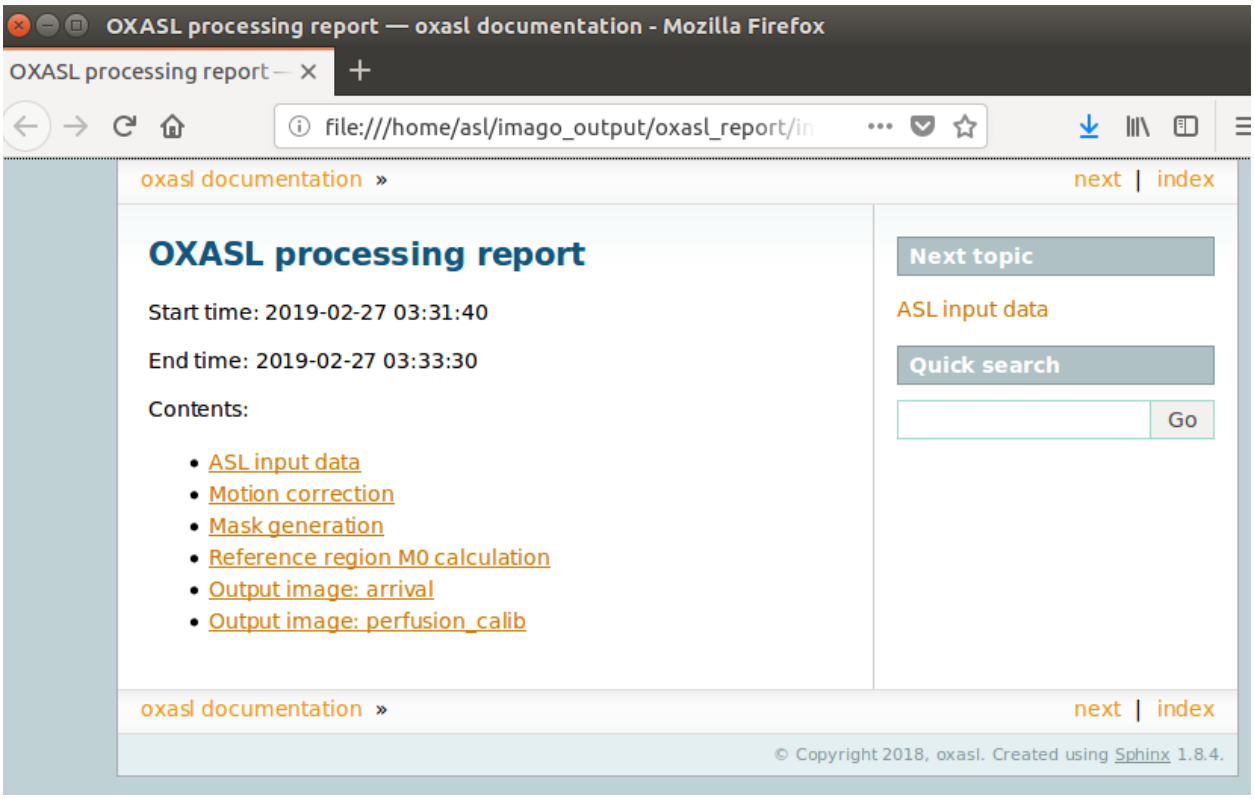
The `perfusion_calib_native` image should look similar to the perfusion weighted image we created initially, however the data range reflects the fact that it is in physical units. To get a clear visualustion set the colour map range to 0 – 60, and clamping to min/max using the Levels button  . You can also select `Only in ROI` as the `View` option just above this so we only see the perfusion map within the selected ROI.

The result should look something similar to below. Notice that you can see a ring of perfusion enhancement near the midline, this is consistent with tumour location, and gadolinium enhancement.

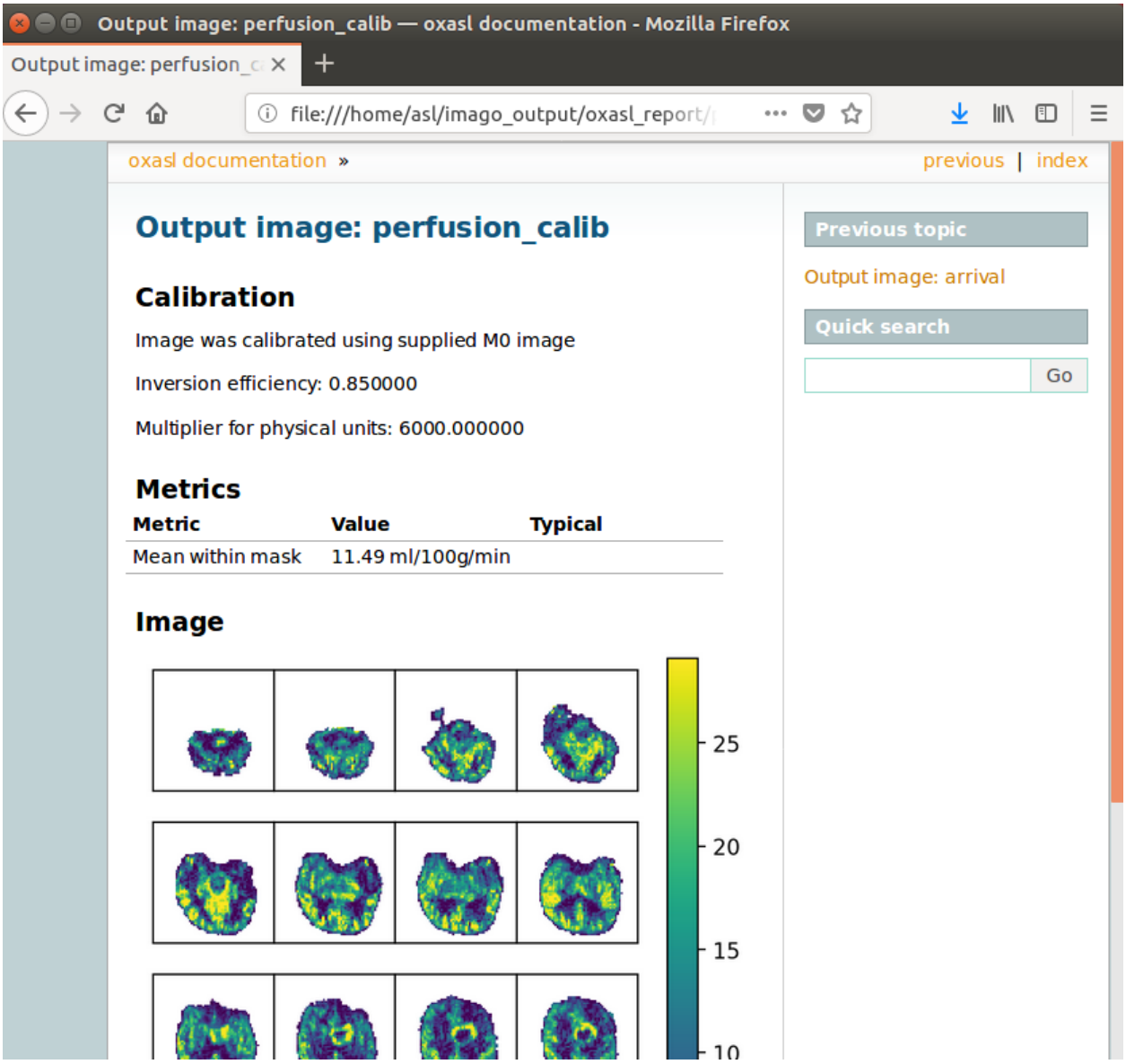


As well as outputting images, Quantiphyse will attempt to open the analysis report in your default web browser when the pipeline has completed, but if this does not happen you can navigate to the directory yourself and open the `index.html` file.

Below is an example of the information included in the report:



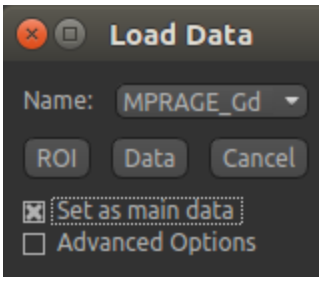
The links are arranged in the order of the processing steps and each link leads to a page giving more detail on this part of the pipeline. For example, if we click on the perfusion image link we get a sample image, which can be to check that the analysis seems to have worked as expected. Here, the mean within mask is not as informative as it might be for a healthy brain, as we are likely averaging in hypoperfused regions.



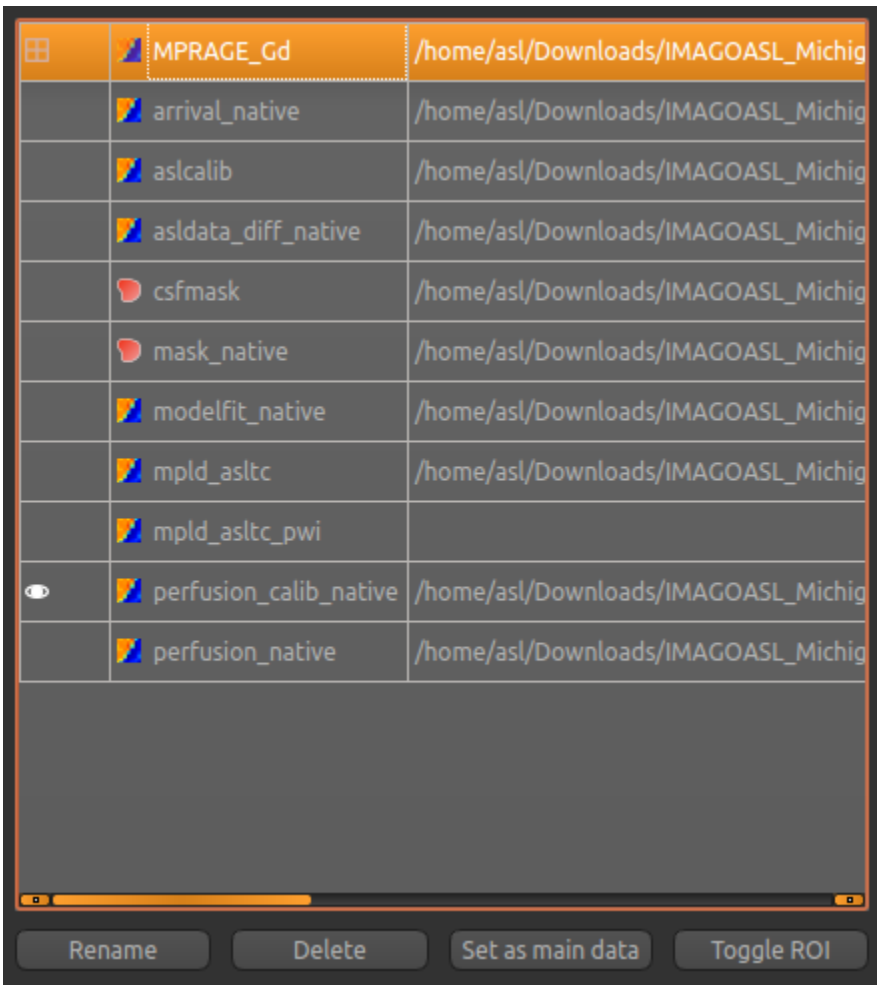
Comparison to structural changes

You may want to see how well the perfusion map corresponds to the tumour visualised on a typical anatomical image. You can load the patient's gadolinium-enhanced T1-weighted scan using `File->Load Data` and `MPRAGE_Gd.nii.gz`.

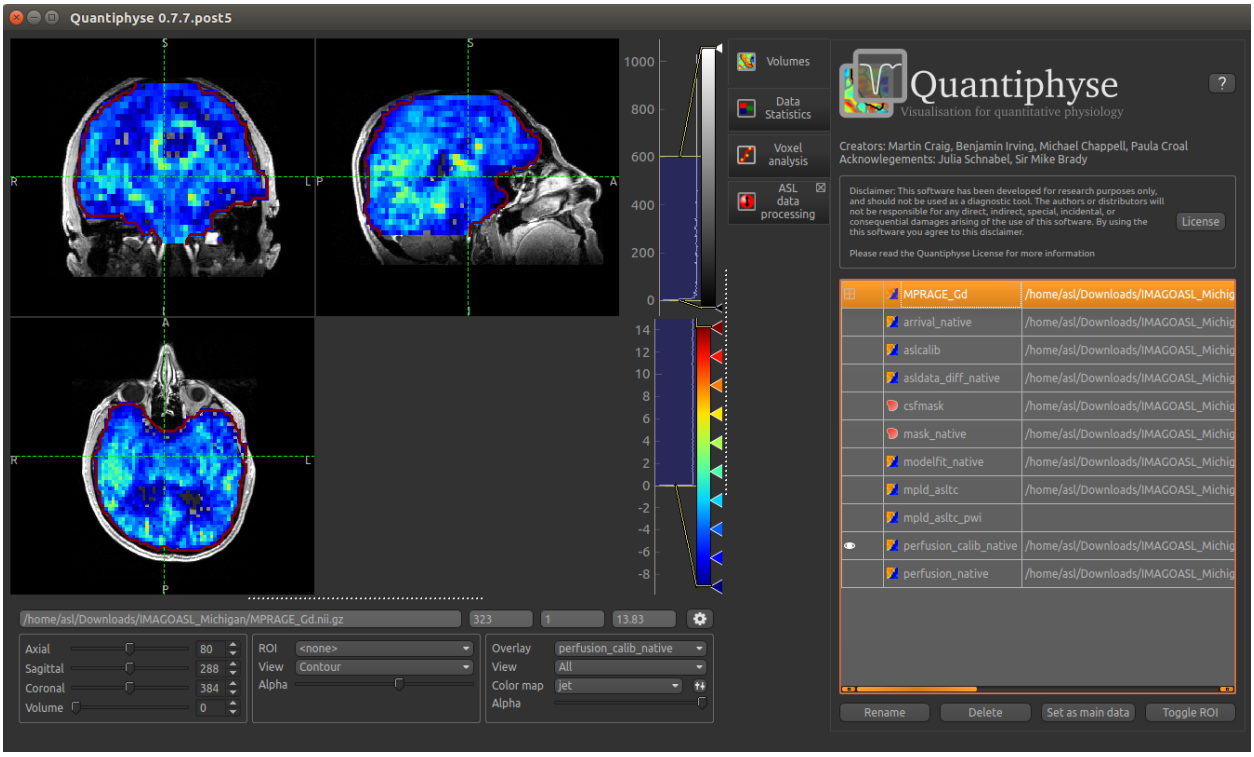
In order to overlay images on top of this structural image, check the `Set as main data` box when loading:



Note: If you forget to do this you can also select the `Volumes` widget, click on the `MPRAGE_Gd` image and click the `Set as main data` button.

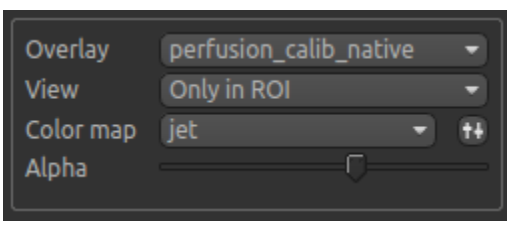


After setting the anatomical image as the main data you, other images selected from the `Overlay` list will be overlaid on top, for example the calibrated perfusion map:



The `Alpha` slider in the overlay box can be used to adjust the transparency of the overlay and compare to the anatomical image underneath.

You should be able to see that the T1 enhancing rim of the tumour corresponds to a region of increased perfusion. We could go on to load ROI's of the tumour and contralateral tissue to quantify this, however it is beyond the scope of this tutorial.



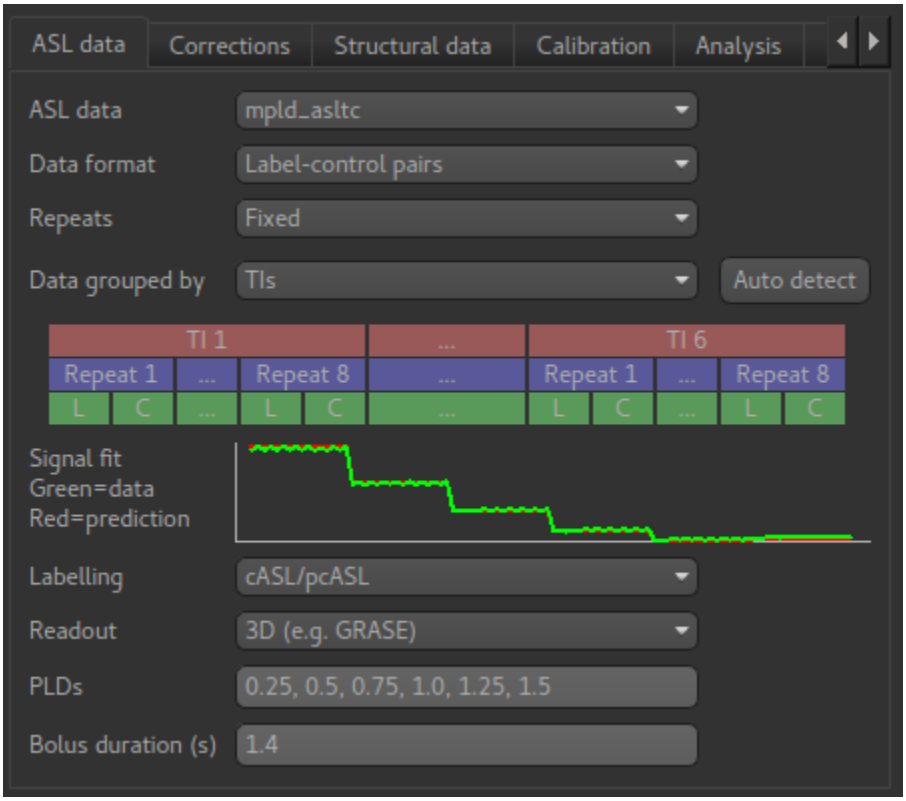
Note: These visualisations work best when `Only in ROI` is selected as the overlay view option.

References

5.2.2 Reference

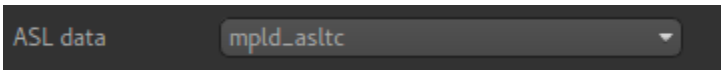
This set of pages goes through each page of the widget in turn an explains the options systematically with some examples.

ASL data tab



This tab describes the structure and acquisition parameters of your ASL data. Once you define the structure of a data set in one ASL widget, others will automatically pick up the same structure when using that data set. In addition, if you save the data set to a Nifti file, the structure information is saved as optional metadata and will be recognised when you load the data back into Quantiphyse.

Start by choosing the ASL data set you want to analyse from the `ASL data` selection box (it must be loaded into Quantiphyse first).

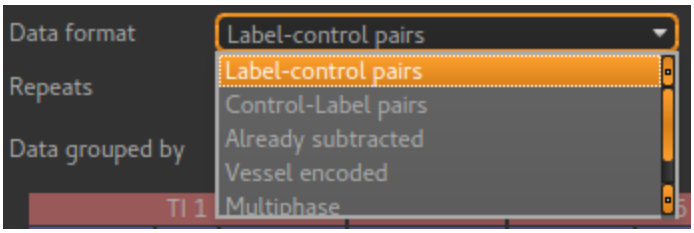


Data format

The data format describes the labelling scheme used for the data and can be described as

- Label-control pairs
- Control-Label pairs
- Multiphase
- Vessel encoded
- already tag-control subtracted or multiphase

Note that currently multiphase data is not supported by this widget, however a multiphase preprocessing widget is provided.

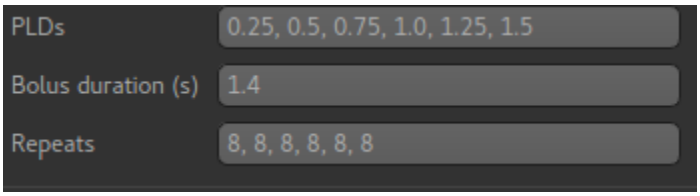


Repeats

By default repeats are fixed. The ASL widget will figure out how many repeats you have.

You can also select variable repeats, in which case each TI/PLD may have a different number of repeats.

The repeats entry is at the bottom with the TIs/PLDs and bolus duration. This is because there needs to be the same number of each so it's sensible to keep them together.



Data grouping/order

This describes the sequence of volumes contained in the ASL data set, and what each volume contains. The two main choices are `Grouped by TIs` and `Grouped by repeats`.

When grouped by TIs, the sequence of volumes would be as follows:

```
1. Tag for first TI
2. Control for first TI
3. Rrepeat tag for first TI
4. Repeat Control for first TI
   ... as above for remaining repeats
11. Tag for second TI
12. Control for second TI
13. Repeat tag for second TI
14. Repeat Control for second TI
   ... etc
```

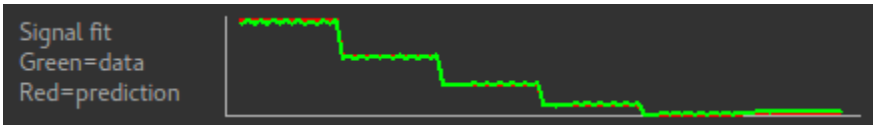
Data structure visualisation

The data structure visualisation shows how the data is grouped inside the file (the volumes in the data increase from left to right).

Starting at the top, this shows that the volumes are divided up into blocks corresponding to the 6 TIs we have defined. Within each block we have 8 repeats of this TI, and each repeat consists of a label and control image (in that order).

Signal fit visualisation

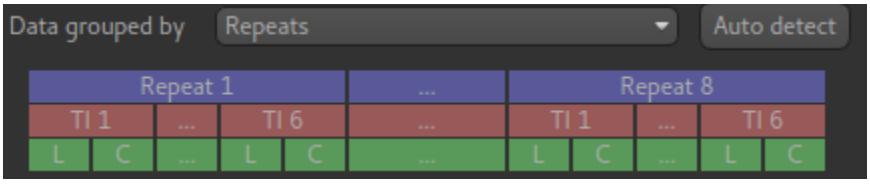
In addition the `Signal Fit` visualisation compares the mean signal from your data with what would be expected for the data grouping you have chosen. In this case they match closely, which is a good check that we have chosen the correct grouping option.



When grouped by repeats, the volume sequence would be as follows:

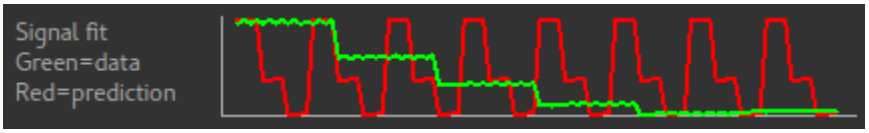
```
1. Tag for first TI
2. Control for first TI
3. Tag for second TI
4. Control for second TI
   ... as above for remaining TIs
11. Repeat of Tag for first TI
12. Repeat of Control for first TI
11. Repeat of Tag for second TI
12. Repeat of Control for second TI
    ... etc
```

And the data structure visualisation looks like this:



Signal fit visualisation - bad fit

In this case the correct grouping order was by TIs, as we saw above. If we select repeats the signal fit will show us that the data do not match what we would expect - this means we have got our grouping option wrong!



Autodetecting the grouping order

The `Auto detect` button tries to guess (based on the closeness of the signal fit) what the best grouping option is for our data. In most cases it will guess correctly, however care should be taken if your data does not fit into one of the standard patterns (see `Custom ordering` below)

Advanced: custom ordering

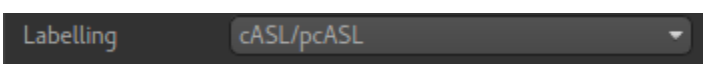
Occasionally, you may encounter ASL data with a different structure. For example if might start with tag images for all TIs and repeats, and then have control images for all TIs and repeats afterwards. In this case you should select `Custom` as the grouping order and enter a string of two or three characters in the text box to define your ordering. The characters should be chosen from:

- `l` for variation in the label (i.e. tag/control or vessel encoding cycles)
- `t` for variation in the TIs/PLDs
- `r` for variation in the repeat number

The characters should be ordered so the first is the *fastest* varying and the last is the slowest varying. For example the two standard 'Grouped by TIs' and 'Grouped by repeats' options would be described by the ordering strings `lrt` and `ltr`. If all the tag images are together and all the control images follow, and within each block the data is grouped by repeats the ordering string would be `trl`.

Labelling

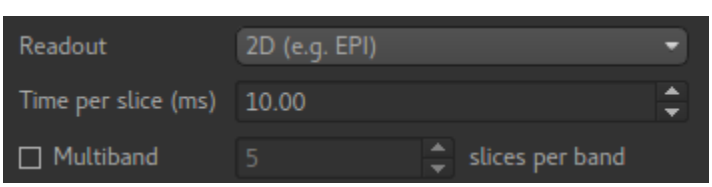
The labelling method is either cASL/pcASL or pASL. In cASL/pcASL, the effective TI for each volume is determined by adding the post-labelling delay (PLD) to the bolus duration. In pASL, the TIs are specified directly.



Readout

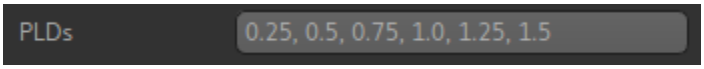
Data acquired with a 3D readout requires no special processing, however if the readout was 2D then each slice will be at a slightly different TI/PLD (the volume TI/PLD in this case is the *initial* TI/PLD).

Selecting 2D readout enables additional options for setting the the delay time per slice so suitable adjustments in the TI/PLD can be made for each slice. It is also possible to specify a multiband readout.

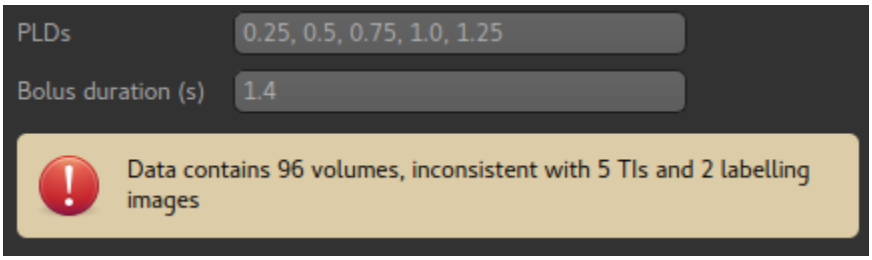


TIs/PLDs

The TIs or PLDs recorded in the ASL data must be specified, with the corresponding bolus durations. Initially data is interpreted as single-TI, however additional TIs can be added by typing their values into the entry box. Values can be separated by commas or whitespace.



If the number of PLDs specified is not consistent with the number of data volumes, a warning is displayed. Here we have removed a PLD so there are only 5 which does not match the data which has 96 volumes.

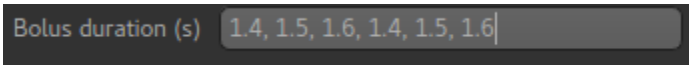


Here we have specified a label-control dataset with 7 PLDs - this means the number of volumes should be a multiple of 14.

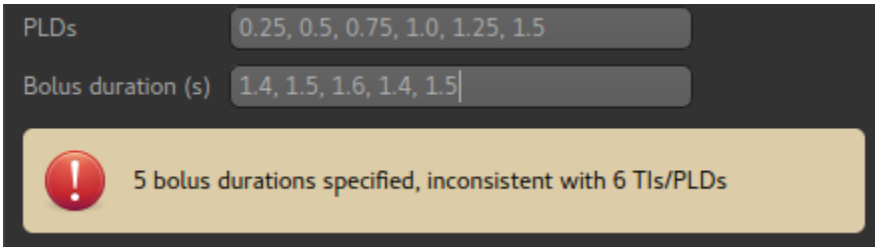
Bolus duration(s)

Most ASL sequences use a single bolus duration whose value should be entered in this box:

It is possible (but unusual) to use a different value for each TI/PLD. In this case a value can be given for each TI/PLD:



The number of values given must match the number of TIs/PLDs:

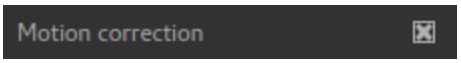


ASL Corrections Tab

Corrections are changes made to the input data before the model fitting is performed. currently four possible sources of corrections are supported:

Motion correction

If enabled this will apply motion correction to the ASL data set using the FSL `MCFLIRT` tool.

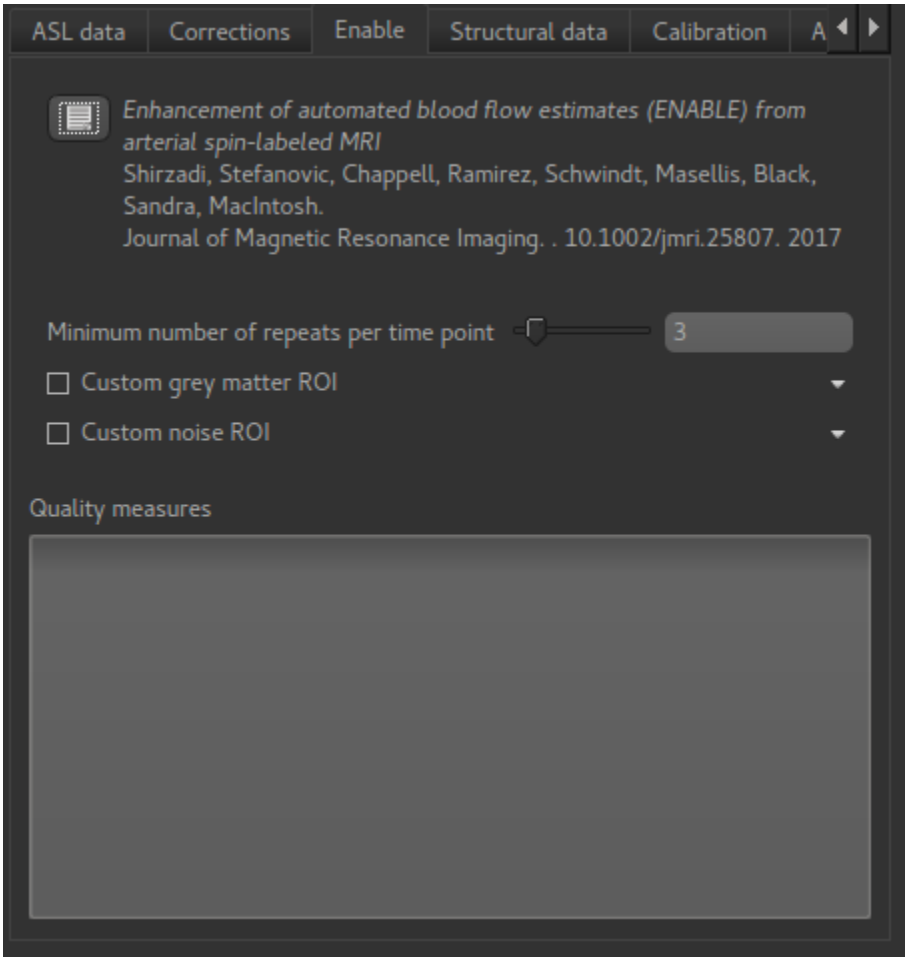


If you prefer, you can do motion correction independently within Quantiphyse (or elsewhere) and disable this option in the ASL processing.

ENABLE volume selection

This uses the ENABLE method to remove 'bad' volumes from the cASL data to improve overall quality. This can be useful if there are a small number of volumes with, for example, major motion artifacts.

When selected, an additional tab appears headed `ENABLE`:



ENABLE options

`Minimum number of repeats per time point`

ENABLE will always leave at least this many repeats of each TI/PLD, so for example if you start out with 8 repeats of each TI/PLD you may end up with 7 repeats of the first TI, 4 of the second, etc. But you will never get just 2 repeats preserved for a TI.

`Custom grey matter ROI`

ENABLE bases its quality measures on signal-noise ratio in grey matter. If you already have a grey matter ROI for your data you can specify it here. Otherwise ENABLE will use the segmentation of the structural data you provide on the `Structural Data` tab.

`Custom noise ROI`

This is an ROI which defines a part of your data to be used to estimate the noise. If you don't specify anything ENABLE will invert the brain mask and use that, which is normally a reasonable choice.

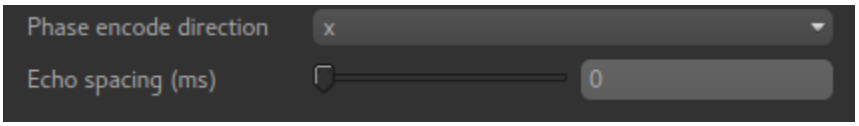
ENABLE Quality measures

The table of quality measures is filled in after ENABLE has run and provides a summary of which volumes in your data were kept and which were removed.

Distortion correction

This corrects for the distortion of the image caused by field inhomogeneities. Two methods are provided: Fieldmap images or a phase-encode reversed (*Blipped*) calibration image.

Generic distortion correction options

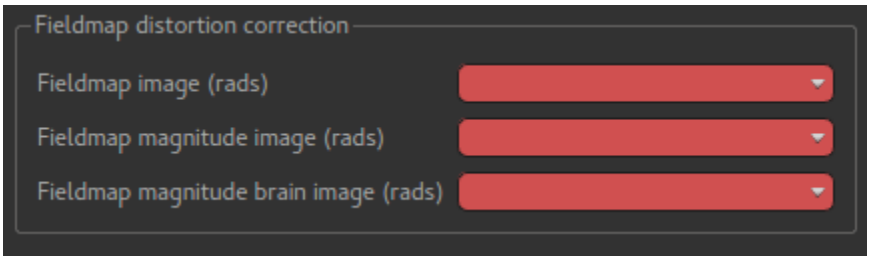


Whichever method you select, you will need to select the phase encoding direction and the echo spacing.

The encoding direction is relative to the raw data axes which normally corresponds to scanner co-ordinates. However you should check the effect of distortion correction visually to ensure that the changes are in the axis that you expect.

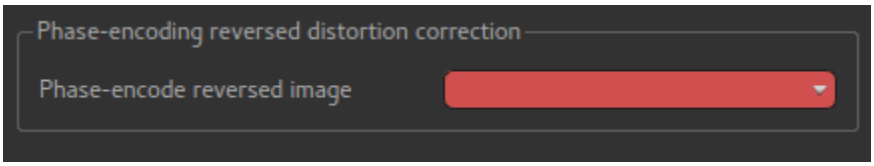
The echo spacing is the true echo spacing (also known as dwell time) and should be specified in ms. The total readout time is this value multiplied by the number of slices (in the phase encoding direction) minus 1.

Distortion correction using fieldmap images



Three images must be provided - you must load them into Quantiphyse first. The first is the fieldmap itself, the second is the magnitude image and the third is the brain extracted version of the second.

Distortion correction using CBLIP images



The CBLIP image must be loaded into Quantiphyse and specified here.

ASL Structural Data Tab

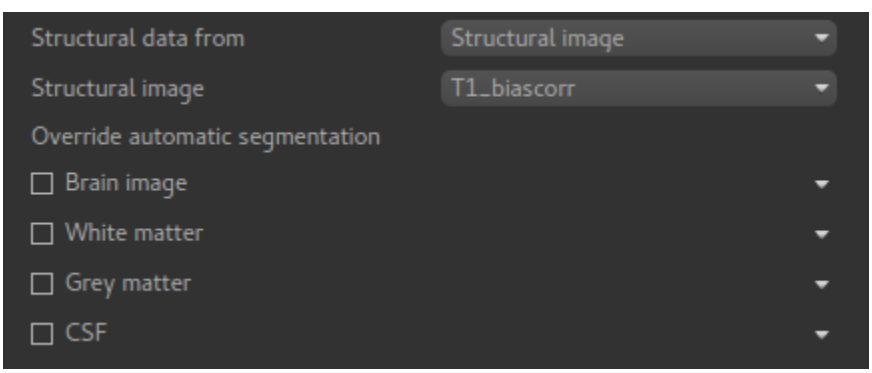
Providing structural data enables the pipeline to do the following:

- Generate a higher quality mask by using the more detailed structural image to identify the brain
- Output data in structural space, enabling it to be easily overlaid onto the structural image
- Automatically segment the structure into tissue types for use in reference region calibration and partial volume correction.

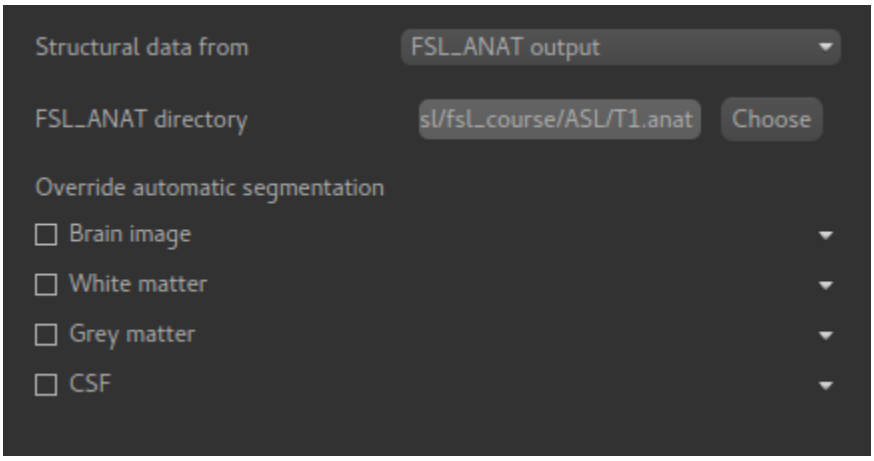
Structural data may be provided in two ways - as a structural image (e.g. T1 weighted) or as an output folder from the `FSL_ANAT` tool. The outcome should be essentially the same but using an `FSL_ANAT` folder is preferable if you have one because it means the segmentation is already done which helps to speed up the pipeline.

In both cases you have the option to override the segementation by providing your own ROIs for different tissue types.

Providing a structural image



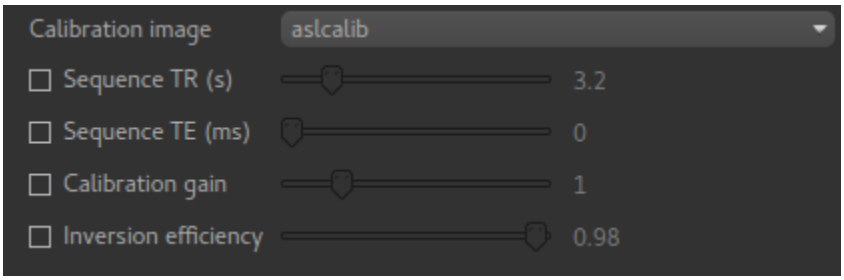
Using an FSL_ANAT output folder



ASL Calibration tab

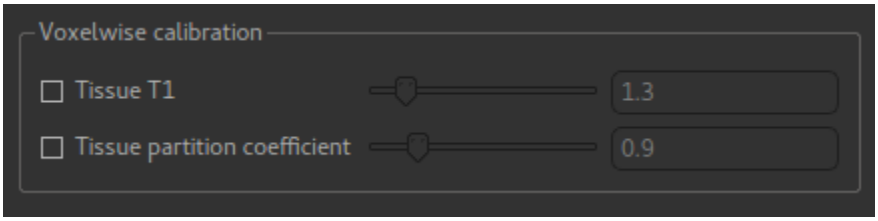
Without calibration, perfusion values from ASL modelling are relative only and cannot be compared between subjects or sessions. By providing calibration data the perfusion can be output in physical units (ml/100g/min) allowing comparisons to be made.

Two calibration methods are provided: *Voxelwise* and *Reference region*. In both cases you must provide a calibration image and may override the default acquisition parameters for this image:



Voxelwise calibration

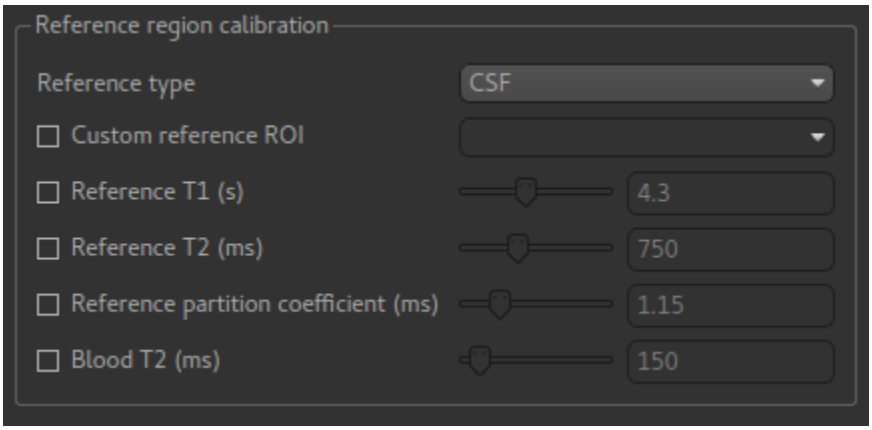
In voxelwise calibration, the calibration image is converted to an M0 image and each voxel in the perfusion data is scaled by the voxelwise M0 value in the M0 image.



This requires two parameters, a notional T1 value for generic 'tissue' and a similar generic partition coefficient. The values given are from standard literature, however they can be modified if needed.

Reference region calibration

In reference region calibration a *single* M0 correction factor is determined for the whole image, by analysing a region of the data containing a single tissue type (typically CSF).

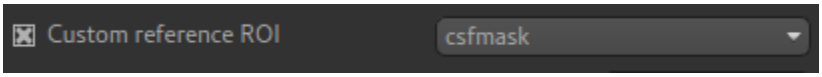


In order to do this, the reference region method requires an ROI which identifies a particular tissue type. By default this is calculated automatically for CSF using the following outline method:

- Obtain the CSF mask from segmentation of the structural image
- Register the structural image to a standard MNI brain image
- Obtain (from standard atlases) the ventricle mask for the standard brain image
- Erode the ventricle mask by 1 voxel and use it to mask the CSF mask from the structural image
- Transform back into structural space and form the reference region mask by conservatively thresholding the ventricle mask at a threshold of 0.9

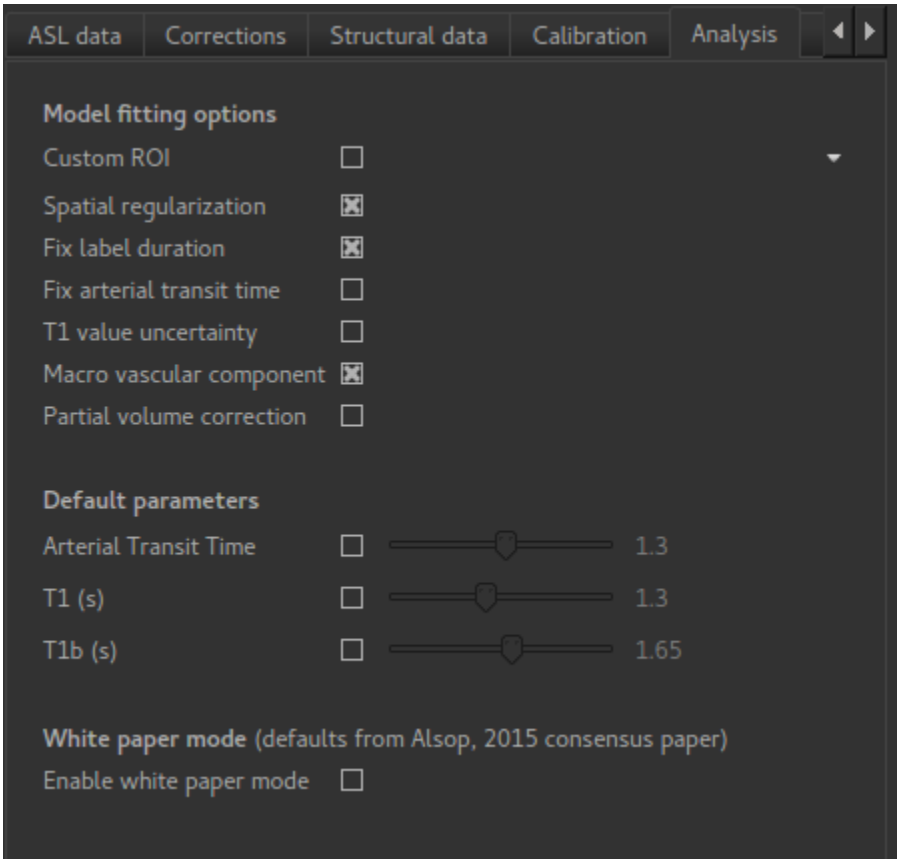
If a tissue type other than CSF is selected, only the first of these steps is performed.

