
BASIL documentation

Michael Chappel, Martin Craig

Aug 24, 2023

Contents

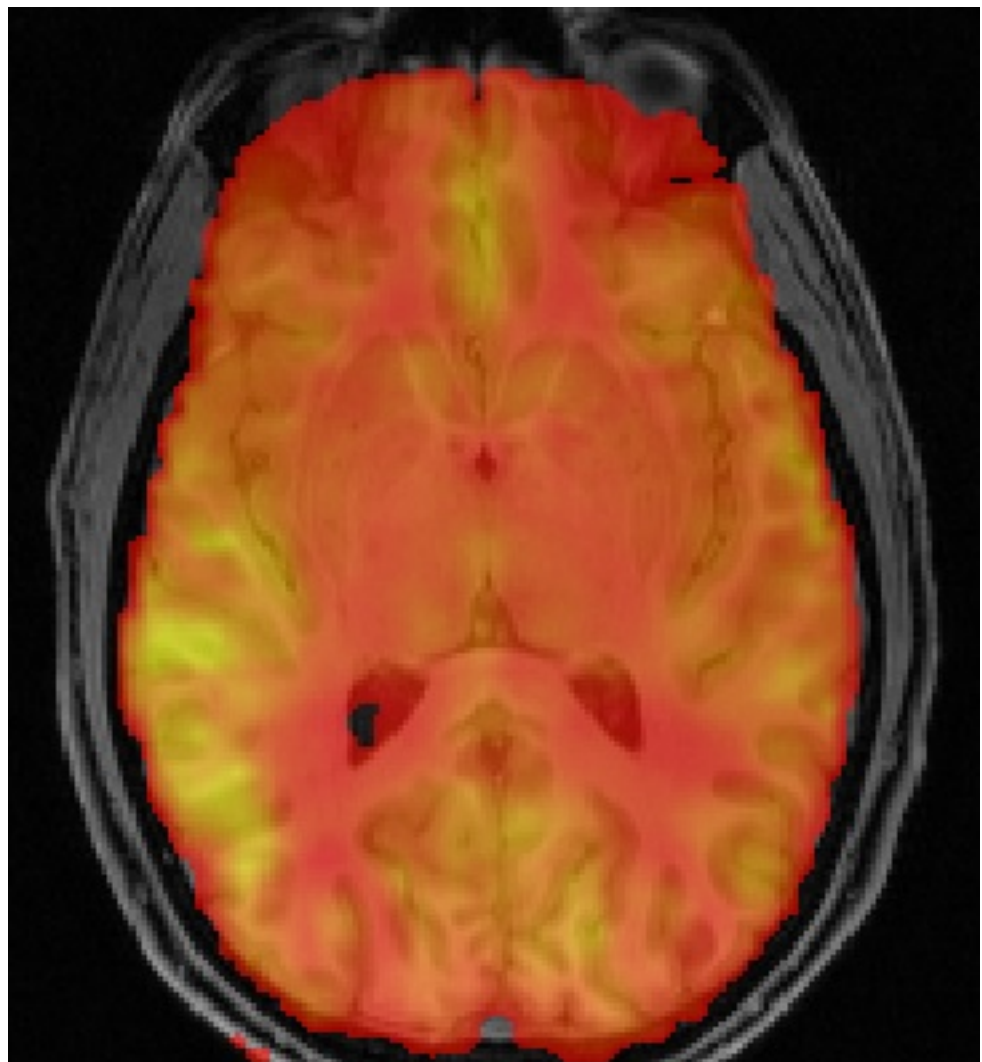
1	ASL Analysis Guide	3
1.1	Perfusion Quantification using Arterial Spin Labelling MRI	5
2	Overview of the BASIL toolset	11
2.1	Higher-level packages	11
2.2	Individual analysis components	11
3	Tutorials	13
3.1	Worked examples from the FSL course	13
3.2	Cerebrovascular Reserve Quantification Using ASL	29
3.3	Command line Tutorial	34
3.4	Preparatory materials	41
4	GUI User Guide	43
4.1	Input Data	44
4.2	Structural	47
4.3	Calibration	48
4.4	Distortion Correction	49
4.5	Analysis	50
4.6	Output	52
5	Command line User Guide	53
5.1	Overview	53
5.2	What you will need	53
5.3	Things to note	54
5.4	Typical Usage	54
5.5	Output	55
5.6	Detailed usage information	56
5.7	Region analysis	60
6	Other components of the BASIL toolset	63
6.1	BASIL (command)	63
6.2	asl_calib - calibration of ASL perfusion	72
6.3	asl_file - preprocessing of ASL data	75
6.4	QUASIL - quantification of QUASAR data	79
6.5	TOAST - quantification of Turbo-QUASAR data	80

BASIL is a collection of tools that aid in the creation of quantitative CBF images from ASL data. It is available within [FSL](#) (v6.0.1 or later is strongly recommended).

BASIL includes complete high-level pipelines for processing ASL data and also individual tools for more bespoke analysis.

CHAPTER 1

ASL Analysis Guide



1.1 Perfusion Quantification using Arterial Spin Labelling MRI

Arterial Spin Labeling (ASL) MRI is a non-invasive method for the quantification of perfusion. This guide introduces the essential concepts you need to understand when acquiring and analysing ASL data, as well as more advanced analysis techniques you might exploit when aiming for higher accuracy and sensitivity when using ASL in a study.

This guide complements the information contained in the [Introduction to Perfusion quantification using Arterial Spin Labelling \(OUP, 2017\)](#). You can perform the analyses shown in the guide using the BASIL toolbox by following the instructions in the [Tutorials](#) section of this website. Alternatively, similar practical activities are also available on neuroimagingprimers.org.

The guide includes five pre-recorded video lectures that are used in the BASIL course (an FSL mini-course).

1.1.1 Preparatory materials

If you are new to Neuroimaging or to FSL tools you might find the following resources helpful before looking at the material on ASL.

If you are not particularly familiar with MRI you might like to read a [Short Introduction to MRI Physics for Neuroimaging](#) available via the neuroimagingprimers.org website.

The [FSL course](#) provides a wide range of videos and practical exercises. Whilst you do not need to have looked at any of this before reading this guide, you might find some of it helpful background for topics that will be covered (especially for the group analysis section).

We would recommend familiarity with:

- Section 1: Image Registration and Distortion Correction (videos 1-5 are all helpful)
- Section 2: Segmentation and Structural Analysis (video 7 on FAST)
- Section 4: Statistics & Task fMRI: Inference and Group Analysis (videos 15-20 provide useful background on group analysis)

1.1.2 Introductory video

In this video we examine what perfusion is and why we might be interested in measuring it in the brain. We briefly consider alternative established perfusion imaging techniques to understand when and where we might use ASL. Finally, we introduce the BASIL toolbox, available as part of FSL.

1.1.3 ASL acquisition

There are various parameters associated with an ASL acquisition. Some of which are important for the quantification/analysis process. Whilst you do not necessarily need to know all of the details of an ASL acquisition to be able to quantify perfusion image, in this section we provide an overview of the key parameters when designing an ASL protocol and that you need to know for successful analysis.

pcASL or pASL

The labelling may either have been pseudo-continuous ASL (pcASL, most common) or pulsed ASL (pASL). There are important differences between these two forms of ASL that affect the quantification, since they determine how the blood-water is labelled.

Post-label delay(s)

After labeling, a delay is left for the labeled blood-water to travel into the brain. For pcASL this is called the Post-Label Delay (PLD) and is the time from the *end* of the label duration (see [Label duration](#)) until imaging. For pASL the labeling process is instantaneous and it is more common to refer to the inversion time (TI). Note that we can define a common delay measure: the inflow time (also TI), the time from the *start* of labeling. This is identical to the inversion time for pASL, but for pcASL is the sum of the label duration and PLD:

pcASL: $TI = PLD + \text{Label duration}$

It is quite common to meet ASL data with multiple repeats/measurements (and thus volumes in the resulting images) that all have the same PLD (or TI) - single delay ASL. It is, however, possible to use a range of different PLD in an acquisition in an attempt to extract more information, or achieve a better SNR - multi-delay (multi-PLD) ASL.

Label duration

The label (or bolus) duration is an important measure of how much labeled-blood water has been delivered to the tissue and is thus important for quantification. For pcASL the value is set by the sequence and thus is something you need to define/know. It is quite common to use a 1.8 second (or longer) label duration with pcASL.

In principle in pASL the label duration is unknown (a spatial region is labeled instead of a known duration of flowing blood). You may find that your pASL acquisition is using Q2TIPS or QUIPSSII, in which case the label duration has been set using extra pulses. Quite often the value of label duration can then be determined from the associated parameter, often called TI2 - a value of 0.7 or 0.8 seconds would be quite normal. Where the label duration is genuinely unknown (e.g. a FAIR pASL acquisition), it is possible to estimate it as long as the data is multi-TI. It is possible even with Q2TIPS/QUIPSSII that the duration will be shorter than expected due to high flow in the labelling region.

Readout

A variety of readouts can be combined with ASL labelling to acquire ASL data. The important distinction is between (multi-slice) 2D and 3D readouts, since in the former the later time of acquisition of more superior slices needs to be accounted for in the quantification.

Background Suppression

It is common for background suppression to be applied in an ASL acquisition to suppress signal not associated with labeled-blood water (static tissue signal) and reduce artefacts arising from motion.

1.1.4 Analysis of ASL data

The very simplest analysis of ASL data requires the subtraction of label and control images in the data to produce a perfusion weighted image. With the addition of kinetic model inversion and calibration (requiring calibration data acquired as part of the ASL dataset) it is possible to produce quantitative perfusion images with conventional usings of ml/100g/min.

Subtraction

Central to ASL analysis is the subtraction of label and control images. Both label and control images will contain some signal from brain tissue - called the static tissue signal (this is true even if background suppression has been used to reduce this contribution). Subtraction of the label-control pair reveals the contribution from labelled blood-water. This image is often referred to as the difference image and is perfusion-weighted, which means it reflects the perfusion in each voxel, but the intensity value in each voxel does not alone provide an absolute measure of perfusion.

To go beyond the perfusion weighted image, and generate quantitative voxelwise measures of perfusion with values in the typical units of ml/100 g/min, we need to use the kinetics of ASL.

Kinetic Model Inversion (Kinetic Modelling)

The voxel intensity in an ASL difference image is directly related to the labelled blood-water. More accurately, it relates to the amount of labelled blood-water that has accumulated in the voxel in the time between creation of the label and the collection of a brain image. This means that it is a measure of delivery and thus perfusion (rather than blood volume or blood flow rate). To be able to say how much labelled blood has been delivered, and thus what the perfusion is, it is necessary to describe the delivery process, as well as what happens to the labelled blood once it has been delivered. This is achieved by means of a kinetic model.

At its very simplest the kinetic model for labelled blood-water in an ASL study needs to account for the delivery of a finite duration (the label duration) of labelled blood-water into the voxel where it accumulates. At the same time as it is being delivered, the label is also decaying away. The tracer decays at a rate defined by the T1 time constant, which is of the order of a second in the brain at typical MRI field strengths. The kinetic model allows the relationship between the signal and perfusion to be expressed as an equation and this can be rearranged to give an equation that takes signal magnitude and returns perfusion, or fit to the data using optimisation techniques.

Calibration

The ASL calculation relies on knowledge of the tracer concentration, strictly the quantity called the equilibrium magnetization of arterial blood, which will vary between individuals and other MRI-related factors (e.g. the main magnetic field strength). The simplest approach for estimating this parameter is by the acquisition of a separate proton-density-weighted image. This can be converted to a measure of arterial magnetization by accounting for the relative density of hydrogen nuclei in tissue and blood (the partition coefficient). Various corrections can be performed where the calibration image is not a pure proton-density weighted image, e.g., where it has a (relatively) short repetition time.

1.1.5 Further Quantification of ASL data

For single delay ASL data kinetic model inversion is relatively trivial and solutions to the standard model have been described in the literature. However, there are various advantages to acquiring ASL data at multiple times post-inversion and fitting the resultant data to a kinetic model. This permits problems in perfusion estimation associated with variable arterial transit time (ATT) to be avoided, since this becomes a parameter of the model whose value is determined from the data. ATT can also be a valuable parameter (describing the passage of blood through the vasculature) in its own right.

The model fitting can be performed by a variety of (non-linear) regression techniques, include two step processes that or least squares algorithms. BASIL uses a (fast) Bayesian inference method for the model inversion, this provides a number of advantages:

- Voxel-wise estimation of perfusion and ATT along with parameter variance (allowing confidence intervals to be calculated).
- Incorporation of natural variability of other model parameters, e.g. values of T1, T1b and labeling/bolus duration.
- Spatial regularization of the estimated perfusion image.
- Correction for partial volume effects (where the appropriate segmentation information is available).

Spatial regularization

BASIL can apply a spatial regularisation to the estimated perfusion image and this is highly *recommended*. This exploits the fact that neighboring voxels are likely to have similar perfusion values, i.e. perfusion variation in the brain is relatively smooth. It brings the advantages associated with the more common pre-processing step of spatially smoothing the data. However, unlike smoothing the data it correctly preserves the non-linear kinetics exploited by the perfusion estimation. It is also adaptive, so that in regions where the data does not support the use of smoothing the perfusion image will not be smoothed.

1.1.6 Group analysis using ASL data

In a study ASL data acquired in individuals can be combined to examine differences or changes in perfusion (or ATT). Group analyses using ASL are similar to that used for other neuroimaging modalities, e.g. BOLD fMRI. In this section we consider specific issues that relate to ASL data, including achieving good alignment between subjects (registration), the influence of the partial volume effect on computing mean grey matter perfusion values, and what we can do with *quantitative* measures.

Registration

Registration of ASL data to the structural image is difficult since the images are low resolution and with limited contrast. The most robust approach appears to be to use the perfusion (or perfusion weighted image) since this has greater tissue contrast and is a closer match to a T1-weighted image than raw ASL (control/label) images. You should *ALWAYS* inspect the results of registration to determine whether it has been effective.

By default in BASIL registration is carried out in multiple steps using the perfusion image directly after the kinetic model inversion, an initial registration having already been done using the raw (undifferenced) ASL data. BASIL now exploits the BBR cost function for registration since this exploits the boundary between grey and white matter seen in the perfusion images. It is possible to use alternative registration strategies and BASIL always produces images in the native space of the data, so that registration can be revisited at a later point.

1.1.7 Advanced Analysis

In the previous sections we have considered what is needed to get a quantitative perfusion image out of ASL data. There are a series of additional techniques that can be used to improve the quality and potentially the interpretability of the results. Whilst these techniques are ‘advanced’, since they go beyond the minimum steps outlined in earlier sections, they are by no means necessarily complicated to perform in practice (being built into BASIL).

Correction for Motion, Distortion and Subtraction Artefacts

Strategies used in other neuroimaging modalities to correct for motion and distortion can also be used with ASL data. A particular source of artefacts for ASL arising in the subtraction of label and control images, giving rise to spurious non-perfusion signal components due to motion related differences. Various (and a growing number) of strategies exist to compensate for these.

Arterial (macrovascular) contribution

There can arise signal from labeled arterial blood in the region of major vessels in ASL data. This is most common in data with short PLD (<1.5 s) or in subjects with particularly prolonged ATT.