You may prefer to supply a ready made reference ROI. This can be done using the `Custom reference ROI` option:



The T1, T2 and partition coefficient for this tissue type are required for calibration. The default values vary according to what tissue type you have selected - so the ones displayed above are appropriate for CSF. These can be modified as required.

ASL Analysis Tab



The analysis tab contains options for the model fitting part of the pipeline.

Model fitting options

Custom ROI

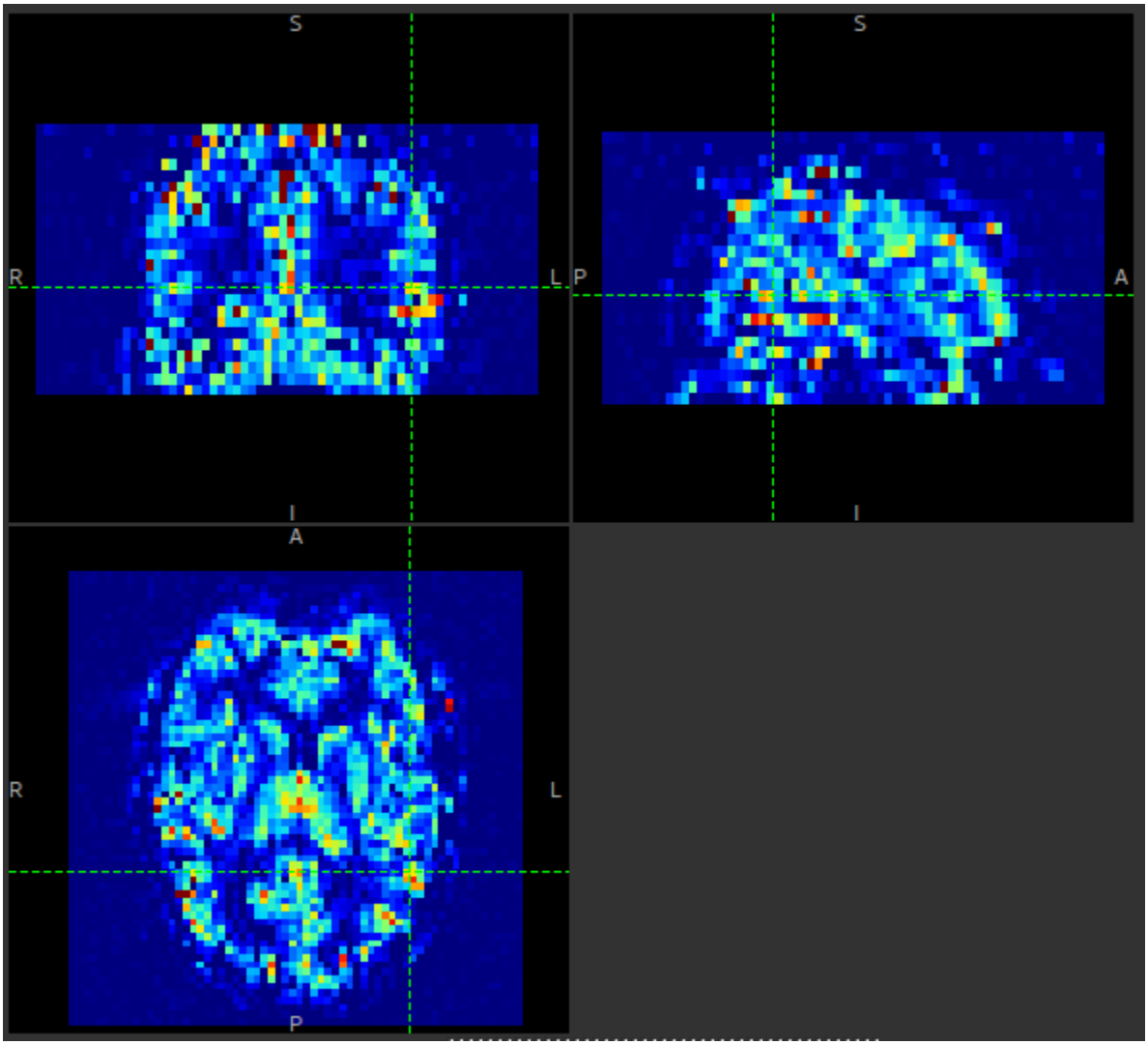
A custom ROI in which to perform the model fitting can be provided - normally this is generated by brain extraction of the structural data (or the ASL data if no structural data is given).

Spatial regularization

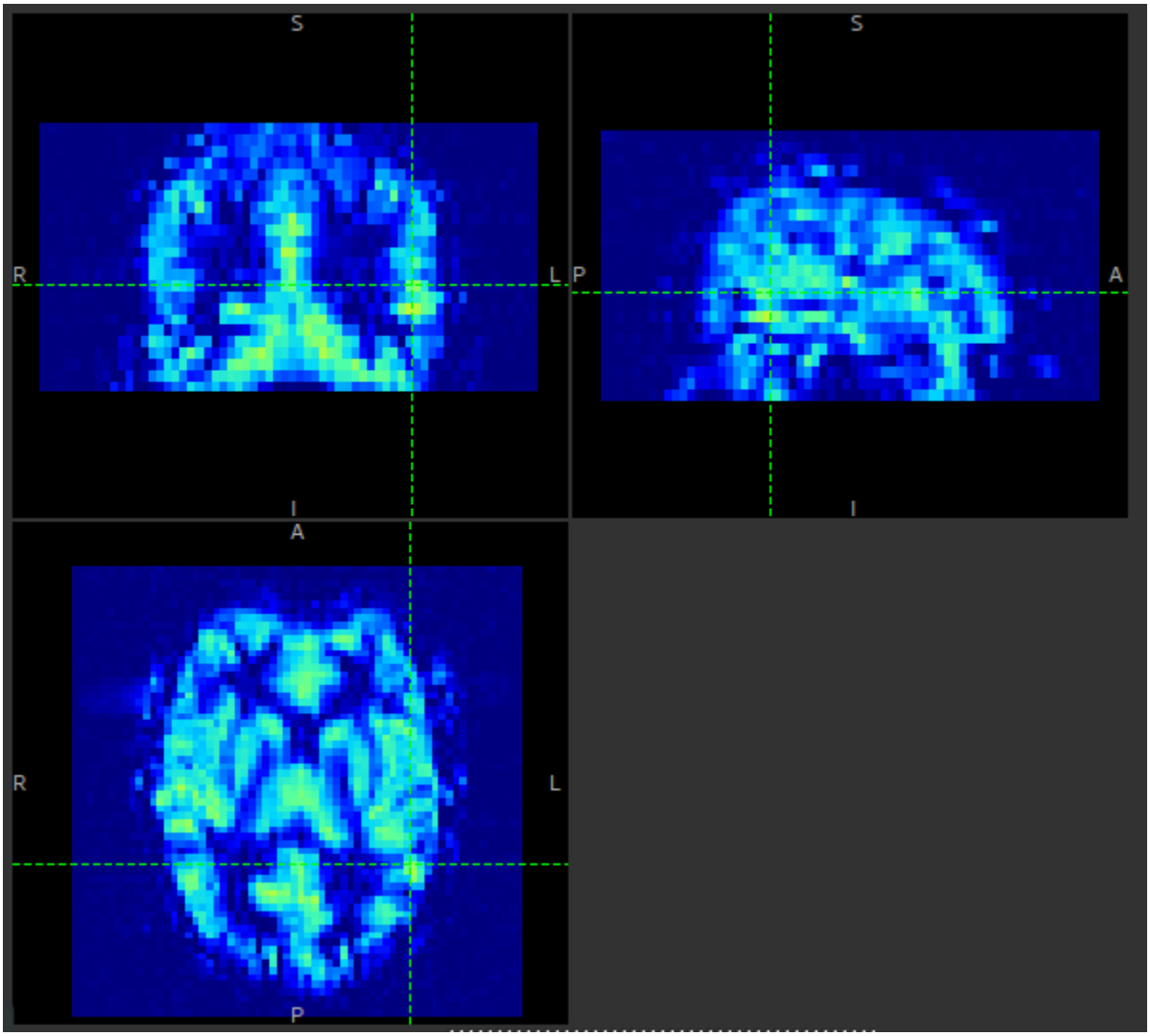
This option will smooth the output data using an adaptive method which depends on the degree of variation in the data. If there is sufficient information in the data to justify fine grained spatial detail, it will be preserved, however if the data is not sufficient this will be smoothed.

The effect is similar to what you would get by applying a smoothing algorithm to the output, however in this case the degree of smoothing is determined by the the variation in the data itself.

An example perfusion map without spatial regularization might look like this:



With spatial regularization turned on, the same data set produced the following perfusion map:



Fix label duration

The label duration (bolus duration) can be allowed some variation to better fit the data. If this option is selected this will not occur. Label duration is fixed by default.

Fix arterial transit time

Similarly to the above, this controls whether the arterial transit time (also known as bolus arrival time) is allowed to vary to fit the data. However, in contrast to the label duration, this is allowed to vary by default with multi-PLD data.

Arterial transit time cannot be accurately estimated with single-delay data.

T1 value uncertainty

This is analagous to the above options but controls whether the T1 value is allowed to vary. By default it is kept constant.

Macro vascular component

Some of the signal in the ASL data will come from labelled blood in arteries as opposed to perfused tissue. This may be a significant contribution in voxels containing a major artery. By adding a macro vascular component this signal can be estimated and separated from the tissue perfusion contribution during the fitting process.

Partial volume correction

If enabled, this will use the GM/WM segmentation to perform an additional modelling step in which the GM and WM contributes will be modelled separately and based on the GM/WM partial volume within each voxel (which will also be modelled as part of the fitting process).

Warning: Partial volume correction adds considerably to the pipeline run time!

Override defaults

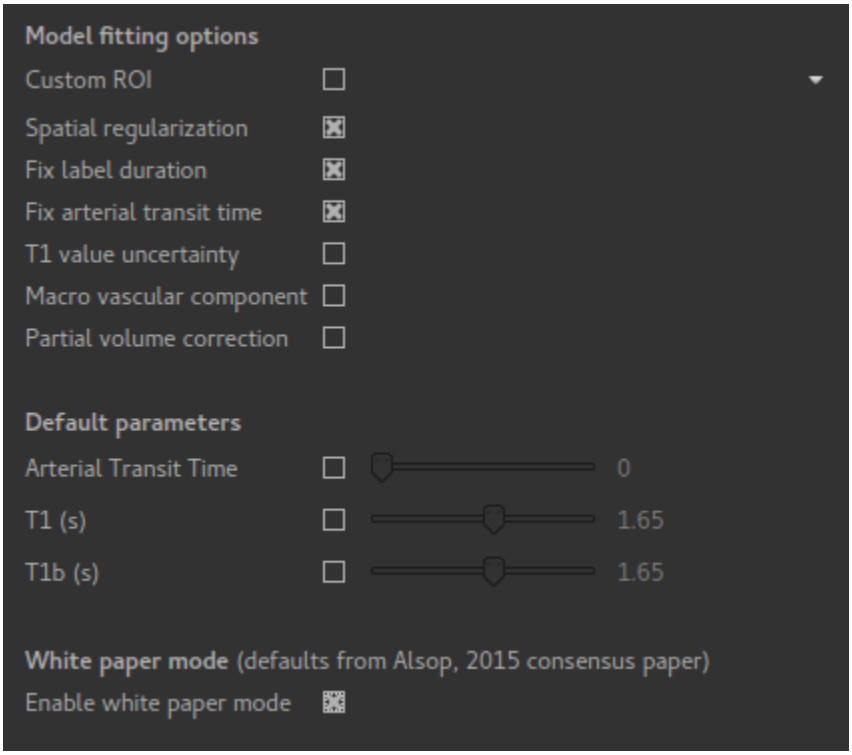
The values given for arterial transit time, T1 and T1b are from the literature, but can be customized if required.

White paper mode

'White paper mode' selects defaults and analysis methods to match the recommendations in Alsop et al (2014)[1]. Specifically this selects:

- Voxelwise calibration
- Arterial transit time of zero (fixed)
- T1 and T1b of 1.65s
- Fixed label duration
- No macrovascular component

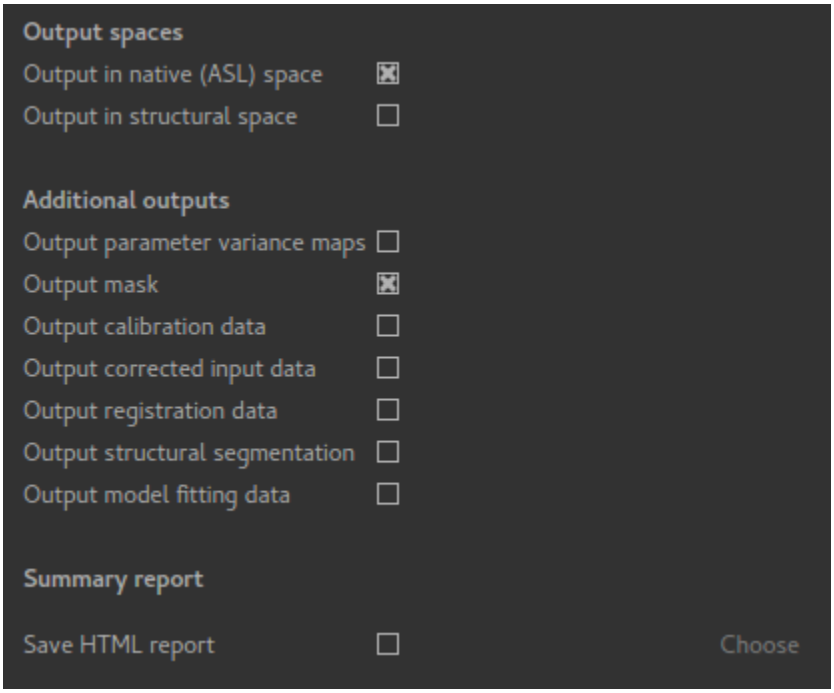
[1] Alsop, D. C., Detre, J. A., Golay, X. , Günther, M. , Hendrikse, J. , Hernandez-Garcia, L. , Lu, H. , MacIntosh, B. J., Parkes, L. M., Smits, M. , Osch, M. J., Wang, D. J., Wong, E. C. and Zaharchuk, G. (2015), Recommended implementation of arterial spin-labeled perfusion MRI for clinical applications: A consensus of the ISMRM perfusion study group and the European consortium for ASL in dementia. Magn. Reson. Med., 73: 102-116. doi:10.1002/mrm.25197



References

ASL Output Tab

This tab controls the output that will be produced.



Output data spaces

Standard data item outputs

The following data items are output:

- `perfusion` Tissue perfusion
- `arrival` Inferred arterial transit time
- `modelfit` Model prediction for comparison with the tag-control differenced data

If `Fix label duration` is *not* specified:

- `duration` Inferred Label duration

If `Fix arterial transit time` is *not* specified:

- `arrival` Inferred arterial transit time

If `Include macro vascular component` *is* specified:

- `aCBV` Macrovascular component

If `Allow uncertainty in T1 values` *is* specified:

- `mean_T_1` Tissue T1 value
- `mean_T_1b` Blood T1 value

If calibration is included, additional calibrated outputs `perfusion_calib` and `aCBV_calib` are also generated.

Data spaces

By default the output is produced in *native* ASL space (i.e. the same space as the input ASL data). These outputs have the suffix `_native`. In addition (or instead of) output can be produced in structural space, in which case the outputs will have a suffix of `_struc`.

Additional outputs

Output parameter variance maps

The Bayesian modelling method used is able to output maps of the estimated parameter variance. This gives a measure of how confident the values in the parameter maps are. These outputs have the suffix `_var`.

Output mask

If selected the mask used to perform the analysis will be output under the name `mask_native`.

Output calibration data

The calibration data would include the reference mask used in reference region calibration and the voxelwise M0 image in voxelwise calibration. These outputs have the suffix `_calib`.

Output corrected input data

This option outputs corrected versions of the input data (ASL and calibration) after motion correction, distortion correction, etc. have been performed. These outputs have the suffix `_corr`.

Output registration data

This option outputs data used as the reference for registration with the suffix `_ref`.

Output structural segmentation

This option outputs the brain extracted and segmented (partial volume and mask) maps from the structural data. These outputs have the suffix `_struc`.

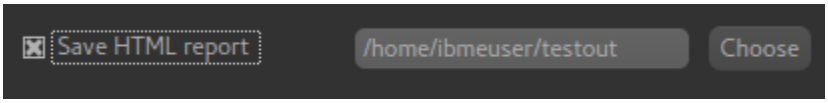
Output model fitting data

This option outputs the full output from the model fitting step. These outputs have the suffix `_fitting`.

> **Warning:** Model fitting is a two-stage multi-step process with a number of intermediate output data files. Selecting this option will generate a large number of output data sets!

Summary report

A summary report in HTML format can be generated - if required you need to select this option and choose an output directory:



Multiphase ASL

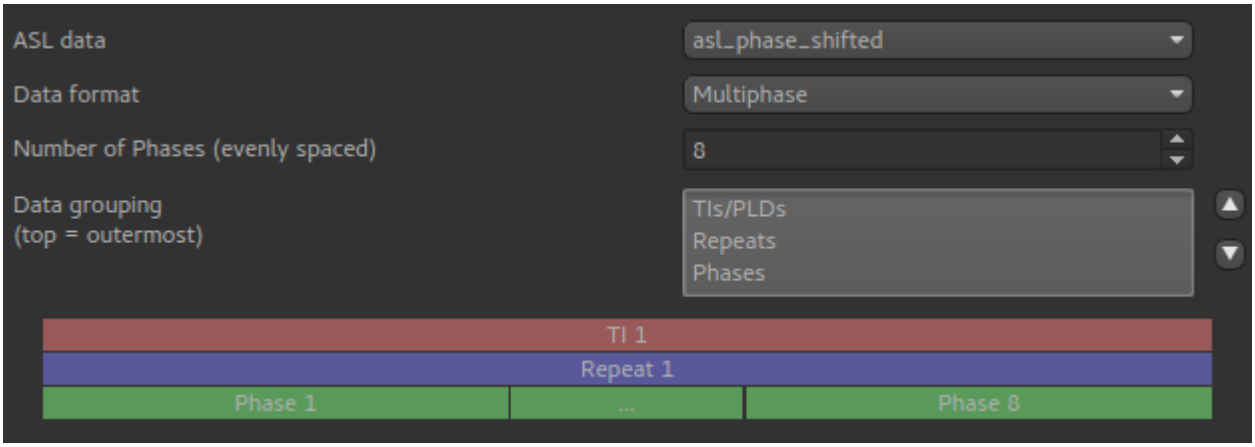
The Multiphase ASL widget is designed for single PLD multiphase ASL data. It performs a model-based fitting process which results in a net magnetisation map which can be treated as differenced ASL data.

Defining the structure

To begin you must define the structure of your ASL data, in the same way as with the other ASL widgets. Multiphase modelling is currently possible only for single PLD data, hence the PLDs entry is not visible. The main things to set are:

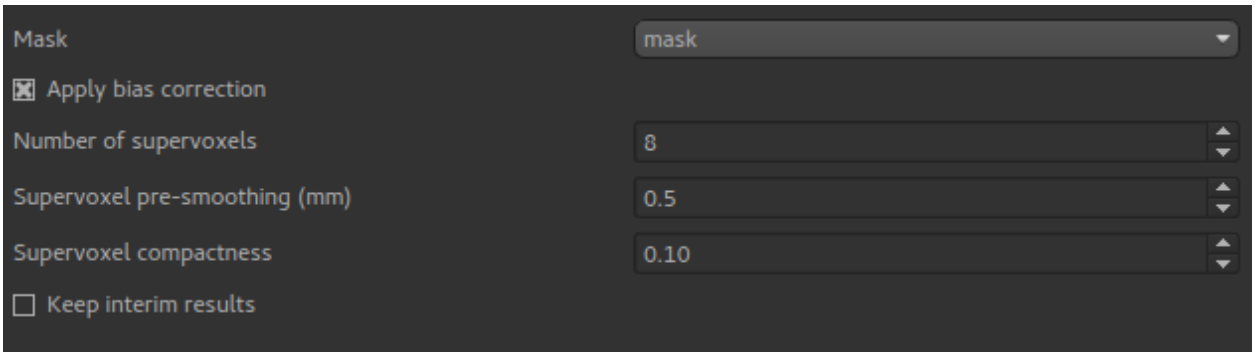
1. The number of phases, which are assumed to be evenly spaced
2. The data ordering, i.e. whether repeats of each phase are together, or whether the full set of phases is repeated multiple times. This is not relevant if the data is single-repeat

This data structure shows a simple single-repeat multiphase data set with 8 phases.



Analysis options

Bias correction



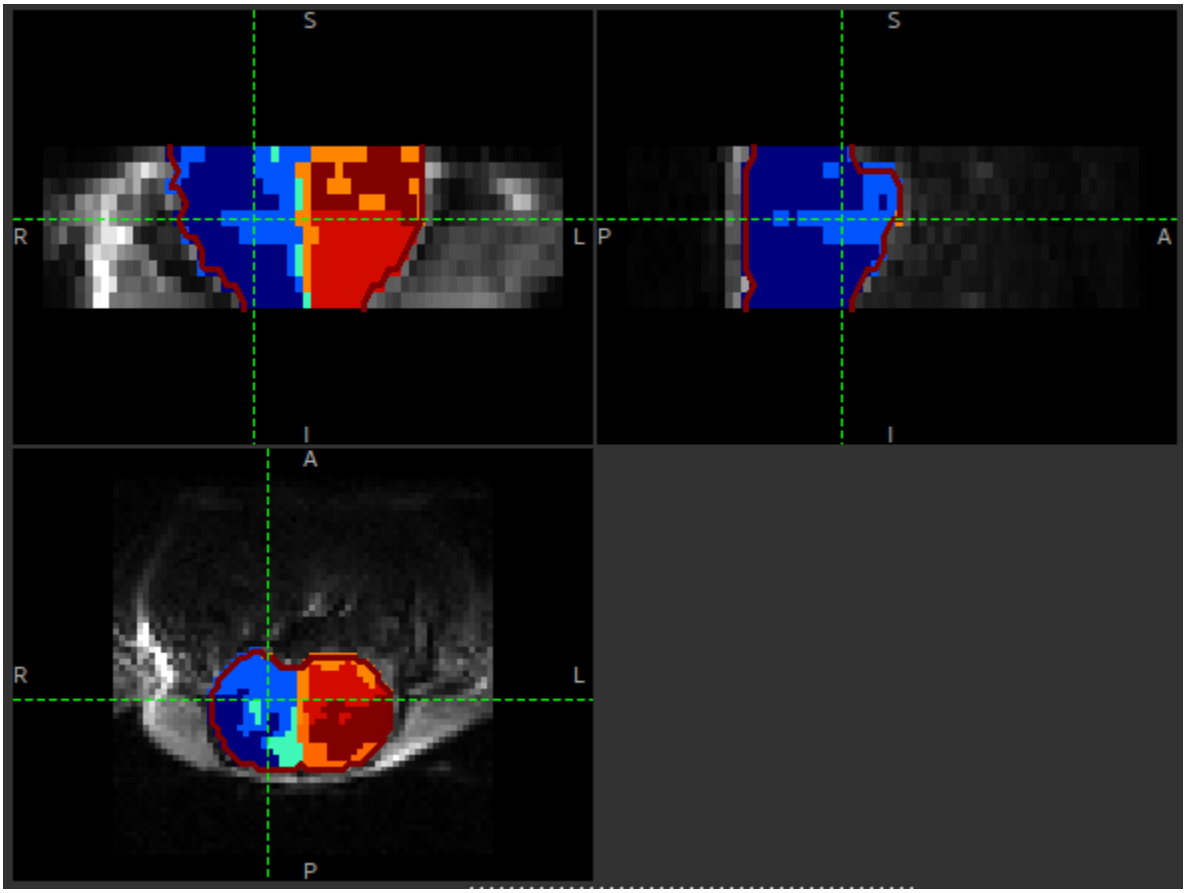
One issue with multiphase modelling is that the estimates of the magnetisation are biased in the presence of noise (in general a lower signal:noise ratio causes an overestimate of the magnetisation. The bias correction option (enabled by default) performs a series of steps to reduce this bias using a supervoxel-based method to first estimate the phase offset in distinct regions of the data. This phase offset is then fixed for the final modelling step, which eliminates the bias which only occurs when magnetisation and phase are both free to vary.

The full set of steps for bias correction are as follows:

1. An initial modelling run is performed, allowing both magnetisation and phase to vary.
2. A set of supervoxels are created based on the output *phase* map only.
3. The signal is averaged in each supervoxel, and a second modelling step is performed. The averaging increases the SNR in each supervoxel to give an improved estimate of the phase in each supervoxel region.
4. A final modelling step is performed with the phase in each supervoxel region fixed to the value determined in step 3. However the magnetisation and overall signal offset are free to vary independently at each voxel (i.e. are not constant within each supervoxel region). With the phase held constant, the biasing effects of noise are eliminated.

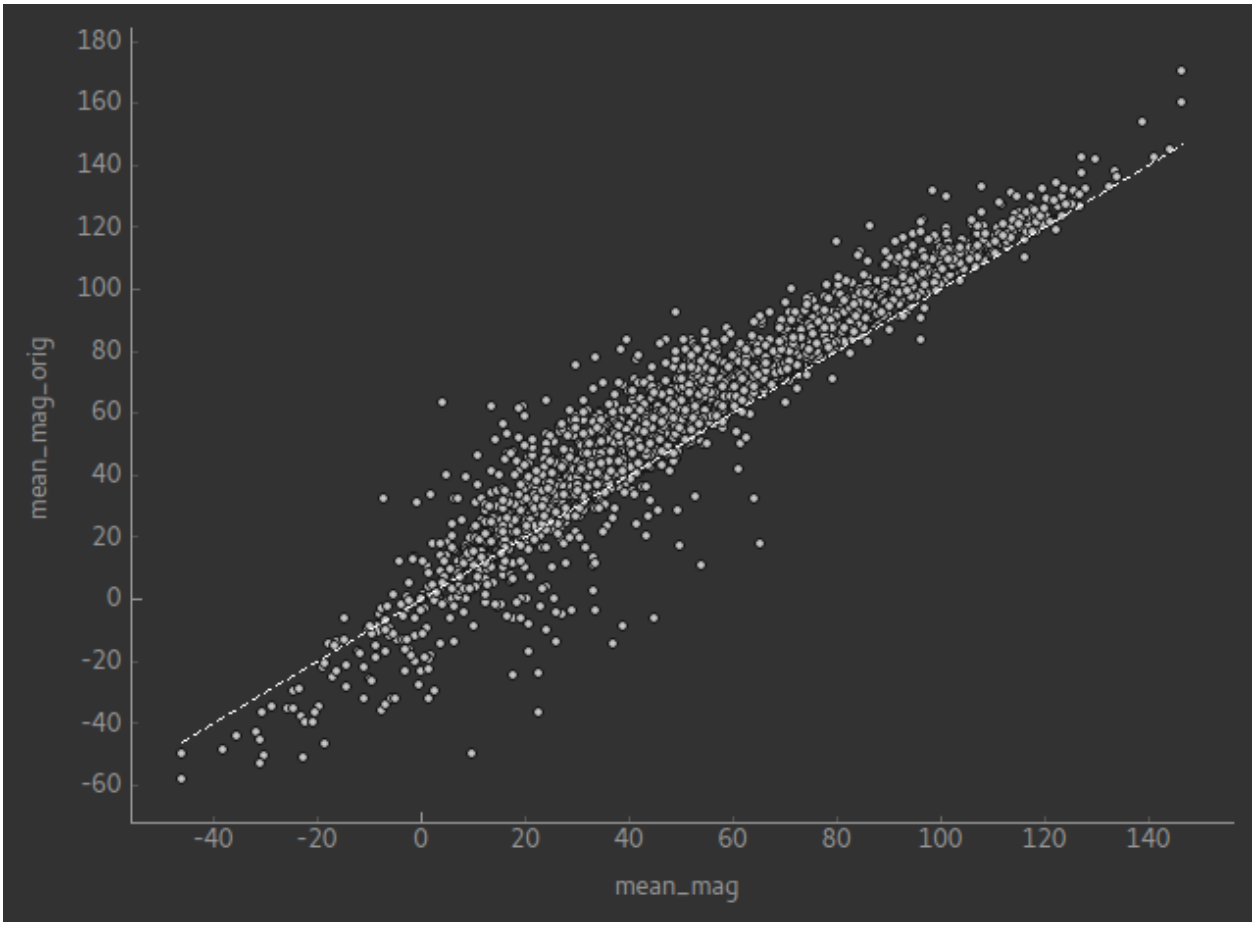
The supervoxels step requires a choice for the total number of supervoxels, the compactness and the degree of pre-smoothing. See the Supervoxels widget for more information about how these options are used.

The justification for assuming a constant phase in distinct regions is that this is related to distinct vascular territories (although there is no assumption of a 1:1 mapping between supervoxels and territories). For example, the following image shows the phase map for a data set with a significant left/right phase difference:



The supervoxels used in the phase mapping can be seen clearly.

This image shows a comparison between the magnetisation map with and without the bias correction. The systematic over-estimation of the magnetisation is clear



By default, only the resulting magnetisation map is returned by the process. By enabling the `Keep interim results` option all the data generated during the process can be preserved. This includes the original uncorrected outputs (suffix `_orig`), the averaged outputs within the supervoxels (suffix `_sv`) and the supervoxels ROI itself (`sv`)

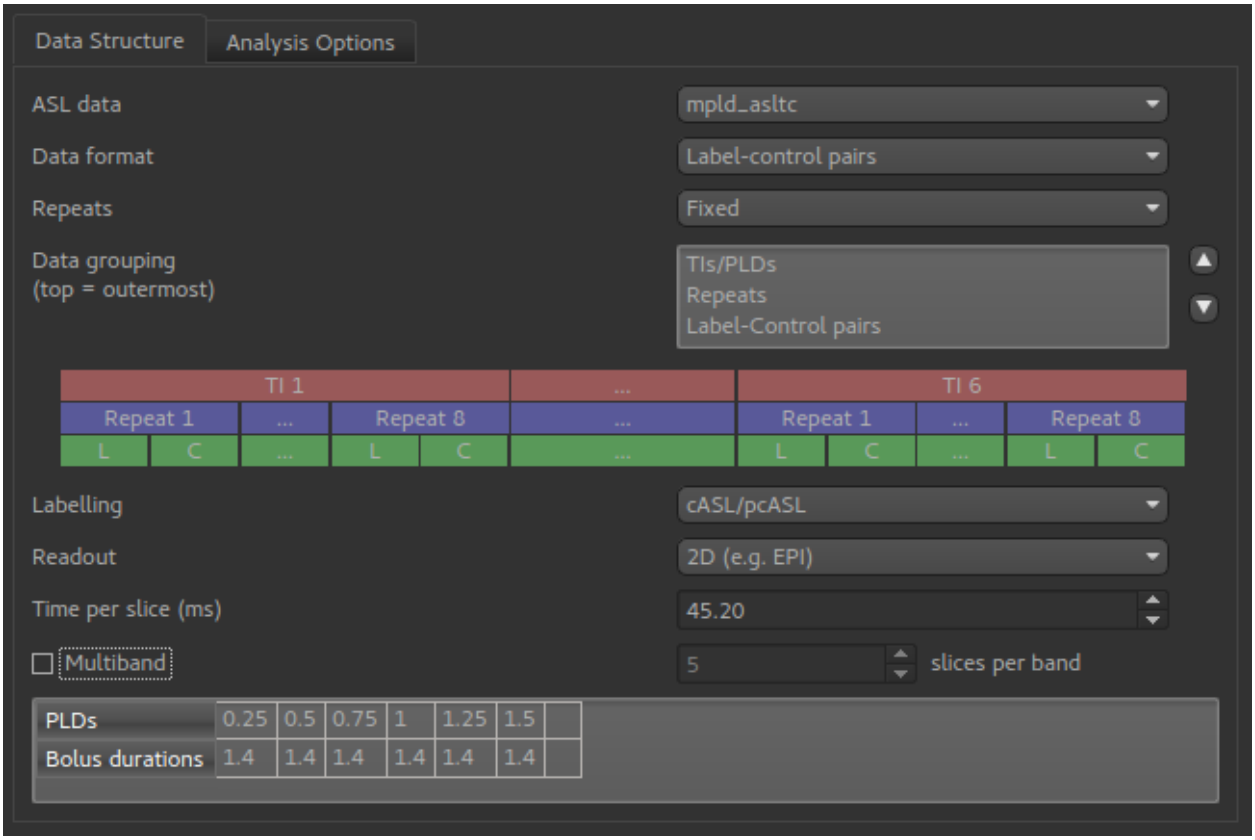
ASL Preprocessing

- *Widgets -> ASL -> Preprocess*

This widget provides simple preprocessing for ASL data.

ASL data structure

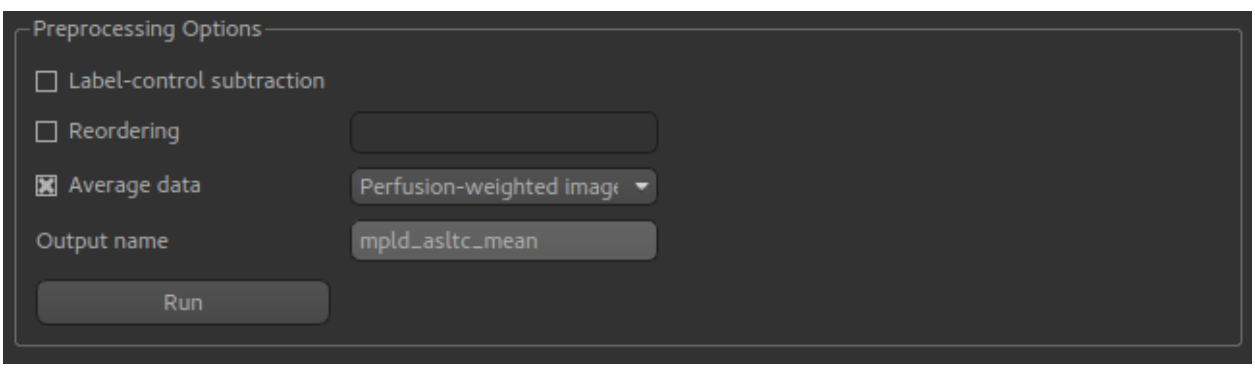
The structure of the ASL data is defined using the ASL structure widget



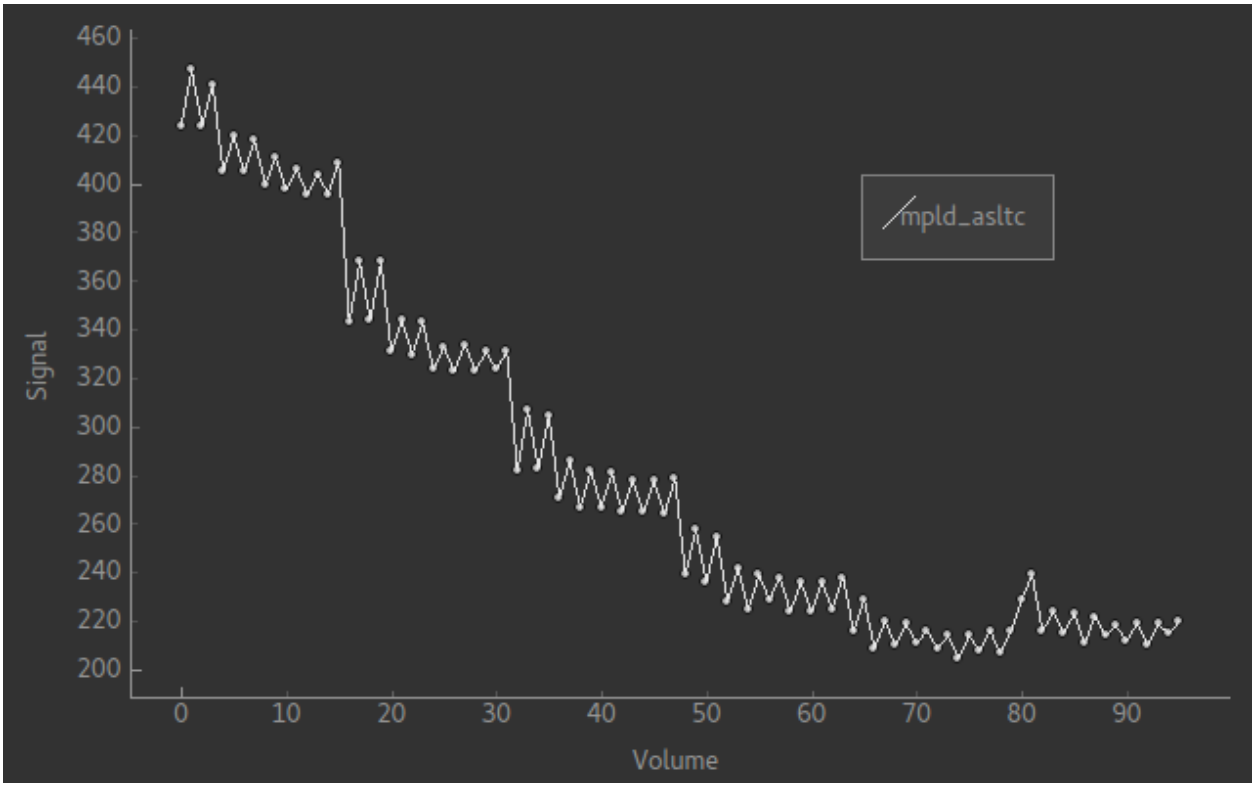
This example shows a multi-PLD PCASL data set with 2D readout.