In single PLD ASL data you will need to examine the perfusion images for signs of arterial contamination (see the ‘White Paper’ for an example of this). This can also be an issue in patients with vascular diseases, where slow flow and thus long ATT are expected and thus longer PLD might be beneficial

For multi delay data the arterial signal can be accounted for by modelling this arterial component, something included in BASIL by default. When the arterial component is included in the analysis then a further parameter, the arterial blood volume, is available in the output images.

Partial volume correction

The low resolution of ASL data typically means that there is substantial partial voluming of grey (GM) and white matter (WM), plus CSF too. Since GM and WM have very different kinetics (WM tends to have lower perfusion and longer arterial transit time) a normal analysis will provide a perfusion value that is a weighted combination of the two tissue types. Partial Volume Correction attempts to automatically correct for the different tissue type using separately supplied estimates of the partial volumes of the tissues. BASIL can do this automatically as long as you supply a structural image that has been already been processed using `fsl_anat` (or if you supply suitable partial volume estimate images).

T1 values

T1 values are important to the kinetic model inversion and should be chosen based on the field strength that data was acquired at, consideration might also need to be taken of the subject in which analysis is being carried out. BASIL by default takes values for 3T and assumes for the tissue only a grey matter value, unless partial volume correction is applied when separate grey and white matter values are specified. By default a separate value for the T1 of blood is used unless operating in ‘white paper’ mode, where the blood T1 value is also used for the tissue.

Commonly it is assumed that T1 values are fixed across the brain in the quantification. However, these values are not absolutely certain and may well vary across the brain and between individuals. BASIL can take this into account by inferring on T1 values, you should still, however, set sensible expected values.

1.1.8 ASL variants

You are most likely to be pcASL data in practice. There are various other variants of ASL which bring particular advantages, a summary of some notable variants is provided here for reference.

Hadamard/Time-encoded ASL

This is a form of pcASL where the labelling is performed via a series of sub-labels with shorter duration. Individual volumes in the ASL acquisition will vary whether for given periods during the label duration labeling is actually taking place or not. This is normally done according to a specific scheme that means that after decoding it is possible to recover multi-PLD data that appears as if it has been collected with a PLD equal to the sub-label duration. Even more advanced versions vary the sub-label durations.

To analyse this data you can first perform the decoding step to reveal the multi-PLD data. Thereafter this can be used in BASIL (and associated tools) treating the data as label-control subtracted and specifying the relevant (sub-) label duration and PLDs.

QUASAR

This is a special version of pASL which combines data with and without vascular signal suppression. QUASAR can be used to separate signal from tissue and macrovascular contamination. It is possible using QUASAR to iso-

late the macrovascular signal and thus estimate an arterial input function, which enables ‘model-free’ deconvolution. QUASAR uses a Look-Locker readout to achieve sampling of different TIs.

Analysis using both ‘model-based’ and ‘model-free’ methods are provided in the QUASIL tool, a version of BASIL optimised for QUASAR data.

Turbo-QUASAR

This is a form of pASL where multiple sub-boluses are created using a series of labelling pulses. It is a variant on QUASAR ASL. The total effective bolus duration is the summation of the duration each sub-bolus, which is equal to the time between each inversion time (TI) of the Look-Locker readout under normal circumstances where the flow velocity of the arterial blood is about 25cm/s. In conditions where the flow velocity is significantly different from this value, an estimation of the flow velocity is needed from a separate phase contrast MR data. Subsequently, the effective bolus duration can be estimated from the flow velocity information.

To analyse Turbo-QUASAR in BASIL, you can the TOAST command line tool.

1.1.9 Further Reading

To learn more about ASL, acquisition choices, the principles of analysis and how perfusion images can be used in group studies you might like to read:

Introduction to Perfusion Quantification using Arterial Spin Labelling, Oxford Neuroimaging Primers, Chappell, MacIntosh & Okell, Oxford University Press, 2017.

Online examples are available to go with this primer using the BASIL tools. These can be found on the Oxford Neuroimaging Primers website: <http://www.neuroimagingprimers.org>

The following book remains a good introduction to functional imaging including perfusion using ASL:

Introduction to Functional Magnetic Resonance Imaging: principles and Techniques. Buxton, Cambridge University Press, 2009.

Overview of the BASIL toolset

2.1 Higher-level packages

These provide a single means to quantify CBF from ASL data, including kinetic-model inversion, absolute quantification via a calibration image and registration of the data. This will generally be the first place to go for most people who want to do processing of ASL data.

- `asl_gui` - The graphical user interface that brings the BASIL tools together in one place.
- `oxford_asl` - A command line interface for most common ASL perfusion analysis.

2.2 Individual analysis components

These are designed to perform a specific task in the analysis of ASL data. In many cases the higher-level tools will call them when required, however they can also be used individually in cases where the high-level processing pipeline is not suitable.

- `basil` (command) - this is the core tool that performs kinetic-model inversion to the data using a Bayesian algorithm. You should only need to use it directly for more custom analyses than that offered by `oxford_asl`/`Asl_gui`.
- `asl_calib` - this tool takes a supplied calibration volume and calculates the magnetization of arterial blood allowing CBF to be quantified in absolute units. The main functionality of `asl_calib` is built into `oxford_asl`, `Asl_gui` and `QUASIL`, but more options are available when using it directly.
- `asl_reg` - this tool is designed to assist in registration of (low resolution) ASL images to structural or standard brain images. The functionality of `asl_reg` is built into `oxford_asl` and `Asl_gui`.
- `asl_file` - a command line tool for the manipulation of ASL data files, particularly designed to cope with the complex structure of interleaved label and control images combined with multiple post-labeling delays.
- `quasil` - A special version of BASIL optimised for QUASAR ASL data, includes model-based or model-free analyses along with calibration.
- `toast` - A special version of BASIL optimised for Turbo-QUASAR ASL data, includes model-based analyses, calibration, and correction for MT effects.

3.1 Worked examples from the FSL course

This tutorial demonstrates some of the common options available in the Basil GUI.

We will be working with single and multi-PLD data from the [FSL tutorial on Arterial Spin Labelling](#).

If you are taking part in an organized course the data will have been downloaded for you and available in:

```
~/fsl_course_data/ASL
```

Note: If you are not taking part in a course you will need to download the following ASL data before following the tutorial:

<https://fsl.fmrib.ox.ac.uk/fslcourse/downloads/asl.tar.gz>

To extract the data from a terminal window you should run:

```
tar -xzf asl.tar.gz
```

The tutorial has been written so that we start with the most basic analysis and gradually add options and show the effect they have on the output. However this is not a complete description of all available options in the GUI.

Contents

- *Perfusion quantification using Single PLD pcASL*
 - *The data*
 - *Launching the GUI*
 - *(Simple) Perfusion Quantification*
- *Improving the Perfusion Images from single PLD pcASL*

- *Motion and Distortion correction*
- *Making use of Structural Images*
- *Different model and calibration choices*
- *Partial Volume Correction*
- *Perfusion Quantification (and more) using Multi-PLD pcASL*
 - *The data*
 - *Perfusion Quantification*
 - *Arterial/Macrovascular Signal Correction*
- *Partial Volume Correction*

3.1.1 Perfusion quantification using Single PLD pcASL

The aim of this exercise is to perform perfusion quantification with one of the most widely recommended variants of ASL. Single PLD pcASL is now regarded as sufficiently simple and reliable, both for acquisition and analysis, that it is the first option most people should consider when using ASL for the first time. Although more can be done with other ASL variants, particularly when acquisition time allows.

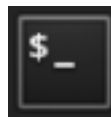
The data

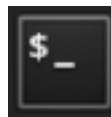
This dataset used **pcASL labelling** and we are going to start with data collected using a **single post-label delay**. This dataset follows as closely as possible the recommendations of the ASL Consensus Paper (commonly called the ‘White Paper’) on a good general purpose ASL acquisition, although we have chosen to use a 2D multi-slice readout rather than a full-volume 3D readout.

The files you will need to begin with are:

- `spld_asltc.nii.gz` - the label-control ASL series containing 60 volumes. That is 30 label and 30 control, in pairs of alternating images with label first.
- `aslcalib.nii.gz` - the calibration image, a (largely) proton-density weighted image with the same readout (resolution etc) as the main ASL data. The TR for this data is 4.8 seconds, which means there will be some T1 weighting.
- `aslcalib_PA.nii.gz` - another calibration image, identical to `aslcalib.nii.gz` 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.
- `T1.nii.gz` - the T1-weighted anatomical of the same subject.

Launching the GUI



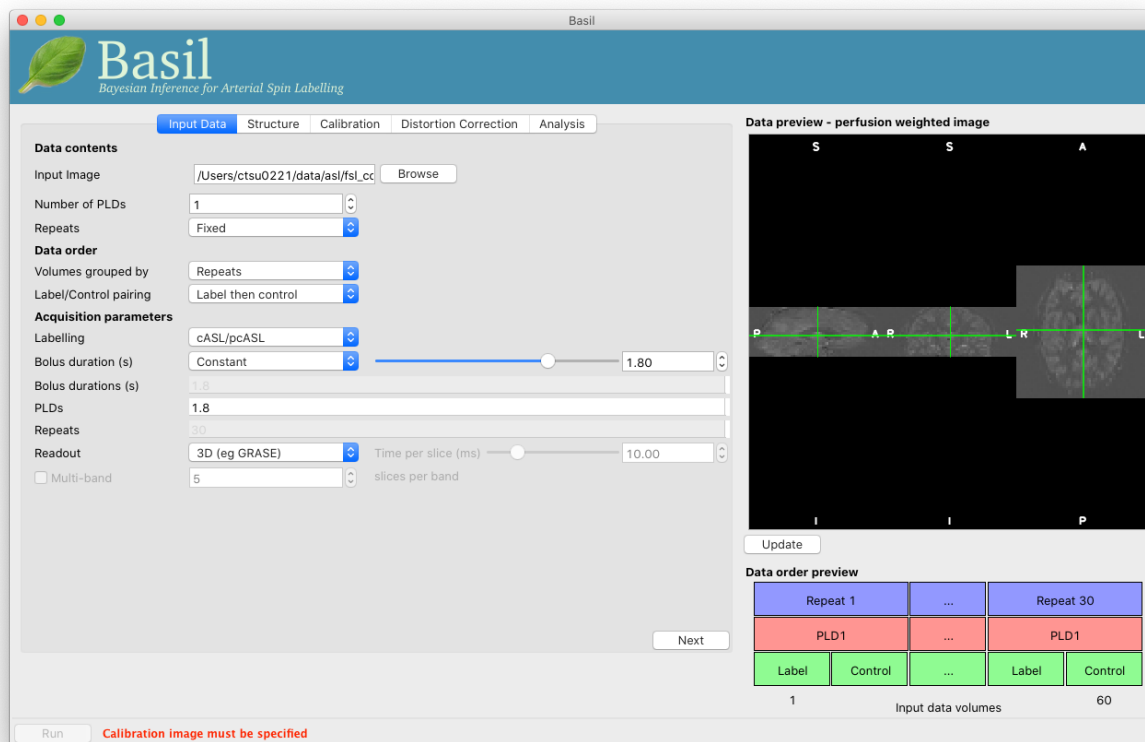
To launch the GUI, open a terminal window by clicking on . At the command line you will need to type `asl_gui`. Don't worry if you see a few 'errors' in the terminal window - this is normal!

Note: Make sure you launch the GUI from a directory you have write permission on - otherwise you may get errors when you try to run the analysis. If you just open a terminal and type `asl_gui` you will be fine.

Once it has launched you will find yourself on the ‘Input Data’ tab, you should:

- Click ‘Browse’ and load the ASL data `spld_asltc.nii.gz` from `~/fsl_course_data/ASL` as the ‘Input Image’.
- Set the ‘Number of PLDs’, which in this case is 1, this is already done by default.
- Click the ‘Update’ button beneath the ‘Data Preview’ pane on the right.

At this point the GUI should look like the screen shot below and a perfusion weighted image will have appeared in the ‘Data Preview’ pane. This this is reassuring, if we didn’t see something that looks roughly like this, we might check if the data order that the GUI is expecting matches that in the data. We could alter the ‘Data order’ settings if needed and update the preview again.



Note also, beneath the ‘Data Preview’, that there is a ‘Data order preview’. The idea of this graphic is to help visually to confirm that the way that the GUI is interpreting the ordering of volumes in the data matches what you are expecting. In this case we have a single PLD repeated 30 times with the label and control images paired in the data (this is pretty common).

In the ‘Data order preview’ diagram, the volumes in the input data are shown on the horizontal axis - in this case from 1 to 60. The blue blocks at the top show that these volumes are divided into 30 repeats. Within each repeat the red boxes show that we have a single PLD and within that PLD the green boxes show a tag and control image (in that order). This pattern is repeated (shown by the ellipsis boxes ...) for the 30 repeats.

You can try a different ‘Data order’ option to see what happens. For example, change ‘Label/Control pairs’ from ‘Label then control’ to ‘Control then label’. This switches the expected order of label and control images within the

pair. If you then update the preview you will find that the contrast reverses, the perfusion now has the wrong ‘sign’.

(Simple) Perfusion Quantification

We have checked the PWI, thus we can proceed to final quantification of perfusion, inverting the kinetics of the ASL label delivery and using the calibration image to get values in the units of ml/100g/min.

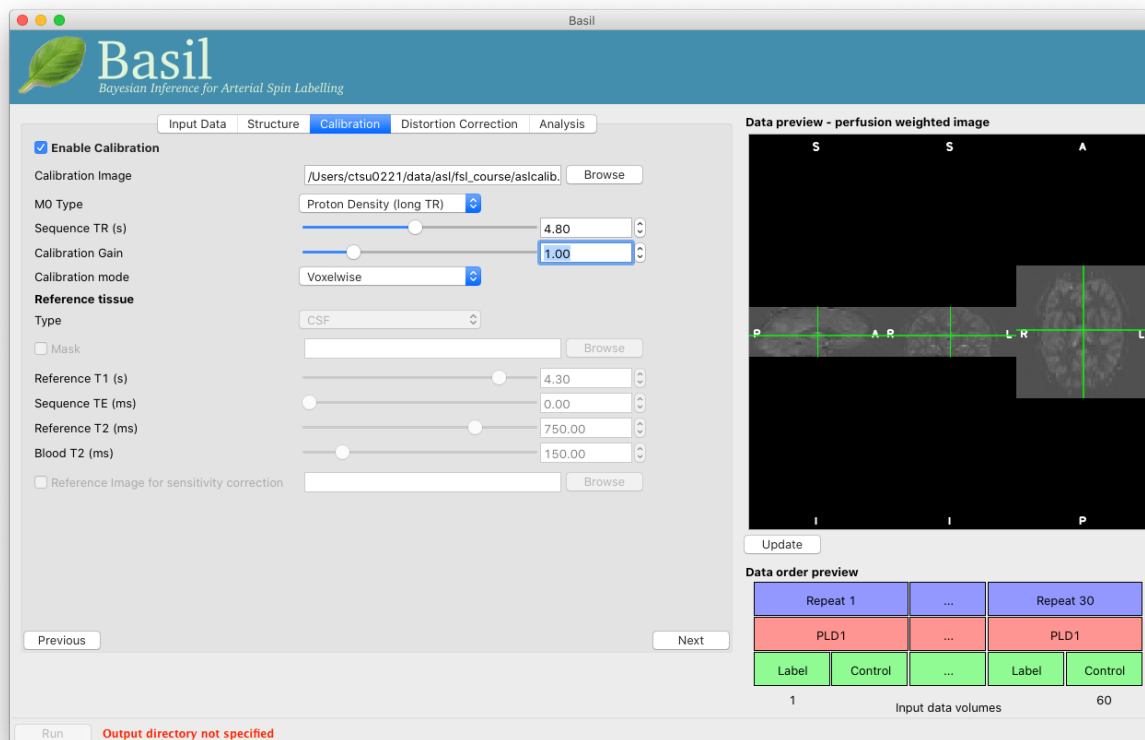
To do this we need to tell the BASIL GUI some information about the data and the analysis we want to perform.

On the ‘Input Data’ tab we need to specify the ‘Acquisition parameters’:

- Labelling - cASL/pcASL (the default option).
- Bolus duration (s) - 1.8 (default).
- PLDs (s) - 1.8 (default).
- Readout - 2D multi-slice (you will need to set this).
- Time per slice (ms) - 45.2 (only appears when you change the Readout option).

You can now hit ‘Next’ and you will be taken to the next tab. For this (simple) analysis we do not want to use a structural image, so we can move on by clicking ‘Next’ again. Or we could skip straight to the ‘Calibration’ tab using the menu across the top.

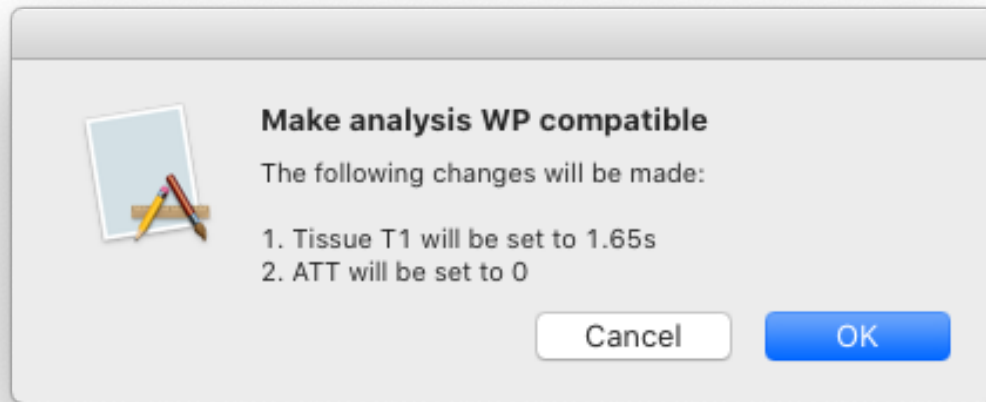
On the ‘Calibration’ tab, ‘Enable Calibration’ first, then load the calibration image `aslcalib.nii.gz`. Change the ‘Calibration mode’ to ‘voxelwise’, and set the ‘Sequence TR (s)’ to be 4.8.



Finally, we need to set the analysis options: either skip to the ‘Analysis’ tab or click ‘Next’ twice.

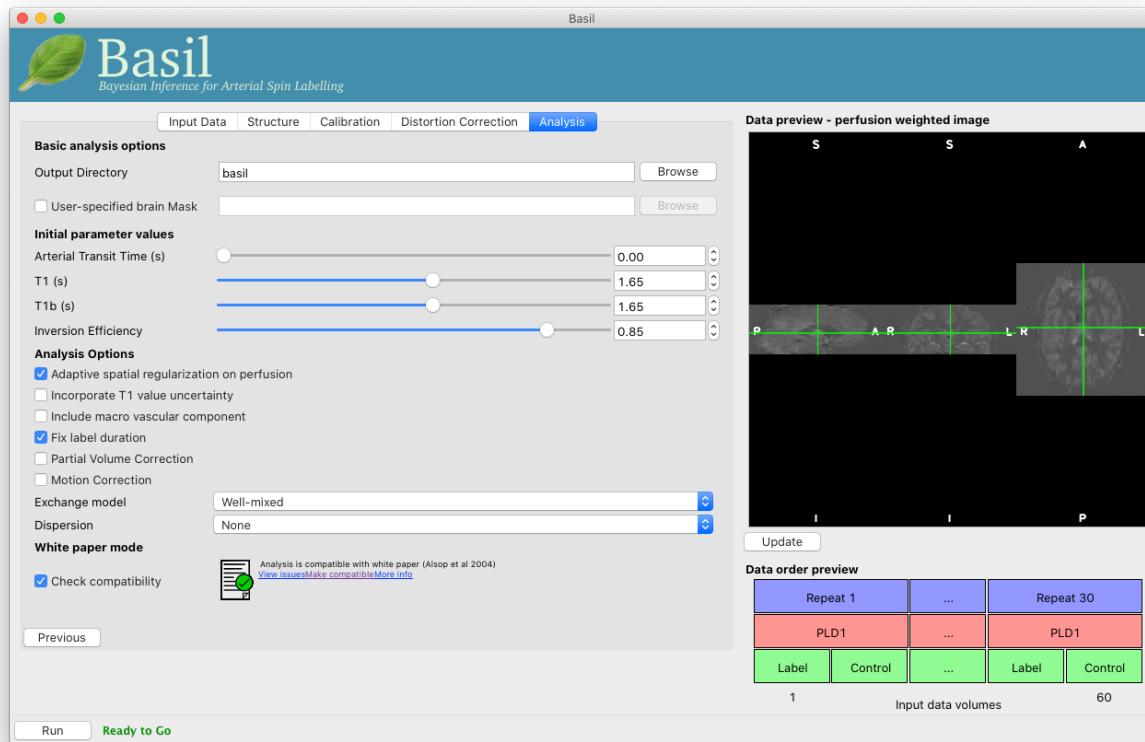
On the ‘Analysis’ tab, choose an output directory name, e.g., `basil`.

We want our analysis to use the same parameter values proposed in the 'ASL White Paper' quantification formula. To check this, select 'Check Compatibility' under the heading 'White Paper Mode'. A status icon will be displayed showing that currently our analysis is not compatible with the white paper. To fix this, click 'Make compatible' - the GUI will first tell you what it is going to change:



Click 'OK' to make these changes and note that the analysis is now marked as white paper compatible.

Note that in the lower left corner the GUI is now telling us that we are 'Ready to Go'. At this point you can click 'Run' to start the analysis.



The output of the `oxford_asl` command line tool is shown in a pop-up window. You can ignore any `erfc underflow` error messages - they are harmless and occur because we haven't provided any structural data.

This analysis should only take a few minutes, but 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, view the final result:

```
fsleyes basil/native_space/perfusion_calib.nii.gz
```

Note that if you just supply a name for the output directory (not a full path), as we have here, this will be placed in the 'working directory', i.e. whichever directory you were in when you launched the GUI.

You will find something that looks very similar to the PWI we viewed before, but now the values at every voxel are in `ml/100g/min`.

You will also find a PWI saved as `basil/native_space/perfusion.nii.gz`. This is very similar to the PWI displayed in the preview pane, except that the kinetic model inversion has been applied to it, this is the image pre-calibration.

Note: If you want to view any of the output log files you can use the `gedit` text editor - just type `gedit` from a terminal window

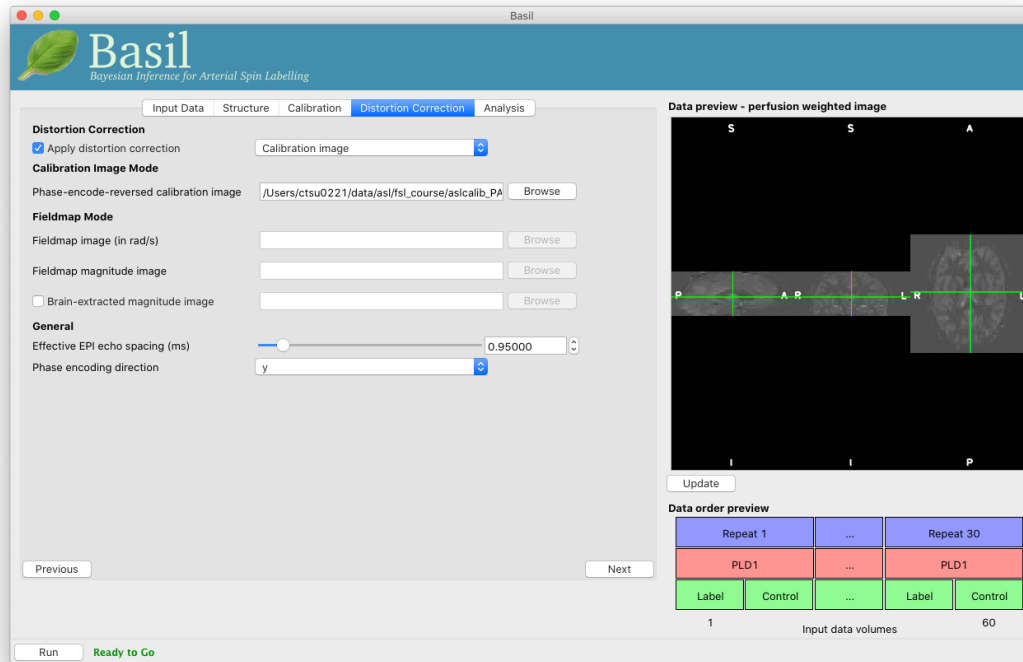
3.1.2 Improving the Perfusion Images from single PLD pcASL

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

Go back to the GUI which should still be setup from the last analysis you did (if you have closed it follow the steps above to repeat the setup - but do not click run).

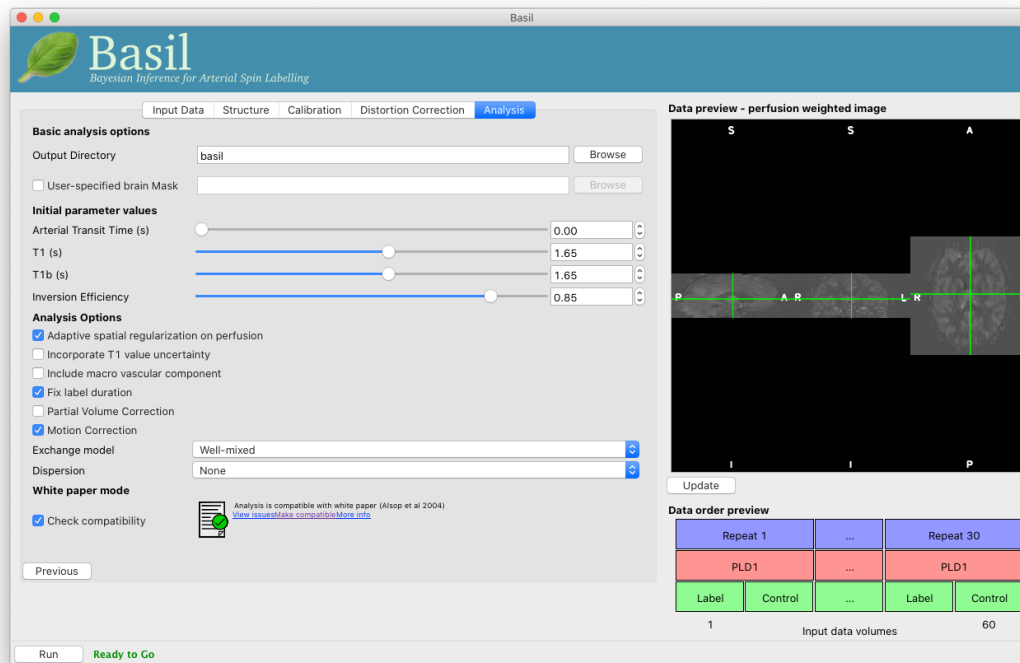
On the 'Distortion Correction' tab, select 'Apply distortion correction'. Load the 'Phase-encode-reversed calibration image' `aslcalib_PA.nii.gz`. Set the 'Effective EPI echo spacing' (also known as the dwell time) to 0.95ms and the 'Phase encoding direction' to 'y'.



On the 'Analysis' tab, select 'Motion Correction'. Make sure you have 'Adaptive spatial regularisation on perfusion' 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.

You might like to change the name of the output directory at this point, so that you can compare to the previous analysis.

Now click 'Run'.



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.

To view the final result:

```
fsleyes basil/native_space/perfusion_calib.nii.gz
```

The result will be similar to the analysis in Example 1 although the effect of distortion correction should be noticeable in the anterior portion of the brain. The effects of motion correction are less obvious, this data does not have a lot of motion corruption in it.

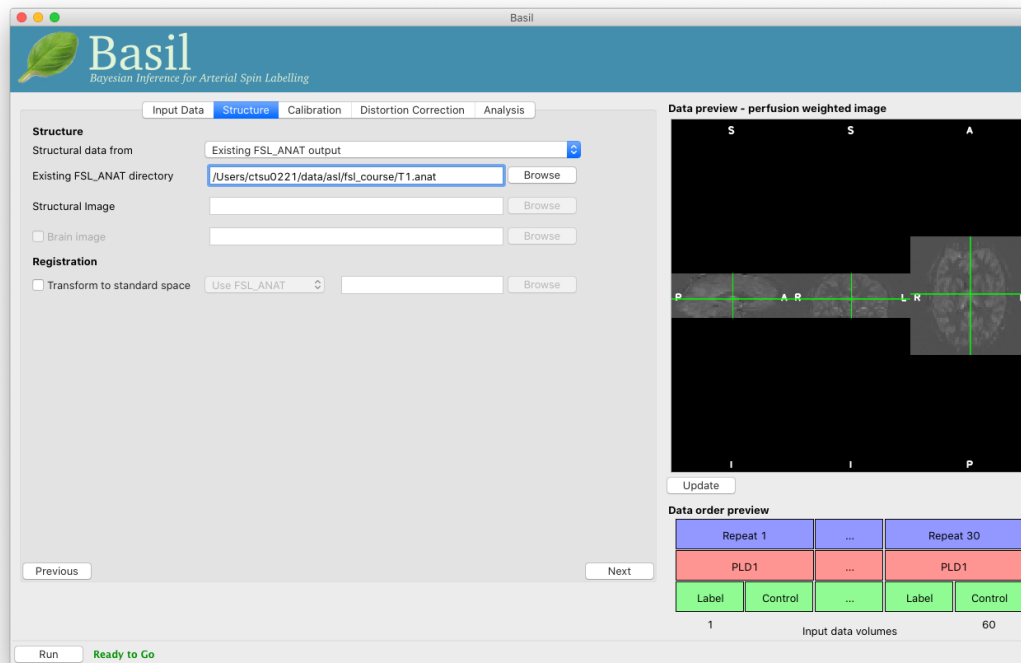
Making use of Structural Images

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 repeat the analysis above but now providing structural information. The recommended way to do this is to take your T1 weighted structural image (which is most common) and firstly process using `fsl_anat`, passing the output directly from that tool BASIL.