Preprocessing options

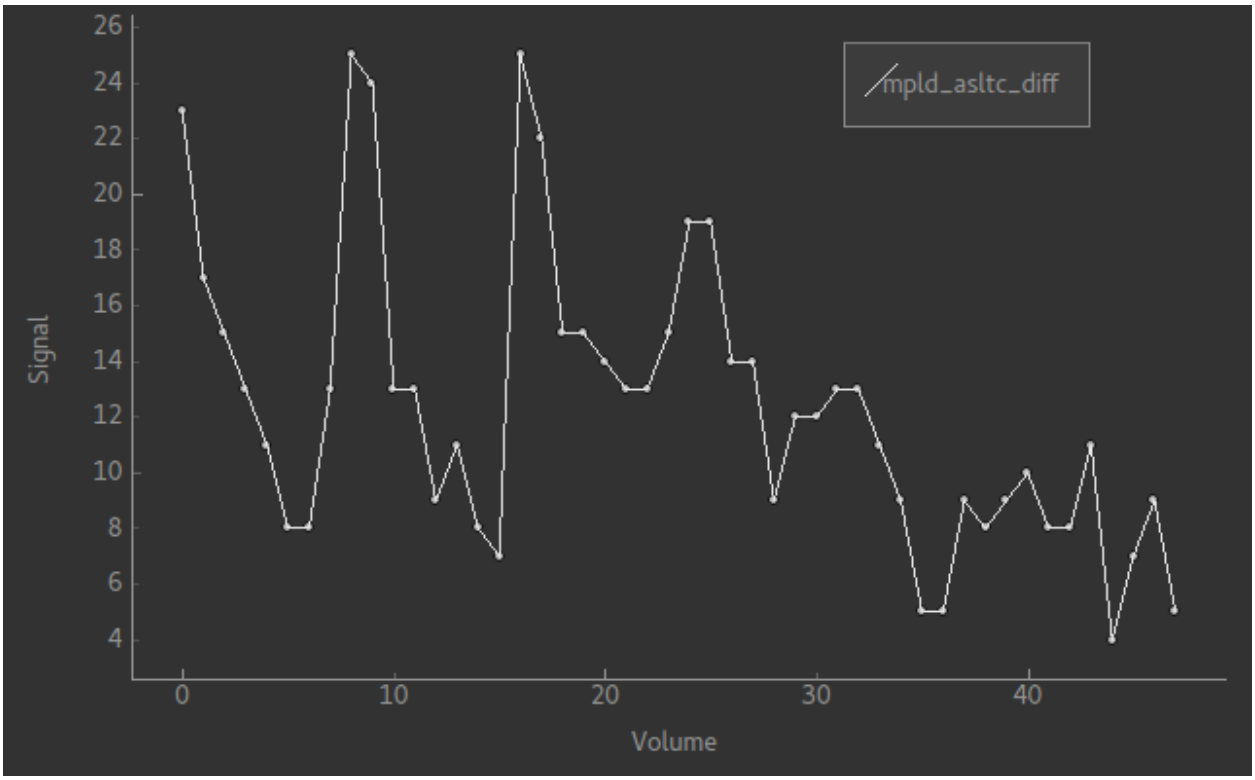
Pre-processing options are shown below the structure definition:



`Label-control subtraction` will convert the data into differenced ASL data, passing on other details of the ASL structure (PLDs, etc) to the output data set. If we view the raw signal using the `Voxel Analysis` widget we see a pattern like this:



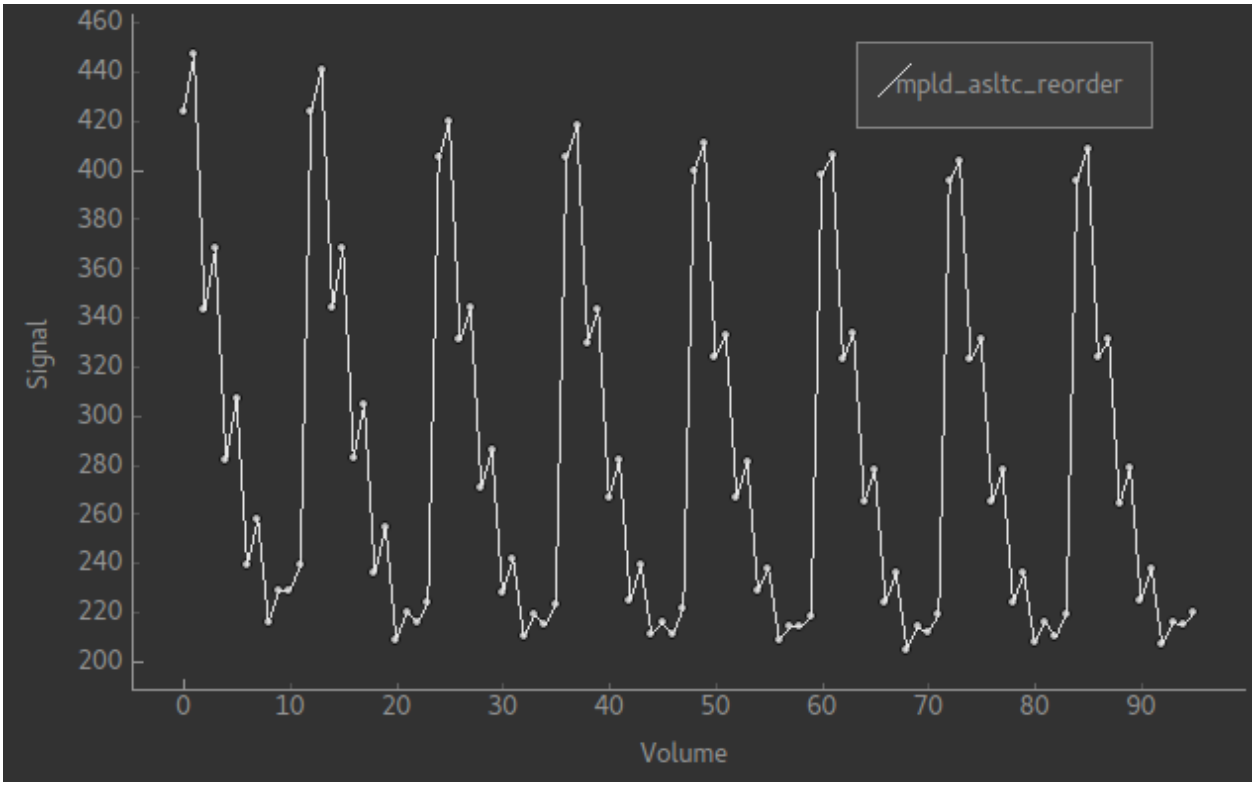
The alternating pattern of the tag-control pairs is clearly visible. After differencing we see the following:



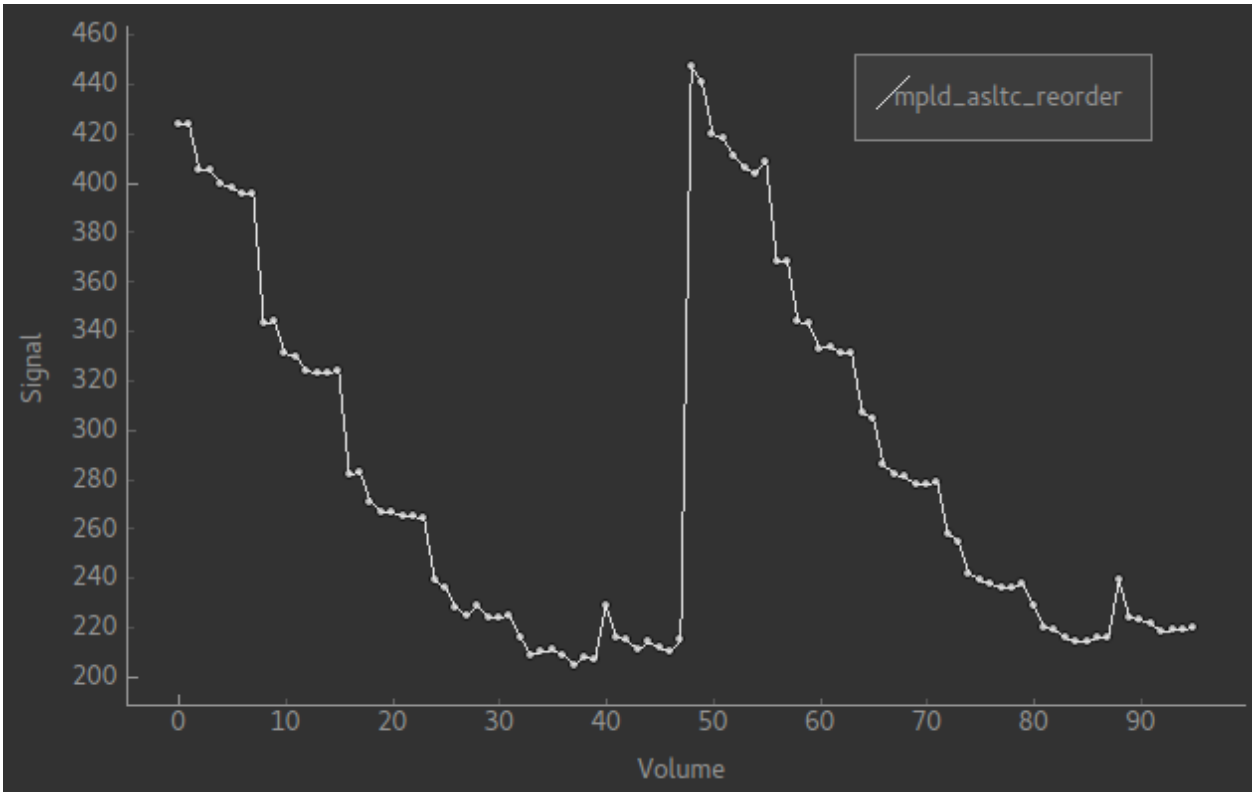
This is harder to read as the data is quite noisy, however note that there are half the number of volumes and the alternating pattern is gone.

`Reordering` changes the grouping order of the ASL data. The re-ordering string consists of a sequence of characters with the innermost grouping first. `p` represents Label-Control pairs, (capital `P` is used for Control-Label pairs), `r` is for repeats and `t` is for TIs/PLDs. For example, the data above is in order `prt` and in the signal pattern above you can see a series of repeat measurements at six PLDs.

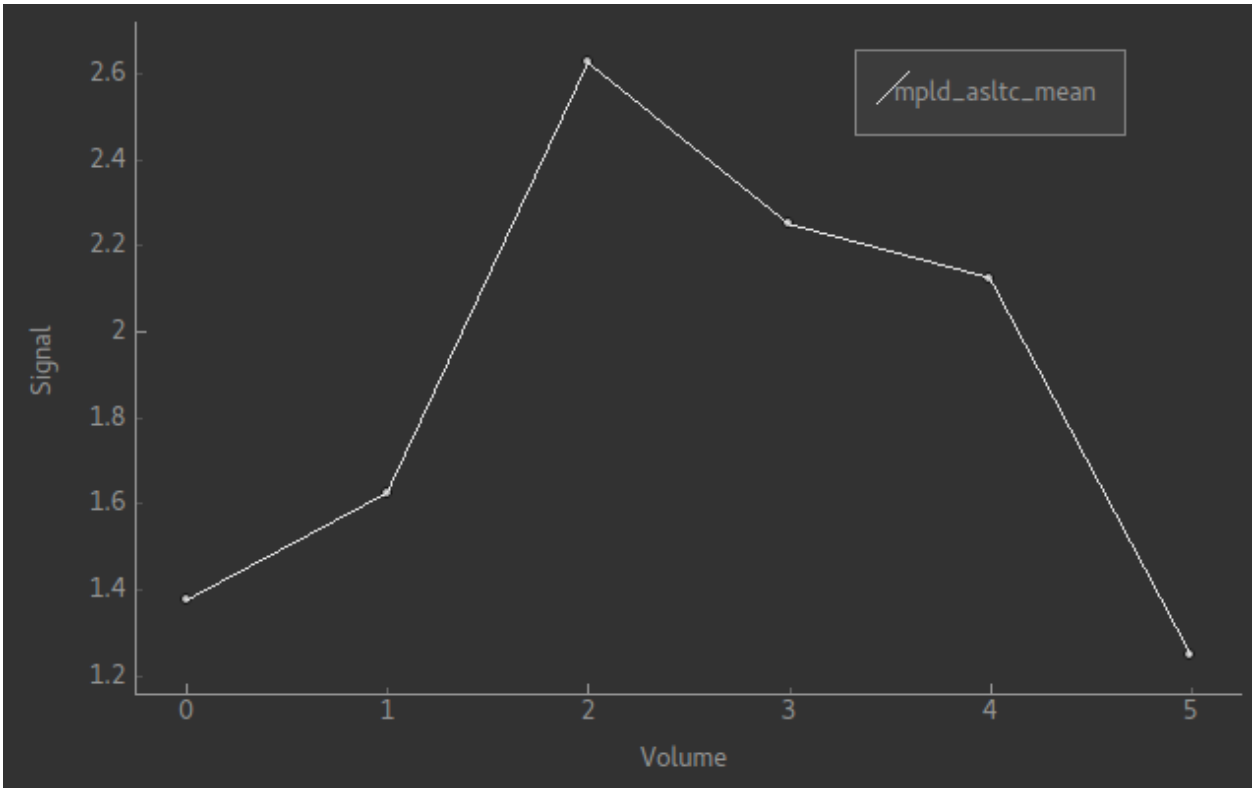
Changing the order to `ptr` will keep the tag/pair alternation but change the data to repeats of sets of all six PLDs:



If you prefer, you could have the original ordering but all the tags first and all the control images afterwards. That would be an ordering of `rtp` and looks like this:

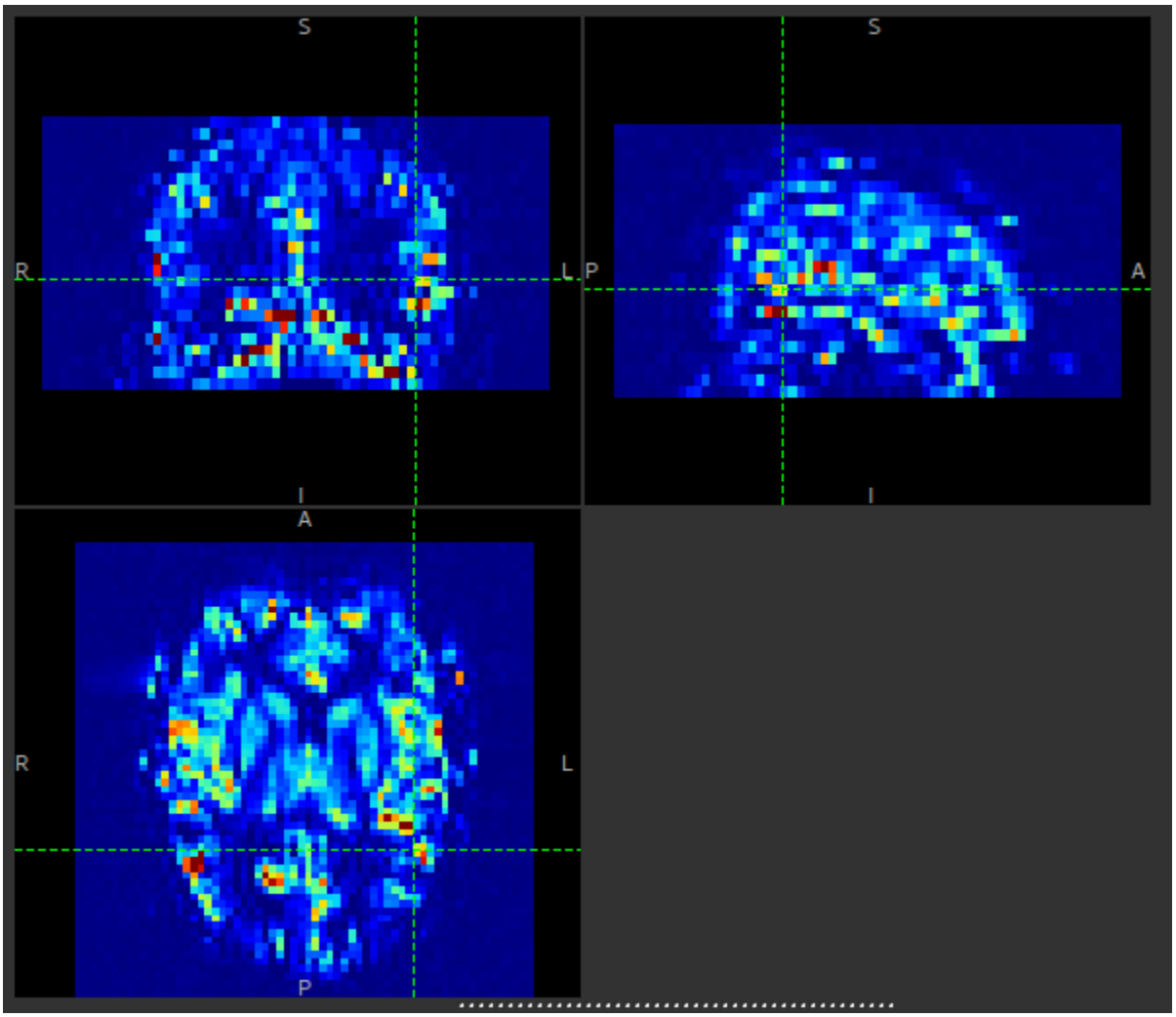


`Average data` can do one of two things. It can take the mean over the repeats at each PLD, resulting in a new multi-PLD data set. This enables the ASL signal to be seen more clearly in multi-repeat data, for example this is a typical signal from the data shown above:



This shows the ASL signal more clearly than the noisy differenced data signal shown above.

Alternatively, for a multi-PLD data set you can take the mean over all PLDs as well to generate a `Perfusion-weighted image`. This is usually similar to the output from model fitting and provides a quick way to check the perfusion. The result of this operation is always a single-volume image. For the data above it looks as follows:



This can be compared to the output shown at the bottom of the ASL model fitting page.

5.2.3 Publications

The following publications are useful citations for various features of the ASL processing pipeline:

- **Bayesian inference method**: *Chappell MA, Groves AR, Whitcher B, Woolrich MW. Variational Bayesian inference for a non-linear forward model. IEEE Transactions on Signal Processing 57(1):223-236, 2009.*
- **Spatial regularization**: *A.R. Groves, M.A. Chappell, M.W. Woolrich, Combined Spatial and Non-Spatial Prior for Inference on MRI Time-Series , NeuroImage 45(3) 795-809, 2009.*
- **Arterial contribution to signal**: *Chappell MA, MacIntosh BJ, Donahue MJ, Gunther M, Jezzard P, Woolrich MW. Separation of Intravascular Signal in Multi-Inversion Time Arterial Spin Labelling MRI. Magn Reson Med 63(5):1357-1365, 2010.*

- **Partial volume correction**: *Chappell MA, MacIntosh BJ, Donahue MJ,Jezzard P, Woolrich MW. Partial volume correction of multiple inversion time arterial spin labeling MRI data, Magn Reson Med, 65(4):1173-1183, 2011.*

5.3 QuantiCEST

- *Widgets -> CEST -> QuantiCEST*

This widget provides CEST analysis using the Fabber Bayesian model fitting framework.

5.3.1 Tutorials

The following tutorial was presented at BCISMRM and provides a walkthrough of a CEST analysis:

QuantiCEST Tutorial

Introduction

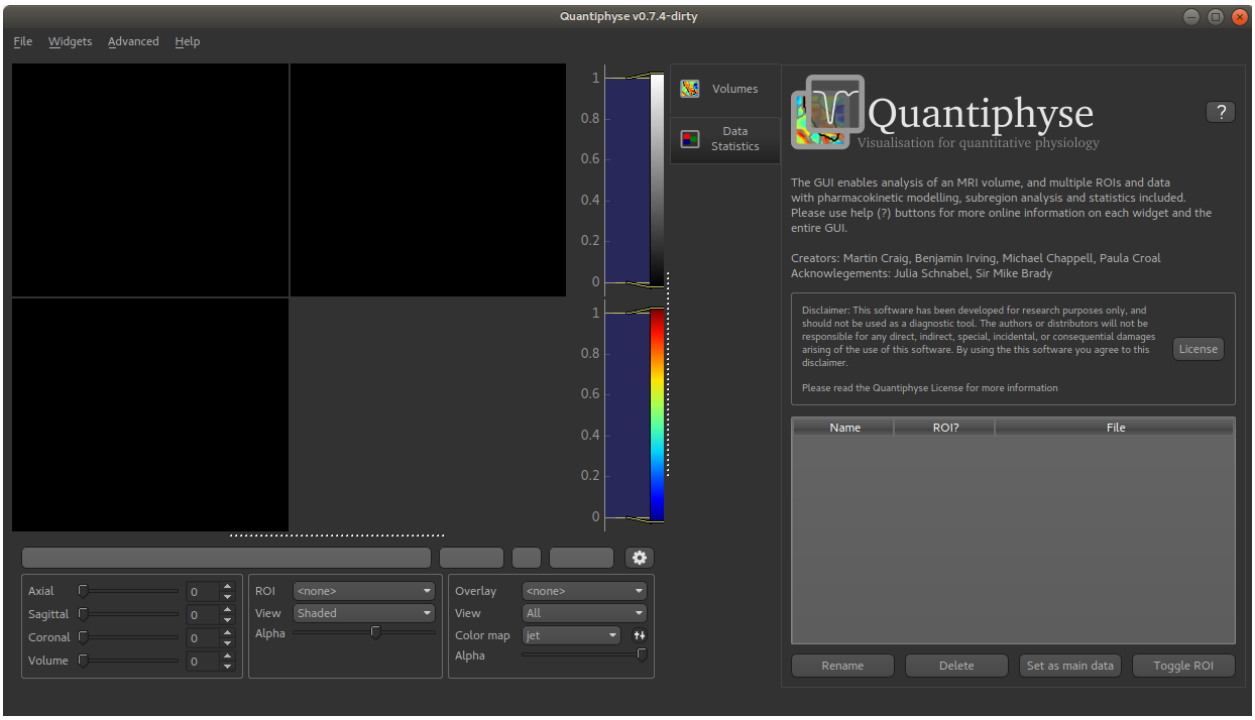
This example aims to provide an overview of Bayesian model-based analysis for CEST[1] using the QuantiCEST widget[2] available as part of Quantiphyse[3]. Here, we work with a preclinical ischaemic stroke dataset using continuous wave CEST[4], however the following analysis pipeline should be applicable to both pulsed and continuous wave sequences acquired over a full Z-spectrum.

Basic Orientation

Before we do any data modelling, this is a quick orientation guide to Quantiphyse if you've not used it before. You can skip this section if you already know how the program works.

Start the program by typing `quantiphyse` at a command prompt, or clicking on the Quantiphyse icon in the menu or dock.

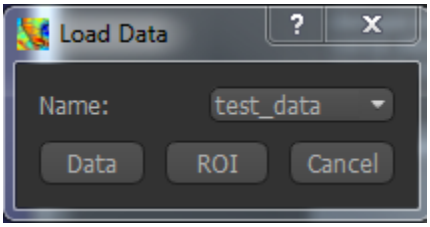
[1] Chappell et al., Quantitative Bayesian model-based analysis of amide proton transfer MRI, Magnetic Resonance in Medicine, 70(2), (2013).
[2] Croal et al., QuantiCEST: Bayesian model-based analysis of CEST MRI. 27th Annual Meeting of International Society for Magnetic Resonance in Medicine, #2851 (2018).
[3] www.quantiphyse.org
[4] Ray et al., Investigation into the origin of the APT MRI signal in ischemic stroke. Proc. Int. Soc. Magn. Reson. Med. 25 (2017).



Loading some CEST Data

If you are taking part in an organized practical workshop, the data required may be available in your home directory, in the `fsl_course/CEST` folder. If not, an encrypted zipfile containing the data can be downloaded below - you will be given the password by the course organizers:

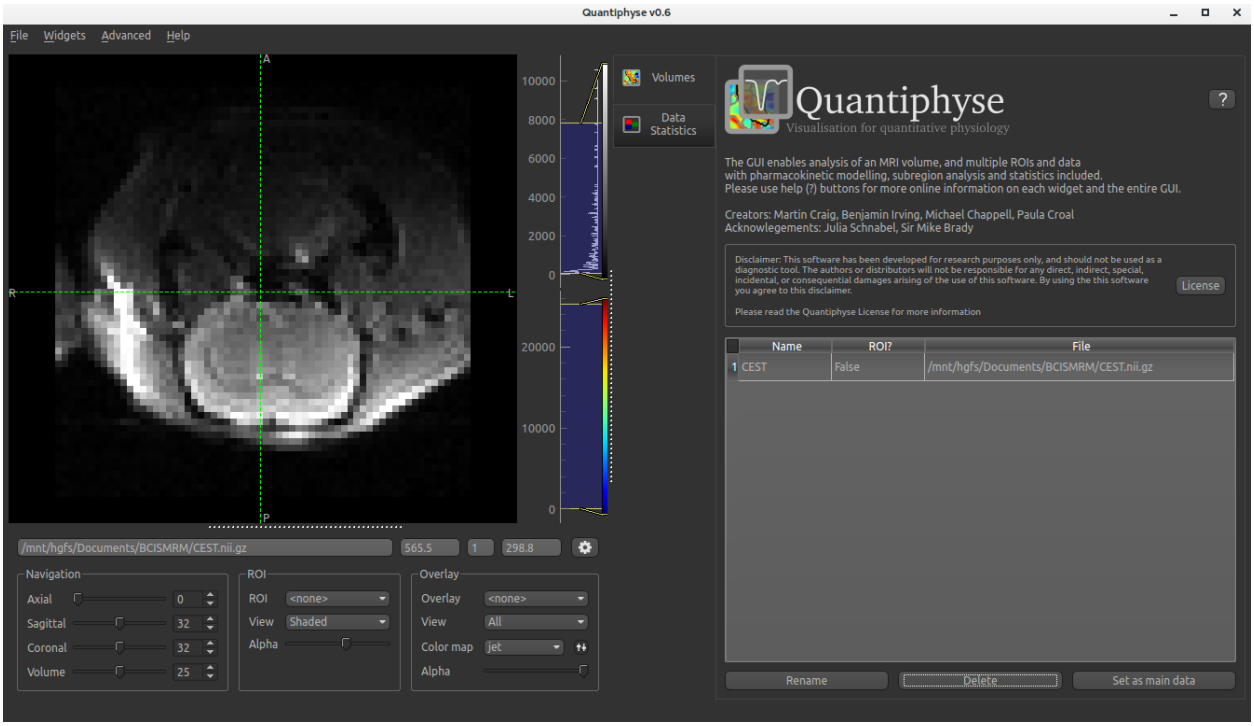
- Self extracting Windows archive
- Encrypted 7zip archive for Unix



Files can be loaded in NIFTI or DICOM format either by dragging and dropping in to the view pane, or by clicking `File` -> `Load Data`. When loading a file you should indicate if it is data or an ROI by clicking the appropriate button when the load dialog appears. Load the following data file:

- `CEST.nii.gz`

The data should appear in the viewing window.



Note: If your single slice CEST NIFTI file is in 3D format rather than 4D, you may need to select `Advanced Options` when loading data and `Treat as 2D multi-volume`.

Image view

The left part of the window normally contains three orthogonal views of your data. In this case the data is a 2D slice so Quantiphyse has maximised the relevant viewing window. If you double click on the view it returns to the standard of three orthogonal views - this can be used with 3D data to look at just one of the slice windows at a time.

- Left mouse click to select a point of focus using the crosshairs
- Left mouse click and drag to pan the view
- Right mouse click and drag to zoom
- Mouse wheel to move through the slices
- Double click to 'maximise' a view, or to return to the triple view from the maximised view.

View and navigation controls

Just below the viewer these controls allow you to move the point of focus and also change the view parameters for the current ROI and overlay.

Widgets

The right hand side of the window contains 'widgets' - tools for analysing and processing data. Three are visible at startup:

- `Volumes` provides an overview of the data sets you have loaded
- `Data statistics` displays summary statistics for data set
- `Voxel analysis` displays timeseries and overlay data at the point of focus

Select a widget by clicking on its tab, just to the right of the image viewer.

More widgets can be found in the `Widgets` menu at the top of the window. The tutorial will tell you when you need to open a new widget.

For a slightly more detailed introduction, see the *Getting Started* section of the User Guide.

Pre-processing

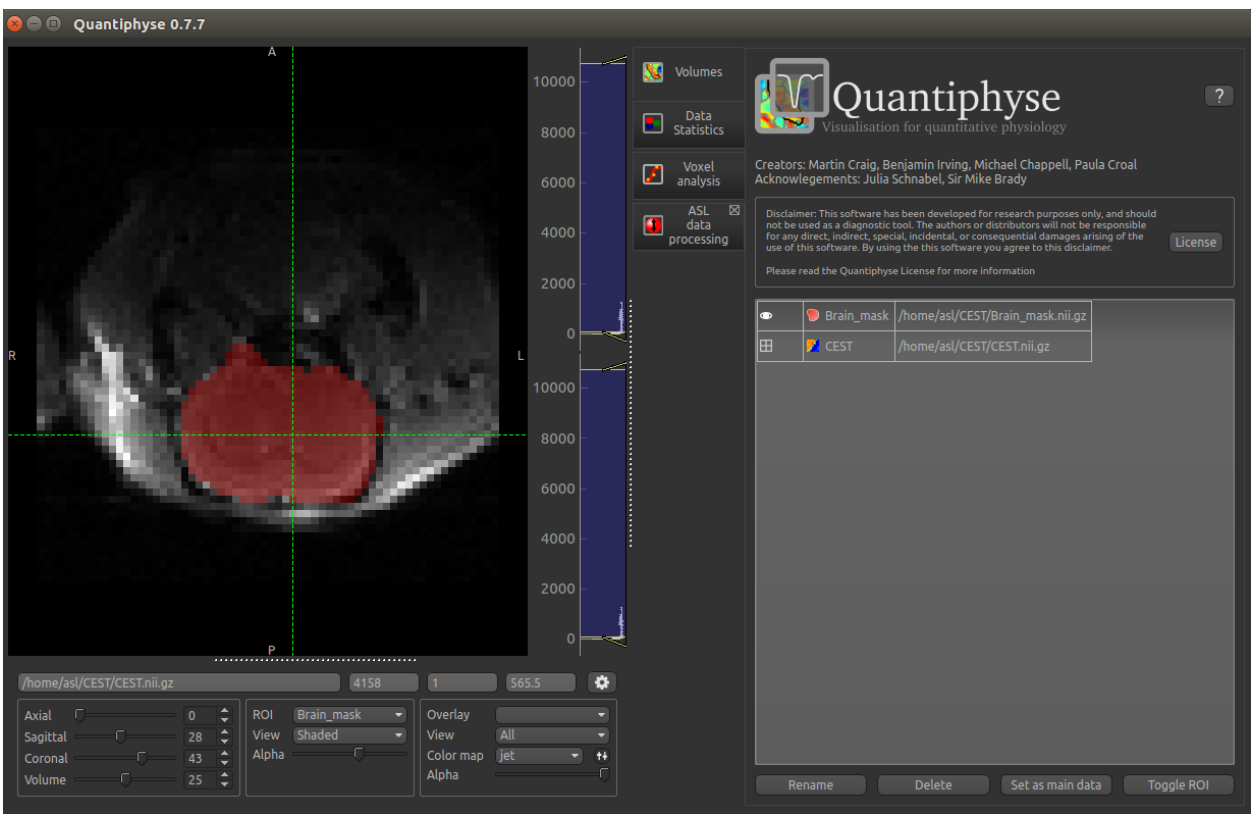
Brain Extraction

For clinical data, we recommend brain extraction is performed as a preliminary step using FSL's BET tool[5], with the `-m` option set to create a binary mask. You can also do this from within Quantiphyse using the FSL integration plugin. It is strongly recommended to include a brain ROI as this will decrease processing time considerably.

In this case we have preclinical data for which BET is not optimised, so we have prepared the brain mask in advance in the following file:

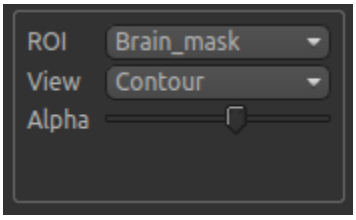
- `Brain_mask.nii.gz`

Load this data set via the `File` menu, and his time select `ROI` as the data type. Once loaded, it will show up in the `ROI` dropdown under the viewing pane, and will also be visible as a red shaded region on the CEST data:



[5] S.M. Smith. Fast robust automated brain extraction. Human Brain Mapping, 17(3):143-155, 2002.

When viewing the output of modelling, it may be clearer if the ROI is displayed as an outline rather than a shaded region. To do this, select `Contour` from the `View` options below the ROI selector:



Note: If you accidentally load an ROI data set as `Data`, you can set it to be an ROI using the `Volumes` widget (visible by default). Just click on the data set in the list and click the `Toggle ROI` button.

Motion Correction

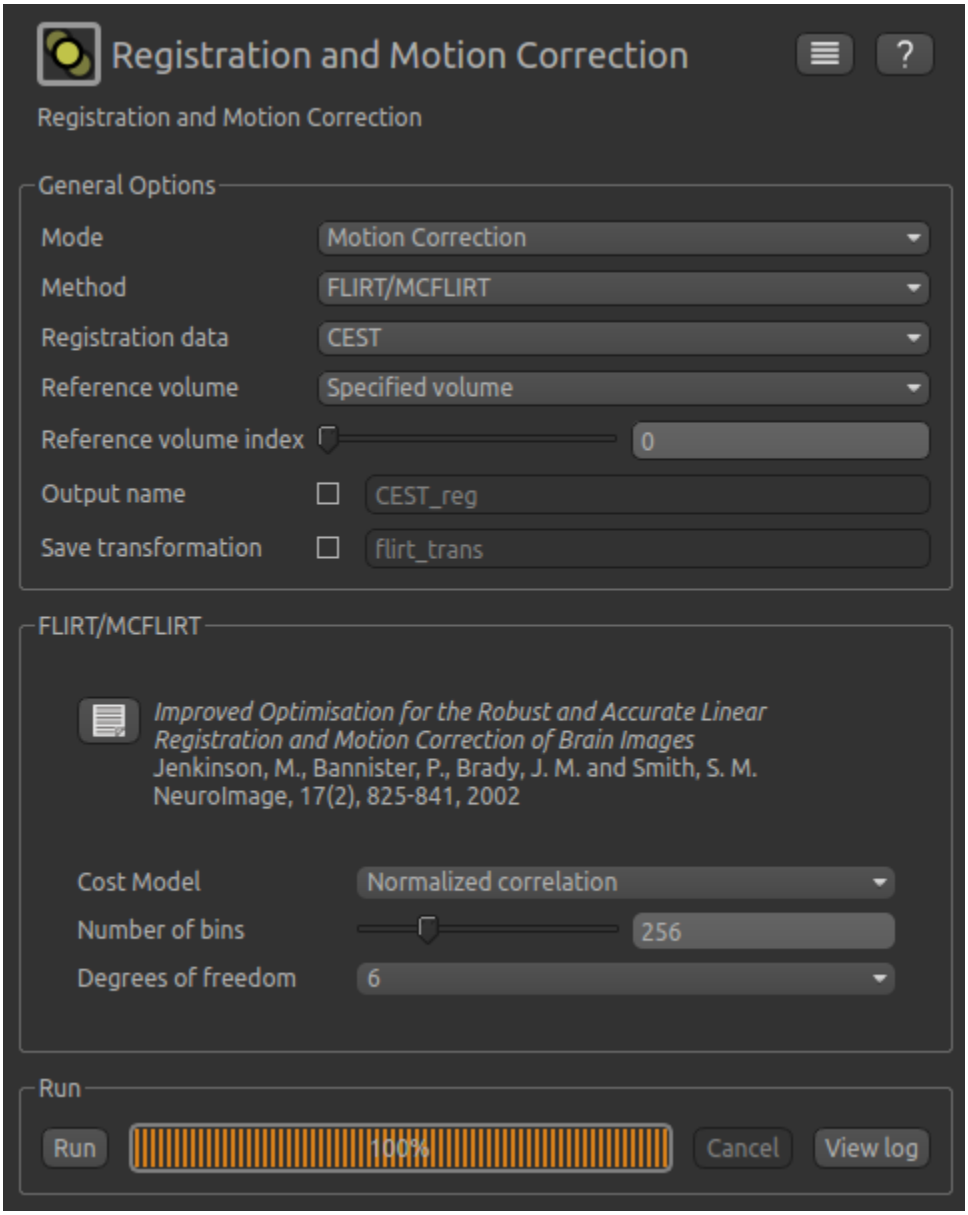
Note: If you prefer you can skip this step - motion correction does not improve this data significantly.

Motion correction can be implemented using FSL's MCFLIRT tool within Quantiphyse, or beforehand using FSL. To run within Quantiphyse, select `Widgets` -> `Registration` -> `Registration`.

To run motion correction on the data, you need to:

- Set the registration mode to `Motion Correction`
- Ensure the method is set to `FLIRT/MCFLIRT`
- Select `CEST` as the `Moving data`
- Select the reference volume as `Specified volume`.
- For CEST data, you probably want the motion correction reference to be an unsaturated image, so we have set `Index of reference volume` to 0 to select the first image in the CEST sequence.
- Set the output name to `CEST_moco`

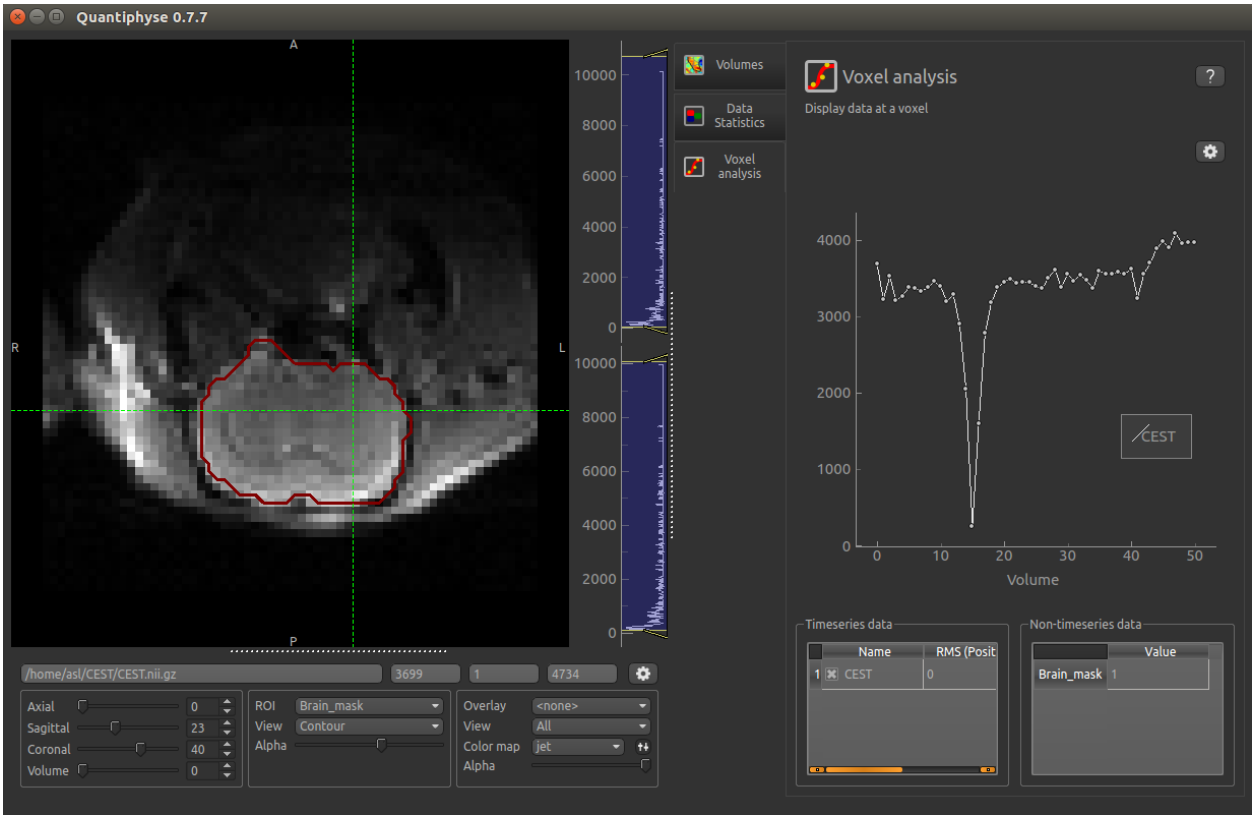
The resulting setup should look like this:



Click `Run` to run the motion correction. The output in this case is not much different to the input as there was not much motion in this data, however if you switch between `CEST` and `CEST_moco` in the `Overlay` selector (below the image view) you may be able to see slight differences.

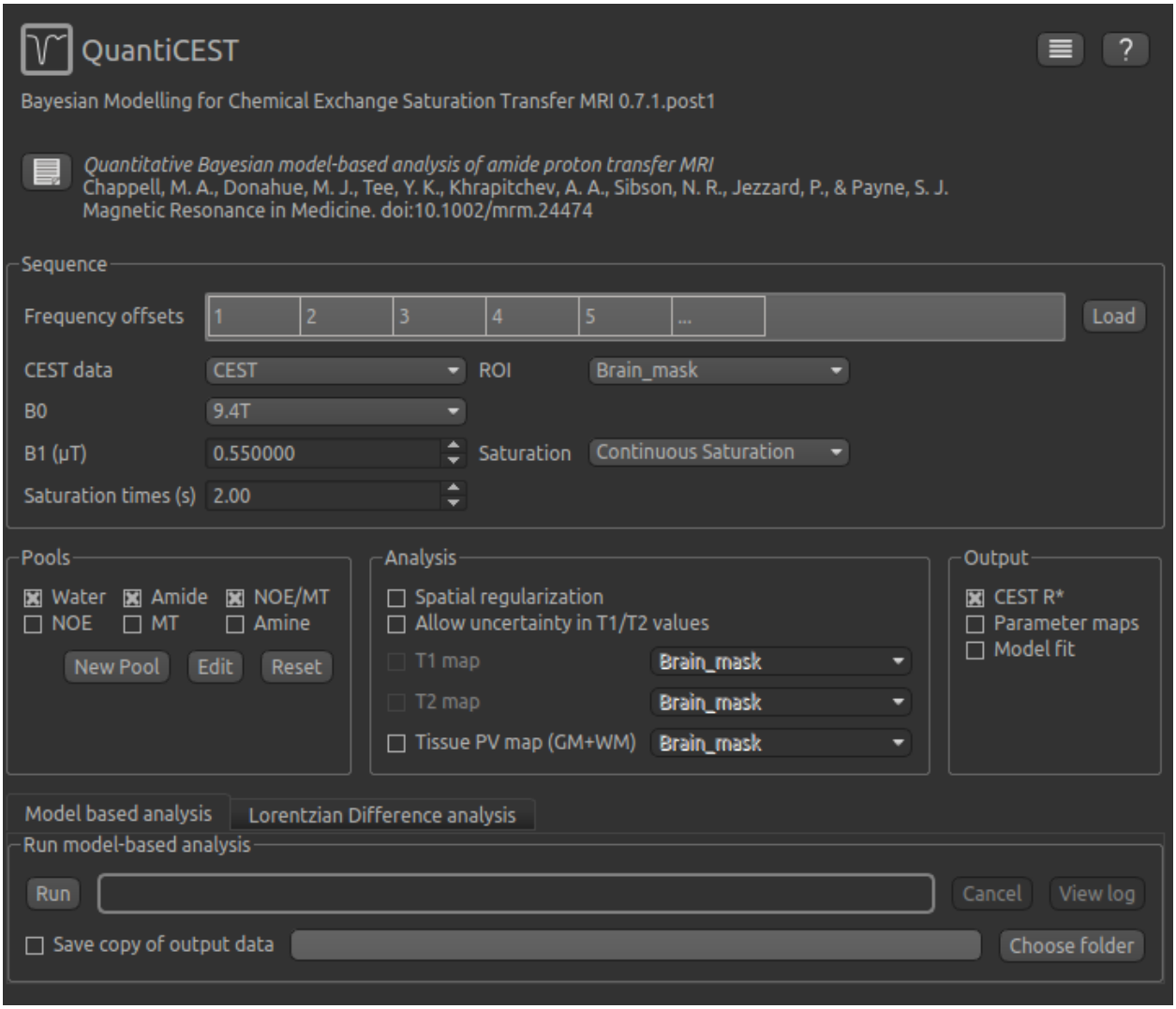
Visualising Data

Select the `Voxel Analysis` widget which is visible by default to the right of the viewing window. By clicking on different voxels in the image the Z-spectra can be displayed:



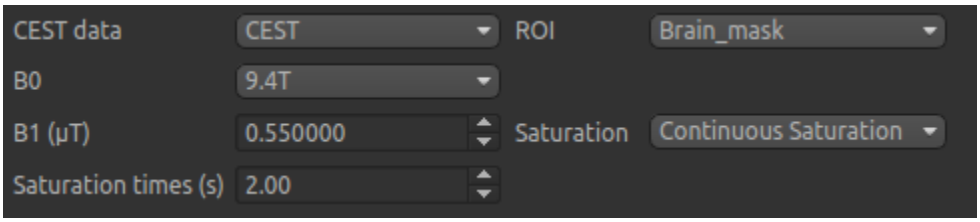
Bayesian Model-based Analysis

To do CEST model analysis, select the QuantiCEST tool from the menu: `Widgets -> CEST -> QuantiCEST`. The widget should look something like this:



Data and sequence section

To begin with, make sure the `CEST` data set is selected as the CEST data, and the `Brain_mask` ROI is selected as the ROI.



The B0 field strength can be selected as 3T for clinical and 9.4T for preclinical studies. This selection varies the pool defaults. If you choose `Custom` as the field strength as well as specifying the value you will need to adjust the pool defaults (see below).

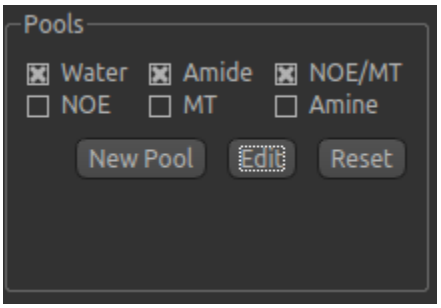
In this case the acquisition parameters do not need altering, however in general you will need to specify the B1 field strength, saturation method and saturation time for your specific setup.

Next we will specify the frequency offsets of your acquisition - this is a set of frequences whose length must match

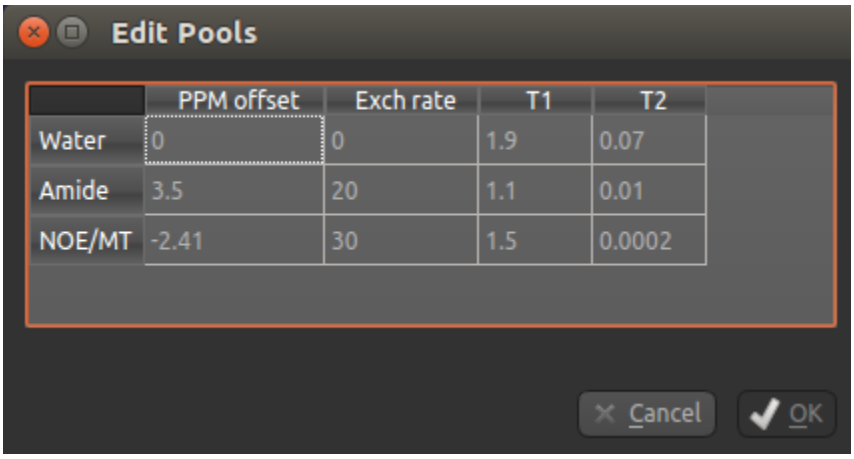
the number of volumes in the CEST data. You can enter them manually, or if they are stored in a text file (e.g. with one value per row) you can click the `Load` button and choose the file.