For this practical `fsl_anat` has already been run for you and you will find the output in the data directory as `~/fsl_course_data/ASL/T1.anat`

Go back to the analysis you have setup above. On the ‘Structure’ tab, for ‘Structural data from’ select ‘Existing FSL_ANAT output’. Then for the ‘Existing FSL_ANAT output’ choose `T1.anat`.



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 results in the output directory:

- `basil/struct_space` - this sub-directory contains results transformed into the same space as the structural image. The files in here will match those in the `native_space` subdirectory of the earlier analysis, i.e., containing perfusion images with and without calibration.
- `basil/native_space/asl2struct.mat` - this is the (linear) transformation between ASL and structural space. It can be used along with a transformation between structural and template space to transform the ASL data into the template space. It was used to create the results in `basil/struct_space`.
- `basil/native_space/perfusion_calib_gm_mean.txt` - this contains the result of calculating the perfusion within a gray matter mask, these are in ml/100g/min. The mask was derived from the partial volume estimates created by `fsl_anat` and transformed into ASL space followed by thresholding at 70%. This is a helpful check on the absolute perfusion values found and it is not atypical to see values in the range 30-50 here. There is also a white matter result (for which a threshold of 90% was used).
- `basil/native_space/gm_mask.nii.gz` - this is a mask that represents areas in which there is some grey matter (at least 10% from the partial volume estimates). This can be useful for visualisation, but mainly when looking at partial volume corrected data.
- `basil/native_space/gm_roi.nii.gz` - this mask represents voxels which are close to 'pure' GM. It is used for the calculation of the mean perfusion in gray matter described above. There is also the associated white matter mask.

Different model and calibration choices

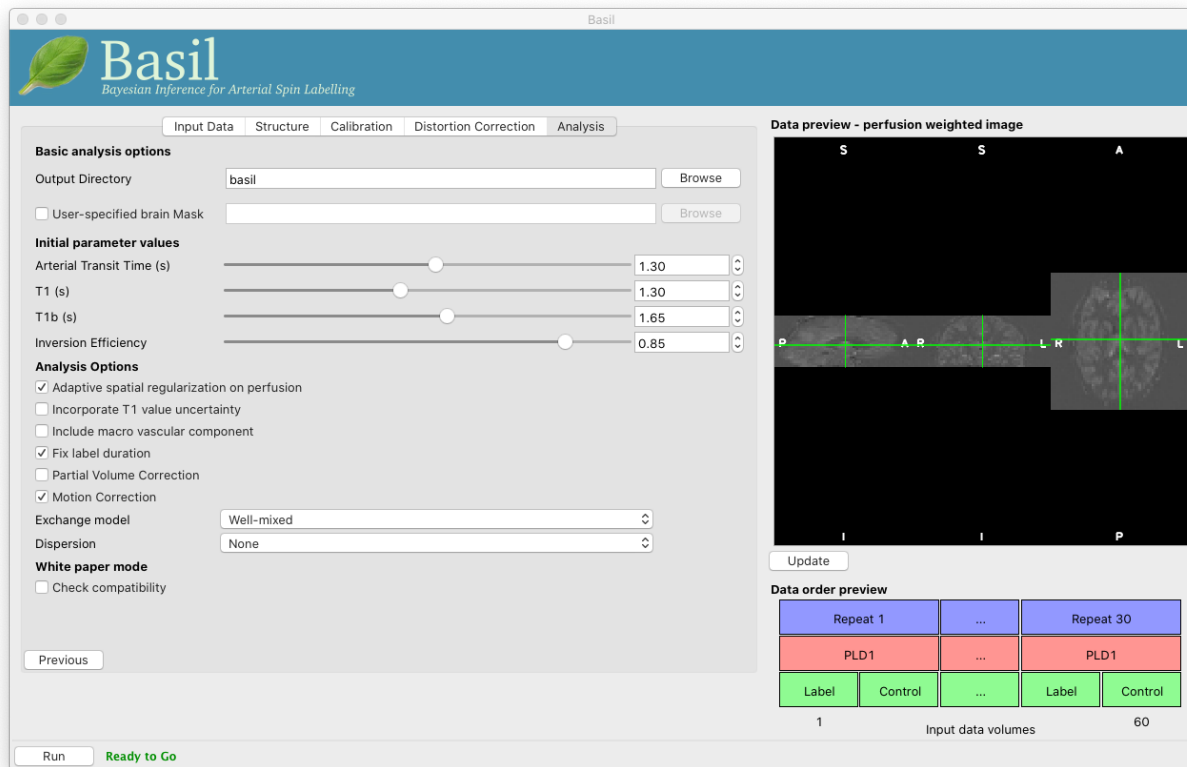
Thus far the calibration to get perfusion in units of ml/100g/min has been done using a voxelwise division of the relative perfusion image by the (suitably corrected) calibration image - so called 'voxelwise' calibration. This is in

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.

Go back to the analysis you have already set up. We are now going to move away from ‘White Paper’ mode to get a potentially more accurate analysis. To do this return to the ‘Analysis’ tab and set the ‘Arterial Transit Time’ to 1.3s (the default value for pcASL in BASIL based on our experience with pcASL labeling plane placement).

Also set ‘T1’ (tissue blood T1 value) to 1.3s, different from ‘T1b’ (for arterial blood) since the Standard (aka Buxton) model for ASL kinetics considers labeled blood both in the vasculature and the tissue.

Note that the ‘White Paper Mode’ is indicating that the analysis is no longer white paper compatible - since we don’t care about this any more we can turn off ‘Check compatibility’ to remove the warning.



Now that we are not in ‘White Paper’ mode we can also change the calibration method. On the ‘Calibration’ tab, change the ‘Calibration mode’ to ‘Reference Region’. Now all of the ‘Reference tissue’ options will become available, but leave these as they are: we will accept the default option of using the CSF (in the ventricles) for calibration.

You could click ‘Run’ now and wait for the analysis to complete. But, in the interests of time we will save ourselves the bother of doing all of the registration all over again. Before clicking run, therefore, do:

- On the ‘Calibration’ tab select ‘Mask’ and load `csfmask.nii.gz` from the data directory. This is a ready prepared ventricular mask for this subject. (in fact it is precisely the mask you would get if you ran the analysis as setup above).

While this is running you might want to read ahead, or if you are keen to keep moving through the examples, then skip this analysis and keep going.

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. You can check on some of the interim calculations for the calibration by looking in the `basil/calib` subdirectory: here you

will find the value of the estimated equilibrium magnetization of arterial blood for this dataset in `M0.txt` and the reference tissue mask in `refmask.nii.gz`. It is worth checking that the latter does indeed only lie in the ventricles when overlaid on an ASL image (e.g. the perfusion image or the calibration image), it should be conservative, i.e., only select voxels well within the ventricles and not on the boundary with white matter.

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 an image of gray matter perfusion directly (along with a separate image for white matter).

This is very simple to do via the GUI. Return to your earlier analysis. You will need to revisit the ‘Structure’ tab and reload the `T1.anat` result as you did above, 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’. That is it! You might not want to click ‘Run’ at this point because partial volume correction takes substantially longer to run.

You will find the results of this analysis already completed for you in the directory `~/fsl_course_data/ASL/basil_spld_pvout`. In this results directory you will still find an analysis performed without partial volume correction in `basil/native_space` as before. The results of partial volume correction can be found in `basil/native_space/pvcorr`. This new subdirectory has the same structure as the non-corrected results, only now `perfusion_calib.nii.gz` is 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.

3.1.3 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.

The data

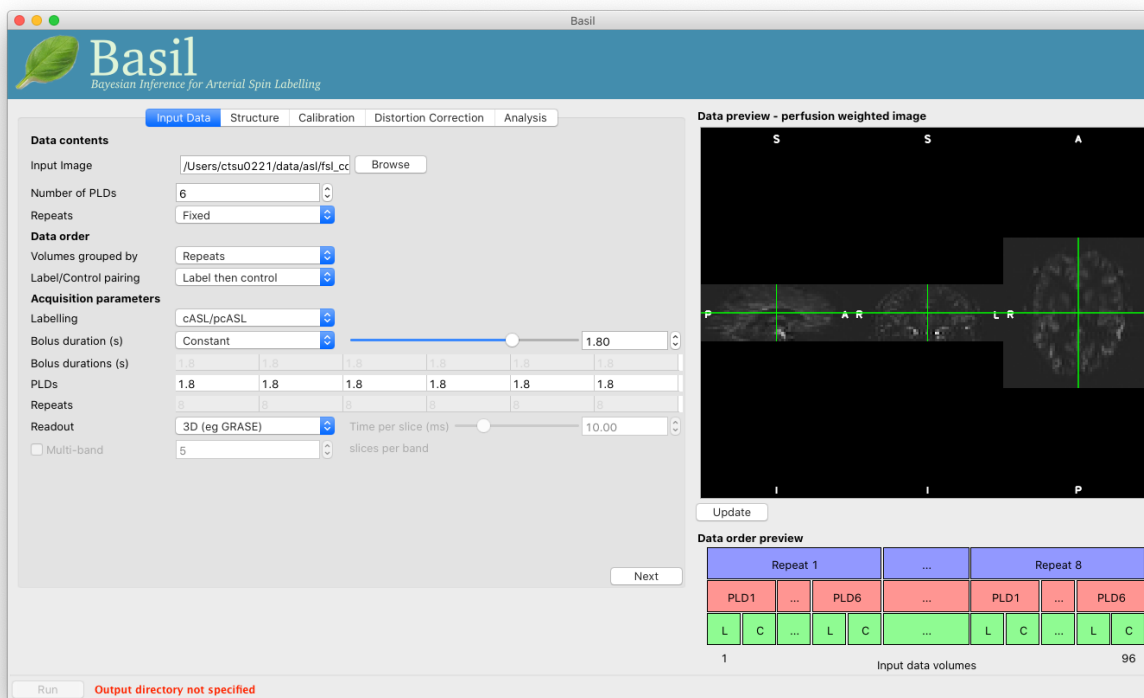
The data we will use in this section supplements the single PLD pcASL data above, adding multi-PLD ASL in the same subject (collected in the same session). This dataset used the same pcASL labelling, but with a label duration of 1.4 seconds and 6 post-labelling delays of 0.25, 0.5, 0.75, 1.0, 1.25 and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

The files you will also now need are:

- `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

Load the GUI (`asl_gui`), it is best to start a whole new analysis as we are moving on to a new set of data and not reuse any GUI you already have open. On the ‘Input Data’ tab, for the ‘Input Image’ load `mpld_asltc.nii.gz`. Unlike the single-PLD data, we need to specify the correct number of PLDs, which is 6. At this point the ‘Number of repeats’ should correctly read 8. Click ‘Update’ below the ‘Data preview pane’. A perfusion-weighted image should appear - this is an average over all the PLDs (and will thus look different to Example 1).



Note the 'Data order preview'. For multi-PLD ASL it is important to get the data order specification right. In this case the default options in the GUI are not correct. The PLDs do come as label-control pairs, i.e. alternating label then control images. But, the default assumption in the GUI is that a full set of the 6 PLDs has been acquired first, then this has been repeated 8 subsequent times.

This is indicated in the preview by the 96 input volumes being divided up into 8 repeats (blue boxes), each containing 6 PLDs (red boxes).



This is quite commonly how multi-PLD ASL data is acquired, but that might not be how the data is ordered in the final image file.

As we noted earlier, in this data all of the repeated measurements at the same PLD are grouped together. Under 'Data order' you need to change 'Volumes grouped by' from 'Repeats' to 'PLDs'.

Note that the data order preview changes to reflect the different ordering. Now the 96 volumes are divided up into 6 PLDs (red boxes now at the top), and within each red PLD box there are 8 repeats (blue boxes). This is now correct.

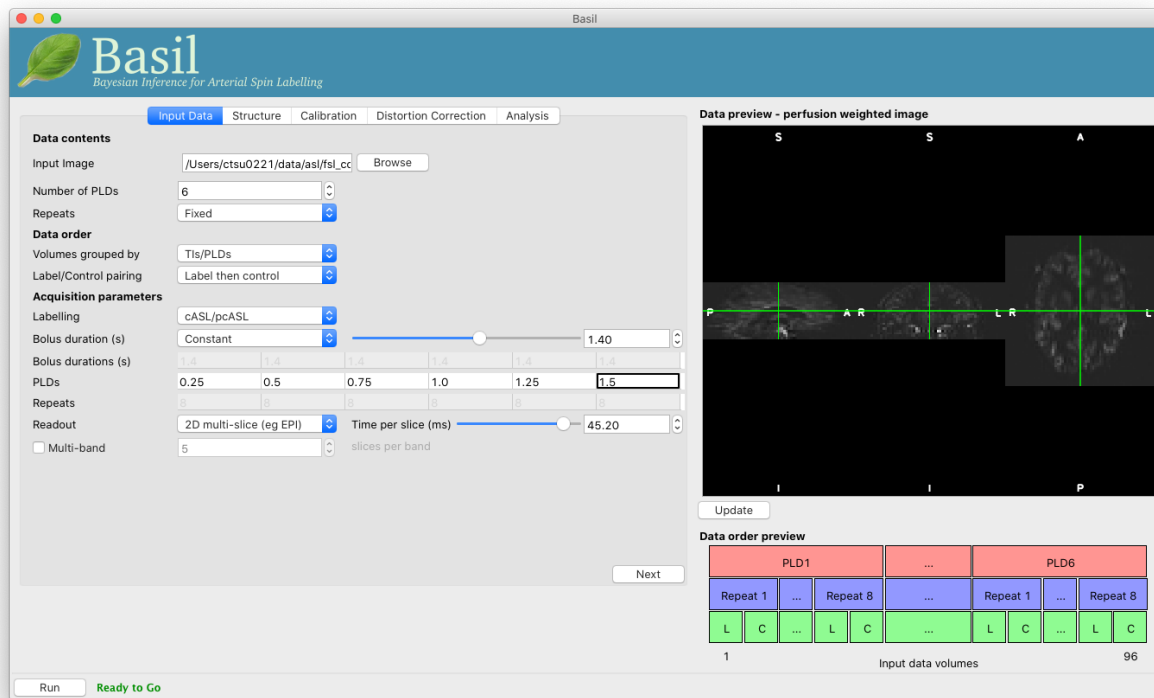


Note that if you were to click ‘Update’ on the ‘Data preview’ nothing changes, the ordering doesn’t affect the (simple) way in which we have calculated the PWI. Getting a plausible looking PWI is a good sign that the data order is correct, but it is not a guarantee that the PLD ordering is correct, so always check carefully. One way to do this, in this case, would be to open the data in `fsleyes` and look at the timeseries: the raw intensity of both label and control images for one PLD are different to those from another PLD (due to the background suppression). The timeseries for the raw data looks like a series of steps, indicating the repeated measurements from each PLD are grouped together (grouped by ‘repeats’).