For this tutorial we have provided the frequency offsets in the file `Frequency_offsets.txt`, so click `Load`, select this file and verify that the values are as follows:

Pool specification



In general, a minimum of three pools should be included in model-based analysis. We provide some of the most common pools to include, along with literature values for frequency offset, exchange rate, and T1 and T2 values for the field strengths of 3T and 9.4T. The data for the pools we have selected can be displayed by clicking the `Edit` button:



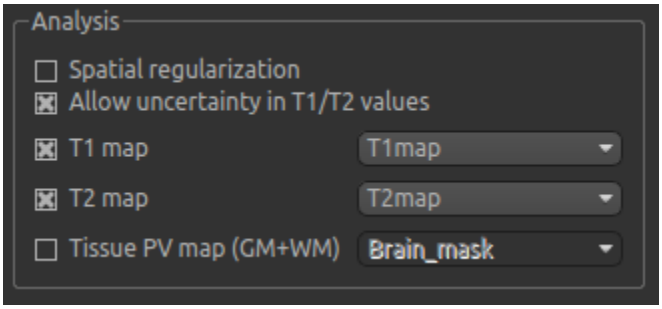
You can also use this dialog box to change the values, for example if you are using a custom field strength. The `Add` button can also be used if you want to use a pool that isn't one of the ones provided.

Analysis section

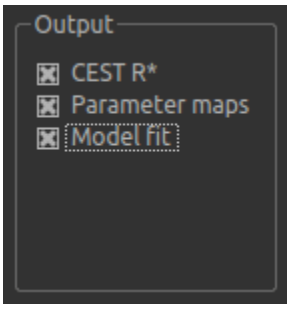
In the analysis section we have the option of allowing the T1/T2 values to vary. We will enable this, but provide T1 and T2 maps to guide the modelling. These maps are stored in the following files:

- `T1map.nii`
- `T2map.nii`

Load both of these files into Quantiphyse using `File->Load Data` as before. Now select the `T1 map` and `T2 map` checkboxes, and select the appropriate data sets from the dropdown menus. The result should look like this:



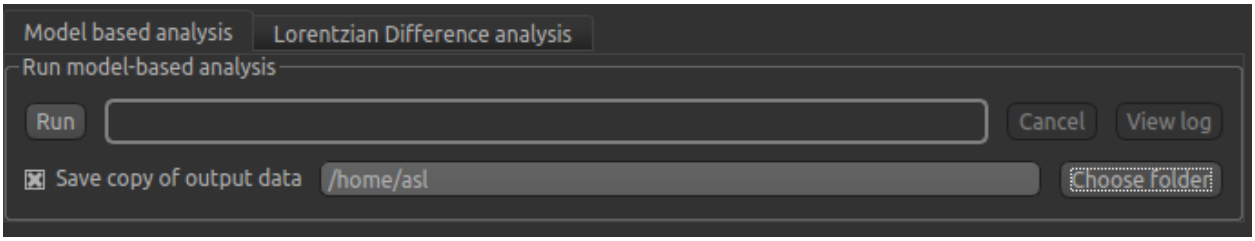
Output section



By default, CESTR* maps will be output, with the added option to output individual parameter maps, as well as fitted curves. As shown above, we have set both of these options, so that fitted data can be properly interrogated.

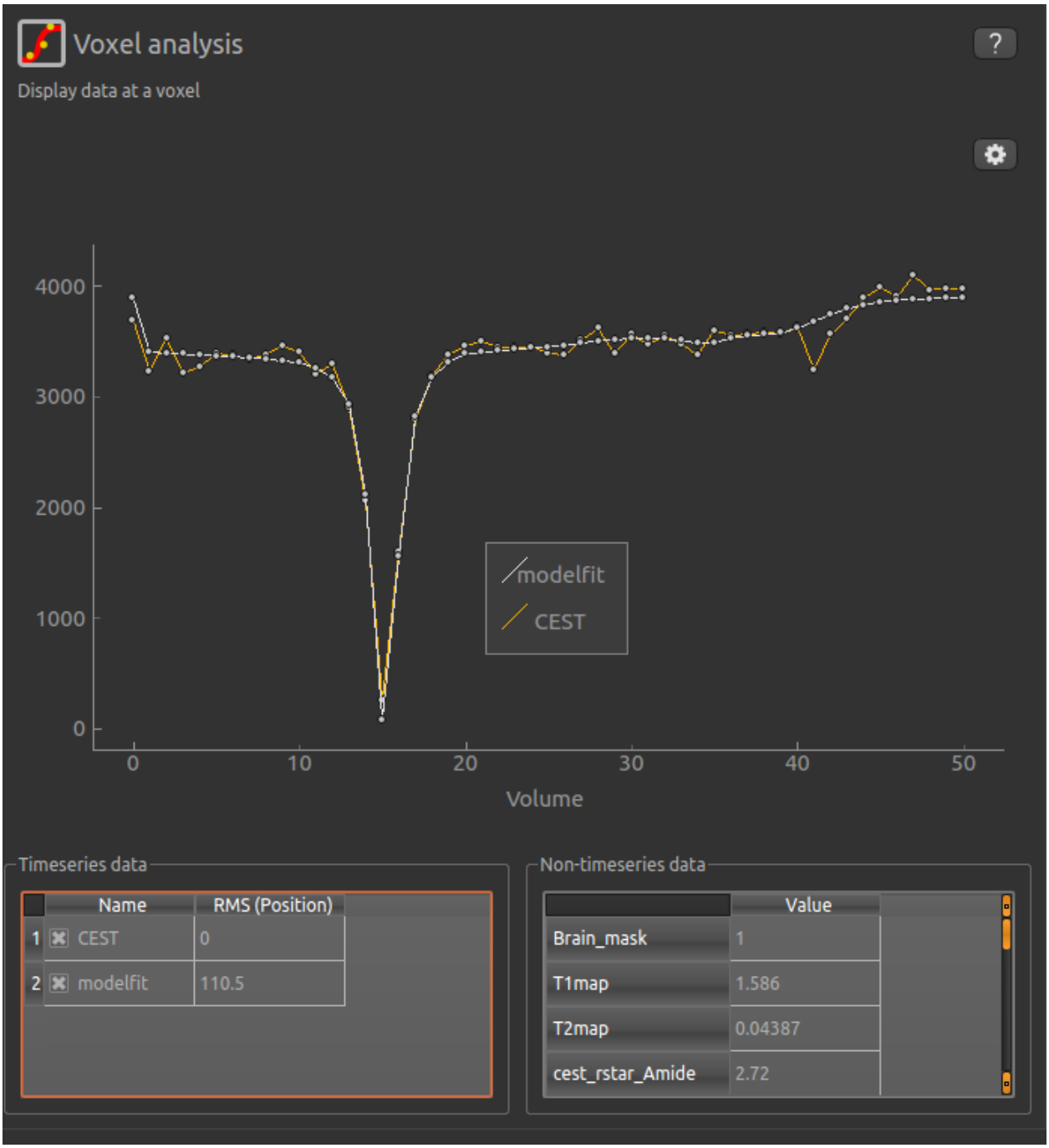
Running model-based analysis

The `Run` button is used to start the analysis. The output data will be loaded into Quantiphyse but if you would also like to save it in a file, you can select the `Save copy of output data` checkbox and choose a folder to save it in.

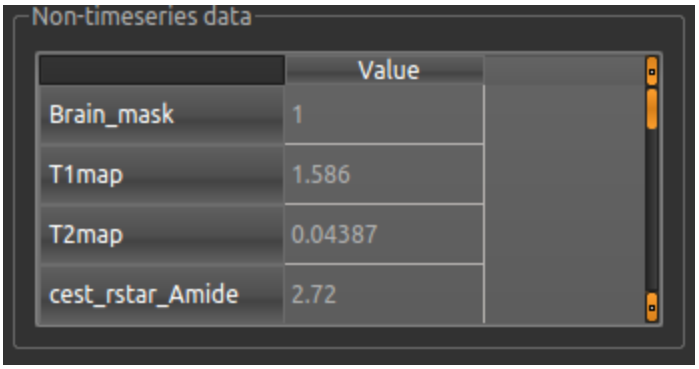


Visualising Processed Data

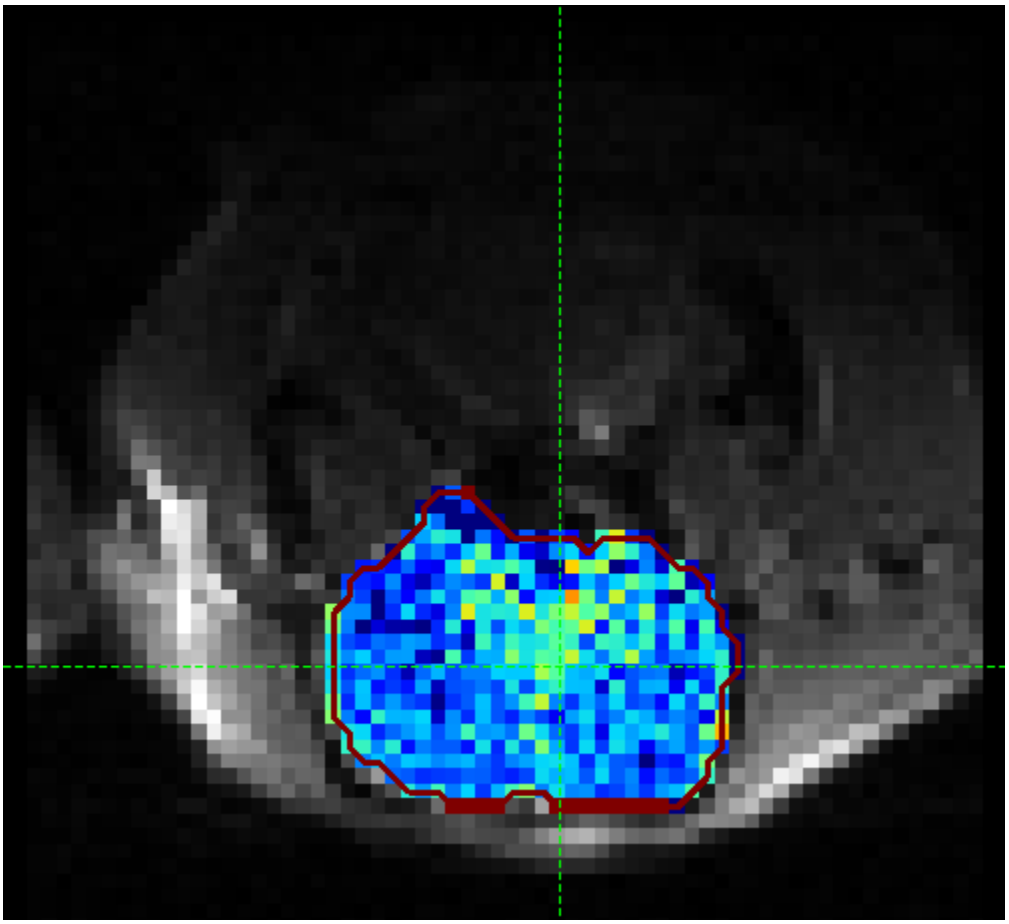
If you re-select the `Voxel analysis` widget which we used at the start to look at the CEST signal in the input data, you can see the model prediction overlaid onto the data. By clicking on different voxels you can get an idea of how well the model has fitted your data.



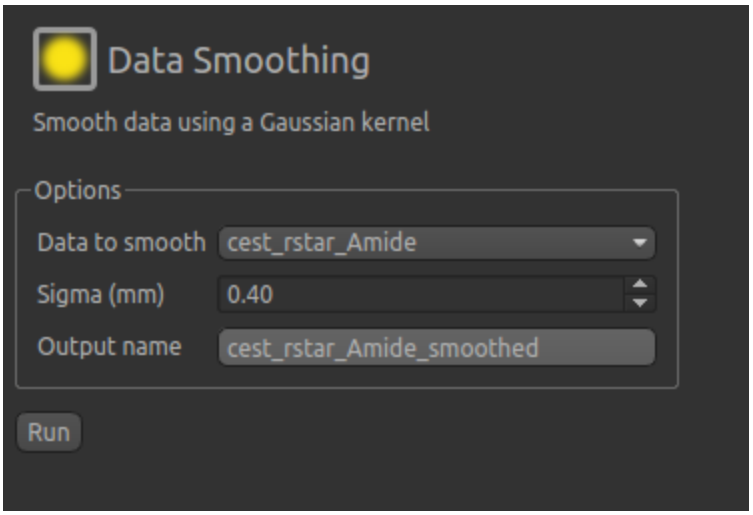
For each non-water pool included in the model there will be a corresponding CESTR* map output (here amide and a macromolecular pool), and these values will be summarised for each voxel underneath the timeseries data.



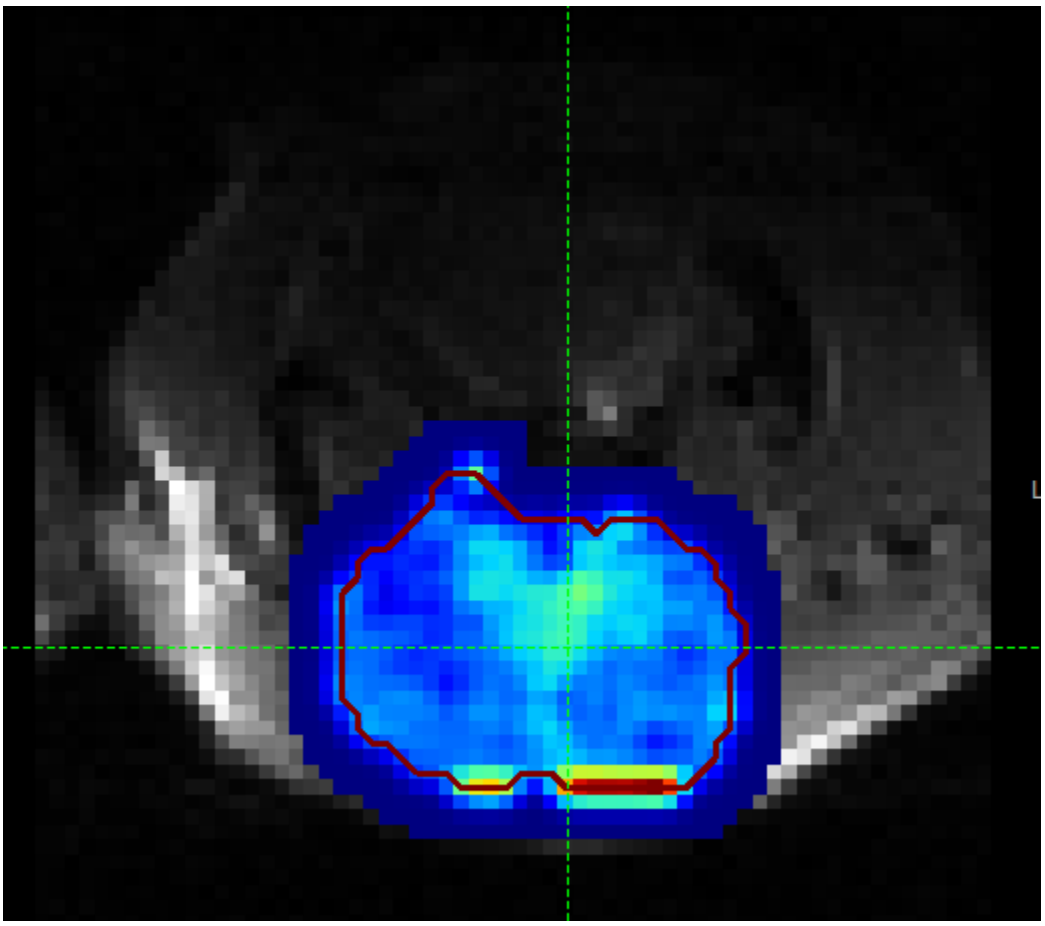
Here we are most interested in the behaviour of the Amide pool; cest_rstar_Amide. In this preclinical example, there is an ischemic region on the right hand side of the brain. If we select `cest_rstar_Amide` from the overlay selector (below the viewing window), a reduced CESTR* is just about visible.



We can extract quantitative metrics for this using regions of interest (ROIs). Before doing this it can help to apply some smoothing to the data. From the menu select `Widgets->Processing->Smoothing` and set the options to smooth `cest_rstar_Amide` with a smoothing kernel size of 0.4mm:



The output of this smoothing appears as follows:



The ischaemic region is a little more visible in this section (to the left of the image, i.e. the right side of the brain).

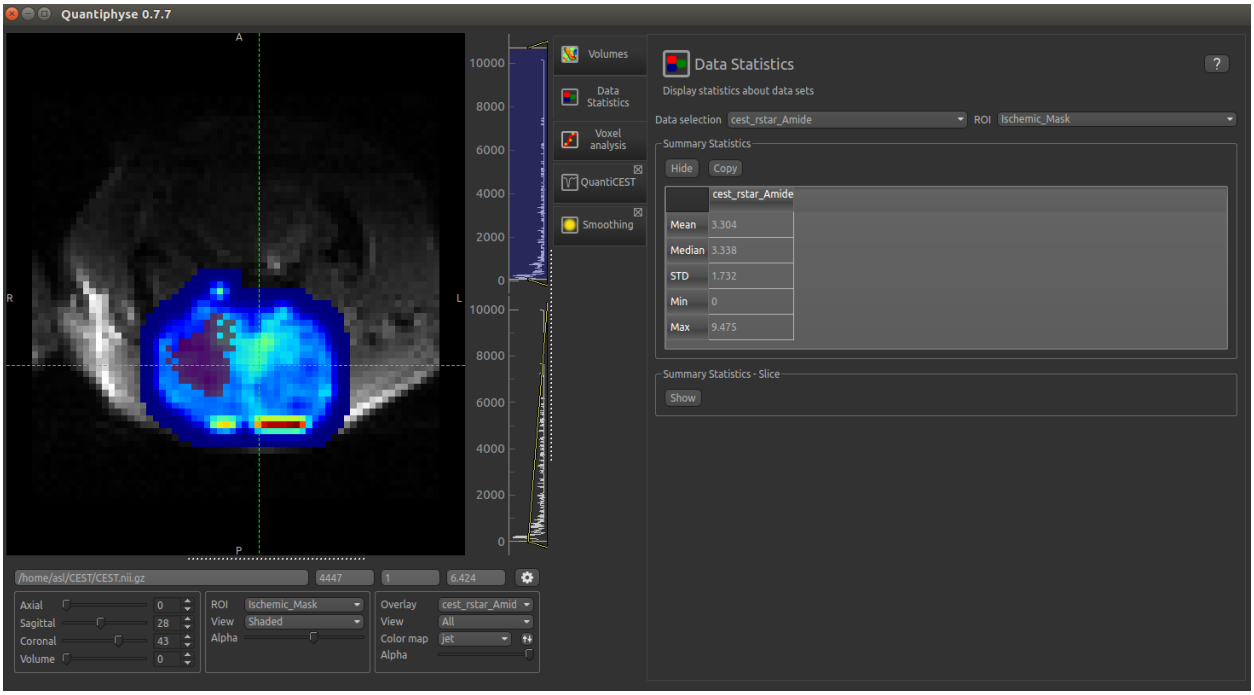
Extracting quantitative Metrics

We have prepared an ROI for the ischaemic region in the file:

- `Ischemic_mask.nii`

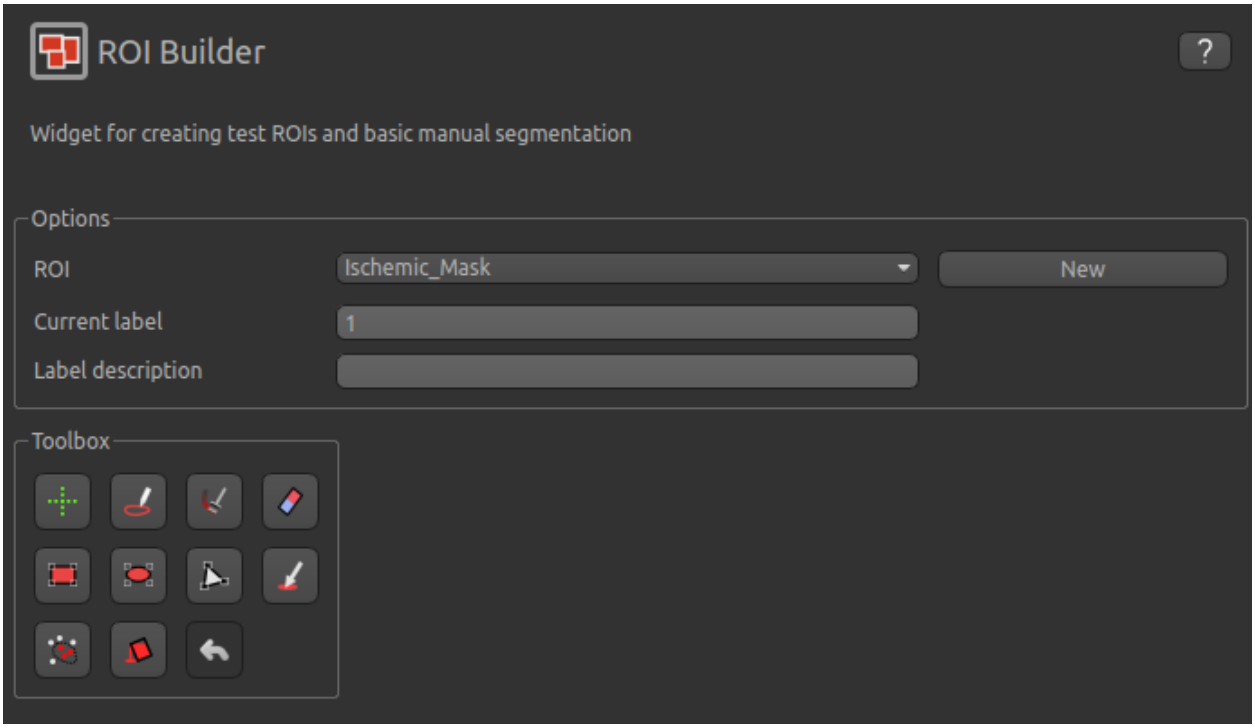
Load this file using `File->Load Data`, selecting it as an ROI.

Now open the `Data Statistics` widget which is visible by default above the `Voxel Analysis` widget. We can now select statistics on `cest_rstar_Amide` within this ROI (click on `Summary statistics` to view):

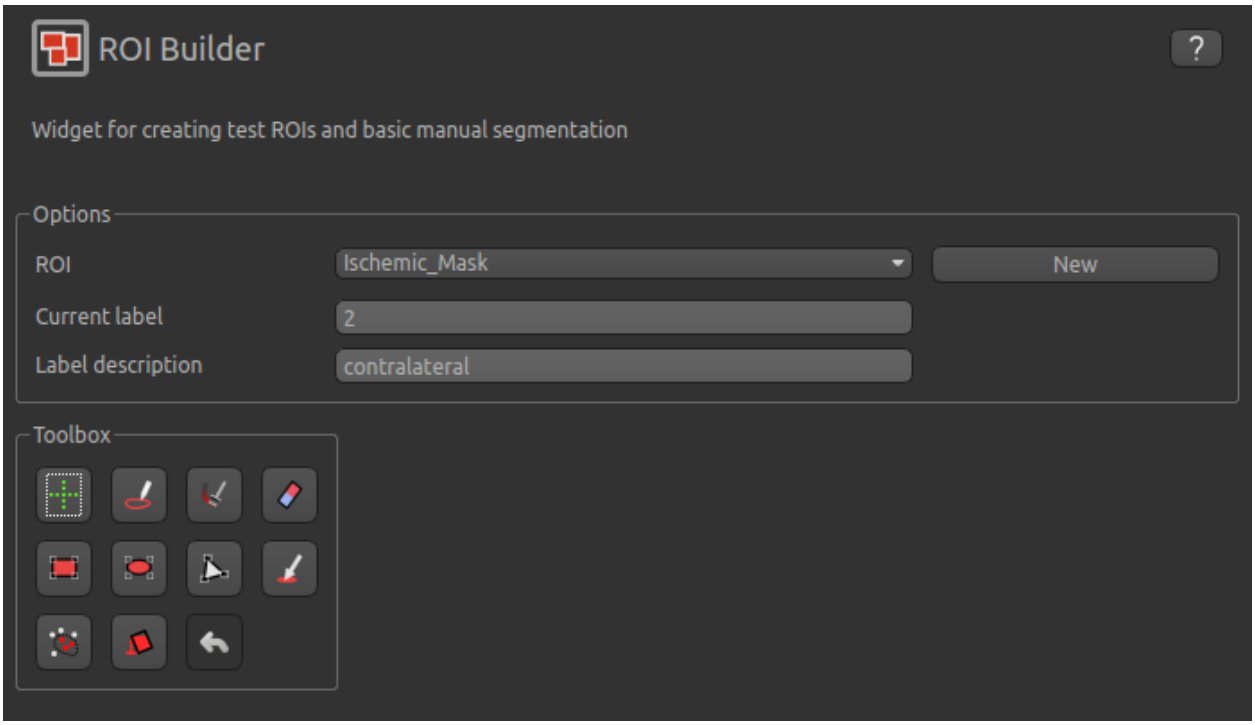


Note that it is possible to display statistics from more than one data set, however here we are just going to look at the CESTR* for the Amide pool.

To compare with the non-ischemic portion, we will now draw a contralateral ROI. To do this, open the `Widgets->ROIs->ROI Builder` and select the `Ischemic_mask` ROI for editing:

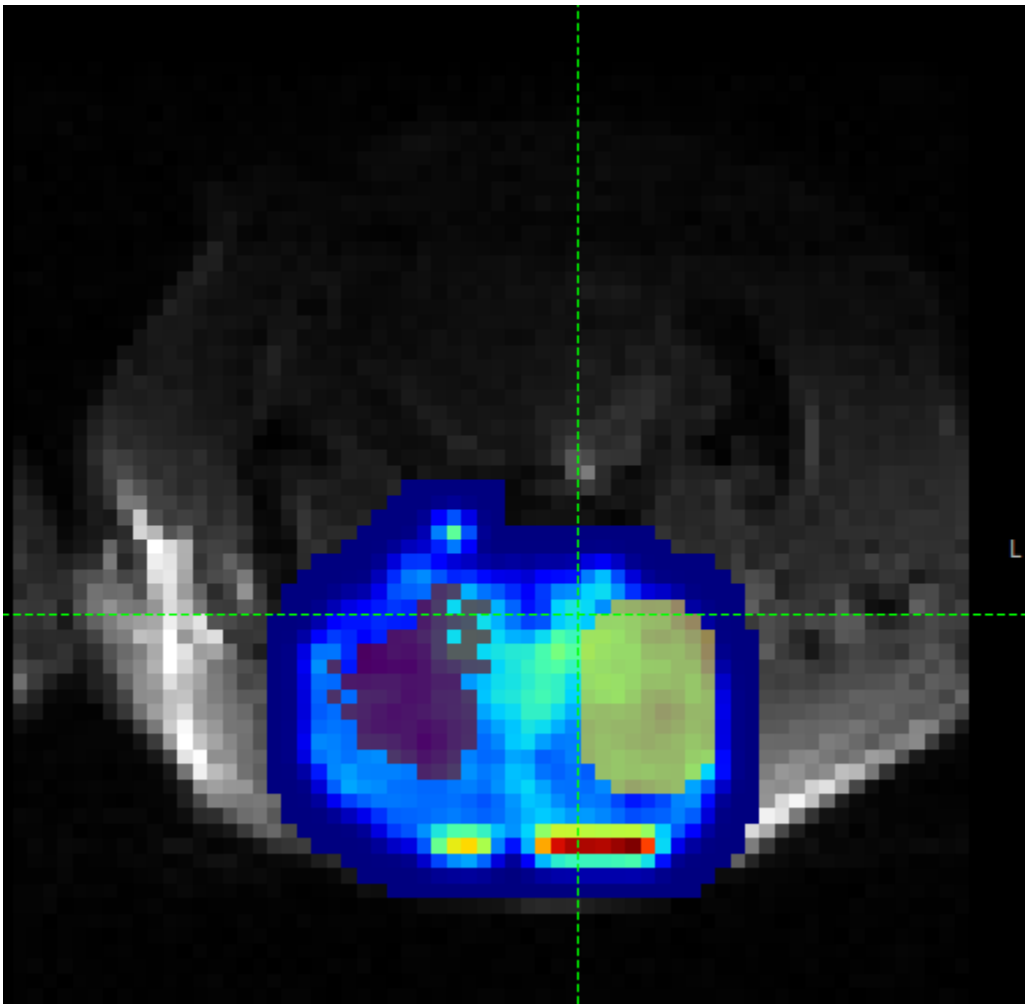


The default label of 1 has been used to label the ischemic core, so type `ischemic` in the `Label description` box. Now enter a new label number (e.g. 2) and change the default name from `Region 2` to `contralateral`:

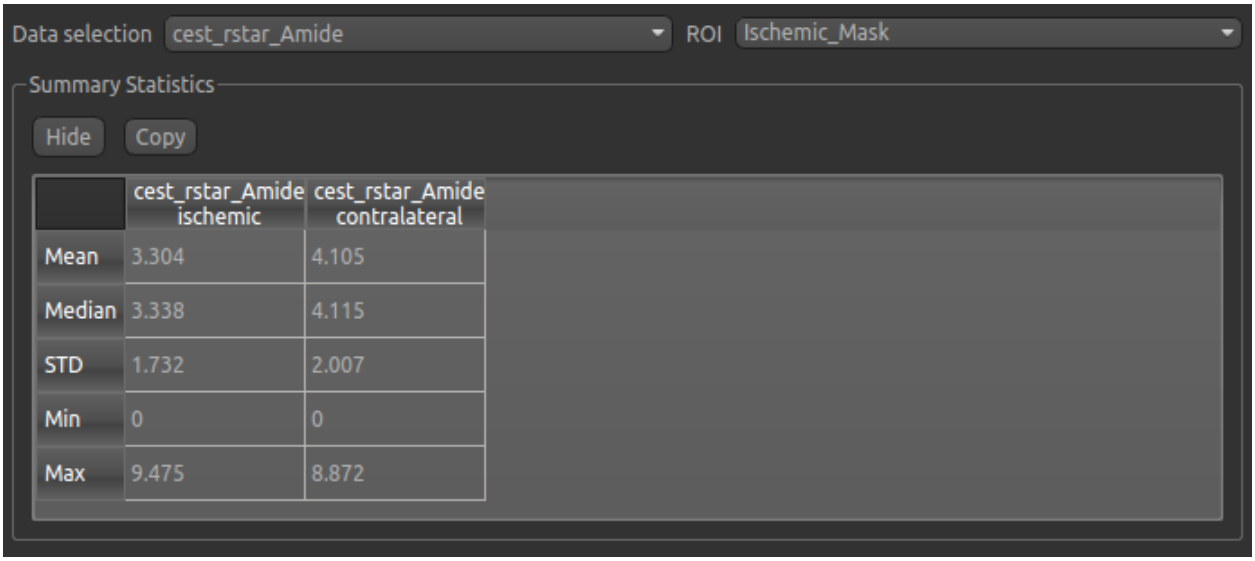


To manually draw a contralateral ROI, use either the pen tool  to draw freehand around a region on the opposite side of the brain, or use one of the other tools to select a suitable region - for example you could draw it as an ellipse

using the  tool. After drawing a region, click `Add` to add it to the ROI. It should appear in a different colour as it is a different label. Here is an example (the new contralateral region is yellow):



Now go back to the `Data Statistics` widget where we can compare the CESTR* in the two regions we have defined. As expected, CESTR* of the amide pool is lower for the ischemic tissue than for healthy tissue.



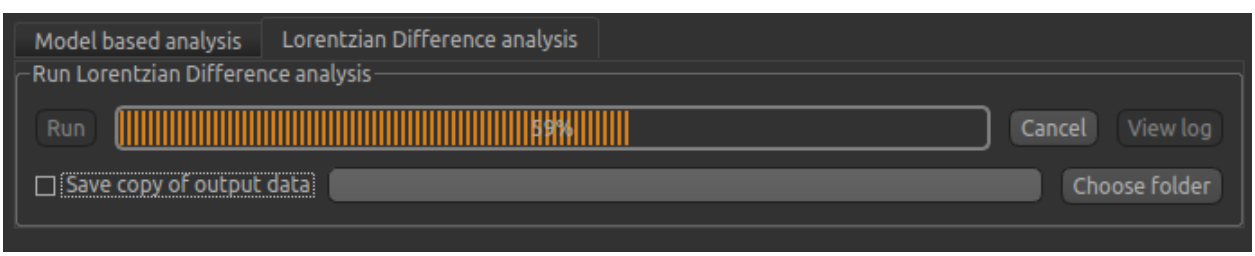
Beyond CESTR*

The minimum outputs from running model-based analysis are the model-fitted z-spectra, and CESTR* maps for non-water pools, as defined in your model setup. If the Parameter Maps option is highlighted then for each pool, including water, there will be additional maps of proton concentration and exchange rate (from which CESTR* is calculated), as well as frequency offset (ppm). For water, the offset map represents the correction for any field inhomogeneities.

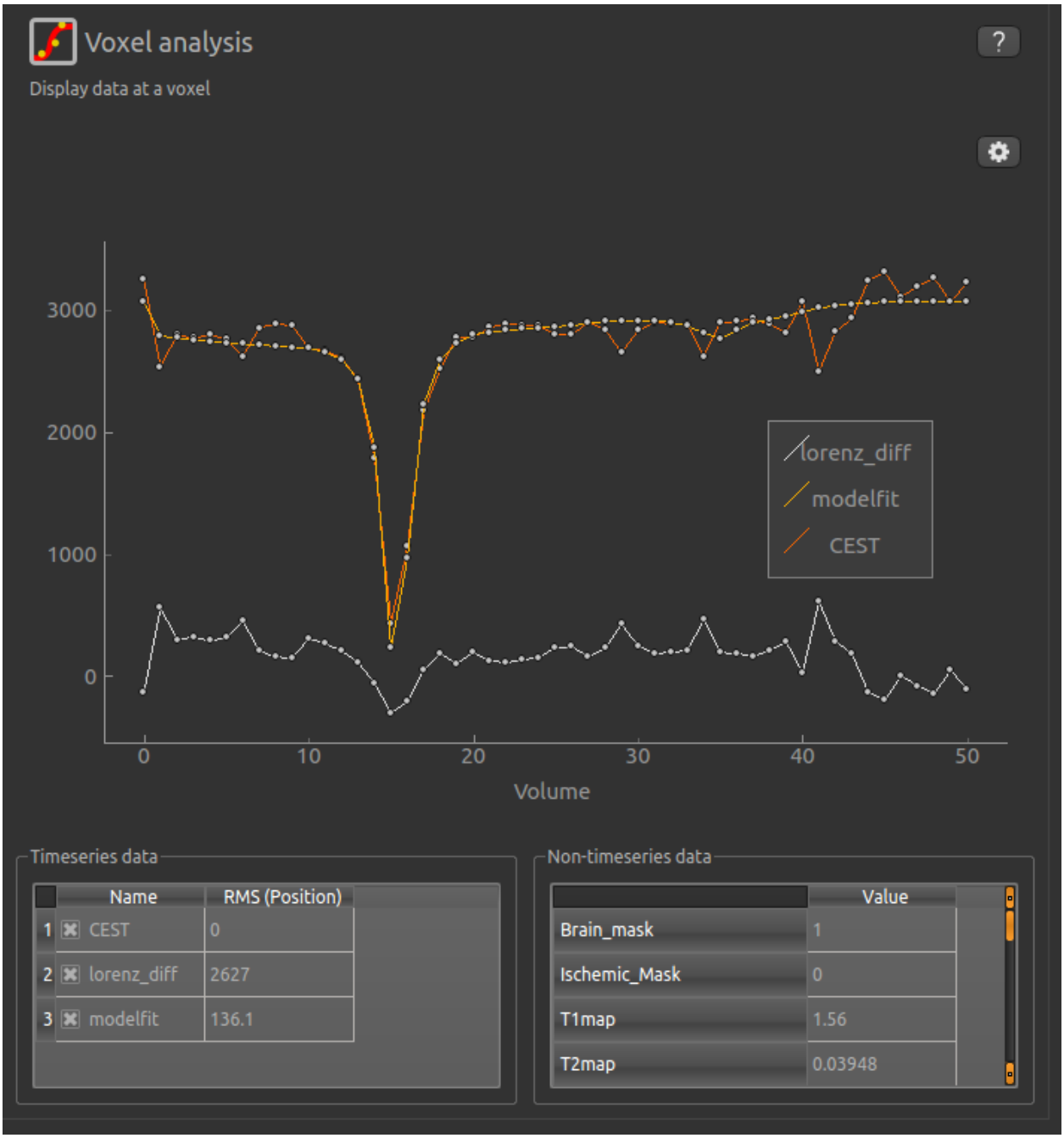
If the `Allow uncertainty in T1/T2 values` is set then fitted maps of T1 and T2 will be available for each pool. Naming conventions follow the order the pools are defined in the QuantiCEST setup panel.

Viewing data without the water baseline

Rather than doing a full model-based analysis as described in section Bayesian model-based analysis, QuantiCEST also has the option simply remove the water baseline from the raw data, allowing you to directly view or quantify the smaller non-water peaks in the acquired CEST volume. Baseline removal is done using the Lorentzian Difference Analysis (LDA) option in QuantiCEST - this is available by selecting the alternative tab in the box containing the `Run` button.



LDA works by fitting a subset of the raw CEST data (within $\pm$1ppm, and beyond $\pm$30ppm) to a water pool (or a water plus MT pool if chosen), and then subtracting this model fit from the data. This leaves behind the smaller non-water peaks in the data, called a Lorentzian Difference spectrum. QuantiCEST outputs this as `lorenz_diff.nii.gz`. This can be viewed in the `Voxel Analysis` widget alongside the data signal and the model-based fit:



Running QuantiCEST from the command line

Here we have covered basic model-based analysis of CEST data using the interactive GUI. If you have multiple data sets it may be desirable to automate this analysis so that the same processing steps can be run on several data sets from the command line, without interactive use.

Although this is beyond the scope of this tutorial, it can be set up relatively simply. The batch processing options for the analysis you have set up can be displayed by clicing on the following button at the top of the QuantiCEST widget  . For more information see documentation for *Batch processing*.

References

5.3.2 Reference

QuantiCEST User Guide

Introduction

To do CEST analysis you will need to know the following:

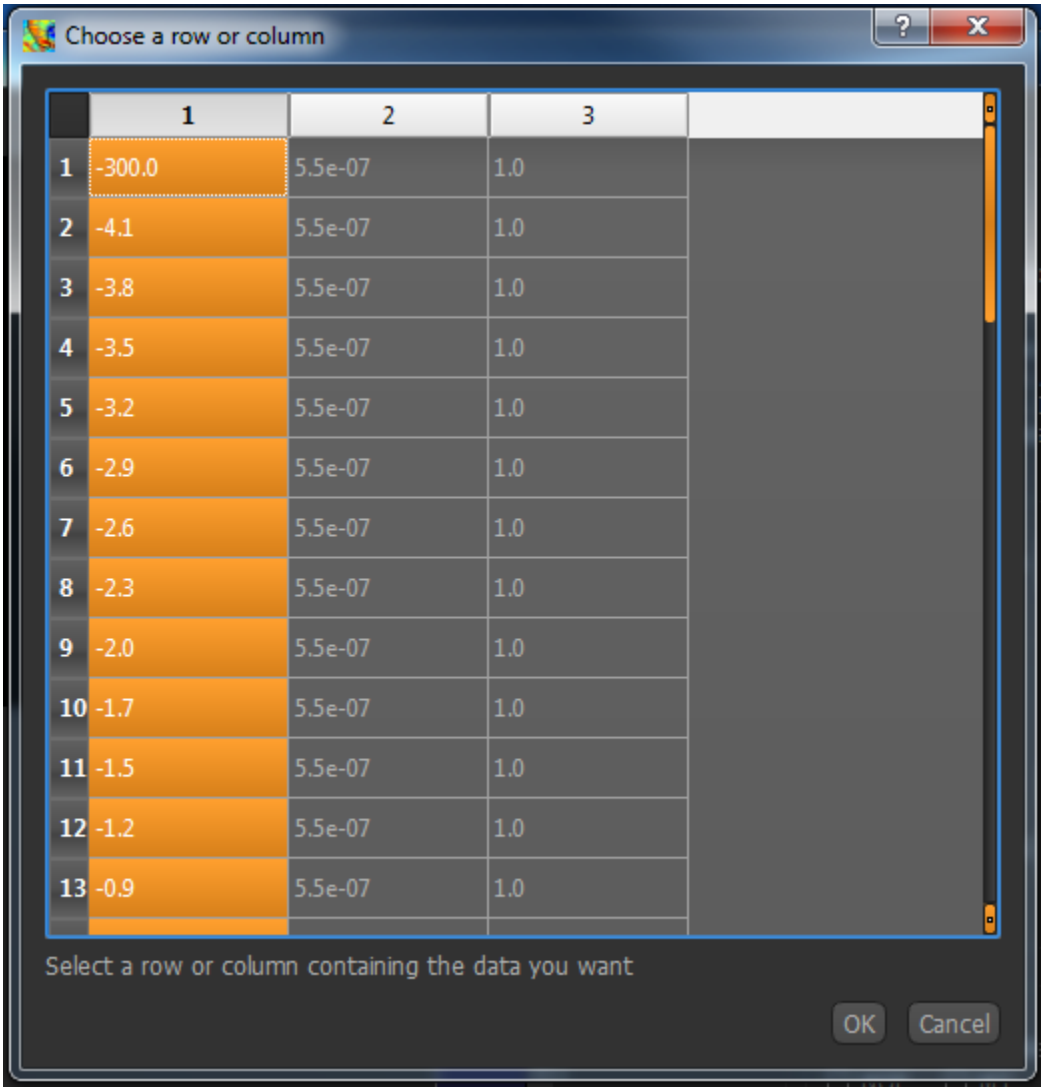
- The frequencies in the z-spectrum you sampled at. The number of frequencies corresponds to the number of volumes in your data
- The field strength - default pool data is provided for 3T and 9.4T, but you can specify a custom value provided you provide the relevant pool data
- The saturation field strength in µT
- For continuous saturation, the duration in seconds
- For pulsed saturation details of the pulse magnitudes and durations for each pulse segment, and the number of pulse repeats
- What pools you want to include in your analysis
- If default data is not available for your pools at your field strength, you will need the ppm offset of each pool, its exchange rate with water and T1/T2 values

Setting the sampling frequencies

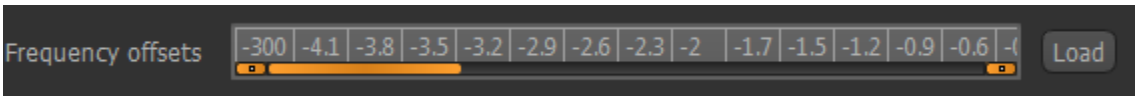
The frequencies are listed horizontally:

You can type in your frequencies manually - the list will automatically grow as you add numbers.

However, you may have your frequencies listed in an existing text file - for example the `dataspec` file if you have been using Fabber for your analysis. To use this, either drag the file onto the list or click the `Load` button and select the file. Quantiphyse will assume the file contains ASCII numeric data in list or matrix form and will display the data found.

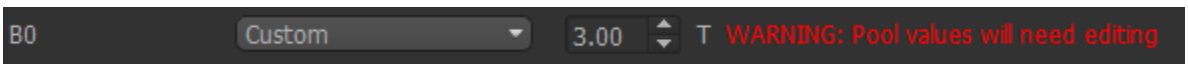


Click on the column or row headers to select the column/row your frequencies are listed in. In this case, we have a Fabber `dataspec` file and the frequencies are in the first column, so I have selected the first column of numbers. Click `OK` to enter this into your frequency list.



Setting the field strengths

Choose the B0 field strength from the menu. If none of the values are correct, select `Custom` and enter your field strength in the spin box that appears

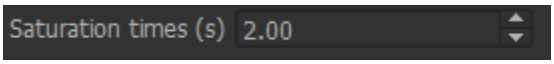


Note that you are being warned that the default pool data will not be correct for a custom field strength and you will need to edit them.

The saturation field strength is set using the `B1 (µT)` spin box below.

Continuous saturation

Select `Continuous Saturation` from the menu, and enter the duration in seconds in the spin box



Pulsed saturation

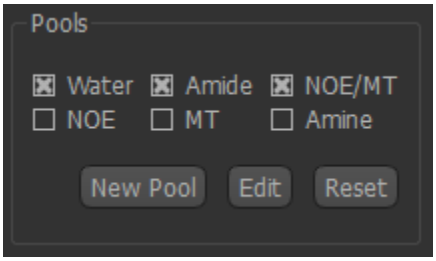
Select `Pulsed Saturation` from the menu.

The pulse magnitudes and durations can be set in the same way as the sampling frequencies, so if you have them in a text file (for example a Fabber `ptrain` file), drag it onto the list and choose the appropriate row/column.

The number of magnitudes must match the number of durations! Repeats can be set in the spin box at the bottom.

Choosing pools

Six built-in pools are provided, with data at 3T and 9.4T, you can choose which to include using the checkboxes.



Each pool is characterized by four parameters:

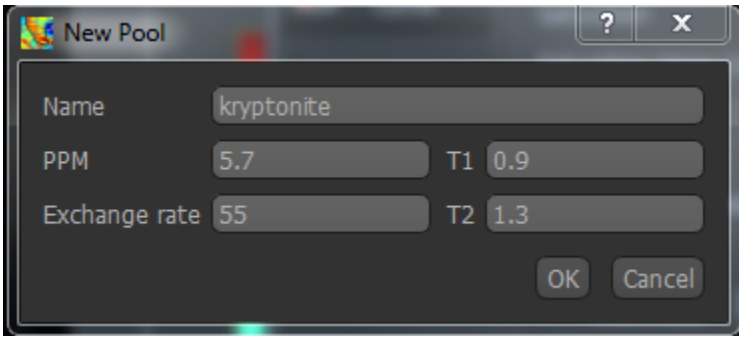
- The ppm offset relative to water (by definition this is zero for water)
- The exchange rate with water
- The T1 value at the specified field strength
- The T2 value at the specified field strength

To view or change these values, click the `Edit` button.

A warning will appear if you change the values from the defaults. Obviously this will be necessary if you are using a custom field strength. If you want to return to the original values at any point, click the `Reset` button. This does not affect what pools you have selected and will not remove custom pools

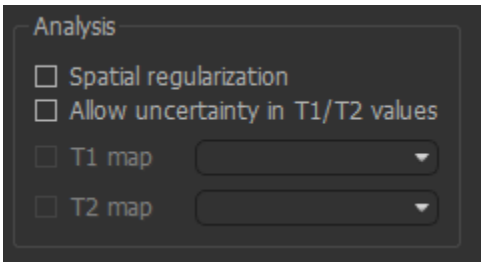
Custom pools

If you want to use a pool which is not built-in, you can use the *New Pool* button to add it. You will need to provide the four parameters above, and your new pool will then be selected by default.



Warning: Currently custom pools, and custom pool values are not saved when you exit Quantiphyse

Analysis options



These affect how Fabber performs the model fitting

- `Spatial regularization` - if enabled, adaptive smoothing will be performed on the parameter maps, with the degree of smoothing determined by the variation of the data

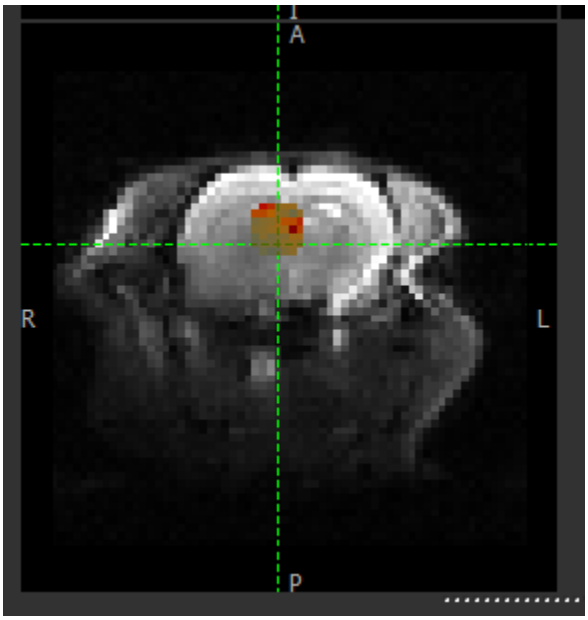
- `Allow uncertainty in T1/T2 values` - T1/T2 will be inferred, using the pool-specified values as initial priors
- `T1 map`/`T2 map` - If inferring T1/T2, an alternative to using the pool-specified values as priors you may provide existing T1/T2 maps for the water pool.

Warning: Spatial regularization prevents Fabber from processing voxels in parallel, hence the analysis will be much slower on multi-core systems.

Run model-based analysis

This will perform the model fitting process.

CEST analysis is computationally expensive, and it is recommended to run on a small ROI before attempting your full data set. The ROI Builder tool is an easy way to define a small group of voxels to act as a test ROI, e.g. as below



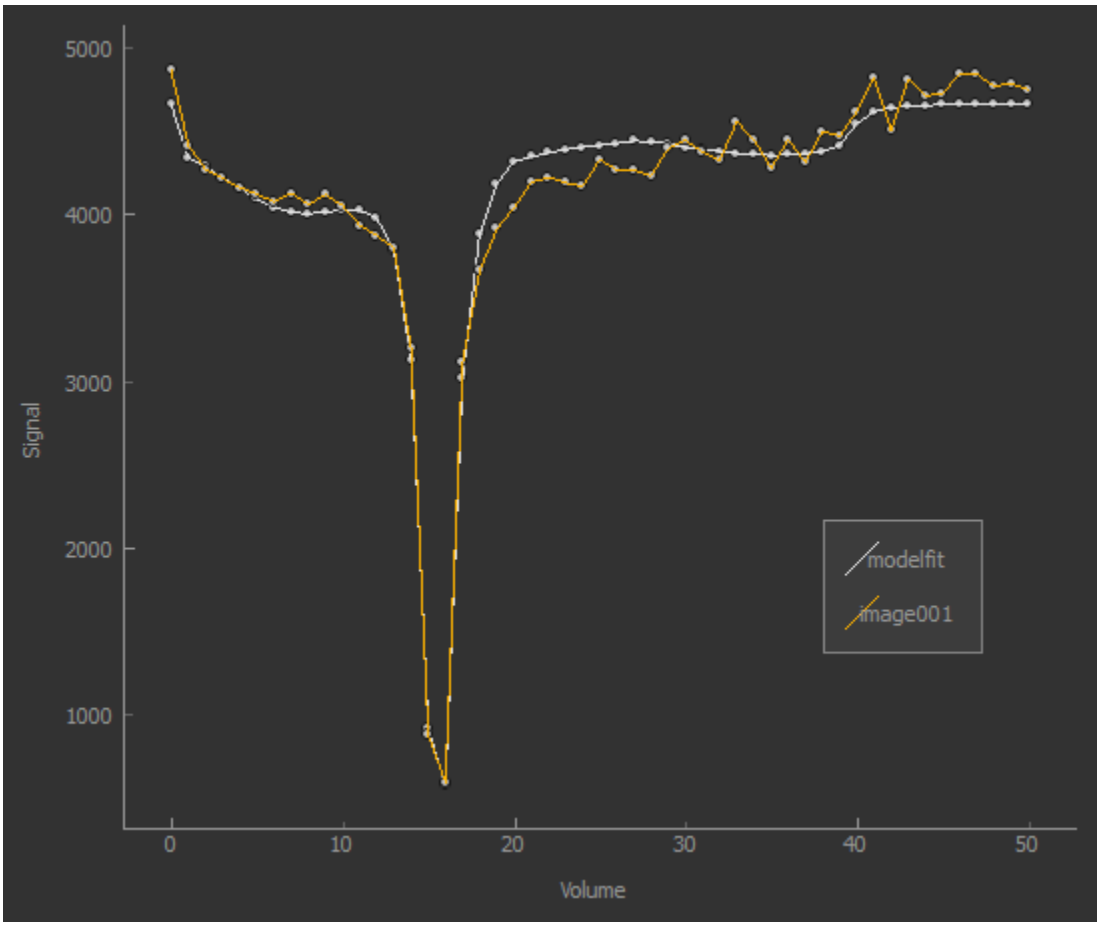
The output of the model-based analysis is a set of data overlays as follows:

- `mean_B1_off` - Model-inferred correction to the specified B1 value
- `mean_ppm_off` - Model-inferred correction to the ppm values in the z-spectrum.
- `modelfit` - Model z-spectrum prediction, for comparison with raw data
- `mean_M0_Water` - Inferred magnetization of the water pool
- `mean_M0_Amine_r`, `mean_M0_NOE_r`, ..etc - Inferred magnetization of the other pools, relative to M0_Water
- `mean_exch_Amine`, `mean_exch_NOE`, ..etc - Inferred exchange rates of non-water pools with water
- `mean_ppm_Amine`, `mean_ppm_NOE`, ..etc - Inferred ppm frequencies of non-water pools
- `cest_rstar_Amine`, `cest_rstar_NOE`, ..etc - Calculation of R* for non-water pools - see below for method

If T1/T2 values are being inferred (`Allow uncertainty in T1/T2 values` is checked), there will be additional outputs:

- `mean_T1_Water`, `mean_T1_Amine`, ..etc - Inferred T1 values for each pool
- `mean_T2_Water`, `mean_T2_Amine`, ..etc - Inferred T2 values for each pool

The screenshot below (from the Voxel Analysis widget) shows the model fitting to the z-spectrum.



CEST R* calculation

The R* calculation is performed as follows:

- After the model fitting process, for each non-water pool, two separate z-spectrum predictions are evaluated at each voxel: - The spectrum based on the water pool only - The spectrum based on the water pool and each other pool individually
- The parameters used for this evaluation are those that resulted from the fitting process, except that: - T1 and T2 are given their prior values - The water ppm offset is zero
- Each spectrum is evaluated at the pool ppm resonance value and the normalized difference to water is returned:

$$R^*_{pool} = \frac{Signal_{water} - Signal_{water+pool}}{M_0}$$

Lorentzian difference analysis

This is a quicker alternative to model-based analysis, however less information is returned.

The calculation is performed using the Fabber fitting tool as previously, in the following way:

- Only the water pool is included, i.e. just fitting a single Lorentzian function to the z-spectrum
- Only data points close to the water peak and unsaturated points are included. Currently this means points with ppm between -1 and 1 are included as are points with ppm > 30 or <-30
- The raw data is subtracted from the resulting model prediction at all sampled z-spectrum points

The output of the LDA calculation is provided as a multi-volume overlay `lorenz_diff`.

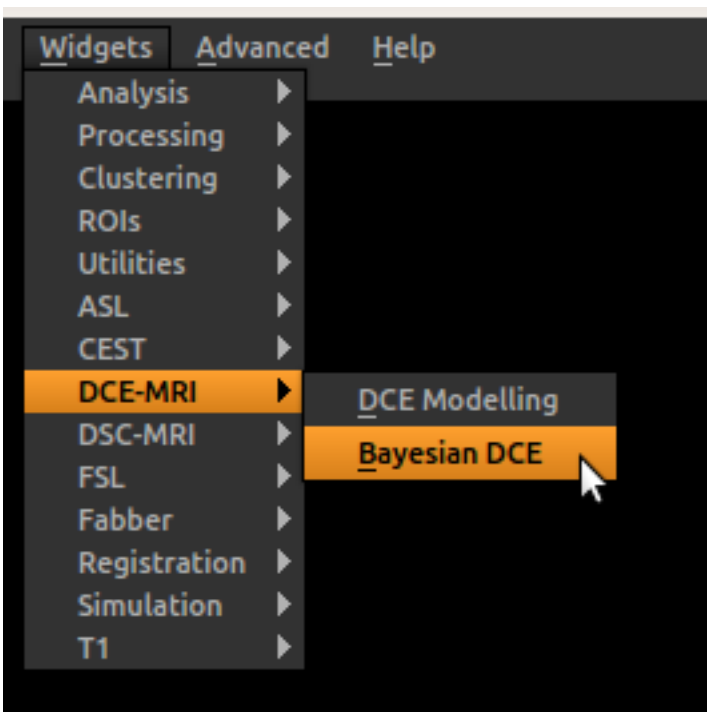
5.3.3 Publications

The following publications are useful citations for the QuantiCEST processing widget:

- **Bayesian inference method**: *Chappell MA, Groves AR, Whitcher B, Woolrich MW. Variational Bayesian inference for a non-linear forward model. IEEE Transactions on Signal Processing 57(1):223-236, 2009.*
- **Bayesian CEST analysis**: *Chappell, M. A., Donahue, M. J., Tee, Y. K., Khrapitchev, A. A., Sibson, N. R., Jezzard, P., & Payne, S. J. (2012). Quantitative Bayesian model-based analysis of amide proton transfer MRI. Magnetic Resonance in Medicine. doi:10.1002/mrm.24474*

5.4 Dynamic Contrast Enhanced (DCE) MRI

Widgets -> DCE-MRI



Dynamic Contrast Enhanced MRI (DCE-MRI) is an advanced MRI technique that captures the tissue T1 changes over time after the administration of a contrast agent. The most common application of DCE-MRI is in monitoring tumours and multiple sclerosis. Thus, DCE-MRI data are typically acquired from patients with cancer or multiple sclerosis. The objective of analysing DCE-MRI data is to quantify a number of haemodynamic parameters (such as Ktrans, perfusion, and permeability), which are important biomarkers to understand the physiology of tumours.

Here, we will explain the steps to analyse DCE-MRI data to quantify the haemodynamic parameters using Quantiphyse. The DCE modelling package allow pharmacokinetic modelling of DCE-MRI using a variety of models. Two independent widgets are provided:

5.4.1 Bayesian DCE modelling

This widget provides DCE model fitting to output image maps of physiological parameters if interest such as K_{trans}, F_p, PS, V_p and V_e. A Bayesian inference approach is used, this has the advantage that prior knowledge about likely parameter values can be incorporated. This allows more complex models to be implemented and, for example, allows a measured T1 value to vary slightly to better fit the data.

Tutorials

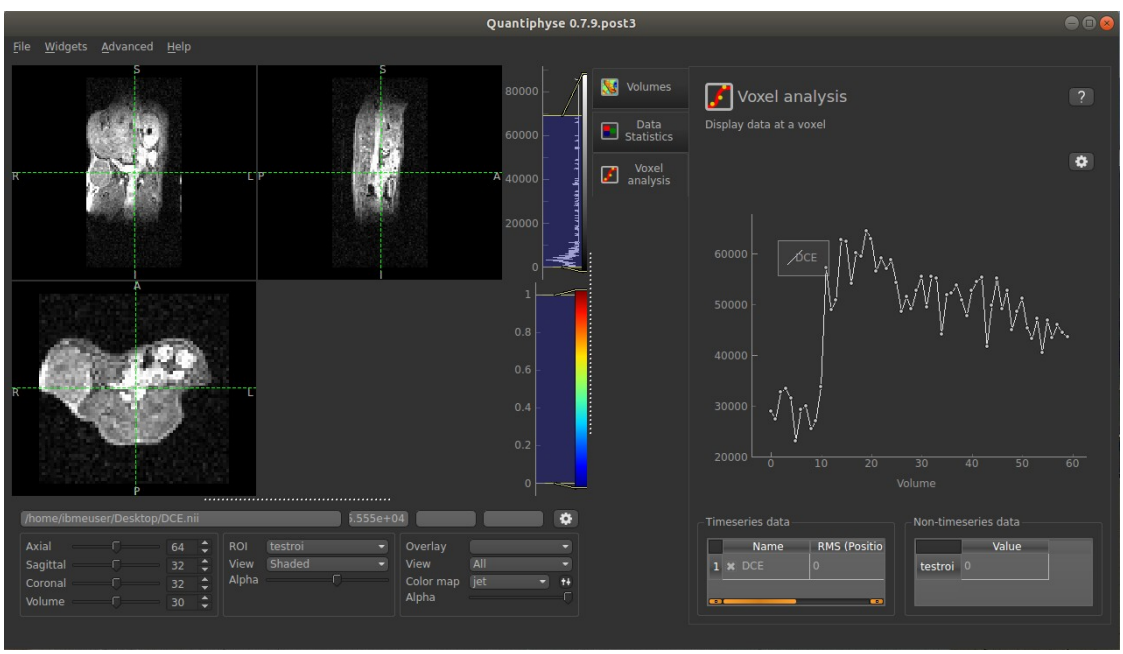
DCE-MRI Data Analysis Tutorial

Introduction

In this tutorial, we are going to explore how to quantify the haemodynamic parameters of DCE-MRI data using Quantiphyse. We will use the Tofts model to perform our analysis.

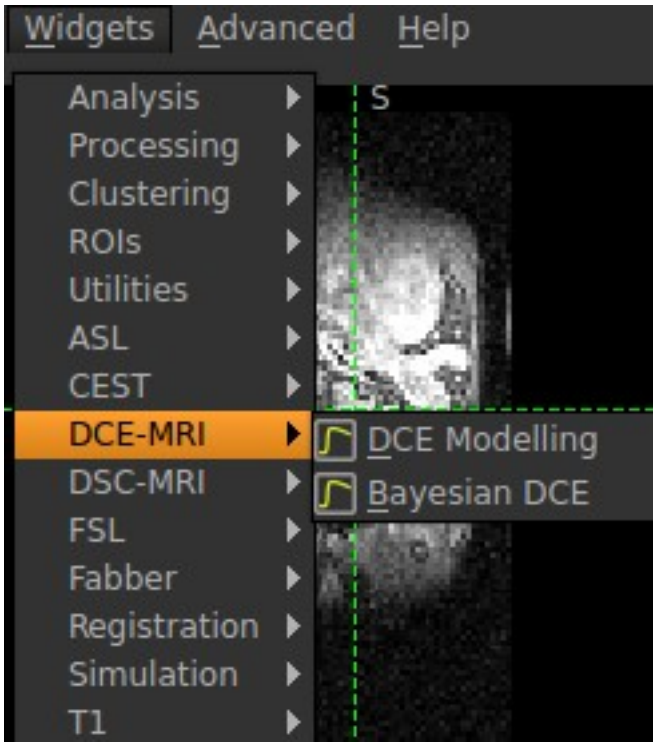
Data Preparation

First, we need to load the DCE-MRI data and the ROI file to Quantiphyse. Be sure to specify the DCE-MRI data as `Data` and ROI file as `ROI`. It is always helpful to check the timeseries of the DCE-MRI data using the `Voxel analysis` Widget:

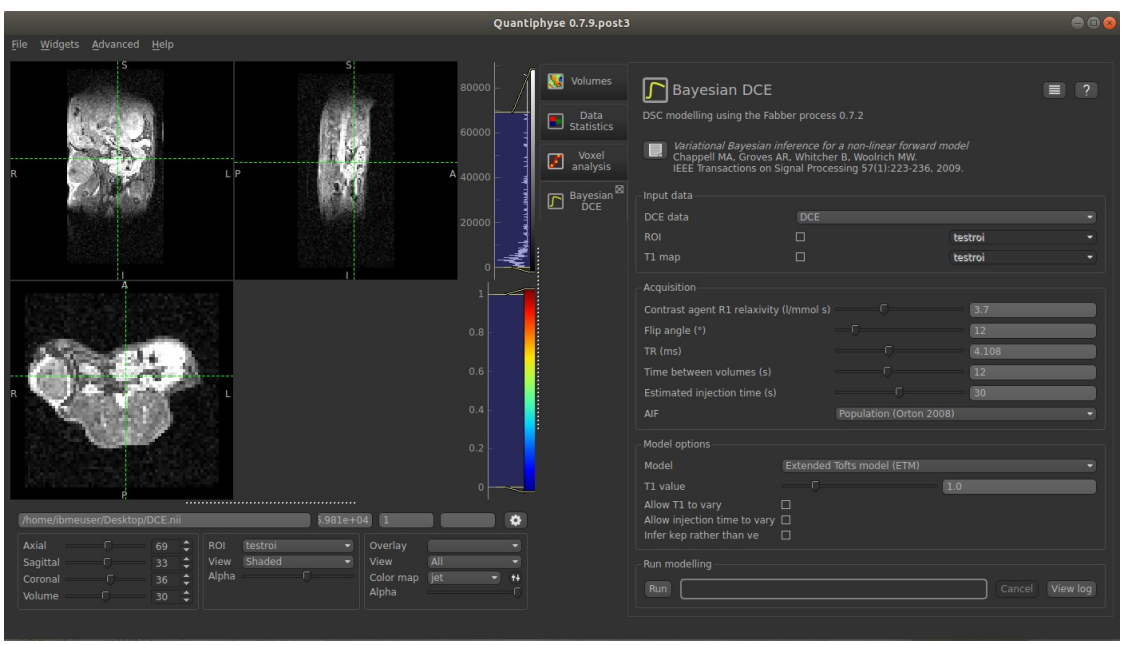


Acquisition Parameters

If the timeseries of the data looks fine, the next step is to specify the sequence parameters for our analysis. Now load the `Bayesian DCE` widget.



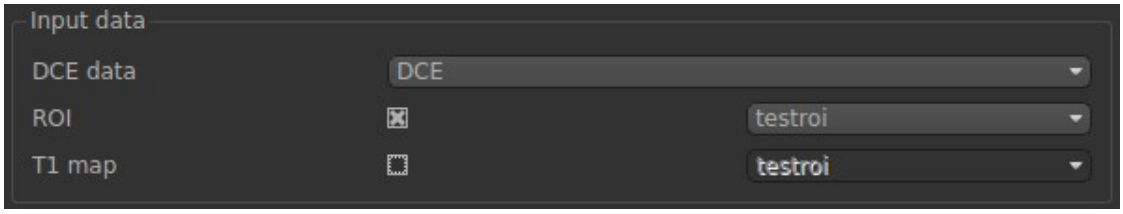
We will see that the Bayesian DCE widget has been loaded onto the right hand side of the Quantiphyse interface.



A full description of the interface can be found in *DCE modelling widget user interface*.

Before we specify the acquisition parameters, we need to make sure that the correct data have been loaded.

In the previous step, we have loaded the DCE-MRI data and a ROI map. In the `Input data` section, we need to tell Quantiphyse the data that we loaded:



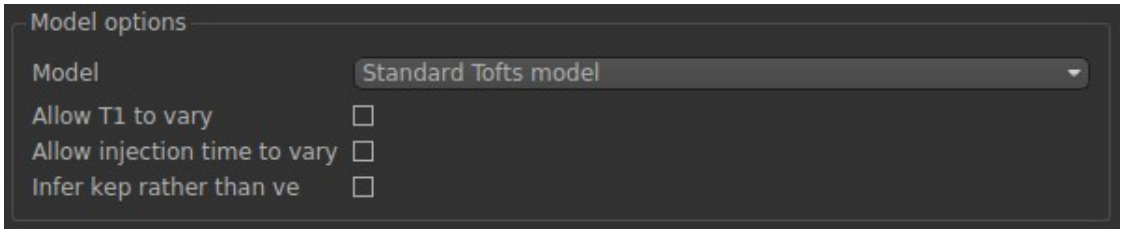
Now we are going to specify the sequence parameter values. These values can be found in the protocol files or the metadata from the scanning session. Note: It is very important to specify these values accurately to ensure the correct analysis. If the data is from an external source, please consult the person who acquired the data.

In our case, we are going to use the following values:

In the AIF option, we are going to select `Population (Parker)`. A detailed explanation on the different AIFs can be found in *AIF options for Bayesian DCE modelling*.

Model Options

After specifying the sequence parameters, the next step is to select the appropriate model to analyse the data. In this example, we are going to use `Standard Tofts Model`. A full description of the different models provided by Quantiphye can be found in *Models available for Bayesian DCE modelling*. We are also going to specify the T1 value. Note: T1 values vary in different tissues. Although is it difficult to have a very precise estimation of the T1 value of our tissue of interest, it is important to specify a value that is close to the actual T1 value to improve the quantification of the haemodynamic parameters in our analysis. We will leave the other options unchecked.



Run Modelling

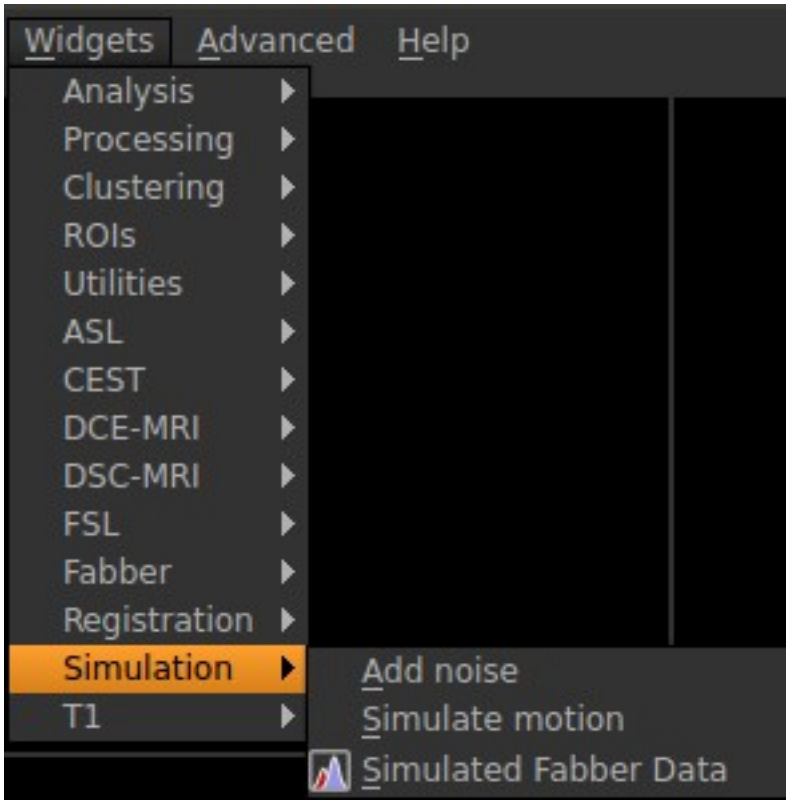
At this point, we have finished the preparation for our analysis. Now click `Run` to start the analysis. Note: it may take a while to complete the analysis.

DCE-MRI Simulation and Analysis

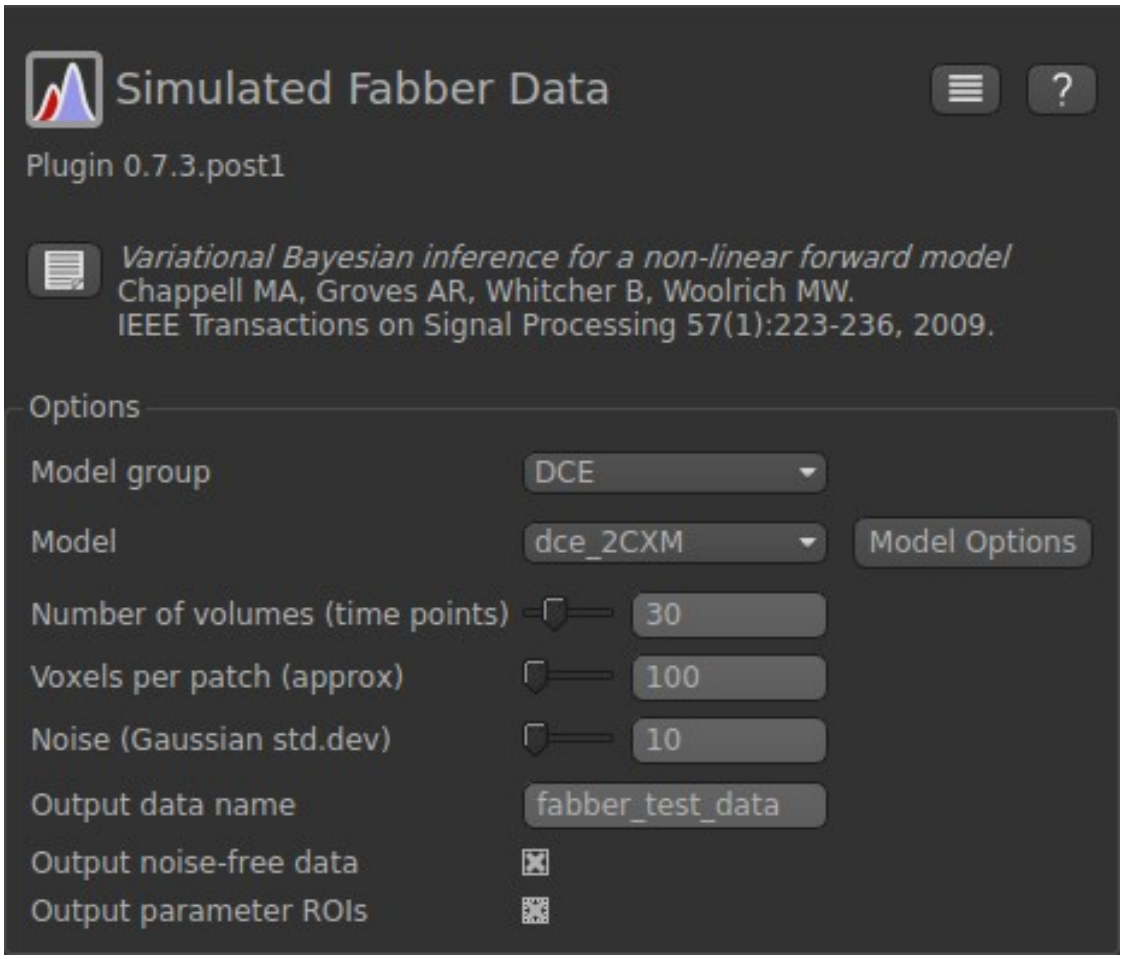
In this tutorial, we will simulate some DCE-MRI data and quantify the simulated parameters using the DCE-MRI analysis pipeline in Quantiphyse.

Simulation

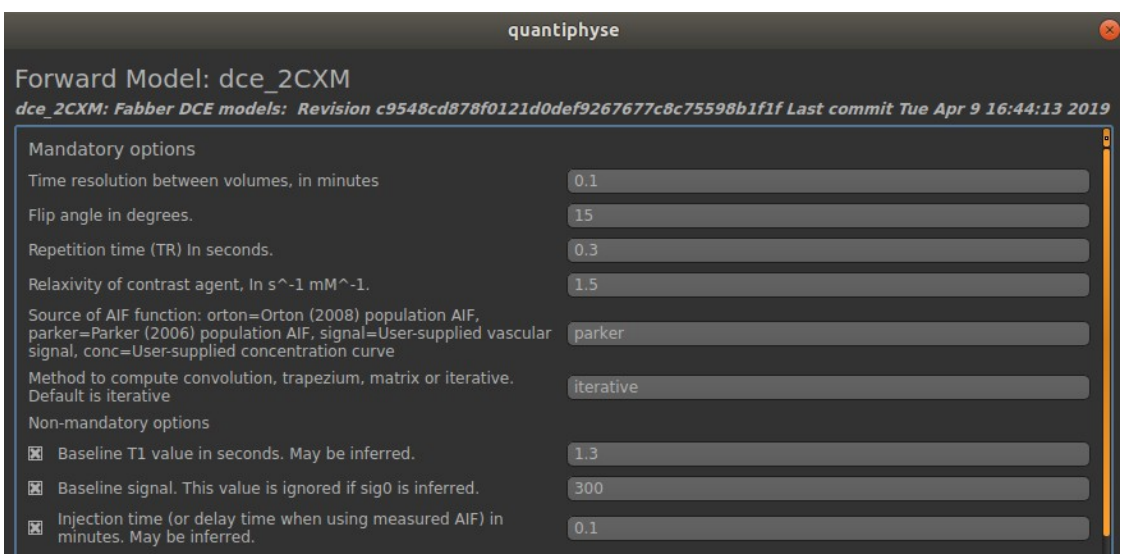
First, launch the widget to simulate DCE-MRI data.



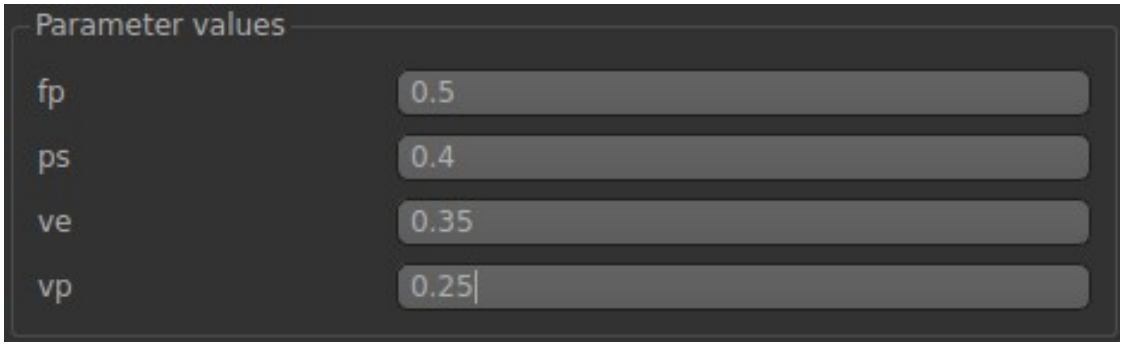
In this example, we will simulate a DCE-MRI data using the two-compartment exchange model (2CXM). First, we specify some basic parameters about the output file. The number of volumes (time points) is related to the total acquisition time and TR. Here we use 30 time points. The voxels per patch indicates the number of voxels (or realizations) to simulate. In this case, we are going to simulate 100 realizations. The noise parameter specifies the noise added to the simulated data. We want to output the noise-free data and parameter ROIs. The setup should look like the following:



Next click on `Model Options` as we are going to set up the sequence parameters in the simulation. In the `Mandatory options` fields, the parameters can take either numerical values or text. The `Non-mandatory options` can also be modified. It is important to remember these simulated parameter values when we check the quantification results. In this example, we are going to use the following parameter values:

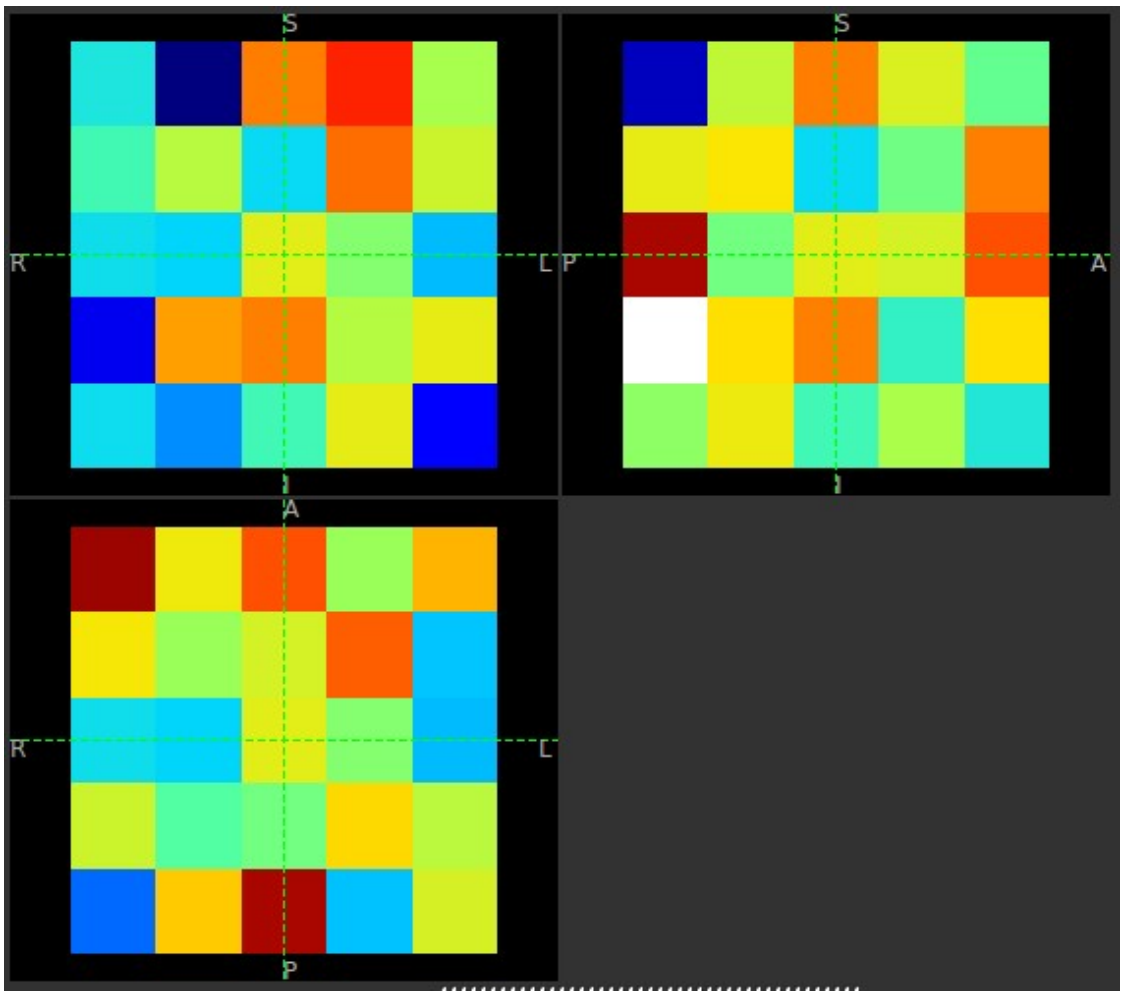


Finally, we are going to specify the hemodynamic parameters. In the 2CXM, there are four hemodynamic parameters: plasma flow (fp), permeability surface area product (ps), volume of extravascular extracellular space (ve), and volume of plasma (vp).

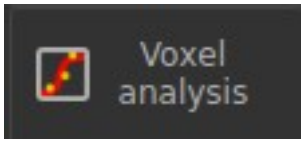


Now click `Generate test data`.

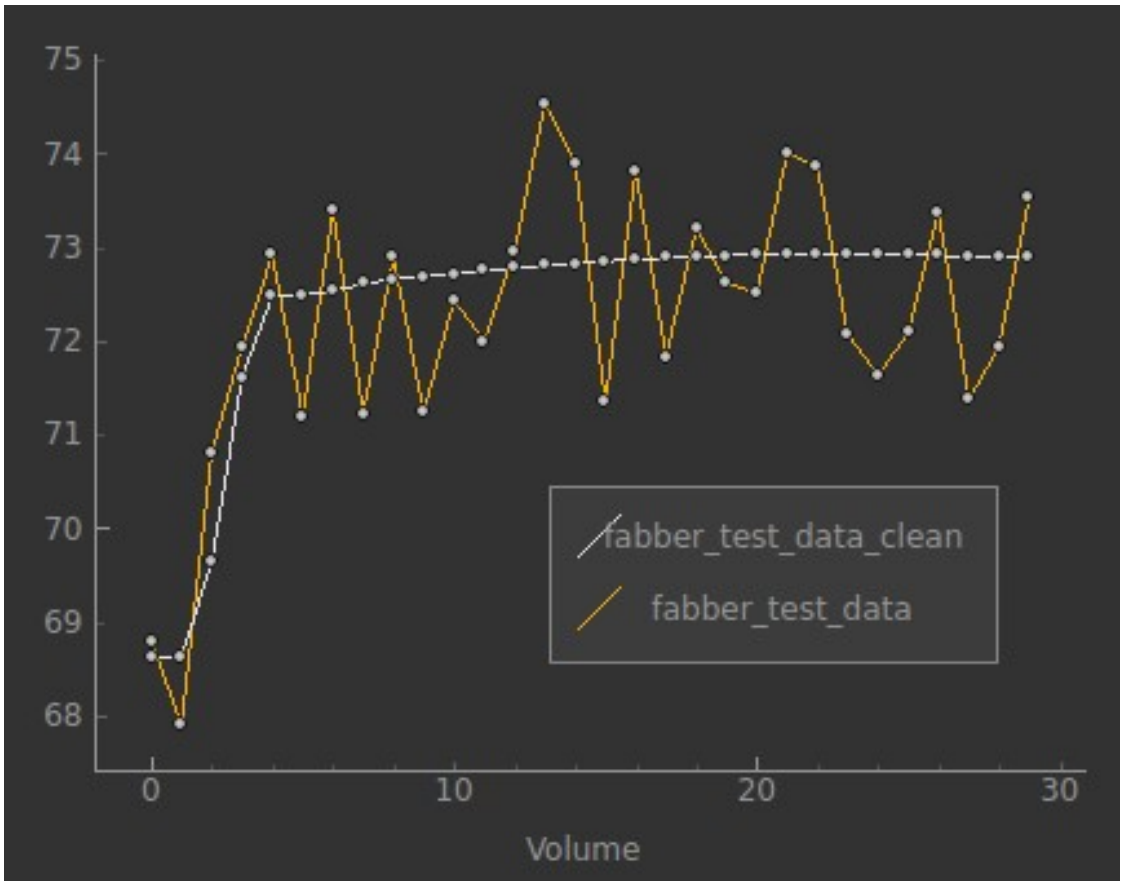
We should be able to see the simulated data shown in the left panel. Your data may be different from the one shown here due to the differences in noise.



Click on `Voxel analysis`

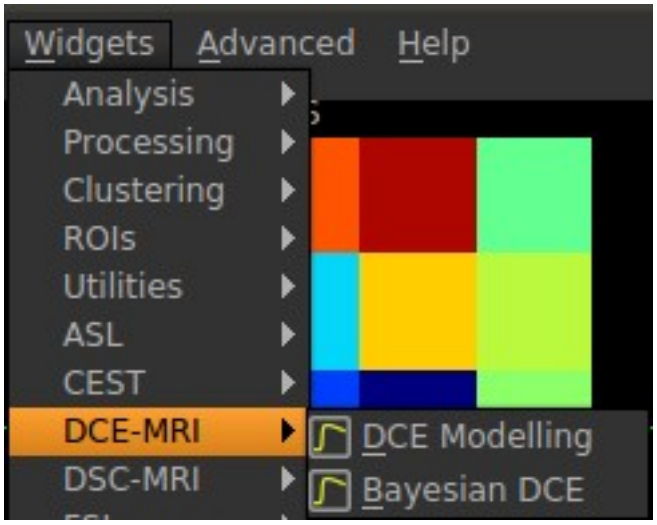


We will be able to see the time series of the noise free (white) and noisy (orange) data in each voxel.