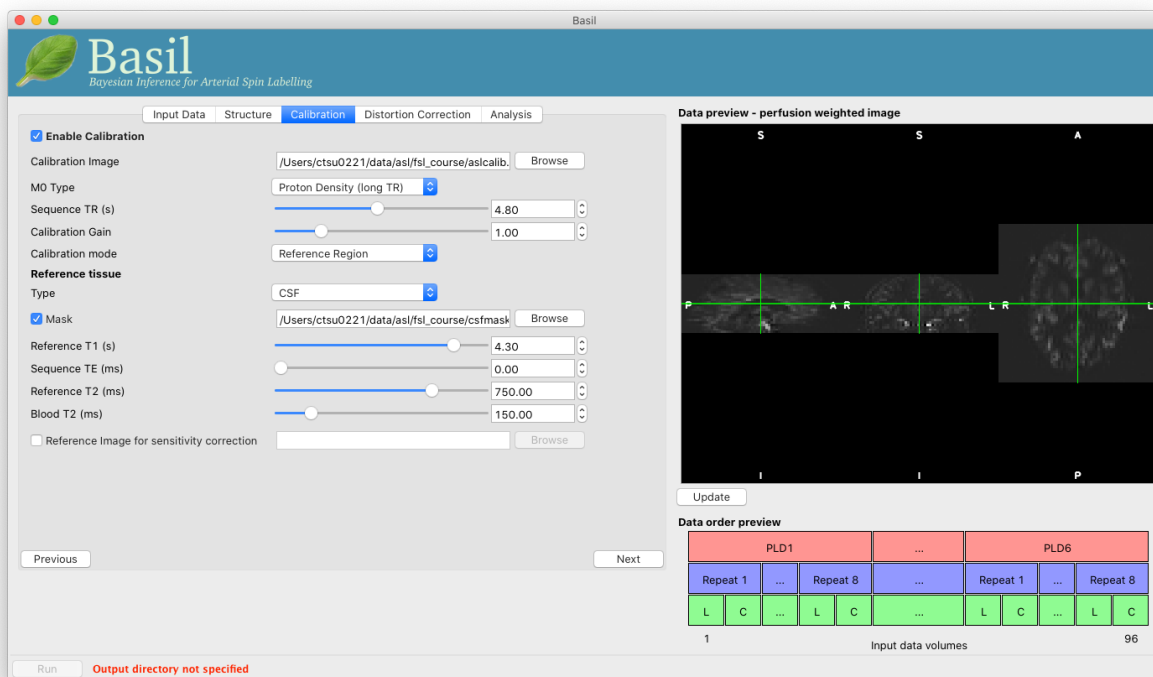
Once we are happy with the PWI and data order, we can set the ‘Acquisition parameters’:

- Labelling - ‘cASL/pcASL’ (default).
- Bolus duration (s) - 1.4 (shorter than the default).
- PLDs (s) - 0.25, 0.5, 0.75, 1.0, 1.254, 1.5.
- Readout - ‘2D multi-slice’ with ‘Time per slice’ 45.2.

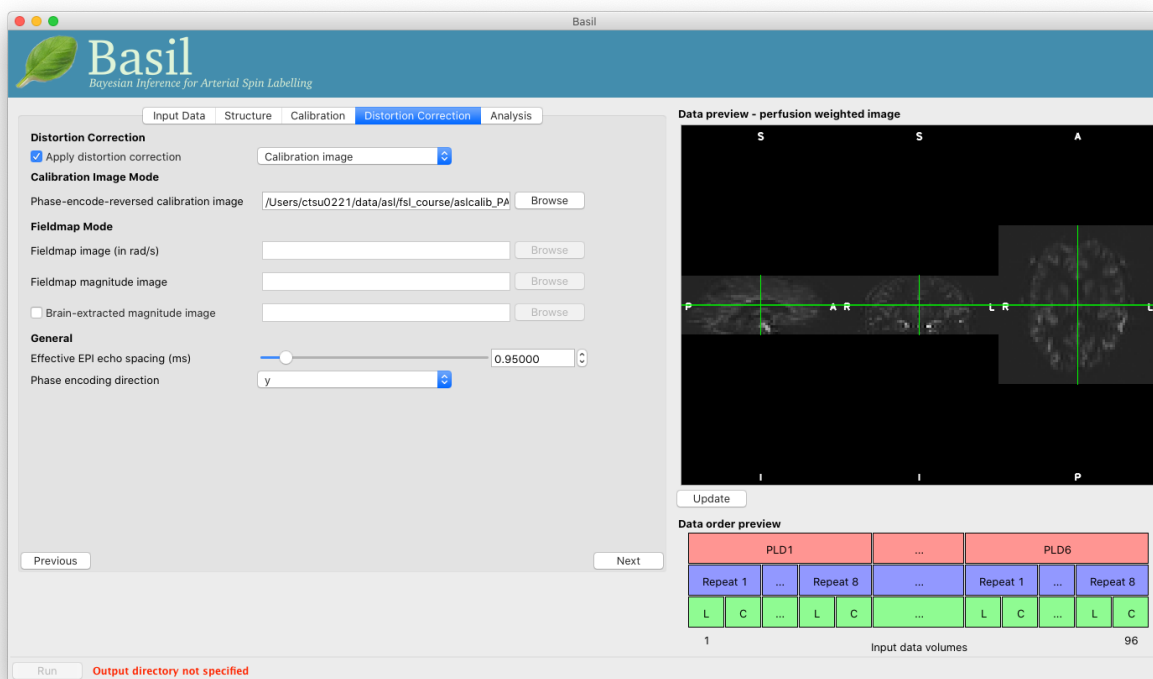
The input tab should appear as follows:



Move to the 'Calibration' tab, select 'Enable Calibration' and as the 'Calibration Image' load the `aslcalib.nii.gz` image from the Single-PLD data (it is from the same subject in the same session so we can use it here too). We have skipped the 'Structure' tab (to make the analysis quicker), this means if we want 'Calibration mode' to be 'Reference Region' we need to supply a mask of the region of tissue to use. Select 'Mask' and load `csfmask.nii.gz`. Set the 'Sequence TR' to be 4.8, but leave all of the other options alone.



Move to the 'Distortion Correction' tab. Select 'Apply distortion correction'. Load the 'Phase-encode-reversed calibration image' `aslcalib_PA.nii.gz` from the Single-PLD pcASL data. Set the 'Effective EPI echo spacing' to 0.95ms again and the 'Phase encoding direction' to 'y'.



Finally, move to the 'Analysis' tab. Choose an output directory, leave all of the other options alone. Click 'Run'.

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 directory 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 `arrival.nii.gz` image. 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.nii.gz` 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 'Include macro vascular component'. Click 'Run'.

The results directory should be almost identical to the previous run, but now we also gain some new results:

- `aCBV.nii.gz` and
- `aCBV_calib.nii.gz`

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. 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.

3.1.4 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. We can do as we did before and load `T1.anat`. 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/basil_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.

The End.

3.2 Cerebrovascular Reserve Quantification Using ASL

3.2.1 CVR quantification using Single PLD pCASL

Introduction

Cerebrovascular reserve (CVR) is defined as the maximum change in perfusion in response to a vasoactive stimulus. CVR has become an important biomarker to assess tissue health and ASL offers a non-invasive technique to measure CVR in vivo. Measuring CVR requires the quantification of perfusion under two different physiological conditions: baseline and (physiologically) stimulated. In the baseline condition, it is common for us to follow the routine procedures where we acquire data while the subject is in a resting state in the scanner. In the stimulus condition, we need to administer a stimulus to manipulate the perfusion of the subject. The choice of the stimulus depends on the availability and the condition of the subject. Nevertheless, the key component in designing a CVR experiment is to change the perfusion of the subject to a different level (from the normal resting state level).

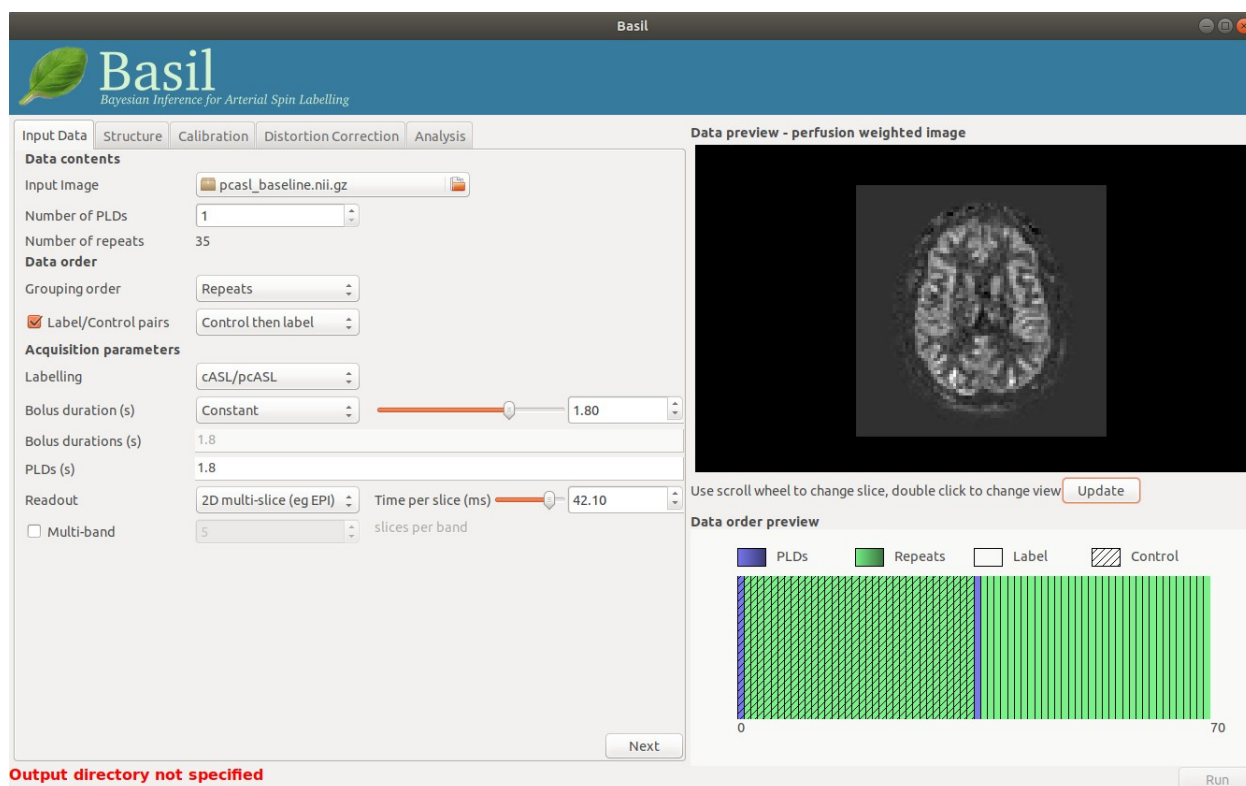
In this tutorial, we are going to illustrate an example CVR study in which CVR was quantified using PCASL and acetazolamide as the stimulus. Sample data can be downloaded [here](#).

ASL Sequence

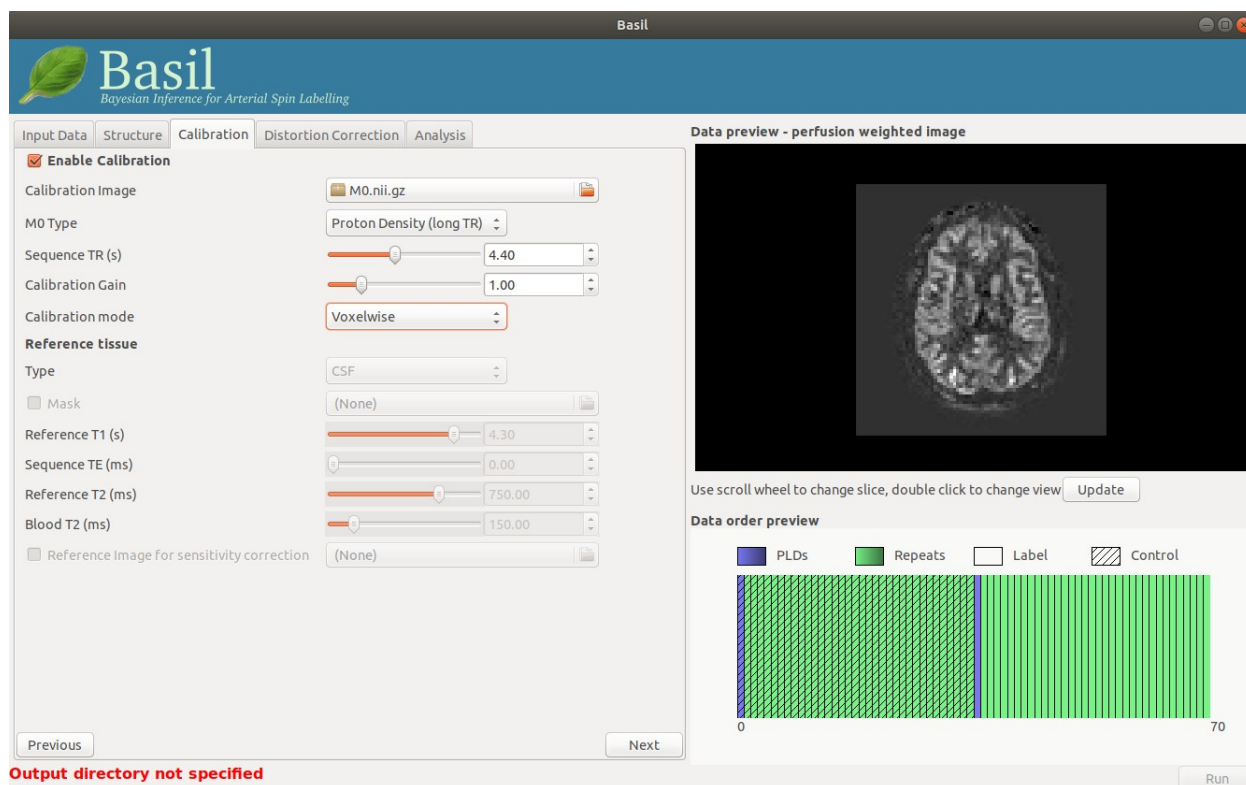
Single-PLD PCASL is used in this experiment, and the sequence parameters are similar to the ones in the ASL white paper. Specifically, the bolus duration is 1800ms, PLD is 1800ms, no background suppression, 2D EPI readout, and the gap between each slice is 42.1ms. There were 140 repeats in this data. The first 35 repeats were collected in resting condition. At repeat 35, acetazolamide was administered. The last 35 repeats were used as the data to quantify CBF in the stimulus condition. The data has already been split into separate data for the resting and stimulus conditions respectively. The full description of the parameters can be found in the reference paper of this tutorial. Calibration data was also acquired using a long TR of 4400ms and 6 repeats.