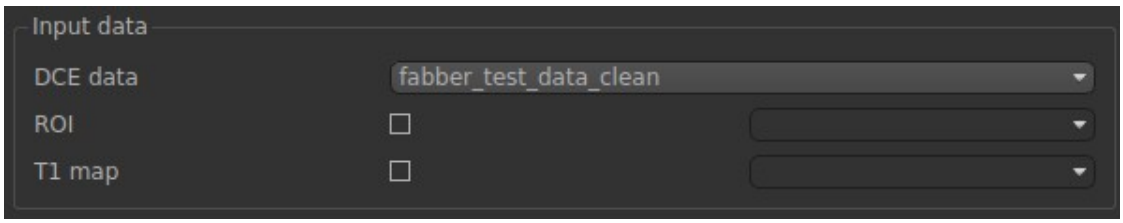


Analysis

In this exercise, we are going to quantify the hemodynamic parameters that we have just simulated. First, bring out the Bayesian DCE-MRI analysis tool.

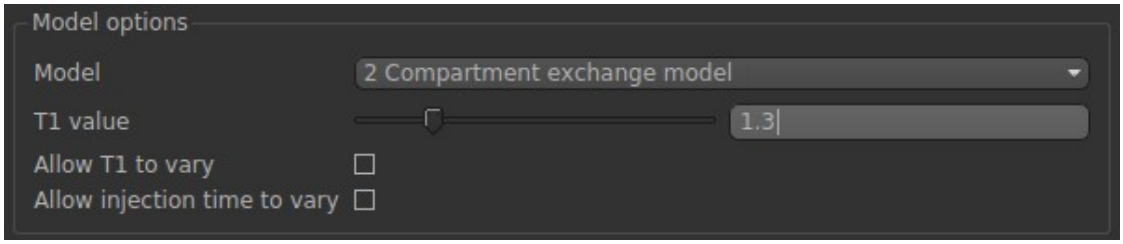


In the input data, we need to select fabber_test_data_clean. This is the noise free data that we have just simulated. Leave the ROI and T1 map empty for now.



In Acquisition, we need to match these parameters with the ones that we used in the simulation in the following:

Finally, in Model options, we need to select 2CXM and specify the T1 value (same with simulations).



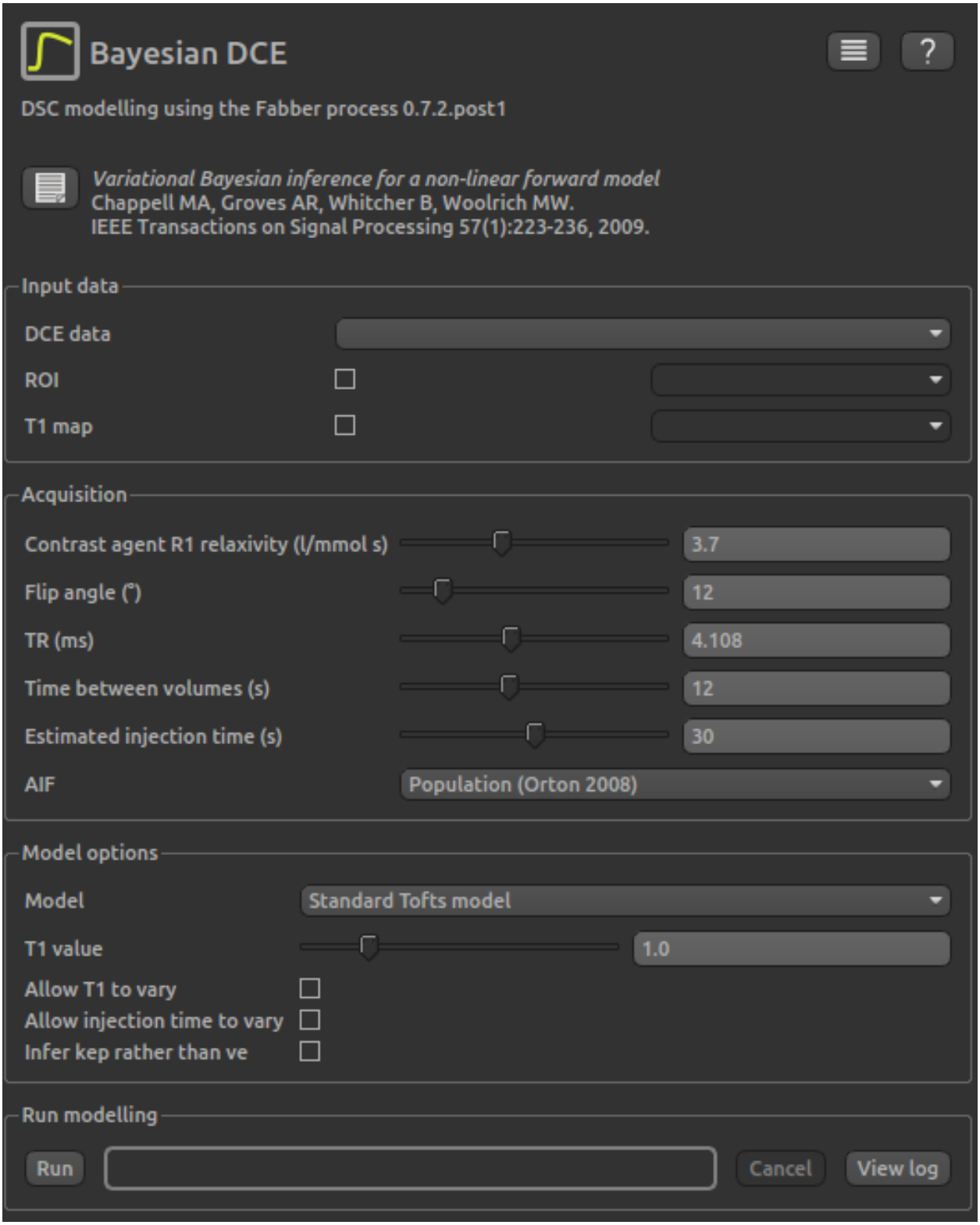
Now. Click Run.

After the analysis is complete, we will be able to see the results on the left panel.

Try to run the analysis on the noisy data from the simulation (fabber_test_data). After the analysis is complete, use the Data Statistics tool to check the quantification results of each parameter.

Reference

DCE modelling widget user interface



Input data

- `DCE data` is used to select the data set containing the 4D DCE time series
- `ROI` is used to select an optional region of interest data set
- `T1` is used to select an optional T1 map (e.g. derived from VFA images using the T1 widget). If a T1 map is not provided a single T1 value must be specified. This can be allowed to vary on a voxelwise basis as part of the fitting process.

Acquisition

- `Contrast agent relaxivity` should be the T1 relaxivity from the manufaturer's documentation. A list of commonly used agents and their relaxivities is also given at List of relaxivities.
- `Flip angle` is defined by the acquisition parameters and should be given in degrees.
- Similarly `TR` is the repetition time for the acquisition and should be given in milliseconds.
- `Time between volumes` should be given in seconds. In some cases this may not be fixed as part of the acquisition protocol, but instead a series of volumes acquired, each with the time at which it was acquired. In this case you must determine a sensible time difference to use, for example by dividing the total acquisition time by the number of volumes acquired.
- `Estimated injection time` is the time delay between the first acquisition and the introduction of the DCE contrast agent in seconds. The latter is often not given immediately in order to establish a baseline signal. The delay time can be estimated as part of the modelling process - this is recommended to account for not just injection delay but also the variable transit time of the contrast agent to different voxels.

Model options

- A selection of models are available - see *Models available for Bayesian DCE modelling* for a full description.
- Similarly the choice of AIF is described in *AIF options for Bayesian DCE modelling*.
- A T1 value in seconds must be given if a T1 map is not provided in the input data section.
- If `Allow T1 to vary` is selected the T1 value (whether from a T1 map or the value given in this section) is allowed to vary slightly on a voxelwise basis. In general if you do not have a T1 map it is recommended to allow the T1 to vary to reflect variation within the region being modelled. If you do have a T1 map you may choose to treat this as ground truth and not allow it to vary in the modelling.
- If `Allow injection time to vary` is selected, then the delay time from the start of the acquisition to the arrival of the DCE tracer is inferred as part of the model fitting. Usually this should be enabled as described above, to account for variable transit times to different voxels.
- For the Tofts model, there is a choice to infer K_{ep} rather than V_e. These are equivalent parameters related by the equation $V_e = K_{trans}/K_{ep}$. Sometimes one choice may be more numerically stable than the other.

Models available for Bayesian DCE modelling

Currently five models DCE models are available in the Bayesian DCE widget. These models are implemented using the Fabber. model fitting framework[1]. More details about the implementation of these models is given in the Fabber DCE documentation.

[1] *Chappell, M.A., Groves, A.R., Woolrich, M.W., "Variational Bayesian inference for a non-linear forward model", IEEE Trans. Sig. Proc., 2009, 57(1), 223–236.*

The standard and extended one-compartment Tofts model[2]

The Extended Tofts model differes from the standard model in the inclusion of the V_p parameter.

The two-compartment exchange model (2CXM)[3]

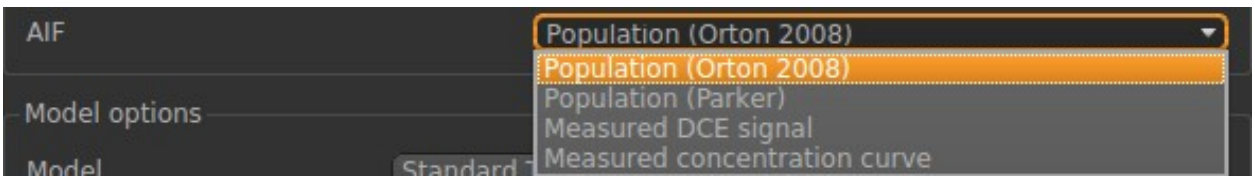
The Compartmental Tissue Uptake model (CTU)[4]

The Adiabatic Approximation to the Tissue Homogeniety model (AATH)[5]

References

AIF options for Bayesian DCE modelling

The arterial input function (AIF) is a critical piece of information used in performing blood-borne tracer modelling, such as DCE-MRI. It describes the arterial supply of contrast agent to the tissue. Quantiphyse supposrts a number of AIF options in the analysis.



The AIF can be described as a series of values giving either the concentration or the DCE signal at the same time intervals used in the DCE acquisition. In this case, the type of AIF is `Measured DCE signal` or `Measured concentration curve`. Note that applying an offset time to the AIF to account for injection and transit time is not required as the model can be given and/or infer a delay time to account for this. This type of AIF is usually measured for the particular subject by averaging the signal in voxels believed to be close to pure arterial voxels, i.e. in a major artery.

Alternatively 'population' AIFs can be used. These are derived from the measurement of AIFs in a large number of subjects and fitting the outcome to a simple mathematical function. This avoids the need to measure the AIF individually for each subject, and avoids additional subject variation associated with this additional measurement. However a population AIF may not reflect the individual subject's physiology particularly when studying a group in which arterial transit may be slower or subject to greater dispersion than the general population.

Two population AIFs are provided as derived by Orton (2008)[1] and Parker (2006)[2]. They can be specified using the `Population (Orton 2008)` or `Population (Parker)` respectively. These are parameterised functions and in our implementation we used the parameter values defined in the respective papers.

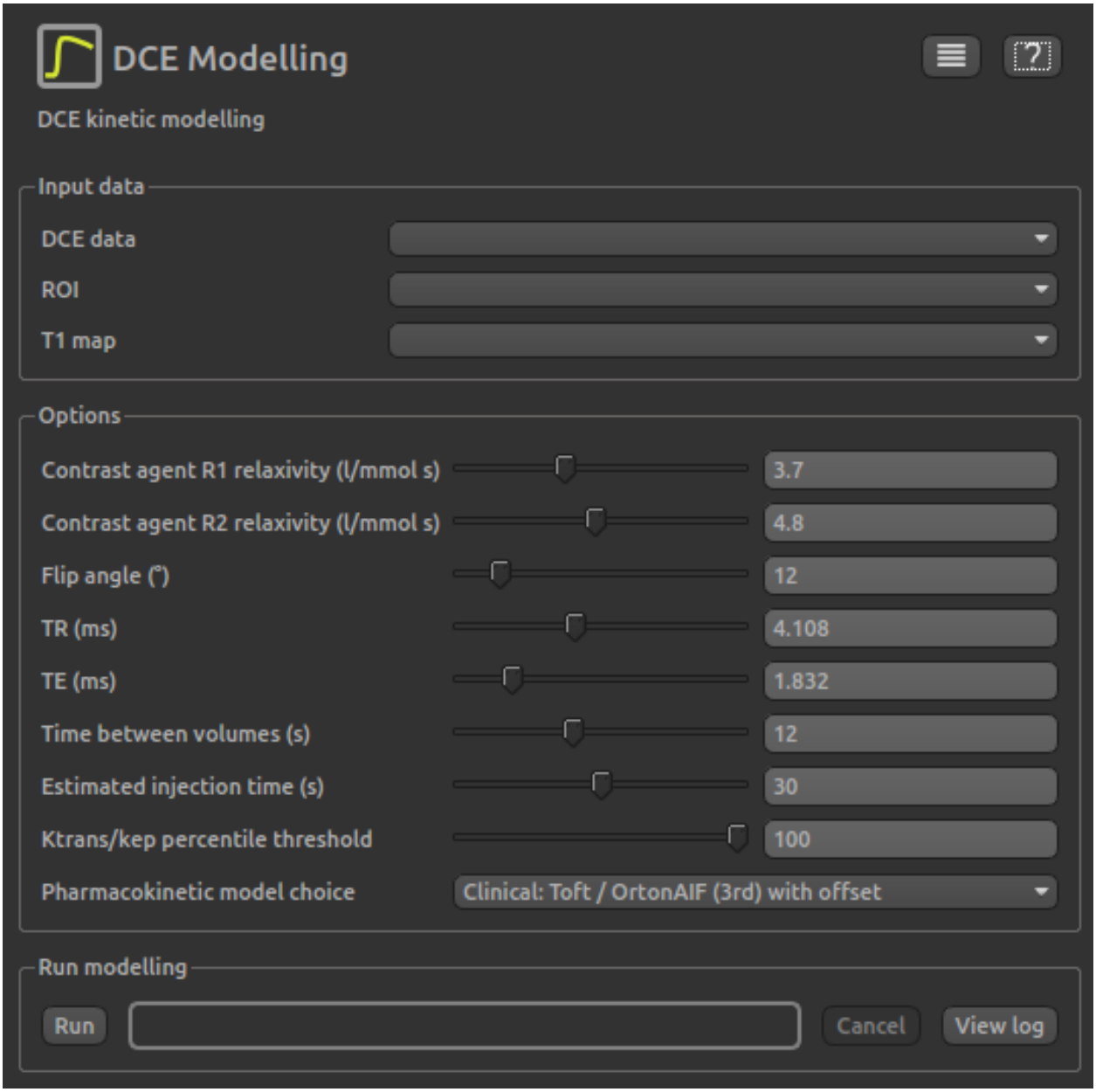
References

The Bayesian DCE widget supports a number of models of varying complexity and a choice of population AIFs or a measured AIF signal.

[2] http://www.paul-tofts-phd.org.uk/DCE-MRI_siemens.pdf
[3] https://onlinelibrary.wiley.com/doi/full/10.1002/mrm.25991
[4] https://onlinelibrary.wiley.com/doi/full/10.1002/mrm.26324
[5] https://journals.sagepub.com/doi/10.1097/00004647-199812000-00011
[1] Matthew R Orton et al 2008 Phys. Med. Biol. 53 1225
[2] https://onlinelibrary.wiley.com/doi/full/10.1002/mrm.21066

5.4.2 Least-squares DCE modelling

The DCE modelling widget performs pharmacokinetic modelling for Dynamic Contrast-Enhanced MRI (DCE) using the Tofts model. Fitting is performed using a simple least-squares technique which limits the range of parameters which can be inferred and the complexity of models which can be implemented. For a more flexible DCE modelling process see *Bayesian DCE modelling*.



Input data

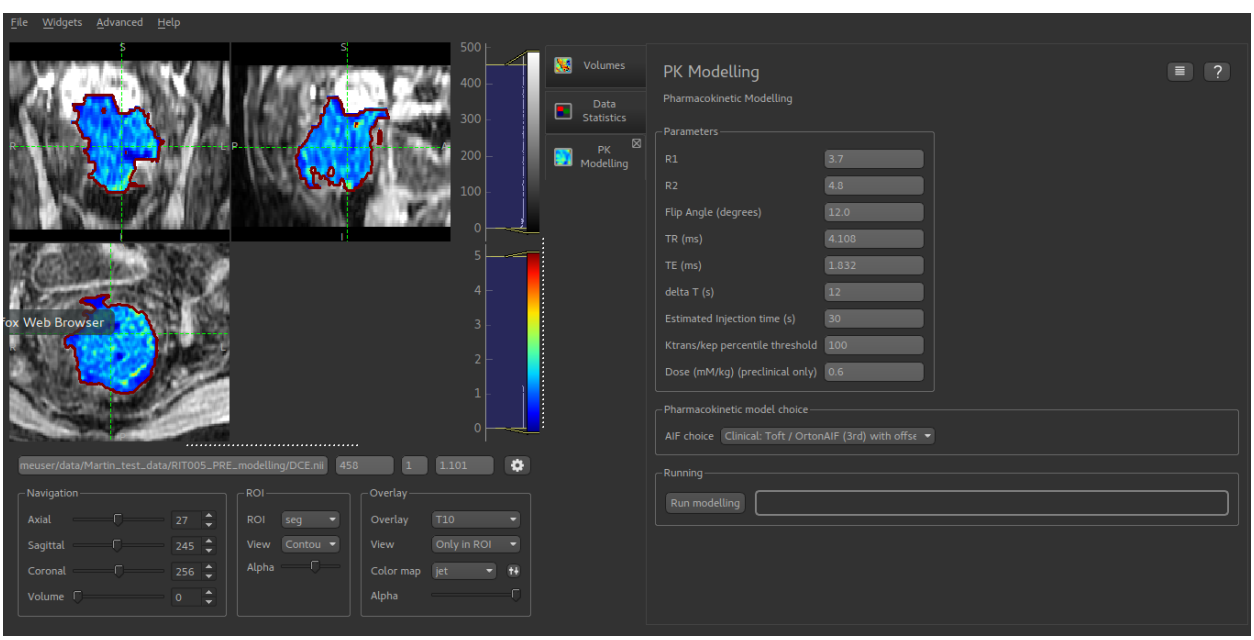
- `DCE data` is used to select the data set containing the 4D DCE time series
- `ROI` is used to select the region of interest data set
- `T1` is used to select a T1 map which is required for the modelling process. This might be derived from VFA images using the T1 widget or by some other method, e.g. saturation recovery.

Options

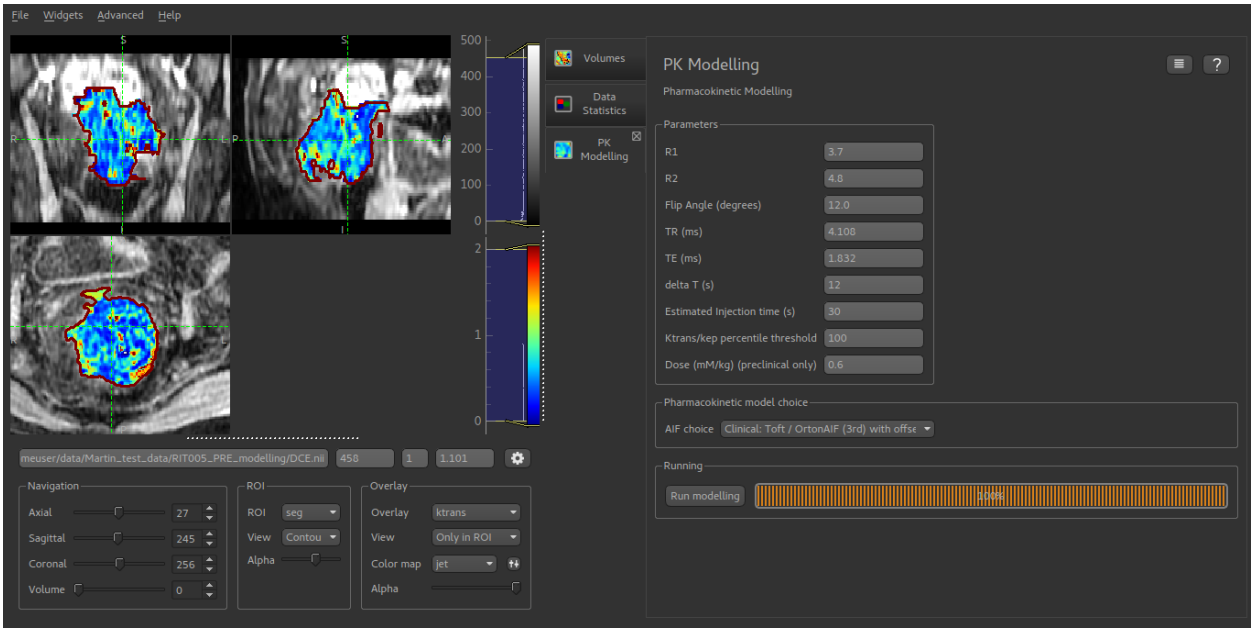
- `Contrast agent R1/R2 relaxivity` should be the T1 and T2 relaxivites from the manufaturer's documentation. A list of commonly used agents and their relaxivities is also given at List of relaxivities.
- `Flip angle` is defined by the acquisition parameters and should be given in degrees.
- `TR` is the repetition time for the acquisition and should be given in milliseconds.
- Similarly `TE` is the echo time for the acquisition and should be given in milliseconds.
- `Time between volumes` should be given in seconds. In some cases this may not be fixed as part of the acquisition protocol, but instead a series of volumes acquired, each with the time at which it was acquired. In this case you must determine a sensible time difference to use, for example by dividing the total acquisition time by the number of volumes acquired.
- `Estimated injection time` is the time delay between the first acquisition and the introduction of the DCE contrast agent in seconds. The latter is often not given immediately in order to establish a baseline signal.
- `ktrans / kep percentile threshold` limits the maximum value of K_{trans}/K_{ep}. This is equal to `V_e` and hence in theory should never exceed 1.0. By reducing this value the effective maximum value of `V_e` can be limited.
- `Pharmacokinetic model choice` selects the combination of model an AIF to use in the modelling. Choices available are:
  - `Clinical Tofts/Orton` - Tofts model using Orton (2008) AIF.
  - `Clinical Tofts/Orton` - As above but with baseline signal offset (recommended)
  - `Preclinical Tofts/Heilman` - Tofts model with preclinical AIF from Heilman
  - `Preclinical Ext Tofts/Heilman` Extended Tofts model with preclinical AIF from Heilman

Screenshots

Start of modelling, showing loaded T10 map



Modelling complete with newly generated :math:'K_{trans}' map



The Least-squares DCE widget supports the basic Tofts model and population AIFs for clinical and preclinical applications.

The interface to the two widgets has been kept as similar as possible to facilitate comparison of the methods, however generally the Bayesian approach is preferred for clinical applications as it provides a greater range of model and AIF options.

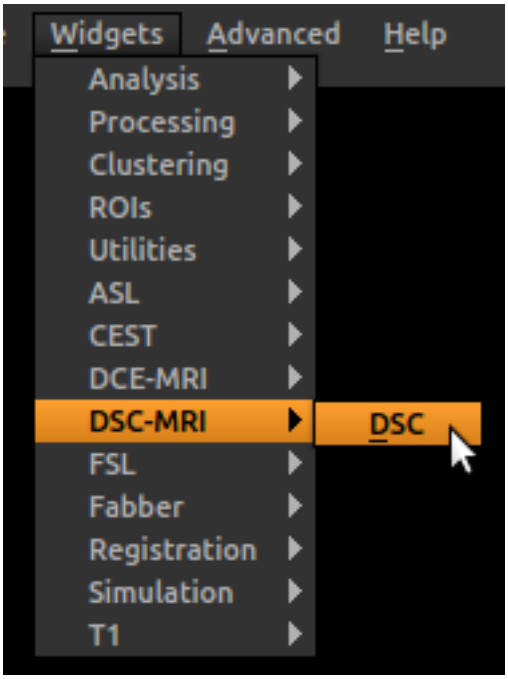
5.4.3 Publications

The following publications are useful citations for the DCE processing widget:

- **Bayesian inference method**: *Chappell MA, Groves AR, Whitcher B, Woolrich MW. Variational Bayesian inference for a non-linear forward model. IEEE Transactions on Signal Processing 57(1):223-236, 2009.*

5.5 Dynamic Susceptibility Contrast (DSC) MRI

Widgets -> DSC-MRI



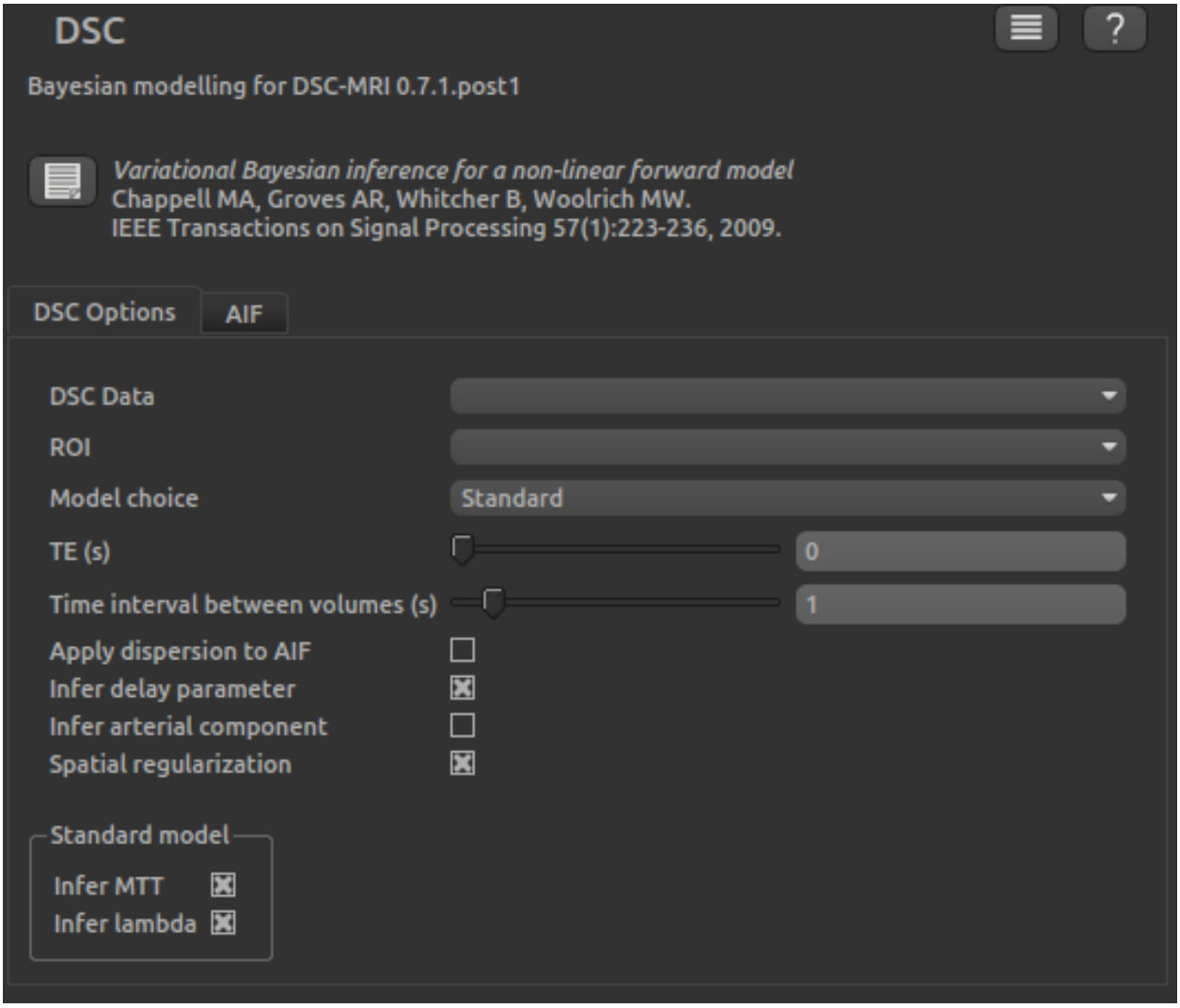
The DSC-MRI package provides a Bayesian modelling tool for quantification of perfusion and other haemodynamic parameters from Dynamic Susceptibility Contrast perfusion MRI of the brain.

5.5.1 Tutorials

- Tutorial coming soon

5.5.2 Reference

DSC modelling widget user interface



DSC options

- `DSC data` is used to select the data set containing the 4D DSC time series
- `ROI` is used to select the region of interest data set
- `Model choice` selects the model to be used for the inference. See *The DSC Vascular Model* for a description of the models available.
- `TE` is the echo time of the acquisition
- `Time interval between volumes` should be given in seconds. In some cases this may not be fixed as part of the acquisition protocol, but instead a series of volumes acquired, each with the time at which it was acquired. In this case you must determine a sensible fixed time difference to use, for example by dividing the total acquisition time by the number of volumes acquired.
- If `Apply dispersion to AIF` is selected, the model is modified to account for dispersion of the tracer within the blood during transit.

- If `Infer delay parameter` is selected, the arrival time of the tracer in each voxel is estimated (recommended).
- If `Infer arterial component` is selected, contamination of the DSC signal by tracer in arteries is included in the model.
- If `Spatial regularization` is selected, adaptive spatial smoothing on the output parameter maps is performed using a Bayesian framework where the spatial variability of the parameter is inferred from the data (recommended).

Standard model options

- If `Infer MTT` is selected the mean transit time of the tracer is estimated

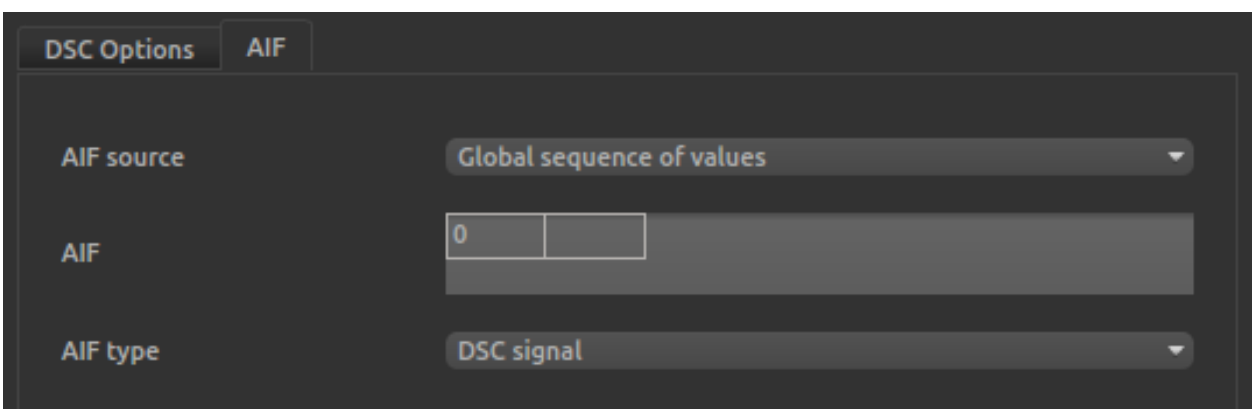
CPI model options

- `Number of control points` selects the number of evenly spaced control points that will be used to model the residue function.
- `Infer control point time position` can be used to allow the control points to move their temporal position rather than being fixed in their original evenly spaced position. This may enable an accurate residue curve with fewer control points, but can also lead to numerical instability.

AIF options for DSC modelling

The arterial input function (AIF) is a critical piece of information used in performing blood-borne tracer modelling, such as DCE-MRI. It describes the arterial supply of contrast agent to the tissue.

The AIF can be described as a series of values giving either the concentration or the DSC signal at the same time intervals used in the DSC acquisition. Note that applying an offset time to the AIF to account for injection and transit time is not required as the model can be given and/or infer a delay time to account for this. This type of AIF is usually measured for the particular subject by averaging the signal in voxels believed to be close to pure arterial voxels, i.e. in a major artery.



The DSC AIF can be supplied in two different ways, selected by the `AIF source` option. In each case the supplied AIF may either be a set of DSC signal values, or a set of tracer concentration values. The `AIF type` option selects between these two possibilities

Global sequence of values

In this case each voxel has the same AIF which is supplied as a series of values giving either the concentration or the DSC signal at the same time intervals used in the DSC acquisition.

The series of values must be pasted into the `AIF` entry widget. A text file containing the values can be drag/dropped onto this entry as a convenient way of entering the values.

Voxelwise image

In this case a 4D image must be supplied which, at each voxel, contains the AIF for that voxel. This allows for the possibility of the AIF varying at each voxel.

The DSC Vascular Model

The standard vascular model

The model used is a specific physiological model for capillary transit of contrast within the blood generally termed the 'vascular model' that was first described by Ostergaard[1][2]. This model has been extended to explicitly infer the mean transit time and also to optionally include correction for macro vascular contamination - contrast agent within arterial vessels[3].

An alternative to the model-based approach to the analysis of DSC-MRI data are 'non-parametric' approaches, that often use a Singular Value based Deconvolution to quantify perfusion.

The CPI model

A key component of the DSC model is the *residue function* which describes the probability that a molecule of tracer that entered a voxel at $t = 0$, is still inside that voxel at a later time t. As constructed this is a monotonically decreasing function whose value at $t = 0$ is 1 and which approaches 0 as $t \to \infty$.

The CPI model[4] describes the residue function by performing cubic interpolation between a set of *control points*. The control point at $t = 0$ has a fixed value of 1 but the remaining control points are limited only by the fact that each cannot take a larger value than the preceding one. By treating the control point values as model parameters to infer we can infer the shape of the residue function without giving it an explicit mathematical form.

By default the CPI model uses a fixed set of evenly spaced control points. In principle we can also allow these points to move 'horizontally', i.e. change their time position. By doing so we might expect to be able to model the residue function realistically with a smaller number of control points. In practice, while this is supported by the model it can result in numerical instability which is linked to the fact that we need to prevent control points from crossing each other, or degenerating to a single point. Adding a larger number of evenly spaced fixed control points can be a better solution in this case.

[1] *Mouridsen K, Friston K, Hjort N, Gyldensted L, Østergaard L, Kiebel S. Bayesian estimation of cerebral perfusion using a physiological model of microvasculature. NeuroImage 2006;33:570–579. doi: 10.1016/j.neuroimage.2006.06.015.*

[2] *Ostergaard L, Chesler D, Weisskoff R, Sorensen A, Rosen B. Modeling Cerebral Blood Flow and Flow Heterogeneity From Magnetic Resonance Residue Data. J Cereb Blood Flow Metab 1999;19:690–699.*

[3] *Chappell, M.A., Mehndiratta, A., Calamante F., "Correcting for large vessel contamination in DSC perfusion MRI by extension to a physiological model of the vasculature", e-print ahead of publication. doi: 10.1002/mrm.25390*

[4] *Mehndiratta A, MacIntosh BJ, Crane DE, Payne SJ, Chappell MA. A control point interpolation method for the non-parametric quantification of cerebral haemodynamics from dynamic susceptibility contrast MRI. NeuroImage 2013;64:560–570. doi: 10.1016/j.neuroimage.2012.08.083.*

References

5.5.3 Publications

The following publications are useful citations for the DSC processing widget:

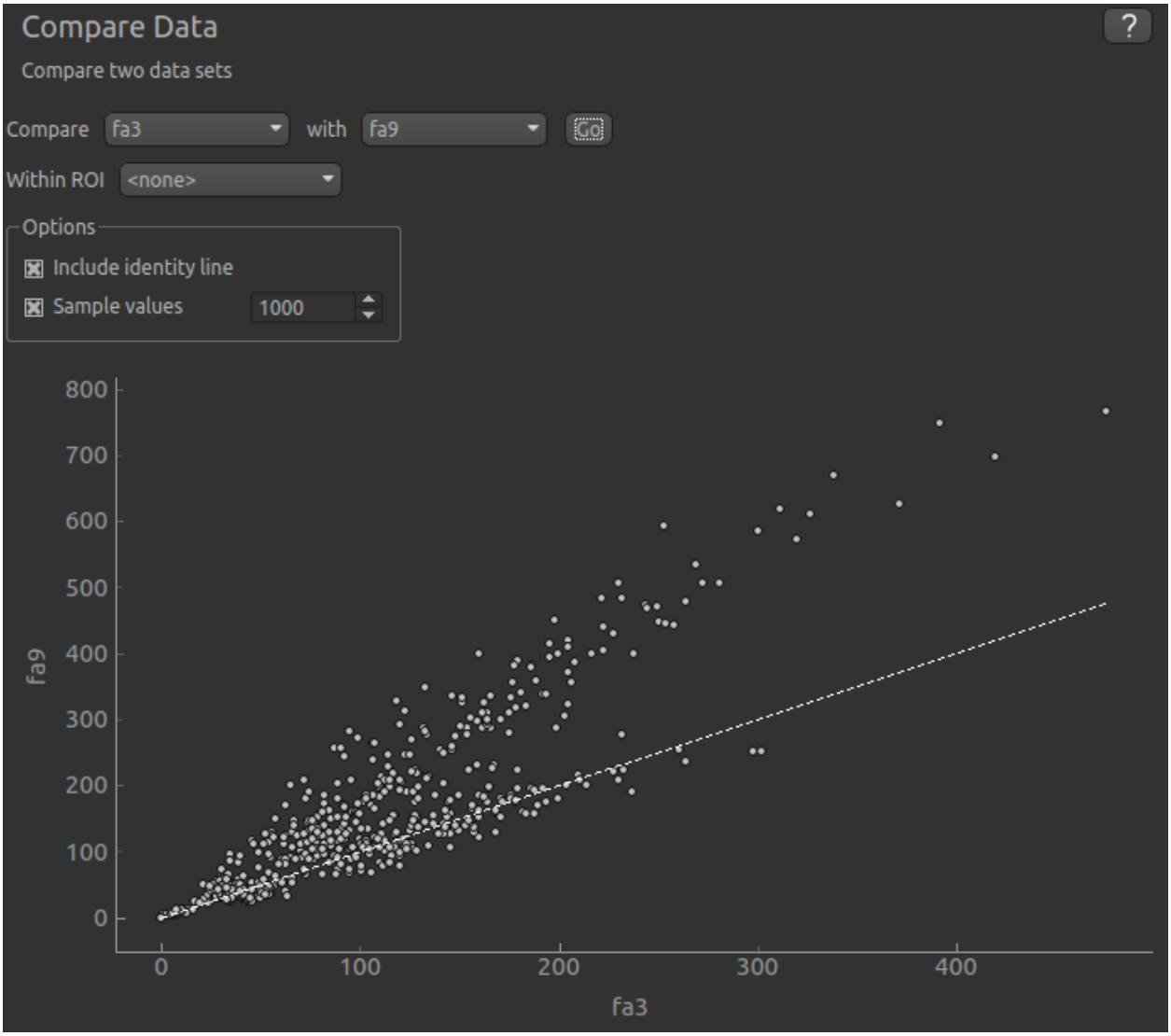
- **Bayesian inference method**: *Chappell MA, Groves AR, Whitcher B, Woolrich MW. Variational Bayesian inference for a non-linear forward model. IEEE Transactions on Signal Processing 57(1):223-236, 2009.*
- **Arterial signal correction**: *Chappell, M.A., Mehndiratta, A., Calamante F., "Correcting for large vessel contamination in DSC perfusion MRI by extension to a physiological model of the vasculature", e-print ahead of publication. doi: 10.1002/mrm.25390*
- **DSC vascular model**: *Mouridsen K, Friston K, Hjort N, Gyldensted L, Østergaard L, Kiebel S. Bayesian estimation of cerebral perfusion using a physiological model of microvasculature. NeuroImage 2006;33:570–579. doi: 10.1016/j.neuroimage.2006.06.015.*
- **CPI model**: *Mehndiratta A, MacIntosh BJ, Crane DE, Payne SJ, Chappell MA. A control point interpolation method for the non-parametric quantification of cerebral haemodynamics from dynamic susceptibility contrast MRI. NeuroImage 2013;64:560–570. doi: 10.1016/j.neuroimage.2012.08.083.*

5.6 Visualisation and processing tools

5.6.1 Compare data

From menu: `Widgets -> Visualisation -> Compare Data`

This widget shows a comparison between two data sets. Select the two data sets you are interested in from the menus and click `Go` to display a scatter plot of corresponding voxel values.



Options

ROI

This option restricts the comparison to voxels within a specified ROI. You should use this option if the data of interest does indeed lie within an ROI, otherwise the sample of points compared will include many irrelevant 'outside ROI' voxels and therefore the comparison will be less reliable.

Sample points

By default only a random sample of 1000 points is displayed. This is because the scatter plot can take a long time to generate otherwise. You can choose the number of points in the sample. If you want to use all values in comparison, turn off the sample, but be aware that the plot may take some time to generate, particularly for large or 4D data sets.

Show identity line

A dotted identity line can be shown in cases where you want to compare the data for equality. In the example above while there is some degree of linear relationship between the data, it is not perfect and the trend does not match the identity line.

Heat map (experimental)

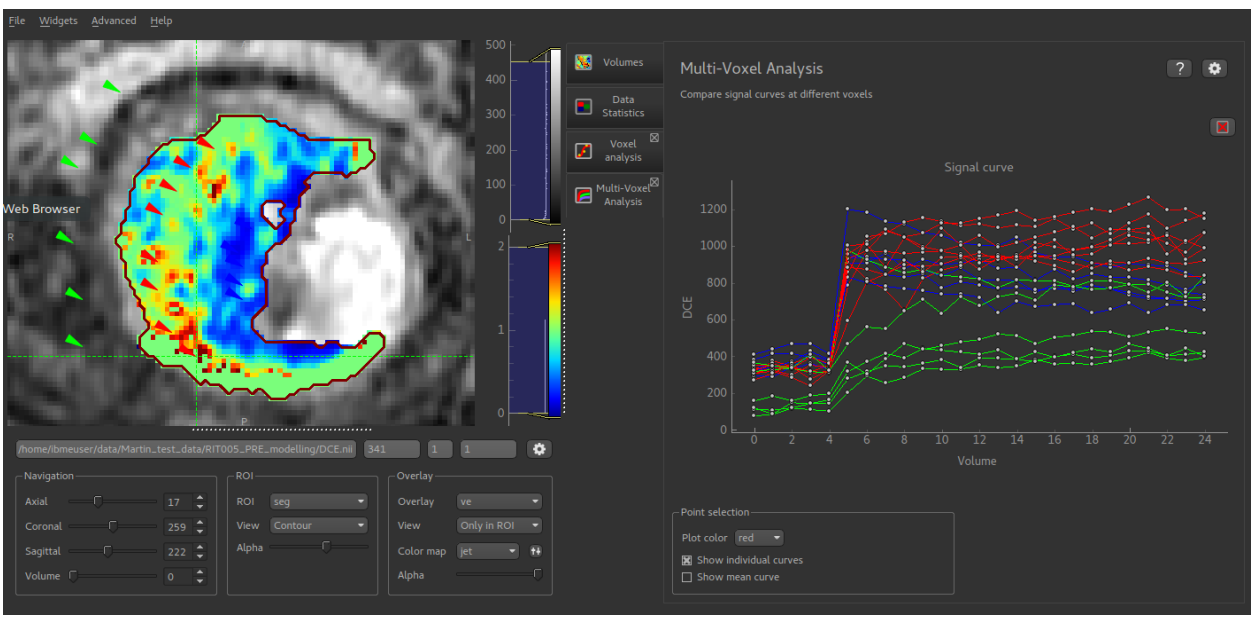
This changes to scatter plot to a heat map. This tool is experimental and you may need to tweak the graph colour map to get a good result. The advantage of a heat map over a scatter plot is that when many points lie very close to each other it can be difficult to tell how much greater the point density is from a scatter plot.

For example, if you are comparing two data sets for equality it may look like there are a large number of inconsistent voxels far away from the identity line. However in practice these may actually be a very small proportion of the total voxels but appear more prominent in a scatter plot because they are not close to other points. Switching to a heat map may show that in fact the vast majority of the data is along the identity line.

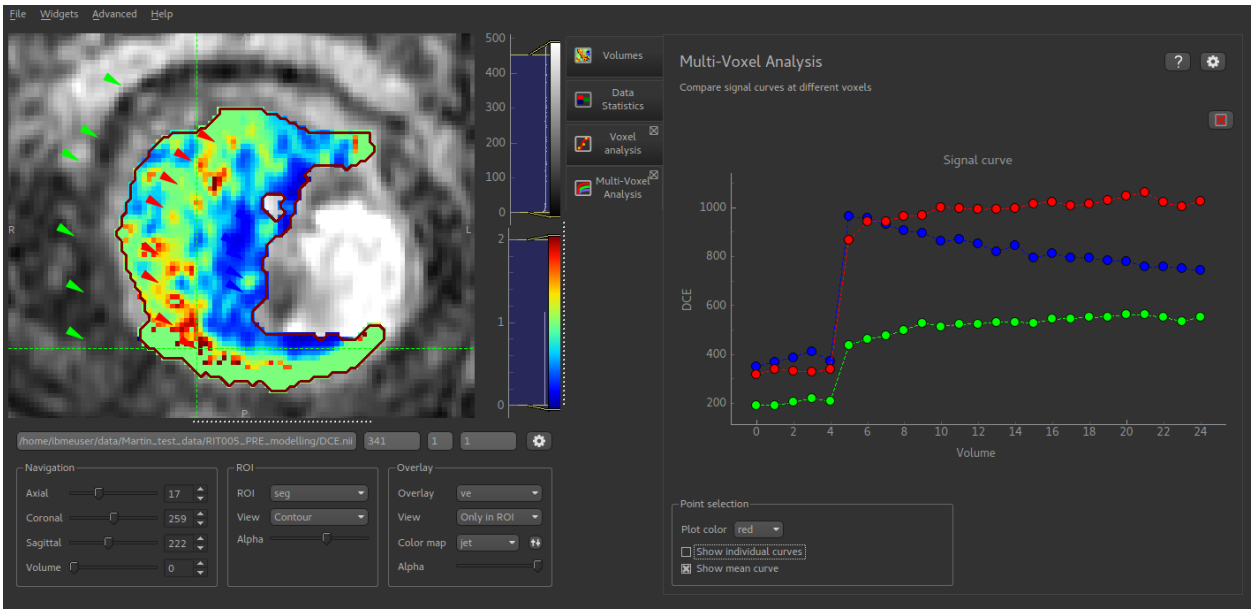
5.6.2 Multi-voxel visualisation

Widgets -> Visualisation -> Multi-voxel

This widget shows the signal-time curve at multiple locations.



- Each click on the image adds a new curve to the plot. By changing the colour, a series of curves can be plotted enabling different parts of the image to be compared
- The plot can be cleared by clicking on the red X at the top right of the window
- The mean curve for each color can also be displayed. This is shown with large circular markers and a dotted line. This can be displayed with the individual curves, or on its own (as below)

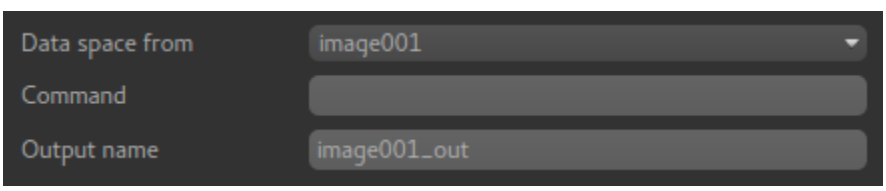


Additional plot options are available by clicking the Options button in the top right.

5.6.3 Simple maths

Widgets -> Processing -> Simple Maths

This widget is a simplified version of the console and allows new data to be created from simple operations on existing data.



The data space for output must be specified by selecting a data set - this is necessary because it's not generally possible to analyse the expression and determine the output space. Usualy the output data space will match the data space of the data sets used in the input.

The `Command` text entered must be a valid Python expression and can include the names of existing ROIs and overlays which will be Numpy arrays. Numpy functions can be accessed using the `np` namespace. Some knowledge of the Numpy library is generally needed to use this widget effectively.

An output name for the data set is also required.

Examples

Add Gaussian noise to some data:

```
mydata + np.random.normal(0, 100)
```

Calculate the difference between two data sets:

```
mydata1 - mydata2
```

Scale data to range 0-1:

```
(mydata - mydata.min()) / (mydata.max() - mydata.min())
```

5.6.4 Registration and Motion Correction

Widgets -> Processing -> Registration

This widget enables registration and motion correction using various methods. Currently implemented methods are:

- DEEDS - a nonlinear fully deformable registration method
- FLIRT/MCFLIRT - a linear affine/rigid body registration method - requires an FSL installation
- FNIRT a nonlinear registration method from FSL

Additional packages may be required to support these methods - you will need to install `quantiphyse_deeds` for the first, while `quantiphyse_fsl` package plus a working FSL installation are required for the second and third.

Registration mode

The registration mode selects between registration and motion correction mode. The difference between the two is that:

- In motion correction mode the reference data is derived from the registration data
- In motion correction mode only 4D data may be registered
- In motion correction mode it is not possible to apply the transformation to other data sets (because there are multiple transformations, one for each 4D volume!)

Registration methods may choose to implement motion correction differently to registration, for example in the latter they might constrain the degree of change to physically plausible movements, or they might skip early rough optimisation steps since motion correction data is usually at least close to the reference data. In the case of `FLIRT/MCFLIRT`, `MCFLIRT` is the motion correction variant of the same basic registration method.

Registration data

This is the data you wish to align with another data set or motion correct. It may be 3D or 4D - if it is 4D, each individual volume is registered with the reference data separately.

Reference data

This is the data set the output should align with. It must be 3D, hence if a 4D data set is chosen, a 3D `Reference Volume` must be selected from it. Options are:

- Middle (median) volume
- Mean of all volumes
- Specified volume index

For motion correction, the reference data is the same as the 4D registration data however a specific reference volume must be chosen as above.

Output space

The output of registration may be generated in one of three ways:

- `Reference`, i.e. at the resolution and field of view of the reference data This is the default for most registration methods, e.g. if we register a low-resolution functional MRI image to a high-resolution structural image we normally expect output at the structural resolution.
- `Registration`, i.e. the same space of the original registration data. This may be implemented by resampling the output in reference space onto the reference space.
- `Transformed` - This is only available for linear registration methods on 3D data, and causes the output voxel data to be completely unchanged, however the voxel->world transformation matrix is updated to align with the reference data. This can be useful as it avoids any resampling or interpolation of the data, however bear in mind that any volumetric processing of the data alongside other data sets may require the resampling to be done anyway to ensure all data is defined on the same grid. In general use of this option followed by a resampling onto the reference or registration image data grid is equivalent to the first two methods.

Not all registration methods may support all output space options.

Output name

A custom output name may be selected for the registered/motion corrected data set.

Apply transformation to other data sets

If selected, this is a list of other data sets which the same transformation should be applied to. Note that these data sets are *not* used in the registration process itself. A common use case for this is when an ROI has been drawn on a data set and it is necessary to align the data set with another. In this case, the ROI can be selected as an additional data set and the transformed ROI will align with the transformed data set.

Save Transformation

If selected, the resulting transformation will be saved under the specified name. There are two possible types of transformation:

- Image transformations where the output is an image data set. This is most common for non-linear registration methods (i.e. a warp field)
- Transformations defined by a linear transformation matrix These are stored as Extra objects.

Saved transformations can be written to files and applied to other data sets using *Registration -> Apply Transform* widget. The method used to derive a transformation is stored as metadata within the transformation since in general the transformation can only be applied by the same method it was created by - e.g. you can't take the output warp field from `FNIRT` and use it with the `DEEDS` method.

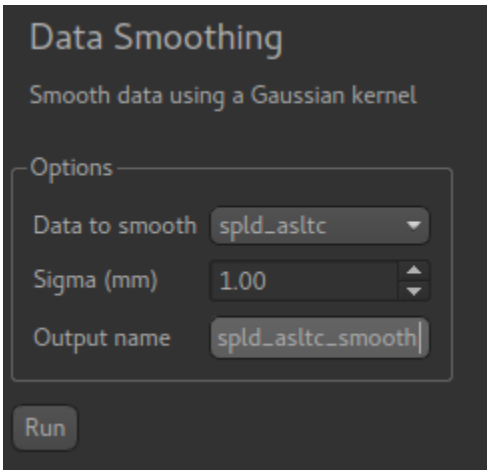
Registration method options

Each registration method has its own set of options which are available when it is selected.

5.6.5 Smoothing

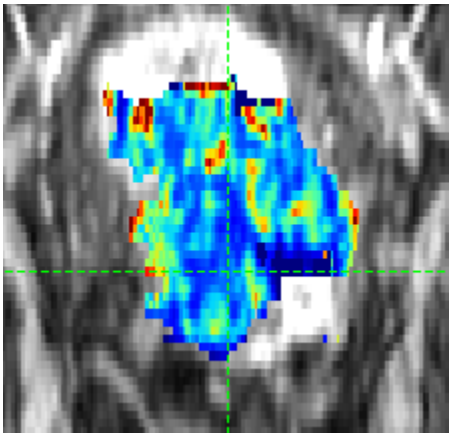
Widgets -> Processing -> Smoothing

This widget provides simple Gaussian smoothing.

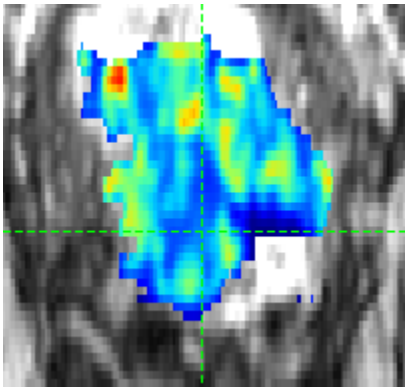


`Sigma` is the standard deviation of the Gaussian used in the convolution in mm. Note that if the voxel size of the data is different in different dimensions then the smoothing in *voxels* will be non-isotropic. For example if you select 1mm as `sigma` with data that consists of 5mm slabs in the Z direction with 1mm resolution in the XY directions, then very little smoothing will be evident in the Z direction, but the XY slices will be visible smoothed.

Sample input



Sample output



5.6.6 K-Means Clustering

Widgets -> Clustering -> KMeans Clustering

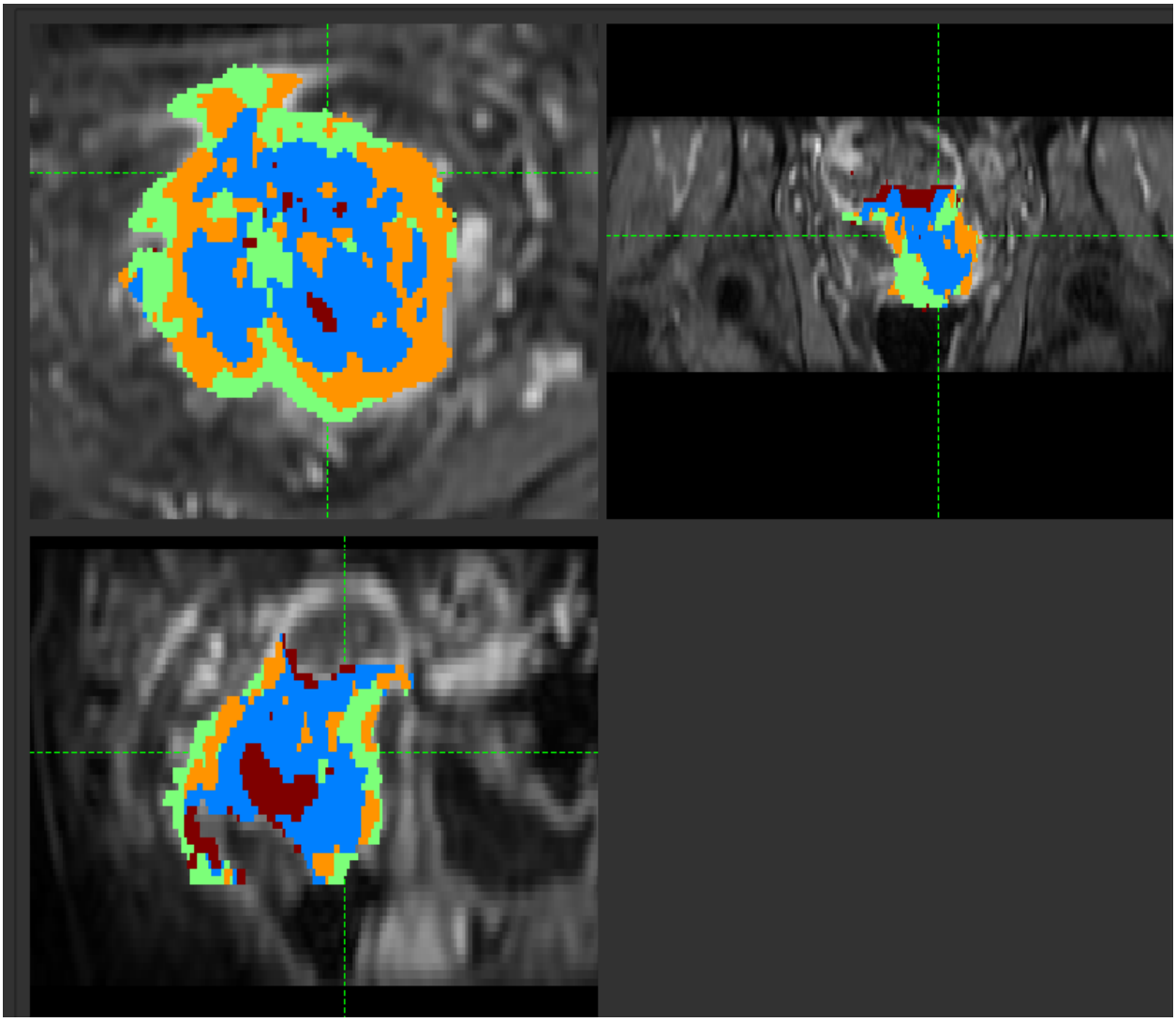
Clustering uses the K-Means algorithm to cluster 3D or 4D data into discrete regions.

When used with 4D data, PCA reduction is used to convert the volume sequence into 3D data before K-Means is applied.

Options

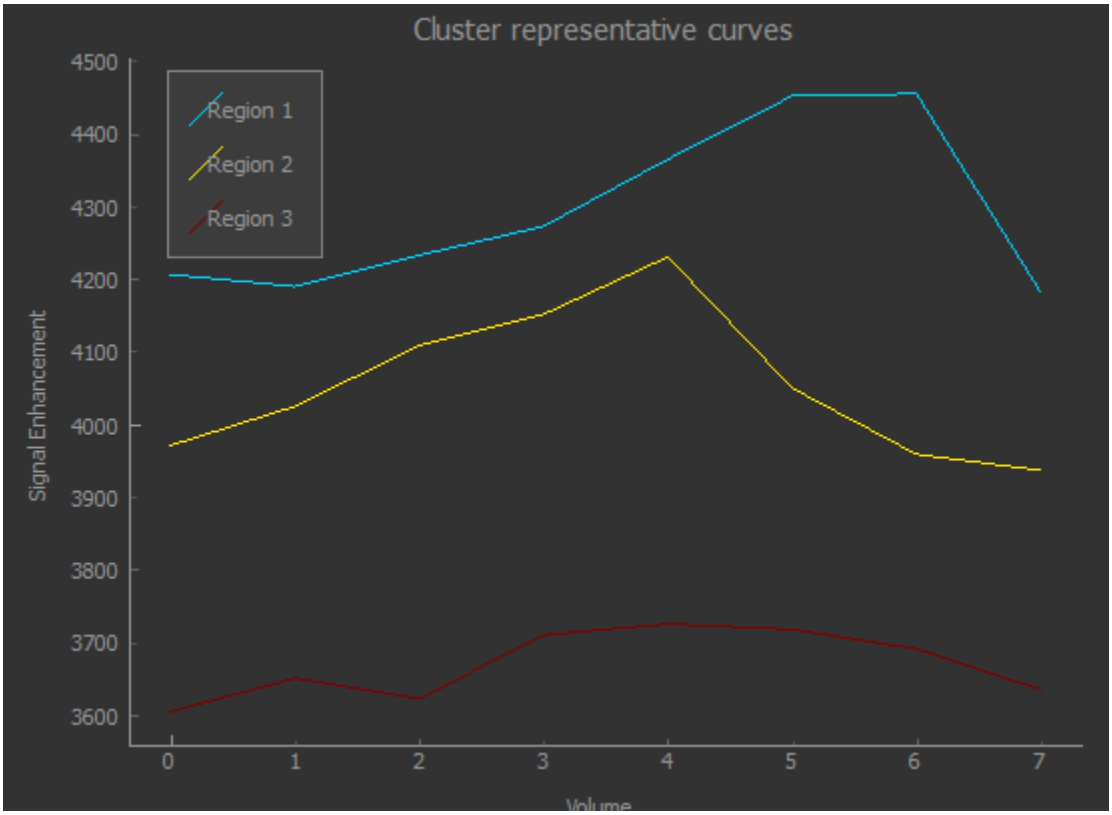
- Data set to use for clustering (defaults to current overlay)
- ROI to cluster within (optional, but recommended for most data)
- Name of the output ROI data set
- Number of clusters, i.e. how many subregions the ROI will be split into
- For 4D data, the number of PCA modes to use for reduction to 3D.

On clicking `Run`, a new ROI is produced with each cluster assigned to an integer ID.



Show representative curves

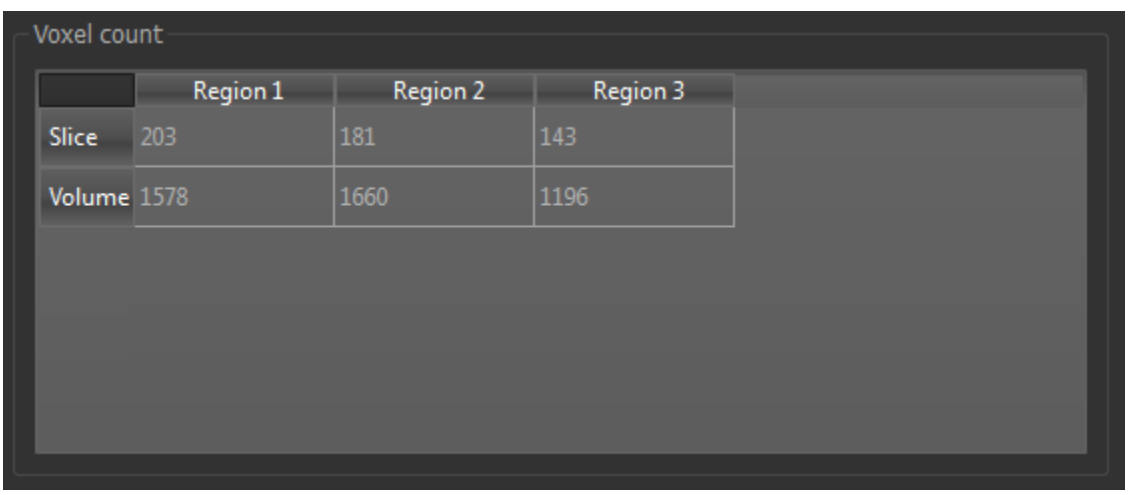
This option is available when clustering 4D data, and displays the mean time-series curves for each cluster.



In this case the clusters correspond to two distinct phase offsets of the signal curve, with a third cluster picking up voxels with weak or no signal.

Show voxel counts

This option shows the number of voxels in each cluster, overall and within the current slice.

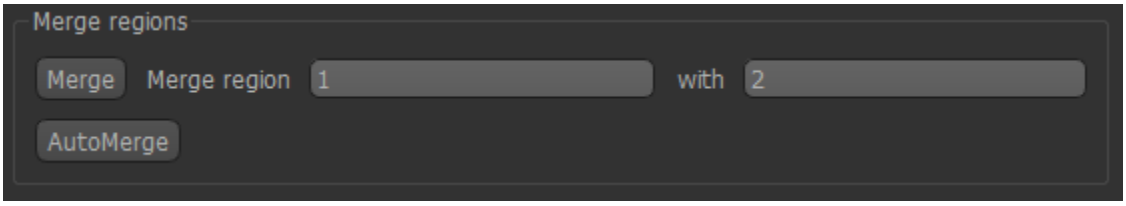


	Region 1	Region 2	Region 3
Slice	203	181	143
Volume	1578	1660	1196

Show merge options

Having generated clusters, it may be desirable to merge some of the subregions - for example if two are very similar, or one contains very few voxels. The `Merge` tool allows you to do this by specifying the two regions to be merged.

An `Auto Merge` tool is also provided to automatically identify subregions for merging.



5.6.7 Supervoxel clustering

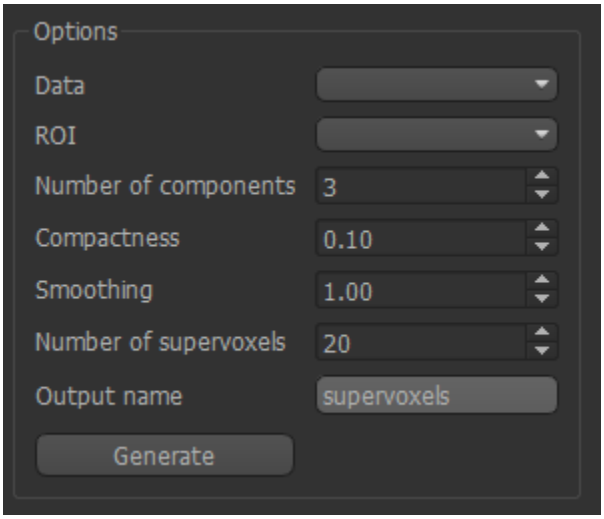
Widgets -> Clustering -> Supervoxels

This widget create supervoxels based a selected data map and a selected ROI.

Supervoxels are collections of voxels which are similar in terms of both data and also spatial location. So, unlike clusters, supervoxels are intended to be connected and localised.

Quantiphyse uses a novel supervoxel method based on SLIC, but modified so that it can be applied sensibly to data within an ROI. For full method details see[1]

Options



The following options are available:

- `Data` can be 3D or 4D data
- `ROI` Select the ROI within which the supervoxels will be generated
- `Number of components` PCA analysis is initially performed to reduce 4D data to a 3D volume, as with the clustering widget. This option controls the number of PCA components and is only visible for 4D data.
- `Compactness` This takes values between 0 and 1 and balances the demands of spatial regularization with similarity of data. A high value will produce supervoxels dominated by spatial location, a low value will produce results similar to clustering with irregular and possibly disconnected supervoxel regions.
- `Smoothing` Degree of smoothing to apply to the data prior to supervoxel generation

[1] *B Irving* maskSLIC: Regional Superpixel Generation with Application to Local Pathology Characterisation in Medical Images https://arxiv.org/abs/1606.09518v2

- `Number of supervoxels` This is the number of seed points which are placed within the ROI, each of which will define a supervoxel.
- `Output name` The data will be output as a new ROI with this name where each supervoxel is a separate numbered region.

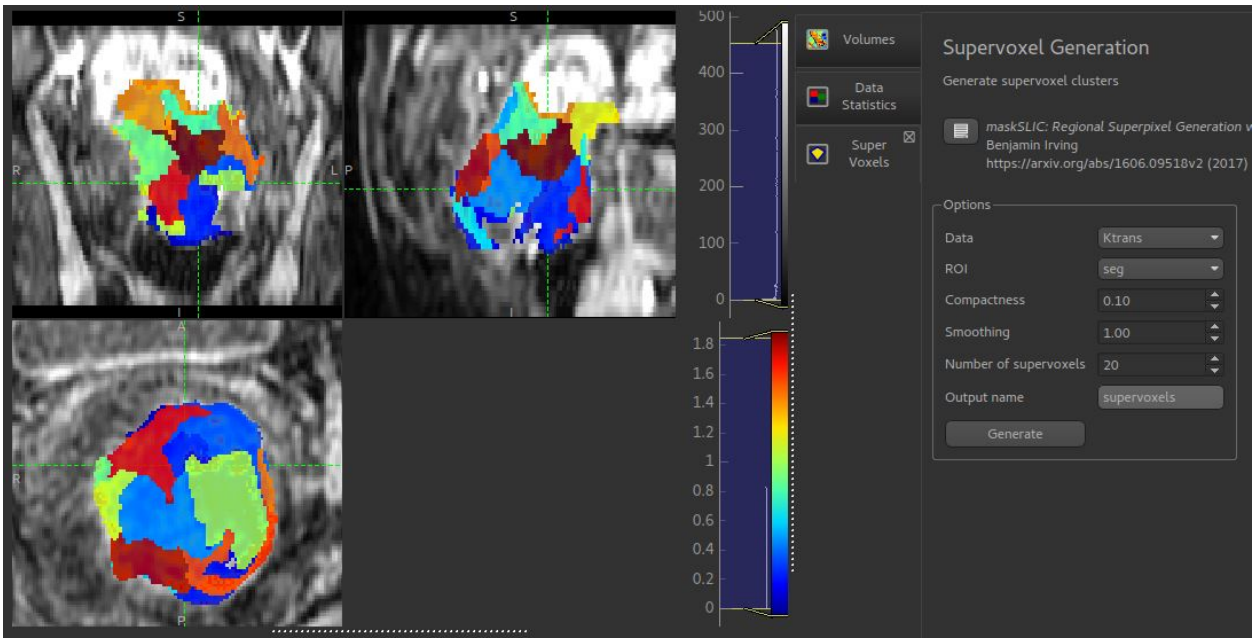
Method

Seed points are placed within the ROI - one for each supervoxel - with initial positions determined by the need to be within the ROI and maximally separated from other seed points and the boundary.

For 4D data, PCA analysis is initially performed as described above. For 3D data, the only preprocessing is a scaling of the data to the range 0-1 to enable parameters such as compactness to have consistent meaning.

The output is an ROI in which each supervoxel is an ROI region. This enables use with for example, the mean values widget, which can replace the data in an overlay with a single value for each supervoxel.

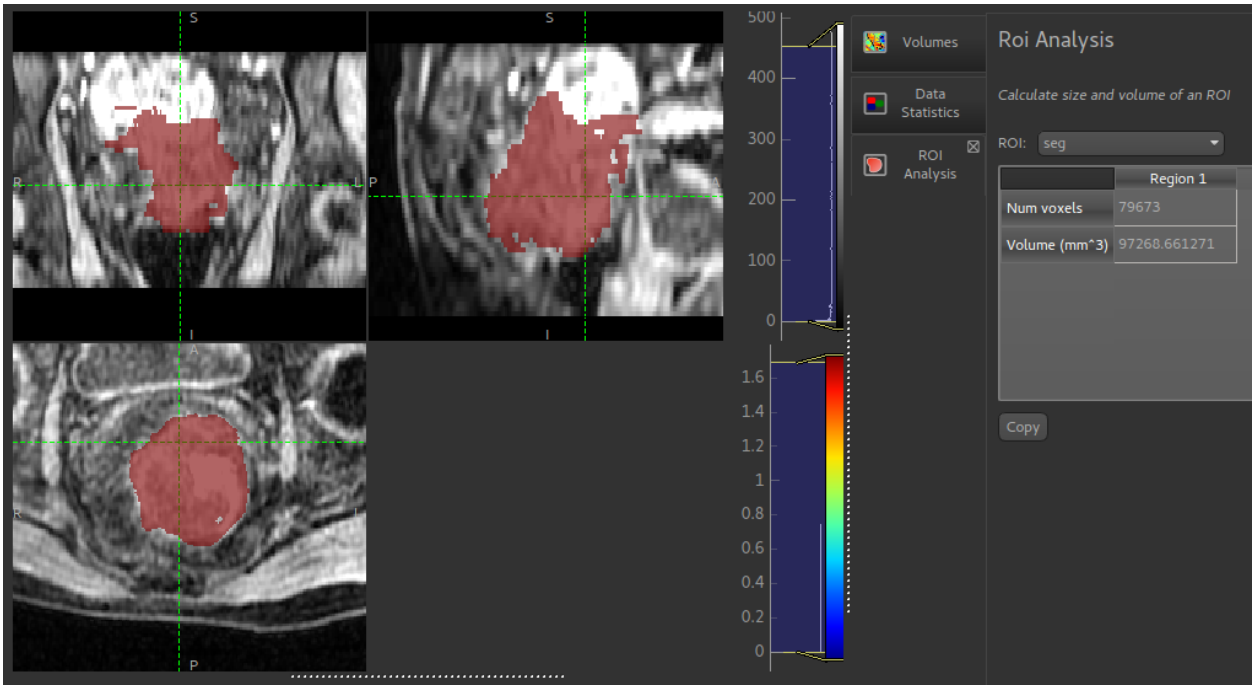
Sample output



5.6.8 ROI Analysis

Widgets -> ROIs -> ROI Analysis

This widget performs a simple calculation of volume and number of voxels within each ROI region.



The output table can be copied and pasted into most spreadsheet applications (e.g. Excel)

5.6.9 ROI Builder

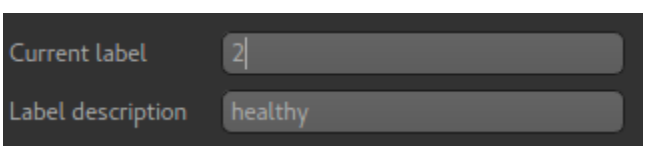
- *Widgets -> ROIs -> ROI Builder*

This widget is designed for simple construction of regions of interest and manual segmentation. It is not designed to be a replacement for sophisticated semi-automatic segmentation tools! However it is very helpful when running intensive analysis processes as you can easily define a small ROI to run test analyses within before you process the full data set.

Basic concept

At the top of the widget, you can choose the ROI to edit. This might be an existing ROI, but often you will want to create a new ROI. To do this, click the `New` button and provide the name of your new ROI, and in addition a data set which defines the data space on which it will be defined.

Labels



All the ROI builder tools either add regions to the ROI or remove them. ROIs can have multiple distinct regions identified by a label (an integer 1 or above - 0 is used to indicate 'outside the ROI'), and an optional name (which appears in some other widgets, e.g. the Data Statistics widget).

The `Current Label` selector can be used to choose what label new ROI regions are added under. You can also edit the name of a label using the `Label Name` edit box. New labels which do not yet exist in the ROI get the default name `Region <n>` where `<n>` is the label identifier.

Tools

Each tool allows you to modify the ROI region - typically (but now always) on a single slice of the image.

Many of the tools work by selecting a region and then have four options as to how to use this region to modify the ROI:

- `Add` means the selected voxels are given the value of the current label
- `Erase` means the selected voxels are removed from the ROI (given a label of 0)
- `Mask` means that voxels *outside* the selected voxels are removed from the ROI (given a label of 0)
- `Discard` means that the selection is removed without affecting the ROI.

Crosshairs

This tool does not modify the ROI at all. Instead, is used to revert to the use of mouse clicks to select points/slices of focus rather than select an ROI region. This is helpful in selecting the slice you are working on without accidentally defining a new ROI region.

Pen

This is a typical tool for manual segmentation. Click and drag to draw a boundary around the region you want to select. Clicking `Add` adds the interior of the region to the ROI. Generally with manual segmentation, you work slice by slice, drawing around the regions as you go. If you are doing this, you may want to maximise one of the viewing windows first. This tool should work with alternative pointing systems, such as touchscreens and drawing tablets.

Paint

With this tool individual voxels can be painted on by clicking and dragging the mouse. The brush size can be modified from 1 (individual voxels) upwards, e.g. `3` will paint a rectangle of 3x3 voxels. Note that painting is 2D only - you cannot paint a 3x3x3 3D block without painting each slice separately.

Eraser

This tool is very similar to the `Paint` tool, only voxels are erased rather than painted. The brush size can be modified to erase larger regions - see the `Paint` tool for more information.

Rectangle

Simple click-and-drag to select a rectangular region. When you are happy, click `Add` to add it to the ROI, or click `Discard` to ignore it.

Ellipse

Identical to the `Rectangle` tool, but selects an elliptical region

Polygon

In this tool, each click on the image adds a vertex of a polygon region. When you click `Add` the last node is connected to the first node to close the polygon, and the interior is selected. Clicks within a different slice window are ignored.

Choose Region

This tool allows you to choose a region of an existing ROI - for example to isolate a particular cluster or supervoxel.

Using the menu, select the existing ROI and then click on a point to choose the region it lies within. The region will be displayed in isolation and you can choose to 'Accept' or 'Cancel' the selection.

Walker

This provides simple automatic segmentation using the random walk algorithm. Mouse clicks select points known to be inside (red flags) or outside (white flags) the region of interest - a menu allows you to change between these modes. When some points have been selected, the `Segment` button will generate an ROI which includes the red flags and excludes the white flags.

This process can be carried out on a slice-by-slice basis, or across the whole 3D volume - the `segmentation Mode` menu allows you to choose which.

Bucket fill

This provides simple 3D 'bucket fill' tool. The image view is clicked to select the seed point, then the tool selects all voxels which can be reached by moving from the seed point within the upper and lower threshold. There is also a max distance control to prevent the fill from progressing too far.

Undo

Most changes can be undone by clicking on the `Undo` button. Generally the last 10 additions or removals can be undone.

5.6.10 Mean value widget

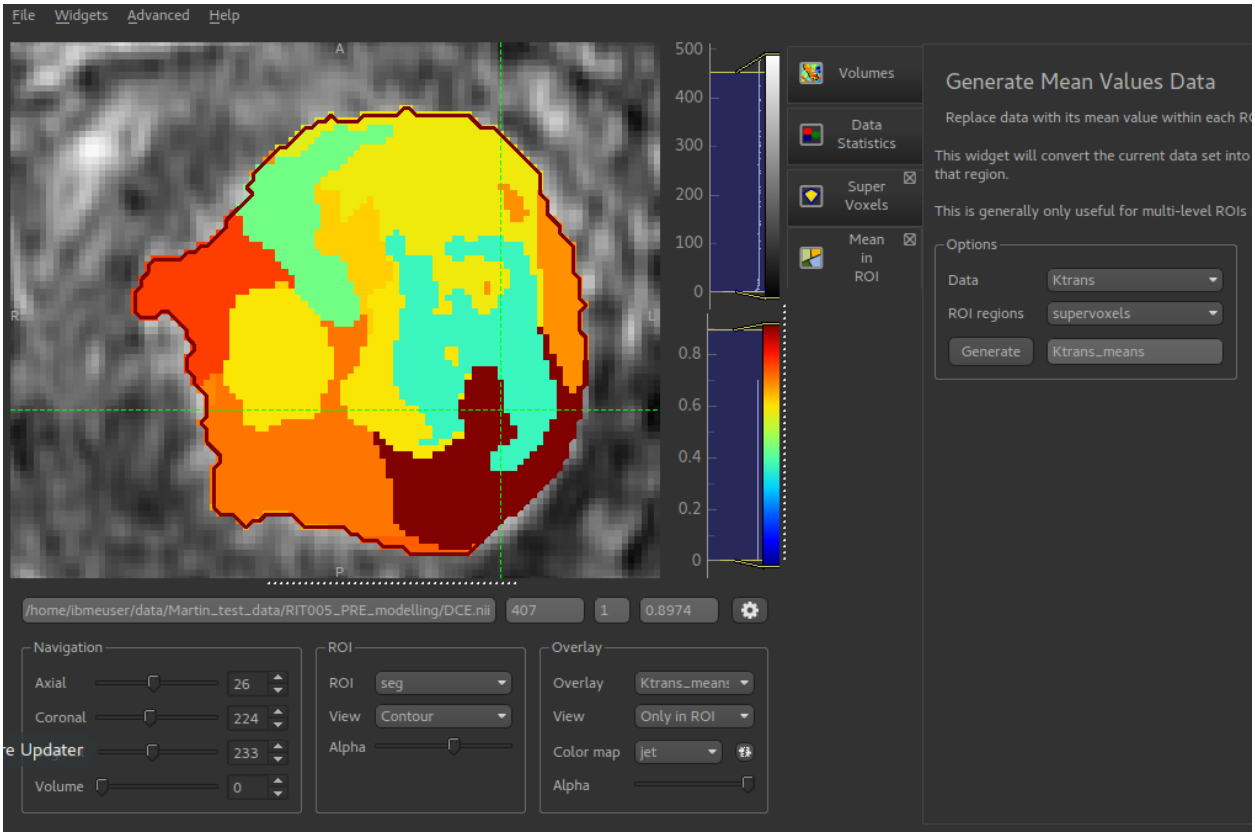
Widgets -> ROIs -> Mean in ROI

The mean values widget takes an overlay and an ROI and outputs a new overlay. Within each ROI region, the new overlay contains the mean value of the original overlay within that region.

This is particularly useful with ROIs generated by clustering or supervoxel methods as it enables the generation of a simplified version of the data where each supervoxel/cluster region has a single value.

The mean values widget also works with 4D data and will output the mean volume series for each ROI region.

Example showing mean Ktrans value of each supervoxel



5.6.11 Histogram widget

Widgets -> Visualisation -> Histogram

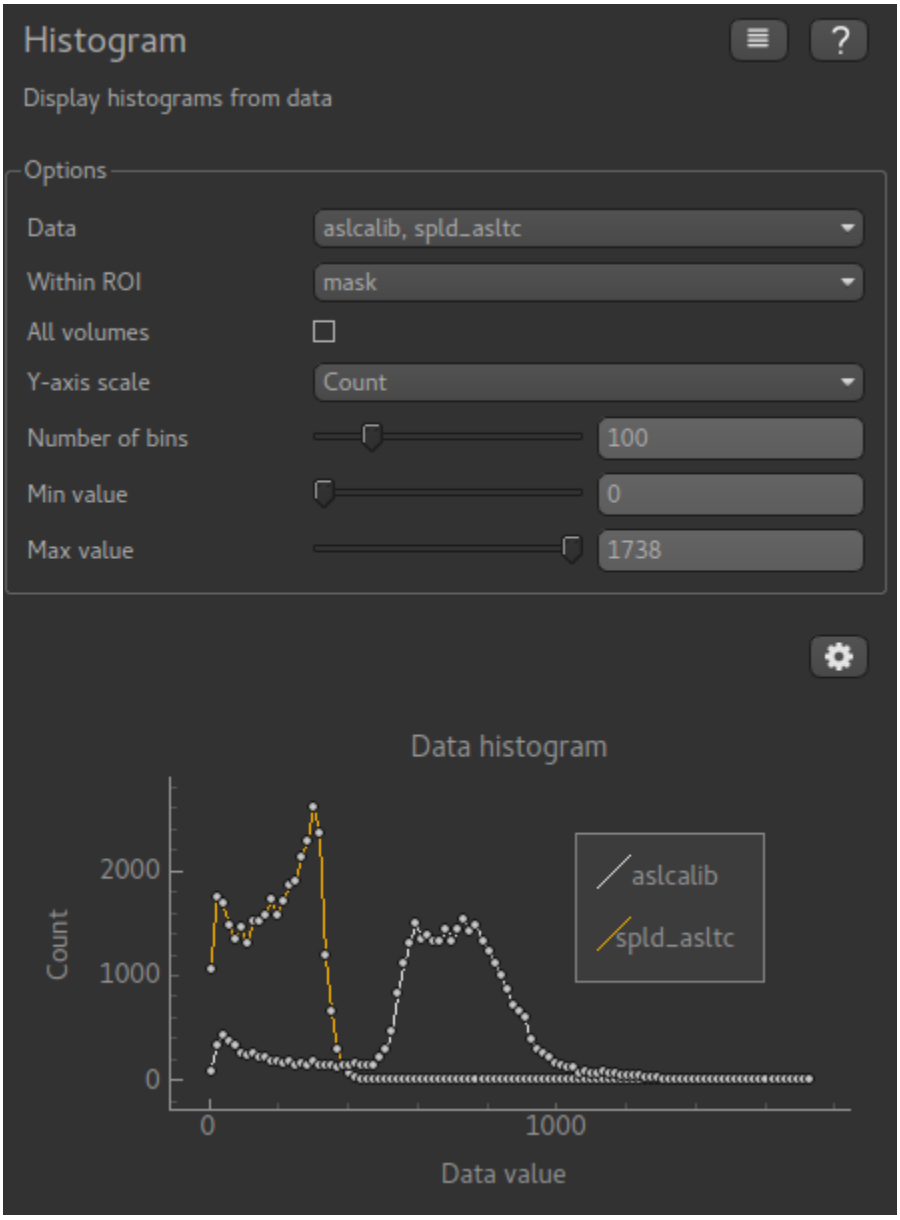
The histogram widget plots a histogram of data values

Any number of data sets can be selected, and the data can be restricted to within a region of interest. Either the frequency or the probability density can be plotted on the Y axis.

The `All Volumes` option applies to 4D data and selects whether the histogram is taken over the full 4D data or just the current visible volume.

Example

This example shows a histogram plotted for two data sets within an ROI `mask`:



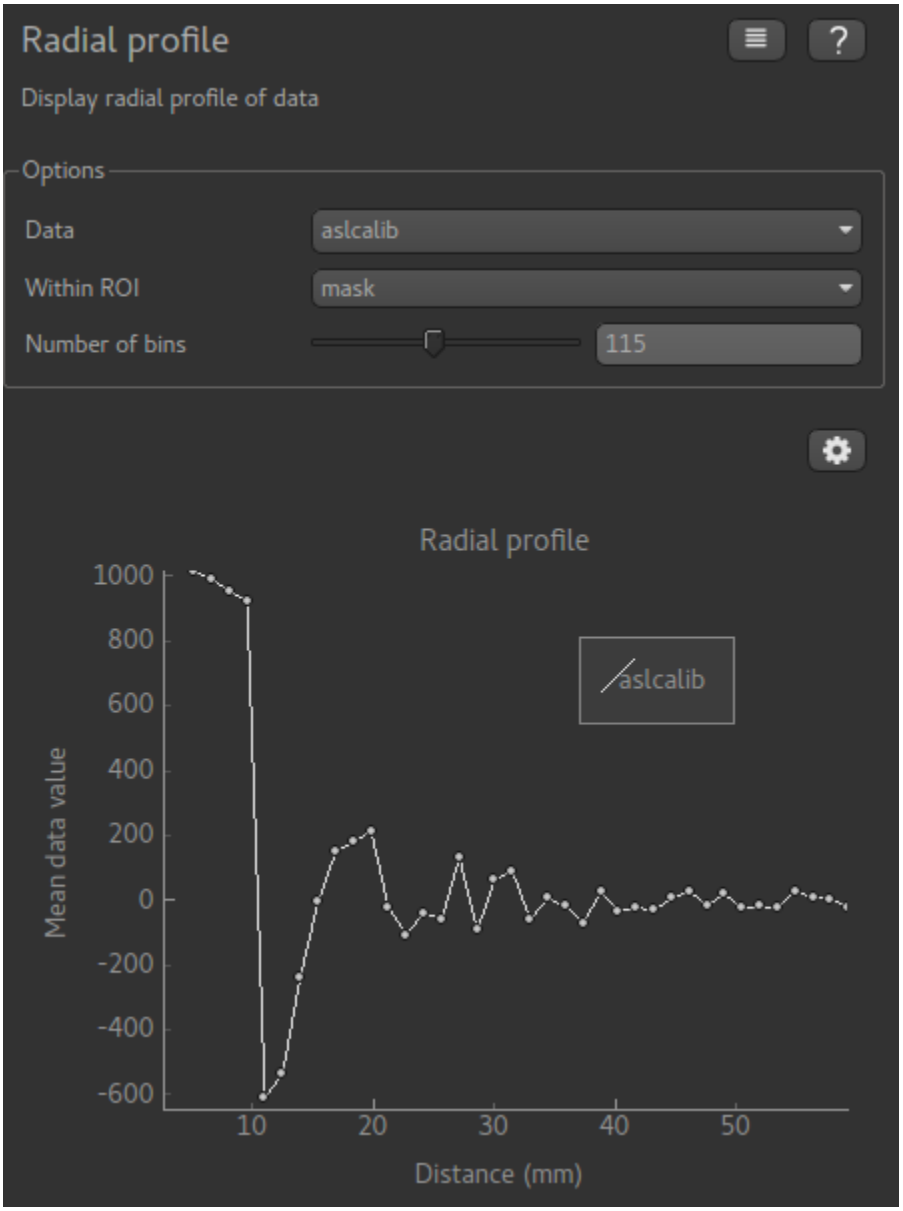
5.6.12 Radial profile widget

Widgets -> Visualisation -> Radial profile

The radial profile widget plots the mean data value as a function of distance from the currently selected point.