Data Analysis: Resting State Perfusion

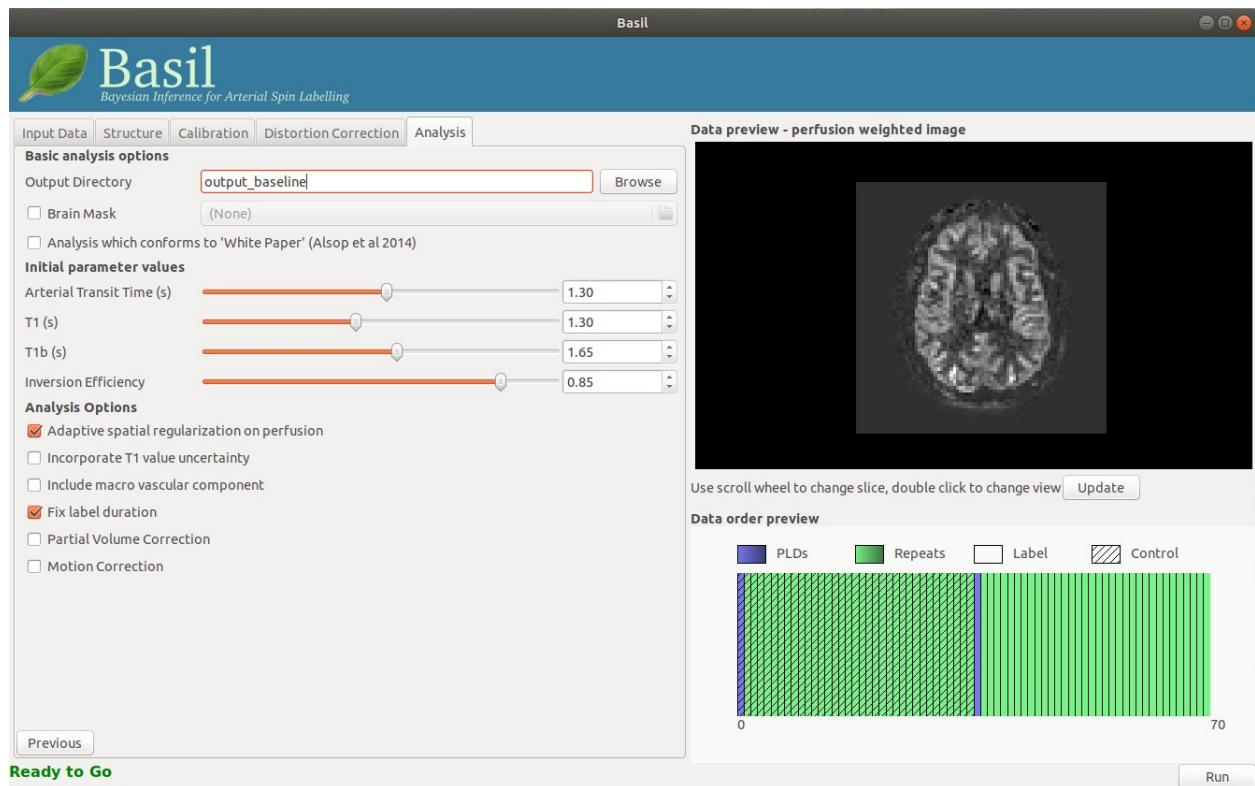
We first quantify perfusion in the resting condition using the PCASL data of the resting state. We can use the BASIL GUI to estimate voxelwise perfusion values in absolute units. We can key in the sequence parameters in the GUI.



In order to calibrate the CBF into absolute units, we need to input the calibration data and select voxel-wise calibration.

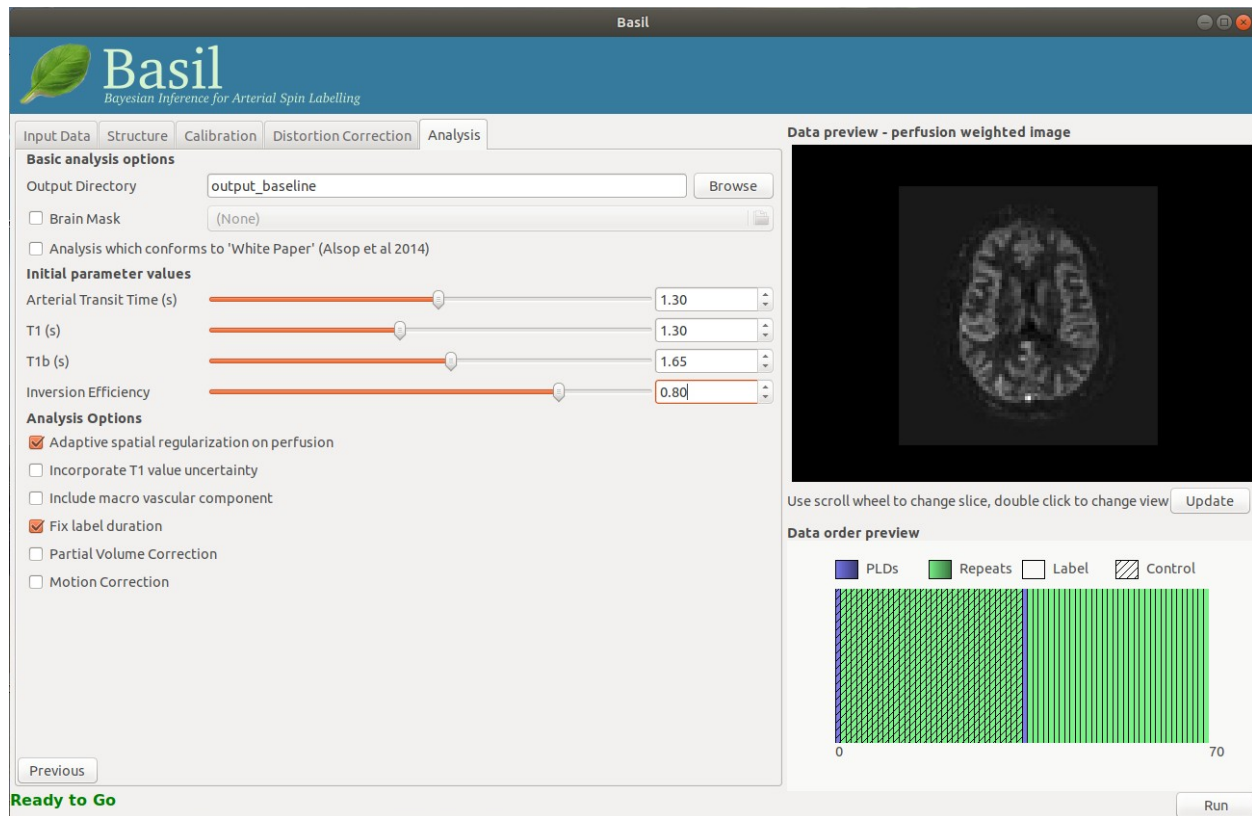


Finally, we need to set up the output director. Now we can click Run.

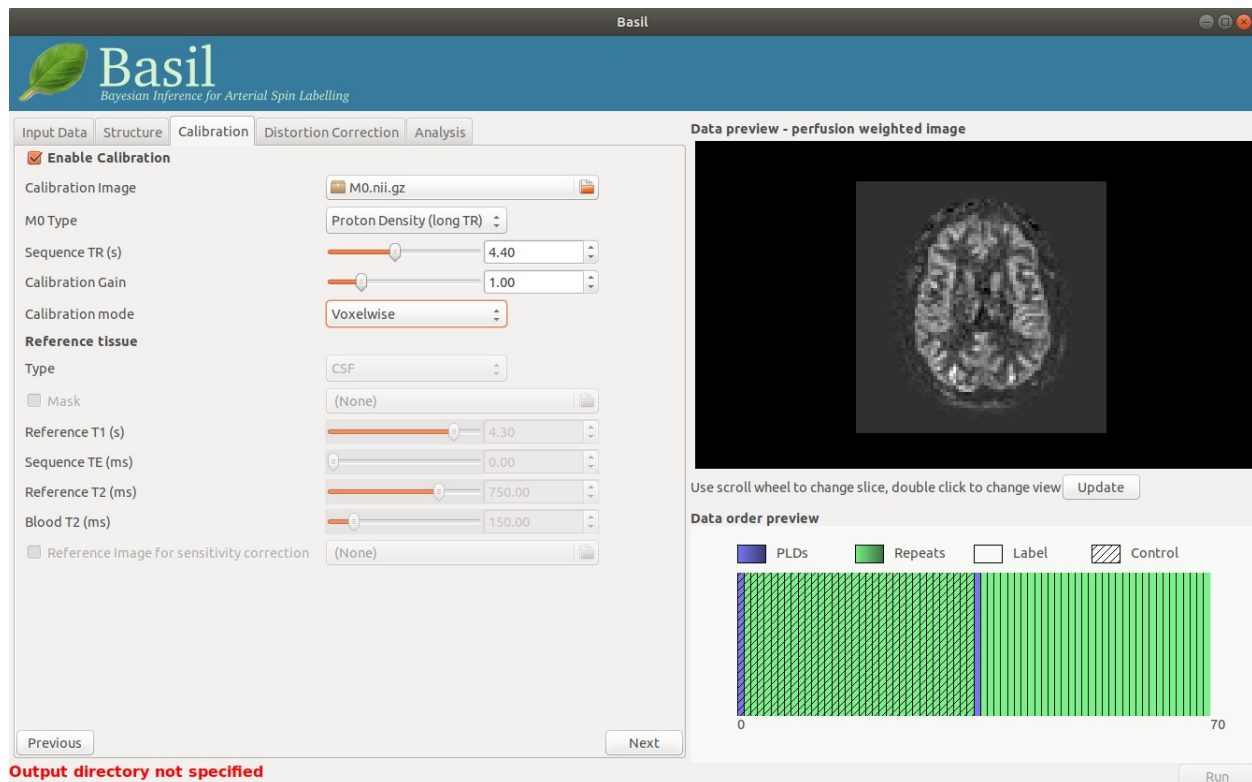


Data Analysis: Stimulus State Perfusion

After quantifying the perfusion of the resting condition, we need to estimate the pwefuaion of the stimulus condition (in this case after the injection of acetazolamide). Since the data of the stimulus condition comes from the same scanning session of the resting state, the sequence parameters are exactly the same:

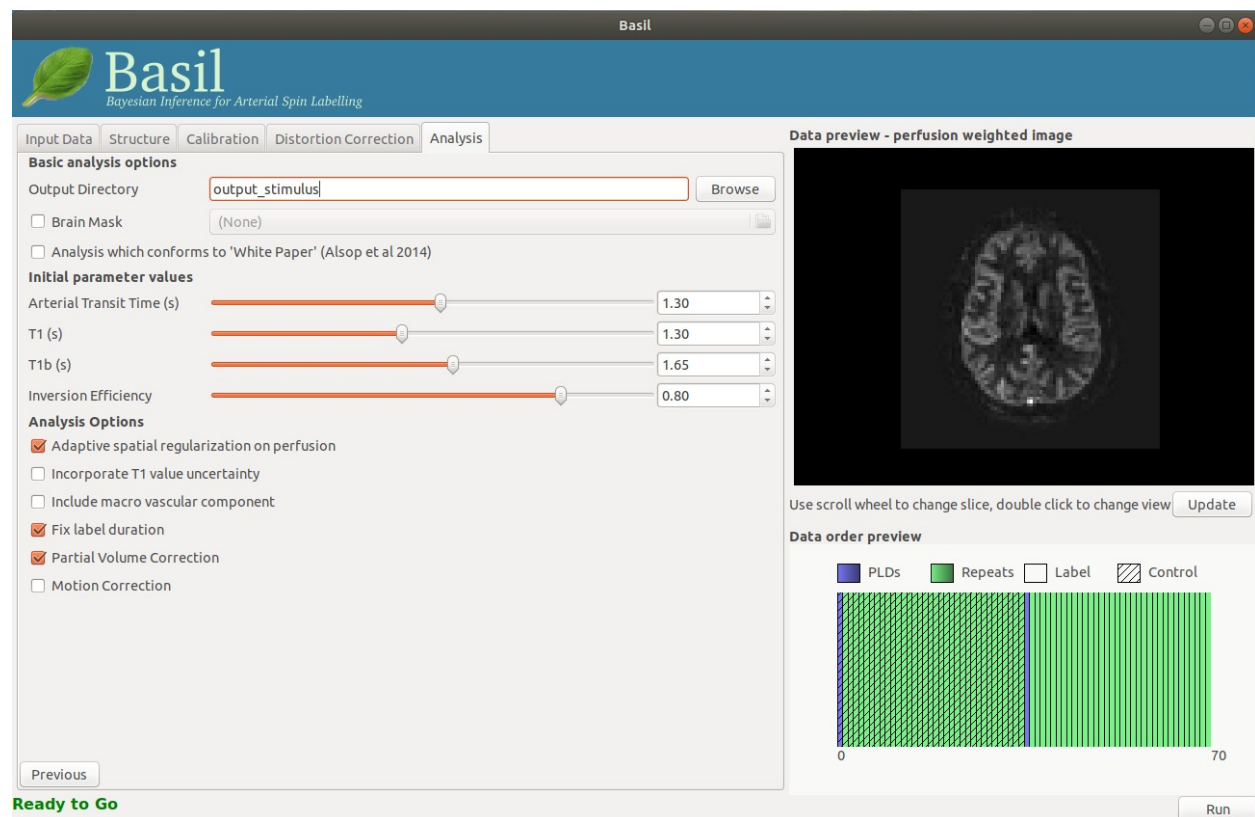


Similarly, we will use the same calibration data to calibrate the estimated perfusion into absolute units.



Before running the analysis, it is important to note that the inversion efficiency of PCASL may vary after the ad-

ministration of acetazolamide. As noted in the ASL white paper, the inversion efficiency of PCASL is affected by the flow velocity of the arterial blood. In this example, the administration of acetazolamide increases the flow velocity, thus changing the inversion efficiency of PCASL. A separate analysis estimating the inversion efficiency post-acetazolamide is needed before quantifying CBF. This can be done by including a phase contrast MRI scan that gives the flow velocity information, which can be used for the estimation of inversion efficiency. The detailed description of this technique can be found in the reference paper of this tutorial. Nevertheless, it is still possible to assume the inversion efficiency to be unchanged if the flow velocity information is unavailable. In this tutorial, we will use a newly estimated (corrected) inversion efficiency value (0.80) to analyse the ASL data after the administration of acetazolamide:



Data Analysis: Quantifying CVR

After we have quantified the absolute perfusion of baseline and stimulus, we are going to apply the following formula to estimate CVR.

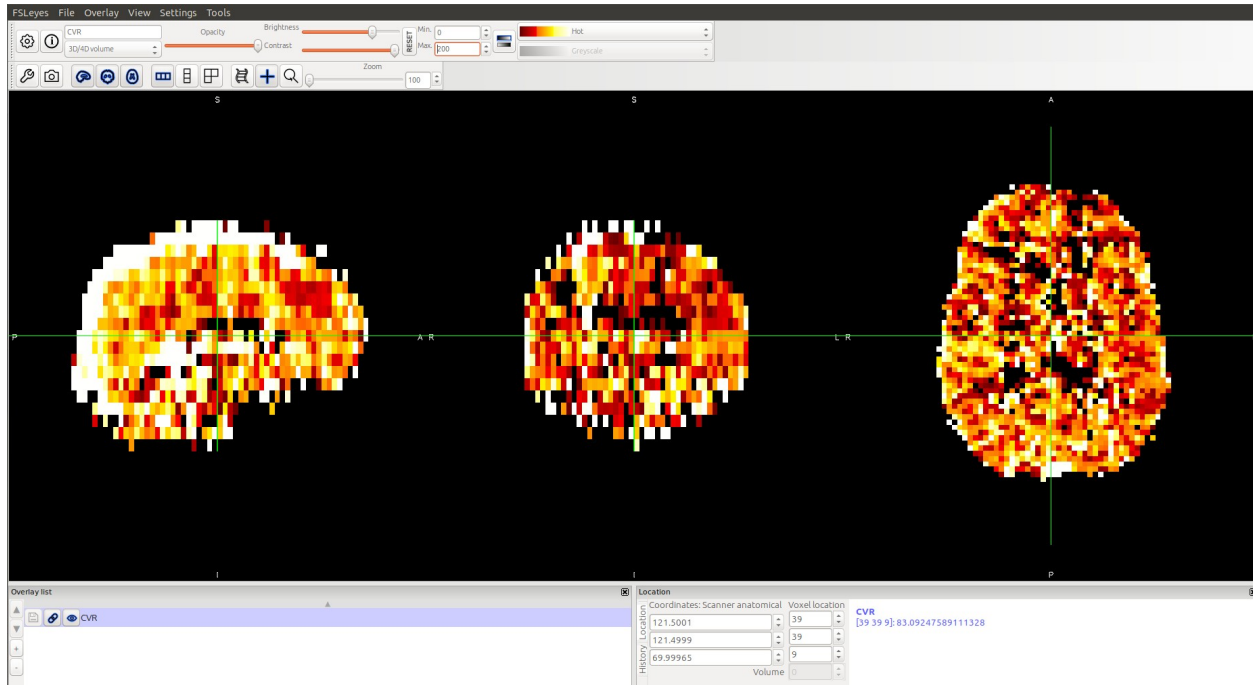
$$CVR = \frac{CBF_{stimulus} - CBF_{baseline}}{CBF_{baseline}} \times 100\%$$

This can be done using the `fslmaths` command:

```
fslmaths output_stimulus/native_space/perfusion_calib -sub output_baseline/native_
↪space/perfusion_calib -div output_baseline/native_space/perfusion_calib -mul 100 CVR
```

Results

Finally, we may use `FSLeyes` to view the CVR results:



3.3 Command line Tutorial

3.3.1 Introduction

In this section you will find a number of tutorials on the analysis of ASL data using the BASIL command line tools. If you are trying to use BASIL on your own data then a good place to begin is to find an example here that is as close to the data you have as a guide on how to proceed.

3.3.2 Single-delay ASL

Dataset 1: pcASL

This tutorial runs through the use of the BASIL command line tools on single-delay ASL data, the example data uses pcASL labelling and can be found here (~35 MB):

- [data_singledelay.zip](#)

Please note that the data is only provided for training purposes and should not be used for research without permission.

Exercise 1.1: Tag-control subtraction

The first (and simplest) thing to do with the data is tag-control subtraction to get a perfusion weighted image. This can be achieved with `asl_file`:

```
asl_file --data=data --ntis=1 --iaf=tc --diff --out=diffdata --mean=diffdata_mean
```

Note that we have told `asl_file` that the data contains only a single delay `--ntis=1`, that we have tag and control pairs `--iaf=tc` (standing for input ASL format). We have asked `asl_file` to perform tag control differencing `--diff` and then output to a file called `diffdata`, we have also asked for a second output in mean the mean has been taken across all the repeats in the data `--mean=diffdata_mean`. If you examine the `diffdata_mean` image you should see

something that looks like an image of perfusion. In fact if we were happy with relative perfusion values we could stop here.

Exercise 1.2: Kinetic model inversion

If we want to quantify perfusion in absolute units (ml/100g/min) then the next step is to invert the kinetic model - the model describes how the rate of delivery of labeled blood (and thus perfusion) is related to the measured signal. This can be achieved using `oxford_asl` (that makes a call to the `basil` command line tool to do the actual model inversion). To do this we now need to know some information about the data, namely:

- Labeling was performed using pcASL (thus we need the `--casl` option).
- Labelling was performed for 1.4 seconds (`--bolus 1.4`).
- The post-labeling delay was 1.5 seconds.

The post-labelling delay corresponds to an ‘inversion time’ of 2.9 seconds, i.e. $1.4 + 1.5$. Thus we set `--tis 2.9` - this is the list of TIs, where we only have one in this case). Here we have data with only a single delay (and BASIL includes various features for multi-delay data) - `oxford_asl` sets a number of other options that are appropriate for single delay data `--artoff --fixbolus`. Note that before FSL 5.0.6 you would need to add the `--singlet` option:

```
oxford_asl -i diffdata -o ex1_2 --tis 2.9 --bolus 1.4 --casl
```

In the `ex1_2/native_space` directory you will find the results: a perfusion image that will look very similar to the one derived in the first exercise, but the magnitude scale will be different. However, this is still only in relative units, we have yet to complete the final step needed to get perfusion in absolute units.

You may also notice an arrival image, this gives the arterial (bolus) arrival time (in seconds) in every voxel. Since this cannot be estimated from single delay data the image will be uniform and default to 0.7 seconds (or whatever value you choose with the `--bat` option).

Exercise 1.3: Calibration

To finally get perfusion in absolute units we also need to calculate the equilibrium magnetization of arterial blood and include that in the calculations. Now we need to know some more information about our data.

- Background suppression was ON.

This should have improved our perfusion contrast by removing as much static tissue signal as possible. However, we with this cannot use the control images to estimate equilibrium magnetization values. Instead there is a separate dataset, called `calib`, which is a series of control images with BGS OFF. We also have `calib_body` data in which control images were acquired with the same parameters but using the body coil. We will assume that the latter has a relatively flat sensitivity and use this to correct for sensitivity variation in the coil we used for the main acquisition. At this stage we also supply a structural image - this will be used to segment out the ventricles which will form the basis of our magnetization estimation, plus the perfusion image will also be transformed into the same resolution as the structural image via registration:

```
bet data_singlet/struct struct_brain
oxford_asl -i diffdata -o ex1_3 --tis 2.9 --bolus 1.4 --casl -c calib --cref calib_
↪body -s struct_brain
```

In `ex1_3/native_space` you should find perfusion which is the same image as in Ex 1.2, but now it will have been joined by `perfusion_calib`, which is perfusion in ml/100g/min. You should also find `ex1_3/structural_space` that has the same results but in a resolution that matches the structural image, the transformation for this process can be found in `ex1_3/native_space/asl2struct.mat`.

The results of the calibration process will be saved in `ex1_3/calib`, here you should find `M0.txt` which is the estimated M_0 value needed to get the perfusion image into absolute units as well as `refmask` that you should inspect to check that it looks like the CSF in the ventricles has been selected. For this data it should have worked fine, but if

you find data for which this has failed you might need to manually mask a region of CSF in the ventricles and supply that to `oxford_asl` using the `--csf` option (see the next exercise).

Note that `oxford_asl` always uses CSF as a reference for calibration. Other regions and alternative methods are possible and can be accessed through `asl_calib`, which can be used to calculate an `M0` value that can then be applied to the perfusion image.

Exercise 1.4: Improving the registration and other options

In the previous example registration was carried out between the raw ASL data and structural image, it should have worked but we would probably like to do better. However, this is a tricky process (due to the low resolution of the ASL data) and is particularly prone to failure if you only have data in which tag-control subtraction has already been done. In the previous example `oxford_asl` used the calibration image as the basis for registration (if this had not been available it would have used the perfusion image), it is possible to supply another image as the basis for registration: here we will use the mean of the raw data:

```
fslmaths data -Tmean data mean
bet datamean datamean_brain
fslmaths data -Tmean datamean
```

Like before we will carry out calibration, but to save repeating the identification of the ventricles we will re-use the mask from before.

Finally, this time we will apply a spatial smoothing prior to the perfusion image. This exploits spatial homogeneity in the perfusion image to improve the estimation, but in an adaptive manner. This is similar to spatially smoothing the data before analysis, but it applies it to the estimated perfusion image and not the data, whilst also estimating the correct degree of spatial smoothing from the data:

```
oxford_asl -i diffdata -o ex1_4 --tis 2.9 --bolus 1.4 --casl
            -c calib --cref calib_body -s struct_brain
            --csf ex1_3/calib/refmask --regfrom datamean_brain
            --spatial
```

The structure of the results will be the same as in the previous exercise. Compare the two and see what difference the choice of registration basis has made and the use of the `--spatial` option, this should be most clear on the `native_space` data.

3.3.3 Multi-delay ASL

This tutorial runs through the use of `oxford_asl` (the command line tool) on a number of different multi-delay ASL datasets. Examples of pASL, pcASL and QUASAR data are included. The data can be found here (~50 MB):

- [basildata.zip](#)

Please note that the data is only provided for training purposes and should not be used for research without permission.

DATASET 1: pulsed ASL

This dataset is resting-state ASL data collected using a single-shot three-dimensional GRASE readout, TR/TE 3110/23 ms, 3.44x3.44x5mm, 22 slices using a matrix size 64x64, FAIR preparation, background suppression. Alternating control and tag pairs were acquired with 10 TIs (400, 620, 840, 1060, 1280, 1500, 1720, 1940, 2160, 2380 ms), each one repeated 10 times. The data is in the directory `data_pasl`.

In this case the tag-control subtraction has already been done and the multiple measurements at each TI have been averaged, this has been put in the file `diffdata.nii.gz`. (In a later exercise we will see how that was done). Have a look at the differenced data using `FSLview`:


```
fslview pasl_data/diffdata &
```

You might like to flick through the different volumes and see if you can spot the label washing in and then decaying away again. Notice that the label arrives in some regions later than others.

Exercise 1.1: CBF estimation

Firstly we are going to do the model-fitting. Try typing this in your terminal:

```
oxford_asl
```

If you call `oxford_asl` without any options you get the usage information. There is a lot of functionality (and we do not want to use all of it now), the three main things it can do are: model-fitting, registration and calibration. We need to do model-fitting to get CBF from the multi-TI data, so we will need:

- The tag-control differenced ASL data - we have this already.
- The TIs that were used to acquire the data - 0.4,0.62,0.84,1.06,1.28,1.5,1.72,1.94,2.16,2.38.
- The duration of the ASL bolus - the acquisition was FAIR, so the bolus duration is determined by the labelling coil (body). In fact the bolus duration for this data is around 1.1 s, so we will use that value, but allow the model fitting to refine that estimate (this is automatically done by `oxford_asl` unless we tell it otherwise).
- A starting guess for the bolus arrival time - we will take the default of 0.7 s, we don't need to be very precise as the model-fitting should work this out for us.
- Values of T1 and T1b for the field strength we used - data was acquired at 3T so use T1 of 1.3 s and T1b of 1.6 s.

We have all the information we need so all we have to do is run this command (check you understand what each bit does):

```
oxford_asl -i data_pasl/diffdata --tis 0.4,0.62,0.84,1.06,1.28,1.5,1.72,1.94,2.16,2.38
-o ex1_1 --bolus 1.1 --bat 0.7 --t1 1.3 --t1b 1.66 --artoff --spatial
```

Notice that we have turned off the estimation of the macrovascular component `--artoff`, we will come back to this. We are using the 'spatial' mode, which is recommended as it exploits the natural spatial smoothness of the estimated CBF image.

In the results directory, `ex1_1`, you will find a `native_space` directory that contains all the estimated images at the same resolution as the original data. You should find in there (and look at using `FSLview`):

- `perfusion.nii.gz` The estimated CBF image in the same (arbitrary) units as the original data.
- `arrival.nii.gz` The estimated bolus arrival time image (in seconds).

Since we would like the estimated CBF in physiological units (ml/100g/min) we also need:

- Calibration data - we have `aslcalib.nii.gz` which was acquired using the same readout but no inversion and no background suppression.
- A reference 'tissue' - in this case we are going to use CSF as our reference (yes it isn't actually a 'tissue').

`oxford_asl` will, if given a structural image, try to automatically segment out the ventricles and use these as a CSF reference for calibration. We want a slightly quicker result, so there is a previously defined CSF mask, `csfmask.nii.gz`, to use. In this case there was a difference in the gain of a factor of 10 used when acquiring the calibration data (no background suppression) and the main ASL data (with background suppression).

Run the command again but with the extra calibration information supplied:

```
oxford_asl -i data_pasl/diffdata --tis 0.4,0.62,0.84,1.06,1.28,1.5,1.72,1.94,2.16,2.38
-o ex1_1 --bolus 1.1 --bat 0.7 --t1 1.3 --t1b 1.66 --artoff --spatial
-c pasl_data/aslcalib --csf pasl_data/csfmask --cgain 10
```

You should now find in the results directory an extra image: `perfusion_calib.nii.gz`, which is the estimated CBF image in ml/100g/min having used the separate calibration information. You should also find a `calib` subdirectory that includes the results of the calibration process, the main one being `M0.txt` that contains the estimated equilibrium magnetization of arterial blood (in scanner units). This `M0` value was used to scale the perfusion image to get it into physiological units.

Exercise 1.2: CBF estimation with a macro vascular component

In the previous exercise we only fit a tissue based kinetic curve to the data. However, the data was not acquired with flow suppression so there should be a substantial contribution from ASL label still within larger vessels. What we should do, therefore, is to add a macro vascular component to account for this:

```
oxford_asl -i data_pasl/diffdata -c aslcalib --csf csfmask
           --tis 0.4,0.62,0.84,1.06,1.28,1.5,1.72,1.94,2.16,2.38
           -o ex1_2 --bolus 1.1 --bat 0.7 --tl 1.3 --tlb 1.66 --spatial
```

Notice that we have run exactly the same command as the previous exercise, we have just removed `--artoff`. By default `oxford_asl` always fits the macro vascular component, even with flow suppression some arterial label can still be present.

In the results directory, `ex1_2`, you will find the perfusion and arrival results again, along with an image called `aCBV.nii.gz`, this is the estimated arterial cerebral blood volume image from the macro vascular component. Compare the images from this exercise with the previous one. Notice that the CBF is lower and arrival time is later where the magnitude of the `aCBV` image is large - around regions where large vessels would be expected.

DATASET 2: pseudo continuous ASL

This dataset is resting-state pcASL data collected using an EPI readout, TR/TE 3750/14 ms, 3.75x3.75x7.5mm, 24 slices using a matrix size 64x64. Alternating control and tag pairs were acquired after 1.4 s of labelling at 5 different post labelling delays (200, 400, 600, 800, 1000 ms), each one repeated 12 times. The data is in the directory `pcasl_data`

We are going to need to know what the inversion times were for each measurement. For pASL this was the time between labelling and readout. For cASL we need the time from the start of labelling to readout, so our TI = labelling duration + post labelling delay. Thus the TIs are: 1.6, 1.8, 2.0, 2.2, 2.4 s.

Exercise 2.1: Tag-control subtraction

The first thing we need to do is take the raw ASL data and do tag-control subtraction to remove the static tissue contribution. We are also going to take the average of the multiple measurements at each TI to make the model-fitting faster (in practice would could skip this as `oxford_asl` could do this for us). We could split the data into separate volumes and do subtraction and averaging of these images before re-assembling it all together, but that would be tedious! Instead we have a command that knows how to deal with ASL data, what we want to do is:

```
asl_file --data=data_pcasl/asl_raw_data --ntis=5 --ibf=rpt --iaf=tc --diff --
↪mean=pcasl_diffdata
```

The command tells `asl_file`:

- Where to find the data.
- How many TIs there are in the file.
- That the data contains repeated measurements (where we have cycled through all the TIs each time).
- That the data is in tag-control pairs.
- That we want to do pairwise subtraction.