Any number of data sets can be selected, and the data can be restricted to within a region of interest.

Example



5.6.13 T1 map from VFA images

Widgets -> T1 -> VFA T1

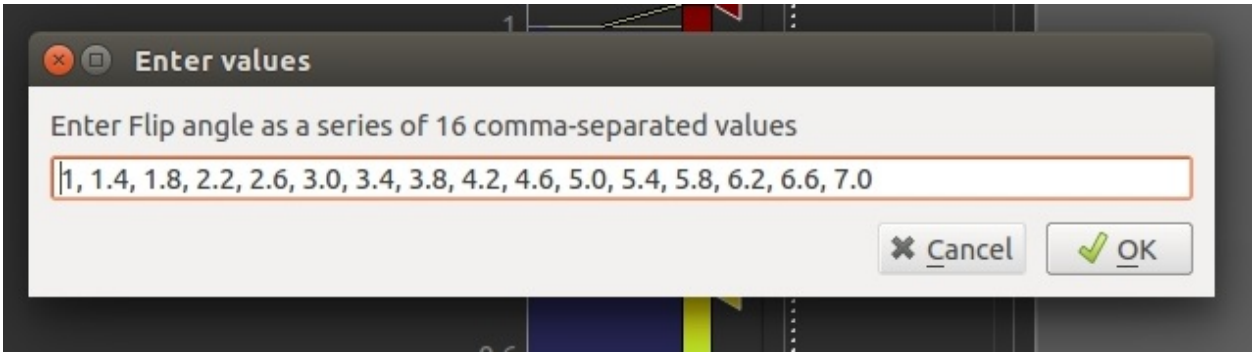
This widget generates T1 maps from variable flip angle images. This is often used as a preprocessing step for kinetic modelling. The VFA scans can be loaded either as separate volumes, one for each flip angle, or a single multi-volume file containing all the flip angles.

The main VFA method is described in[1].

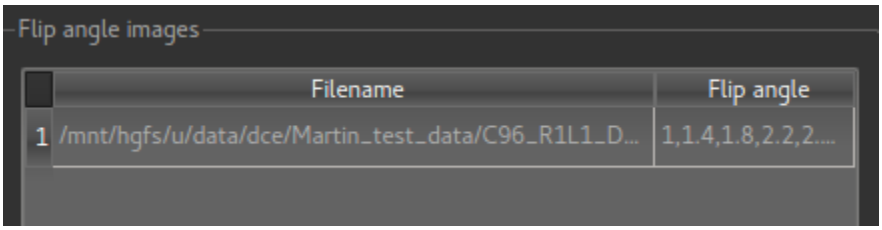
[1] Fram et al., Magn Reson Imaging 5(3): 201-208 (1987)

Using a single 4D volume with multiple flip angles

Click `Add` to and select the data file containing the VFA images. You will then need to enter the flip angles as a comma separated list which must match the number of volumes in the data set.

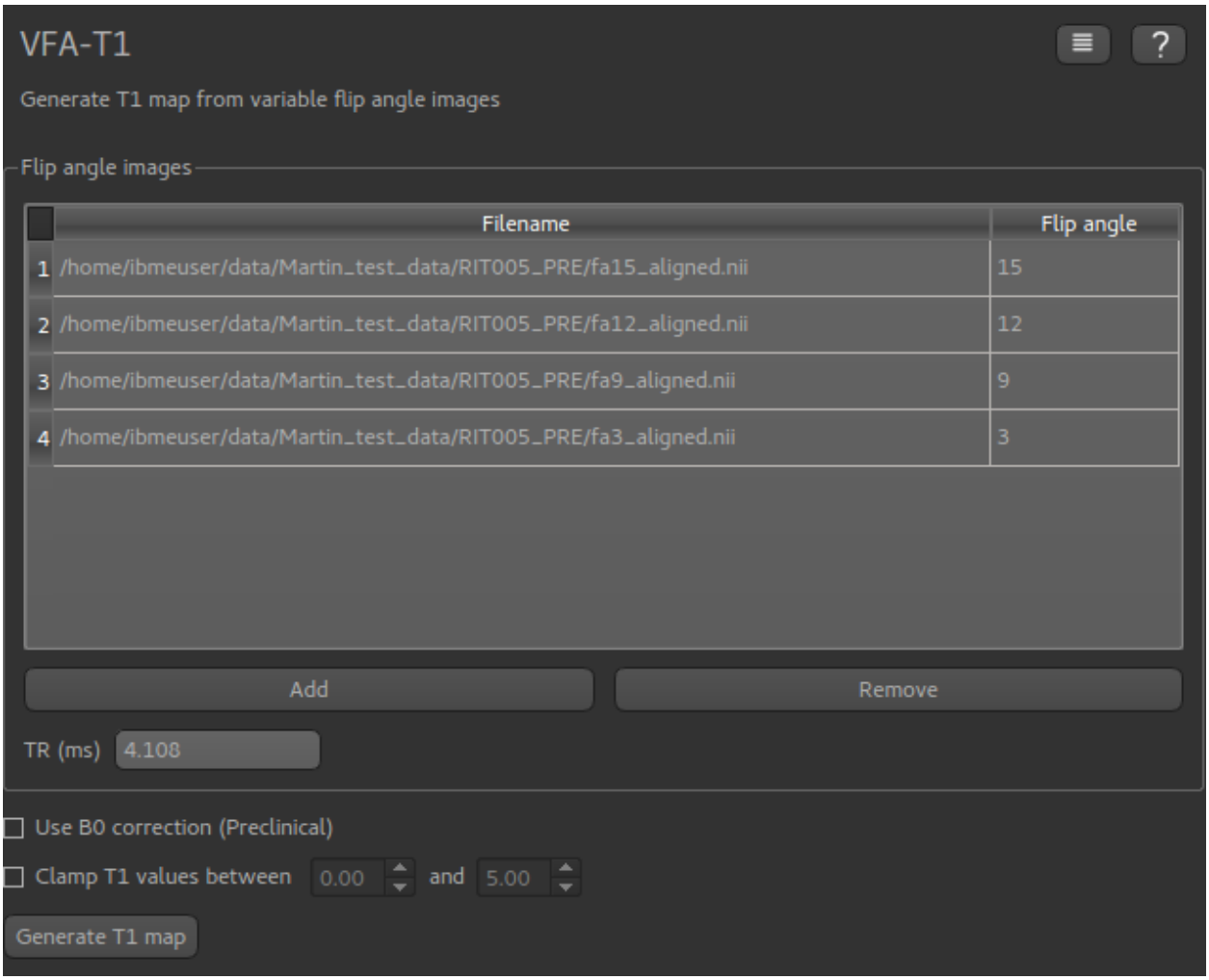


Once loaded, the file will be added to the list:



Loading muliple flip angle volumes

It is also common for the images at different flip angles to be stored in separate files. In this case, simply load each one separately, entering the flip angle for each (the widget will try to guess the flip angle if it is part of the file name, but ensure it has guessed correctly!)



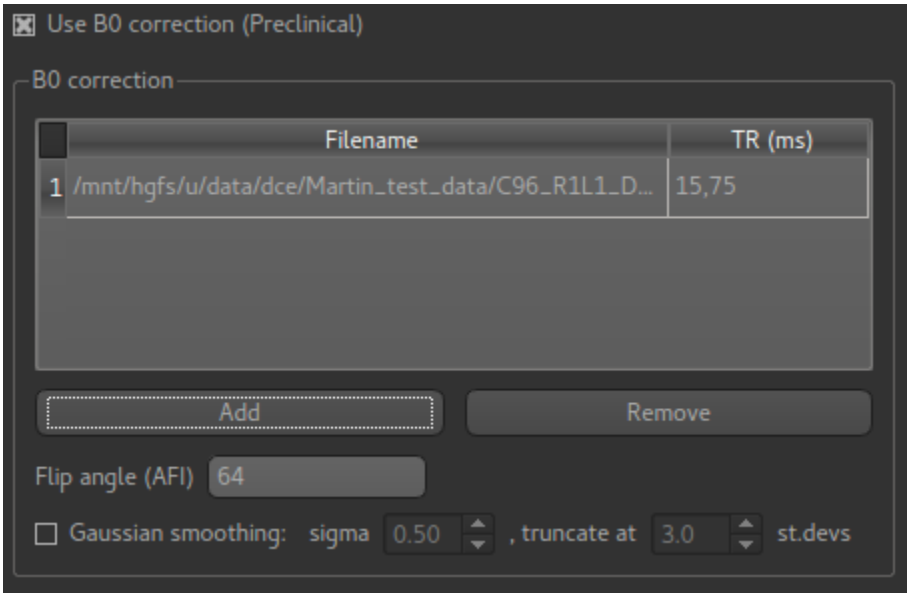
Using a B1 correction

This option allows for correction of field inhomogeneity and B1 effects using Actual Flip Angle Imaging (AFI) data sets. This is particularly common where high field strengths are used, for example in preclinical applications. The method is described in[2]

These are loaded in the same way as the VFA data described above, however in this case you must enter the TR value for each file in ms (or a sequence of TR values if the AFI data is stored in a single 4D data set).

The flip angle used for the AFI sequence is also required.

[2] Yarnykh, V. L. (2007), Actual flip-angle imaging in the pulsed steady state: A method for rapid three-dimensional mapping of the transmitted radiofrequency field. Magn. Reson. Med., 57: 192-200. doi:10.1002/mrm.21120



Applying Gaussian smoothing to the T1 map

An optional postprocessing step is to apply smoothing to the output T1 map. The `sigma` value is standard deviation of the Gaussian used as the smoothing kernel, and is measured in voxels.

The `Processing->Smoothing` widget can also be used to apply smoothing to the output. This allows the kernel size to be specified in physical units (mm).

Clamping the T1 values

The output T1 values may be clamped between limits if required - this may be useful to eliminate unrealistic values in a small number of voxels.

References

5.7 Advanced Tools

5.7.1 Batch processing

Often it is useful to be able to run a set of analysis / processing steps on a whole set of files, without needing to manually load and save the files separately within the GUI. Quantiphyse provides a simple batch processing system which gives access to most of the processing steps available from the GUI.

Batch files are written in YAML syntax. Below is a simple example.

```
# Example config file for running Fabber

OutputFolder: out
Debug: True

Processing:
    - Load:
```

(continues on next page)

(continued from previous page)

```
        data:
          mri.nii:
        rois:
          roi.nii: mask

    - Fabber:
        method: vb
        max-iterations: 30
        model:  poly
        noise: white
        degree: 2
        save-mean:

    - Save:
        mean_c0:
        mean_c1:
        mean_c2:
        mask:
        mri:

Cases:
    Subj0001:
        InputFolder:   c:\mydata\0001
    Subj0003:
        InputFolder:   c:\mydata\0003
    Subj0003:
        InputFolder:   c:\mydata\0003
```

The batch file is divided into three main sections.

Defaults section

First we have statements to set up default options which will apply to all cases:

```
OutputFolder: out
Debug: True
```

In this example we are putting all output data in the `out` folder and enabling Debug messages. Options defined in the defaults section can be overridden for a specific case.

Processing section

This defines a series of processing steps. This usually starts with a Load statement and typically ends with Save:

```
Processing:
    - Load:
        data:
          mri.nii:
        rois:
          roi.nii: mask

    - Fabber:
        method: vb
        max-iterations: 30
```

(continues on next page)

(continued from previous page)

```
        model:  poly
        noise: white
        degree: 2
        save-mean:

    - Save:
        mean_c0:
        mean_c1:
        mean_c2:
        mask:
        mri:
```

In this case we are loading a data file and an ROI which have the same filename for each of our cases, however this file name is interpreted **relative to the individual case folder** so different data is loaded each time.

After loading the data we run the Fabber modelling tool. Options to provide to the tool are given here.

Finally we save the three output data sets generated by the Fabber process, as well as the main data and ROI. These files will be saved in a subdirectory of the output folder specific to the case.

Cases section

This section contains a list of `Cases`. The processing steps will be run separately on each case and the output saved in separate subdirectories of the output folder:

```
Cases:
    Subj0001:
        InputFolder:   c:\mydata\0001
    Subj0003:
        InputFolder:   c:\mydata\0003
    Subj0003:
        InputFolder:   c:\mydata\0003
```

Here we have three cases with input data stored in three separate folders.

Output

The output from processing is stored in `OutputFolder` in a subdirectory named by the case identifier (e.g. `Subj0001`). Processing steps may also generates a log file (e.g. `Fabber.log`) in the same subdirectory. In the above example we would expect the following output structure:

```
out/Subj0001/mri.nii
out/Subj0001/mean_c0.nii
out/Subj0001/mean_c1.nii
out/Subj0001/mean_c2.nii
out/Subj0001/mask.nii
out/Subj0001/Fabber.log
out/Subj0002/mri.nii
out/Subj0002/mean_c0.nii
out/Subj0002/mean_c1.nii
out/Subj0002/mean_c2.nii
out/Subj0002/mask.nii
out/Subj0002/Fabber.log
out/Subj0003/mri.nii
out/Subj0003/mean_c0.nii
```

(continues on next page)

(continued from previous page)

```
out/Subj0003/mean_c1.nii
out/Subj0003/mean_c2.nii
out/Subj0003/mask.nii
out/Subj0003/Fabber.log
```

Overriding processing options within a case

If a particular case needs options to be varied, you can override any of the toplevel options within the case block. For example:

```
Cases:
    Subj0001:
        InputFolder:   c:\mydata\0001
        # This case does not converge in 30 iterations
        Fabber:
            max-iterations: 100
```

Or, if one case is causing problems we might enable debug mode just for that case:

```
Debug: False

Cases:
    Subj0005:
        InputFolder:   c:\mydata\0005
        # What's going on here?
        Debug: True
```

Multiple processing steps

The `Processing` block contains a list of steps, which will be performed in order. For example this example performs motion correction on the main data, followed by PK modelling:

```
Processing:
    - Moco:
        method: deeds
        replace-vol: True
        ref-vol: 14

    - PkModelling:
        model:      1
        fa:         30     # degrees
        tr:         5.0    # ms
        te:         2.2    # ms
        dt:         0.5    # temporal resolution (s)
        r1:         3.7    # T1 Relaxivity of contrast agent
        r2:         4.8    # T2 Relaxivity of contrast agent
        ve-thresh:  99.8   # Ktrans/kep percentile threshold
        tinj:       60     # Approximate injection time (s)
```

Extras

Extras are data created by processing modules which are not voxel data, but can be saved to a text file. They can be saved in the same way as data using the `SaveExtras` command. For example the `CalcVolumes` process calculates

the volume of each region of an ROI and outputs a table extra:

```
OutputFolder: out

Processing:
    - CalcVolumes:
        roi: mask
        output-name: roi_vols

    - SaveExtras:
        roi_vols:

Cases:
    Subject1:
        Folder:   c:\Users\ctsu0221\build\data
        Load:
          data:
             test_data.nii
          rois:
             test_mask.nii : mask
```

In this case, the volume data will be saved in `out/Subject1/roi_vols.txt`. In this case the output is a tab-separated file which can be loaded into a spreadsheet.

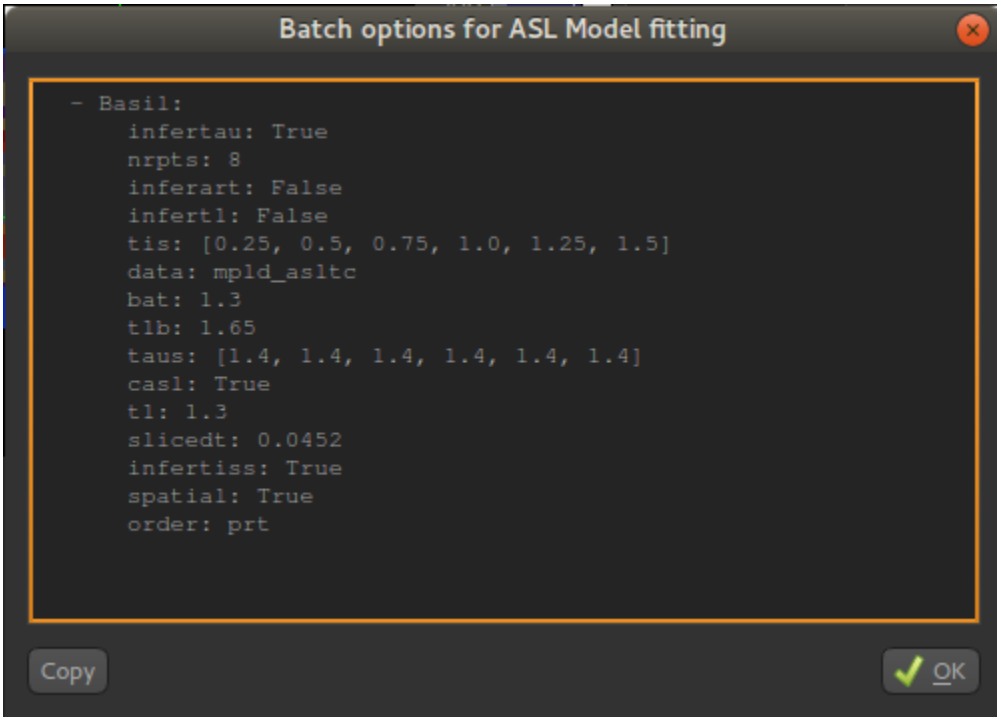
Building batch files from the GUI

It can be convenient to build up a batch process during the course of an interactive session, for example to try out processing steps on a sample dataset and record the selected steps for later application to a group of cases. Quantiphyse provides some basic features to facilitate this.

The *Batch Button*

Many widgets support a *Batch Button* which is normally located in the top right corner, level with the widget title:

Clicking the batch button pops up a window containing the batch file code for the analysis process currently defined by the widget's GUI controls. For example, here is the result of clicking the batch button on the ASL model fitting widget after we have set up a multi-PLD analysis:

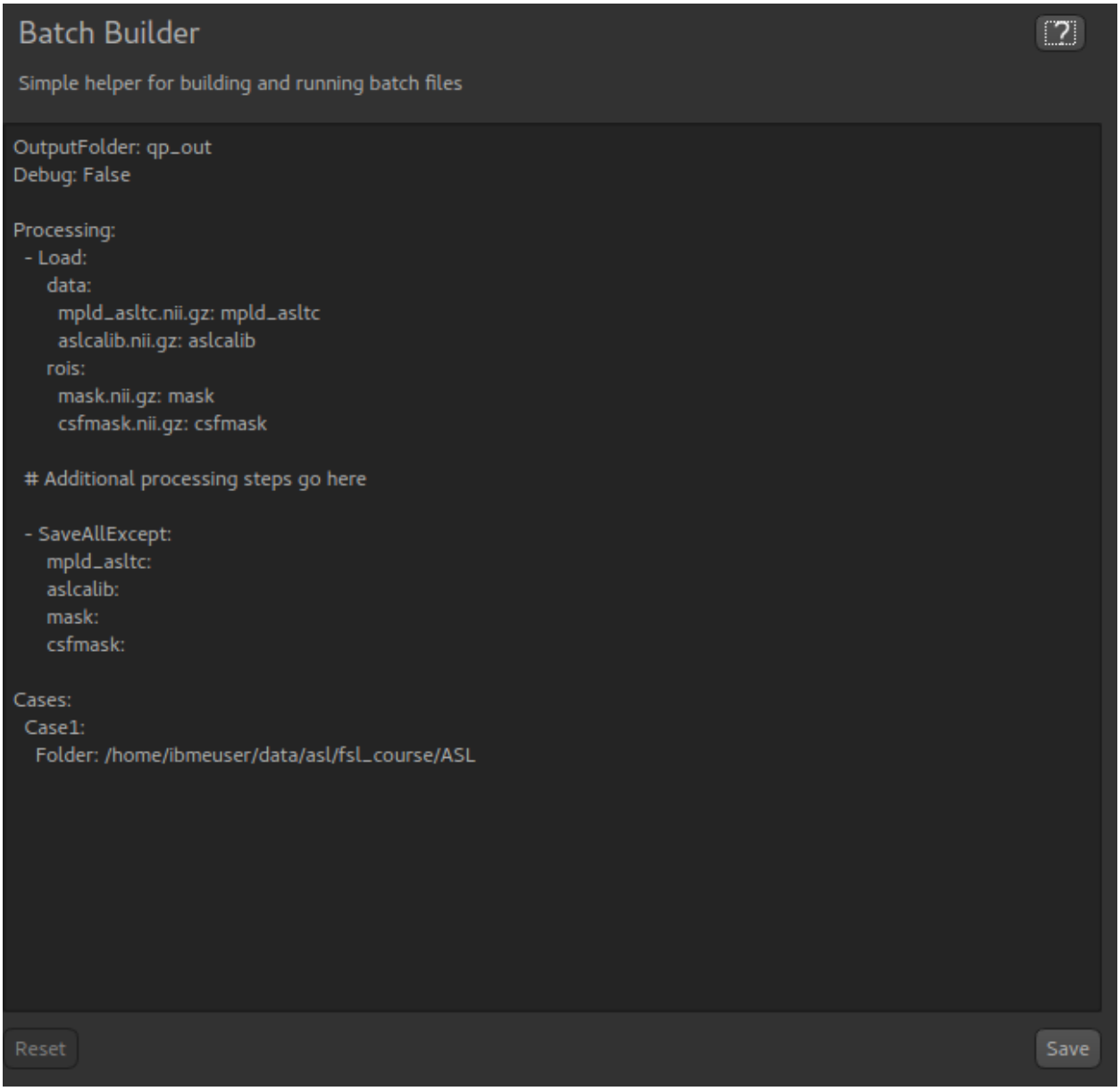


The `Copy` button copies this code to the clipboard where it can be pasted into a batch script that you are creating in a text editor, or using the `Batch Builder` widget (see below).

The *Batch Builder* widget

This widget is available from the 'Utilities' menu and gives a simple editor for batch scripts.

When first opened, a skeleton batch script will be generated which loads all currently opened data files and then saves all new data created during the batch script (using the `SaveAllExcept` process). Here's an example after we've loaded some ASL data:

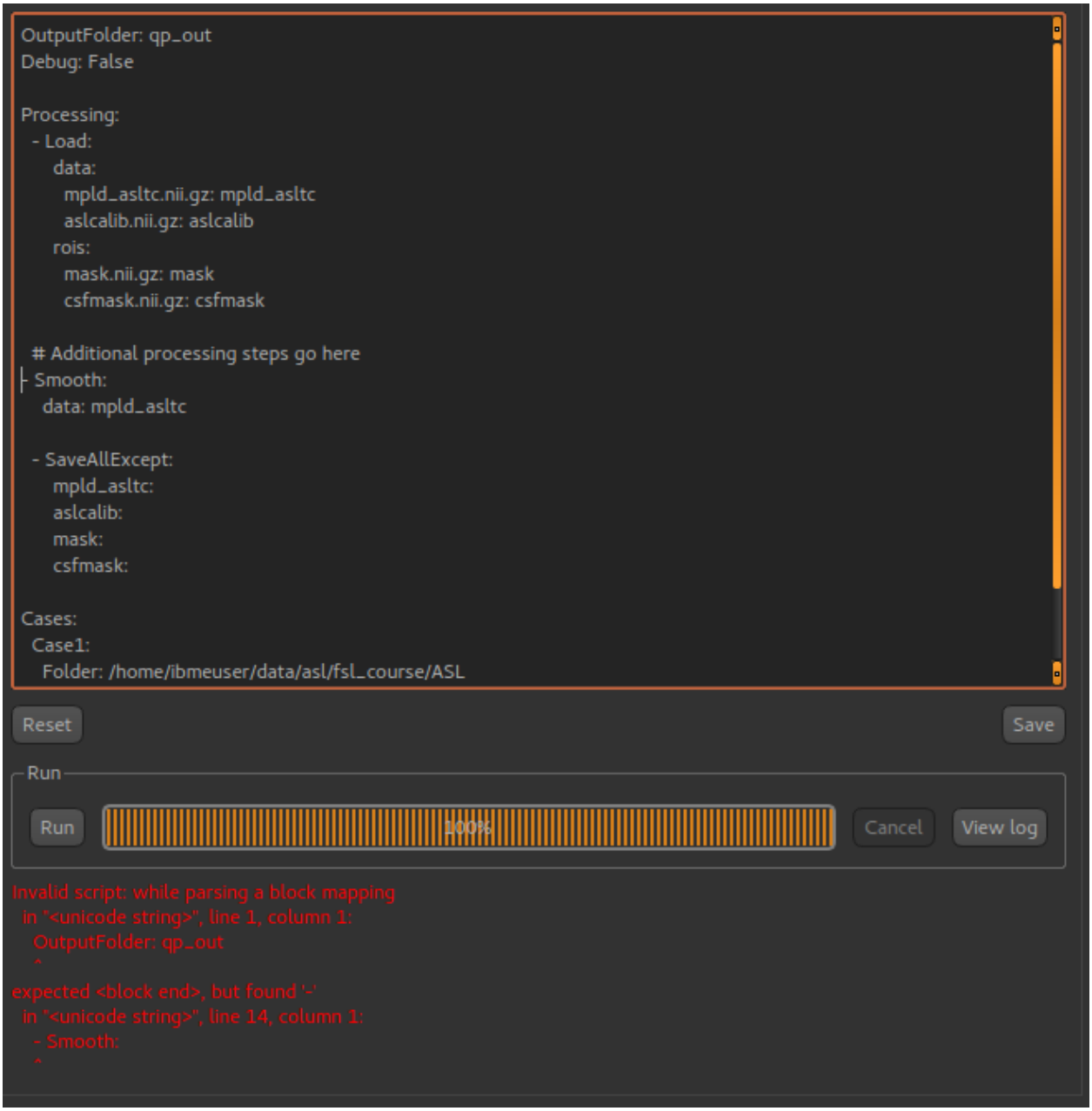


This script will not do anything else, however we can copy batch code from widgets using the batch button and paste it where the script says: `# Additional processing steps go here`. So we could paste the ASL analysis code shown above:

```
OutputFolder: qp_out
Debug: False

Processing:
 - Load:
    data:
      mpld_asltc.nii.gz: mpld_asltc
      aslcalib.nii.gz: aslcalib
    rois:
      mask.nii.gz: mask
      csfmask.nii.gz: csfmask

 # Additional processing steps go here
 - Basil:
    infertau: True
    nrpts: 8
    inferart: False
    infert1: False
    tis: [0.25, 0.5, 0.75, 1.0, 1.25, 1.5]
    data: mpld_asltc
    bat: 1.3
    t1b: 1.65
    taus: [1.4, 1.4, 1.4, 1.4, 1.4, 1.4]
    casl: True
    t1: 1.3
    slicedt: 0.0452
    infertiss: True
    spatial: True
    order: prt

 - SaveAllExcept:
    mpld_asltc:
```

This batch script can be `Run` to test it, and then we use `Save` to save it to a file when we're happy. You can add cases and other processing as required. `Reset` will return to the 'skeleton' batch script with no custom processing.

The batch builder will indicate if your file contains any syntax errors, for example if we don't indent our processing steps correctly:



One common issue is the use of tabs in a batch file which is not allowed but can cause difficult to interpret errors. Therefore, if you use a tab character in the batch builder it will check and simply give a warning of `Tabs detected`.

Future extensions

The batch system may be extended in the future, however it is *not* intended to be a programming language and basic facilities such as loops and conditionals will not be implemented. If your processing pipeline is complex enough to require this the suggested method is to write the process in Python, using Quantiphyse modules directly, for example:

```
from quantiphyse.volumes import ImageVolumeManagement
from quantiphyse.analysis.io import LoadProcess, SaveProcess

ivm = ImageVolumeManagement()
```

(continues on next page)

(continued from previous page)

```
load = LoadProcess()
load.run({"data" : {"mydata.nii" : "data"}, "rois" : {"mask_43.nii.gz" : "roi"}})

# The std() method returns the data on the standard RAS grid derived from the main␣
↪data
numpy_data = ivm.data["data"].std()

# ...Do my processing here which may involve running additional Quantiphyse processes
#    alongside custom Python manipulations...

ivm.add_data(output_data, name="output_data")
save = SaveProcess()
save.run({"output_data":"output_data.nii.gz"})
```

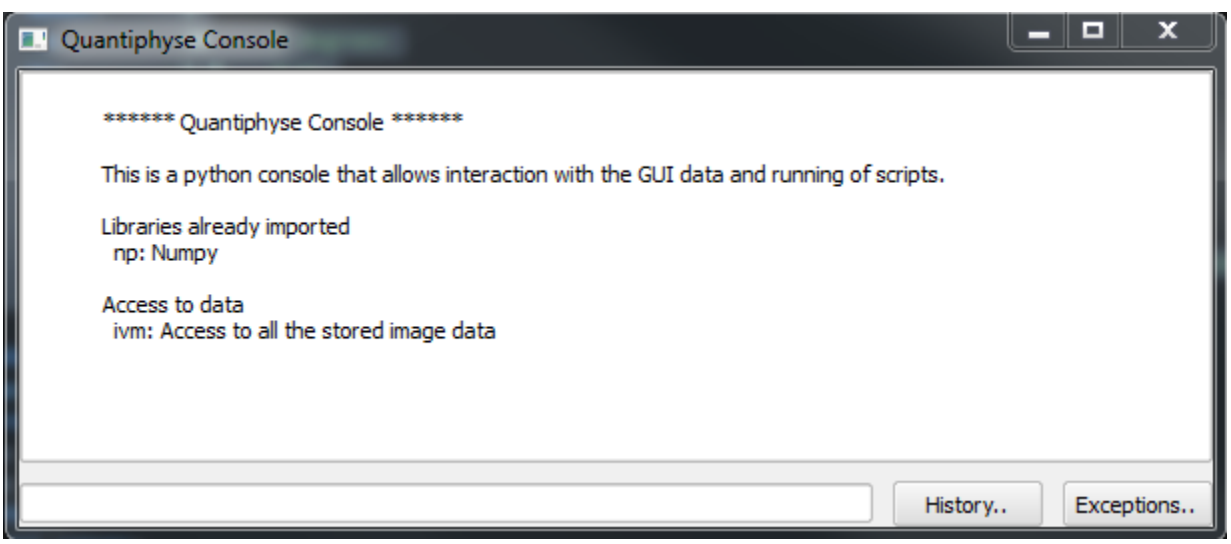
The processing modules available in the batch file are all included in the quantiphyse.analysis package. They all operate on data stored in the ImageVolumeManagement object. Data can be added to this object using the `add_data` and `add_roi` methods, which can take a Numpy array, provided it's dimensions are consistent with the current main data. This means that you can load data independently or generate it programmatically if this is required.

> **Warning:** The volume management and analysis process APIs are *not* currently stable and you will need to read the code to see how to use them - a stable API may be defined in the future for this purpose.

5.7.2 Using the console

The console is an advanced tool which allows you to interact directly with the data structures within the program. You might use this to perform processing steps which don't have a predefined widget, using the full power of Python and the Numpy and Scipy libraries.

To open the console, select `Console` from the `Advanced` menu.



Objects provided

The main predefined variables are:

- `ivm` - The volume management object. It provides the `add_data` and `add_roi` methods you need to get data into the viewer
- Each existing data item is a named variable - for example if you have an overlay named `T10` there will be a variable `T10` which contains the data.

The following namespaces are predefined:

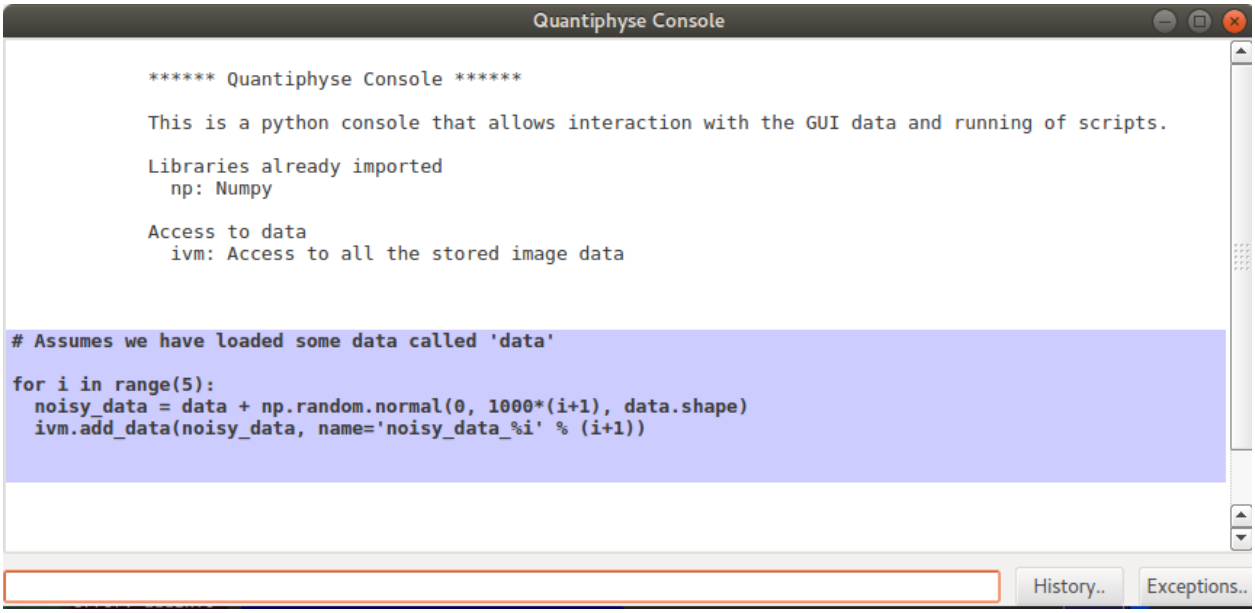
- `np` - The Numpy module

Working with data

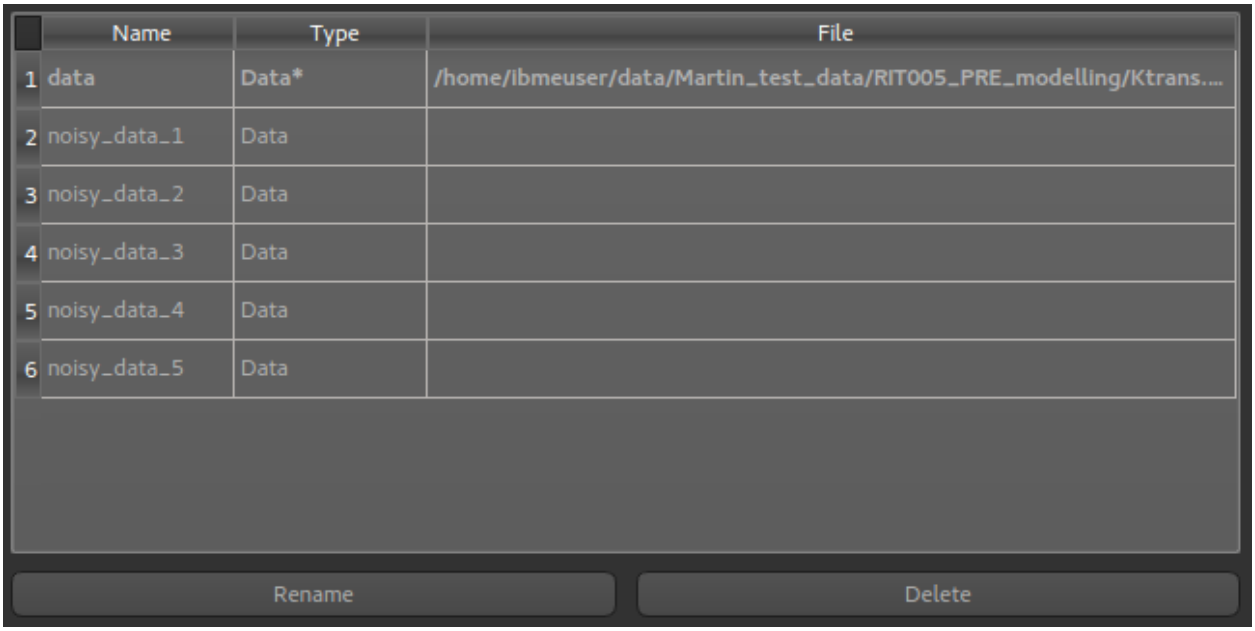
Data objects are subclasses of Numpy arrays and can support any operations on them. To add new data into the viewer you use the `add_data()` or `add_roi()` methods.

Examples

- Create a series of data objects by adding varying levels of Gaussian noise to an existing data set



This creates 5 new data sets containing the original test_data plus random Gaussian noise with mean 0 and standard deviations between 1000 and 5000.



5.7.3 NIFTI metadata extension

Quantiphyse stores various metadata about its data sets which it would be useful to persist across loading and saving. The NIFTI format provides for this in the form of 'header extensions'.

Each header extension is identified by a code number so software can choose to pay attention only to header extensions that it knows about. Quantiphyse has been assigned the code `42` for its header extensions.

Quantiphyse extensions will be stored as strings in YAML format for easy serialization/deserialization to Python and because YAML is already used as the basis for the batch format.

There has been suggestion that `nibabel` may add its own metadata as a NIFTI extension. This might enable some of the Quantiphyse metadata to be deprecated, however this is not available at present. It may also be possible to align this metadata with the BIDS standard in the future.

The following set of metadata is an initial proposal, however any widget can save its own metadata by adding a YAML-serializable object to the data sets `metadata` dictionary attribute. Hence this list is not exhaustive.

Generic metadata

```
Quantiphyse:
    roi : True/False          # Whether the data set should be treated as an ROI
    regions :                 # ROI regions (codes and names)
        1 : tumour
        2 : nodes
    raw-2dt : True            # Indicates that 3D data should be interpreted as 2D+time
    dps: 3                    # Suggested number of decimal places to display for values
```

ASL data set structure

```
AslData:
  tis : [1.4, 1.6, ...]      # List of TIs
  plds : [2.5, 2.6, ...      # List of PLDs, alternative to TIs
  rpts : [4, 4, 4, ...]      # Repeats at each TI/PLD
  phases : [0, 45, 90, ...] # Phases in degrees for multiphase data
  nphases : 8                # Alternatively, number of evenly-spaced phases
```

CEST data set structure

```
CestData:
  freq-offsets : [-300, -30, 0, 10, 20, ...] # Frequency offsets
  b0 : 9.4                                   # Field strength in T
  b1 : 0.55                                  # B1 in microT
  sat-time : 2                               # Continuous saturation time in s
  sat-mags : [1, 2, 3, 4, ...]               # Pulsed saturation magnitudes
  sat-durs : [1, 3, 2, 4, ...]               # Pulsed saturation durations in s
  sat-rpts : 1                               # Pulsed saturation repeats
```

5.7.4 Quantiphyse plugins

Some Quantiphyse functionality requires the installation of plugins. The following plugins are currently available:

- `quantiphyse-dce` - DCE modelling
- `quantiphyse-fabber` - Bayesian model fitting - required for various specialised tools
- `quantiphyse-fsl` - Interface to selected FSL tools (requires FSL installation)
- `quantiphyse-cest` - CEST-MRI modelling (requires `quantiphyse-fabber`)
- `quantiphyse-asl` - ASL-MRI modelling (requires FSL installation and `quantiphyse-fabber`)
- `quantiphyse-dsc` - DSC-MRI modellingg (requires `quantiphyse-fabber`)
- `quantiphyse-t1` - T1 mapping (requires `quantiphyse-fabber`)

Plugins are installed from PyPi, e.g.:

```
pip install quantiphyse-dce
```

They will be automatically detected and added to Quantiphyse next time you run it. The packages available on the OUI software store have all plugins included which were available at the time of release.

5.8 Frequently Asked Questions

5.8.1 Installation

Errors when installing from pip because modules not available

Usually these problems are not related directly to Quantiphyse but involve dependencies which require specific versions of a module.

If you encounter these types of problems, you might want to try using `conda` instead of pip which we generally find is more reliable for packages which include native (i.e. non-Python) code. Instead of `pip install` try `conda install` for the packages which are causing trouble, then try `pip install quantiphyse` afterwards.

Error on startup after installing plugins

One known issue can be identified by starting quantiphyse from the command line. If it fails with an error message that ends as follows:

```
pkg_resources.ContextualVersionConflict: (deprecation 2.0.6 (/usr/local/lib/python2.7/
↪dist-packages), Requirement.parse('deprecation<=2.*,>=1.*'), set(['fslpy']))
```

This can be fixed with:

```
pip install deprecation==1.2 --user
```

The cause is an apparently invalid requirements specification in a dependency package.

5.8.2 Running

On Windows the data viewer and graphs do not work properly

The symptoms of this problem include:

- The image viewer windows only update when you drag on them with the mouse
- Graph plots (e.g. in voxel analysis) do not appear

This seems to be an issue with PySide which affects the pyqtgraph library on Windows. We have found that installing PySide and pyqtgraph using `conda` rather than `pip` can help.

Fabber modelling widgets do not work (e.g. CEST/ASL/DCE/DSC)

These functions require an up to date version of Fabber. We expect **FSL 6.0.1** to include sufficiently up to date versions of this code - this should be available very soon. If you can't wait for this, please contact the maintainers and we will explain how to install an interim version which will work.

This does not affect the packages downloaded from the OUI Software Store which include prebuilt versions of Fabber and the required models.

CHAPTER 6

Bugs/Issues

Bugs may be submitted using the Github issue tracker for Quantiphyse.

For any other comments or feature requests please contact the current maintainer:

CHAPTER 7

Contributors

- Martin Craig (Current maintainer)
- Ben Irving (Original author)
- Michael Chappell
- Paula Croal

CHAPTER 8

Acknowledgements

- Julia Schnabel
- Sir Mike Brady

Images © 2017-2019 University of Oxford