- That it should take the mean of each TI and save that as the output file in the current directory: `pcasl_diffdata`.

Have a look at the data `pcasl_diffdata` in FSLview as we did for the pASL data. This set will look a bit different as we only have 5 TIs and these are all placed so that they will be near the peak of the kinetic curve. So we don't see the nice clear wash in of the label as we did before.

Exercise 2.2: CBF estimation

We will do CBF estimation in a very similar way to the pASL data. However, this time we will:

- Use a cASL model with the `--casl` option.
- Set the bolus duration to 1.4 s - the length of labeling. Since the cASL label is well defined we won't try to estimate its duration, so we add the `--fixbolus` option.
- Supply a structural image, which means that `oxford_asl` will try to register the ASL data to the structural image and give the CBF results in the same space as the structural. By default `oxford_asl` will try to register the estimated CBF image to the structural, this can be problematic as there may not be excellent contrast for this. The raw ASL data is a much better basis for registration so we will instruct `oxford_asl` to use this with the `--regfrom` command.
- Not supply a CSF mask. We will let `oxford_asl` automatically identify the CSF using the structural image.

Firstly we do a little pre-processing of the supporting images - mainly brain extraction:

```
bet data_pcasl/struc struc_brain
fslmaths data_pcasl/asl_raw_data -Tmean asl_raw
bet asl_raw asl_raw_brain
```

The full command we need is (again see if you can identify what each term does):

```
oxford_asl -i pcasl_diffdata -c data_pcasl/calibration_head
--tis 1.6,1.8,2.0,2.2,2.4 -o ex2_2
--bolus 1.4 --bat 0.7 --tl 1.3 --tlb 1.66
--artoff --fixbolus --spatial --casl -s struc_brain
--regfrom asl_raw_brain
```

In the results directory, `ex2_2`, you will find a `native_space` set of results, but also the same results at the resolution of the structural image `struct_space`. As with the pASL results there are perfusion and bolus arrival time images. Since we only have 5 tightly spaced TIs we won't expect our arrival time images to be as good. You will also notice from the arrival time image that the mask generated by `oxford_asl` wasn't perfect - it includes all the brain, but some non brain too. We could have made our own mask and supplied it to `oxford_asl` with the `-m` option if we had wanted to. It is also worth looking at `ex2_2/calib/refmask.nii.gz` as this is the mask that was used to identify the CSF in the calibration image, you should check that it looks like voxels within the ventricles have been identified.

When we analysed the pASL data we also added a macro vascular component into the model. However, we won't do that here since all the TIs we have come quite late and we are likely to have missed most of the early arriving arterial based label.

DATASET 3: QUASAR

The QUASAR variant of ASL makes use of a combination of flow suppressed and non suppressed multi-TI data to allow for a better separation of the tissue and macro vascular signals. This aids model-based analysis and also permits 'model-free' analysis similar to that used in DSC-MRI. QUASAR data also has all the information within it to do the calibration step. Because the QUASAR sequence is well defined we don't have to worry about all the options in `oxford_asl`, in fact there is a special version specifically designed for QUASAR data called `quasil`. Again just trying the command brings up the usage - there are not many options this time!

Exercise 3.1: Model-based analysis

Firstly we are going to do a model-based analysis, just like we did in exercise 2, but tailored for QUASAR data. The command we want is:

```
quasil -i data_quasar/data -o ex3_1
```

In the results directory, `ex3_1`, you should find perfusion and aCBV images to examine.

Exercise 3.2: Model-free analysis

Now we are going to compare the model-based results with numerical deconvolution (this is the method proposed in Petersen's original paper). `quasil` will also do this using the `--mfree` option:

```
quasil -i data_quasar/data -o ex3_2 --mfree
```

Like the model-based analysis both perfusion and aCBV images are produced. Compare the model-based and model-free results, you should find that the model-free perfusion values are generally lower than the model-based results, primarily due to the underestimation of the numerical deconvolution.

DATASET 4: Turbo-QUASAR

Turbo-QUASAR achieves full brain coverage and improves the SNR of QUASAR by using multiple labelling pulses to create a longer effective bolus duration while retaining the other characteristics of QUASAR. Due to the frequent labelling pulses, MT effects can be an issue affecting both calibration and CBF quantification. The analysis pipeline `toast` includes options to either correct the MT effects or use a separately acquired calibration data, in addition to quantifying the main hemodynamic parameters such as perfusion, arterial transit time, and arterial blood volume.

This tutorial runs through the use of TOAST (the command line tool) on Turbo-QUASAR dataset. The data can be found here (~25 MB):

Exercise 4.1: Calibration by correcting for MT effects

The command to quantify the hemodynamic parameters by correcting for MT effects in calibration:

```
toast -i Turbo_QUASAR_data -o ex4_1 --infert1 --corrcal
```

The option `--infert1` indicates that MT effects are corrected. The optional step `--corrcal` indicates that the partial volume effects on the edge of the brain are corrected.

Exercise 4.2: Calibration by using a separately acquired

Calibration can also be performed using a user-provided M0 image from a separate scan in the same session. The TR of the calibration image needs to be specified. A structural image needs to be provided in order to register the calibration image to the ASL image. The command is:

```
toast -i Turbo_QUASAR_data -o ex4_2 --calib M0 --tr 4.4 --struct structural --corrcal
```

Exercise 4.3 Quantify arterial blood volume

Turbo-QUASAR can also quantify arterial blood volume (ABV or aCBV) from the data using the `--inferart` option. We could use either of the calibration methods. The command is:

```
toast -i Turbo_QUASAR_data -o ex4_3_1 --infert1 --corrcal --inferart
```

or:

```
toast -i Turbo_QUASAR_data -o ex4_3_2 --calib M0 --tr 4.4 --struct structural --  
→corrcal --inferart
```

3.3.4 Acknowledgments

Thanks are due to Tom Okell, Brad MacIntosh, Dan Gallichan, Michael Kelly, Esben Petersen, Xavier Golay, Lena Václavů, and Aart Nederveen for the provision of the ASL data used in these exercises.

A set of examples of the use of BASIL (for pcASL), which is similar to those above, is also available as part of the primer:

Introduction to Perfusion Quantification using Arterial Spin Labelling, Oxford Neuroimaging Primers, Chappell, MacIntosh & Okell, Oxford University Press, 2017.

The examples themselves are freely available online at the primer website: [neuroimagingprimers.org](http://www.neuroimagingprimers.org), you can access the ASL examples directly [here](#).

<http://www.neuroimagingprimers.org/examples/introduction-to-perfusion-quantification-using-asl/>

3.4 Preparatory materials

If you are new to Neuroimaging or to FSL tools you might find the following resources helpful before looking at the material on ASL.

If you are not particularly familiar with MRI you might like to read a [Short Introduction to MRI Physics for Neuroimaging](#) available via the [neuroimagingprimers.org](http://www.neuroimagingprimers.org) website.

If you are new to FSL we recommend these three short introductory FSL practicals on:

- [FSLeyes](#)
- [BET](#)
- [FSL Utils](#)

CHAPTER 4

GUI User Guide

The graphical user interface to the BASIL tools can be accessed by typing `asl_gui` at the command line. It should provide most of the options required for analysis of ASL data including the majority of the more advanced features of BASIL.

Note: if you are using a release of BASIL that you have installed separately from FSL you may need to specifically call `asl_gui` where it is installed: e.g. `/Users/{blah}/Downloads/oxford_asl/asl_gui`. This page documents the GUI available in FSL v6 (also available via the pre-release website), there are some differences with the GUI found in FSL v5.

The GUI largely provides a more approachable interface to the `oxford_asl` command line tool (and even produces a command line call for `oxford_asl` for you to reuse separately if you like).

The GUI has five tabs, whose function are fairly self-explanatory. Each tab represents one step in the (notional) workflow for your ASL analysis and you can work through them using the prev/next buttons in the lower right of each tab. For many analyses you will be able to accept many of the default options and you might find you do not need to even visit some of the later tabs.

- **Input Data:** Specify details of the ASL data here, this is also where you have to record details about the acquisition used.
- **Calibration:** Specify your calibration image (if you have one), to be used to produce images of absolute perfusion (in ml/100g/min). You can also choose to set options relating to the method for estimation of the equilibrium magnetization of arterial blood.
- **Structural:** Specify a structural image, or the output of a previous run of `fsl_anat`, to be used in the analysis process and for registration. Analysis can proceed without a structural image, but if you have one it is recommended that you include it.
- **Distortion correction:** If you have suitable images to do distortion correction of your ASL data you can include them here.
- **Analysis:** Options relating to the analysis, primarily the kinetic model and the estimation process.

More details on each tab are provided below. Note that some of the options available on each tab depend on what information is present in your data (set on the data tab). Thus your view of the tab might differ from the one shown here slightly.

4.1 Input Data

Basil
Bayesian Inference for Arterial Spin Labelling

Input Data | Structure | Calibration | Distortion Correction | Analysis

Data contents

Input Image: (None)

Number of PLDs: 1

Repeats: Fixed

Data order

Volumes grouped by: Repeats

Label/Control pairing: Label then control

Acquisition parameters

Labelling: cASL/pcASL

Bolus duration (s): Constant 1.80

Bolus durations (s): 1.8

PLDs: 1.8

Repeats: 1

Readout: 3D (eg GRASE) Time per slice (ms): 10.00

☐ Multi-band 5 slices per band

Update

Data order preview

Repeat 1	
PLD1	
Label	Control
1	2

Input data volumes

Run **Input data must be specified**

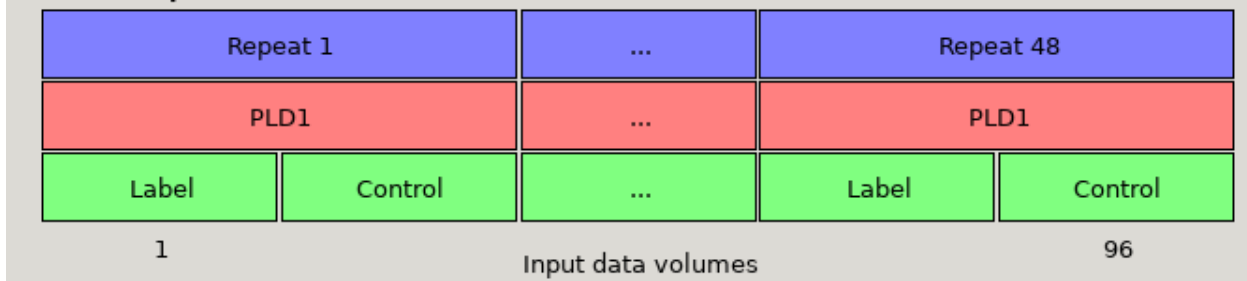
Data contents

- *Input Image*: A single 4D nifti file of the main ASL data (either label-control pairs or subtracted images), with the individual measurements in the 4th dimension.
- *Numer of PLDs*: Set the number of post-label delays (or inversion times/inflow times) in the data.
- *Number of repeats*: A value will be calculated here based on the number of PLDs you entered in the box above. Use this as a sanity check again the number of repeats you expect from your acquisition.

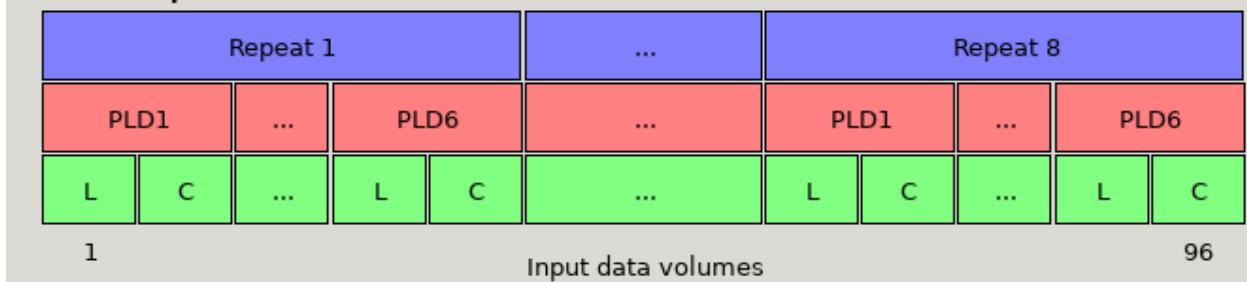
Data order

- *Volumes grouped by*: For multi-PLD data, this specifies whether the acquisition consisted of a full set of all PLDs, with the whole block repeated multiple times (Volumes grouped by *Repeats*), or if the first PLD was repeated multiple times, followed by all repeats of the second PLD, and so on (Volumes grouped by *TIs/PLDs*).
- *Label/control pairing*: This specifies whether the label image was before the control or vice versa, or if the data is already subtracted.

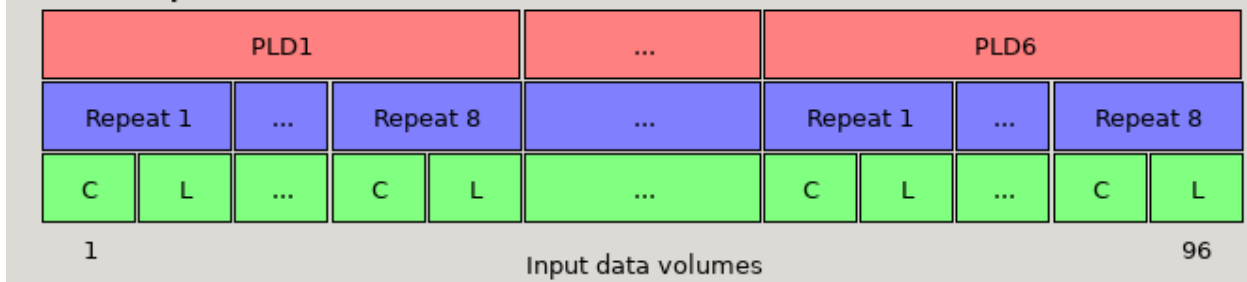
The choices selected here are reflected in the **data order preview** view, at the bottom right of the window. This gives a visual representation of the volumes in the input data. The horizontal axis represents the sequence of volumes in your data, and the boxes above shows the PLD, repeat and label/control corresponding to each volume.

Data order preview

Data preview for single-PLD data with `Label` then `control` ordering and 48 repeats. Note that the ‘Volumes grouped by’ option makes no practical difference for single PLD data.

Data order preview

Data preview for multi-PLD data with 6 PLDs, with `Label` then `control` ordering and volumes grouped by `Repeats`. The first set of volumes contains a single repeat of all the PLDs, followed by another block of all PLDs, etc.

Data order preview

*Data preview for multi-PLD data with 6 PLDs, with `Control` then `label` ordering and volumes grouped by `PLDs`. Note that the order of the label and control images (the green `L` and `C` boxes) has changed, and the first set of volumes is 8 repeats of PLD1, followed by 8 repeats of PLD 2, etc.

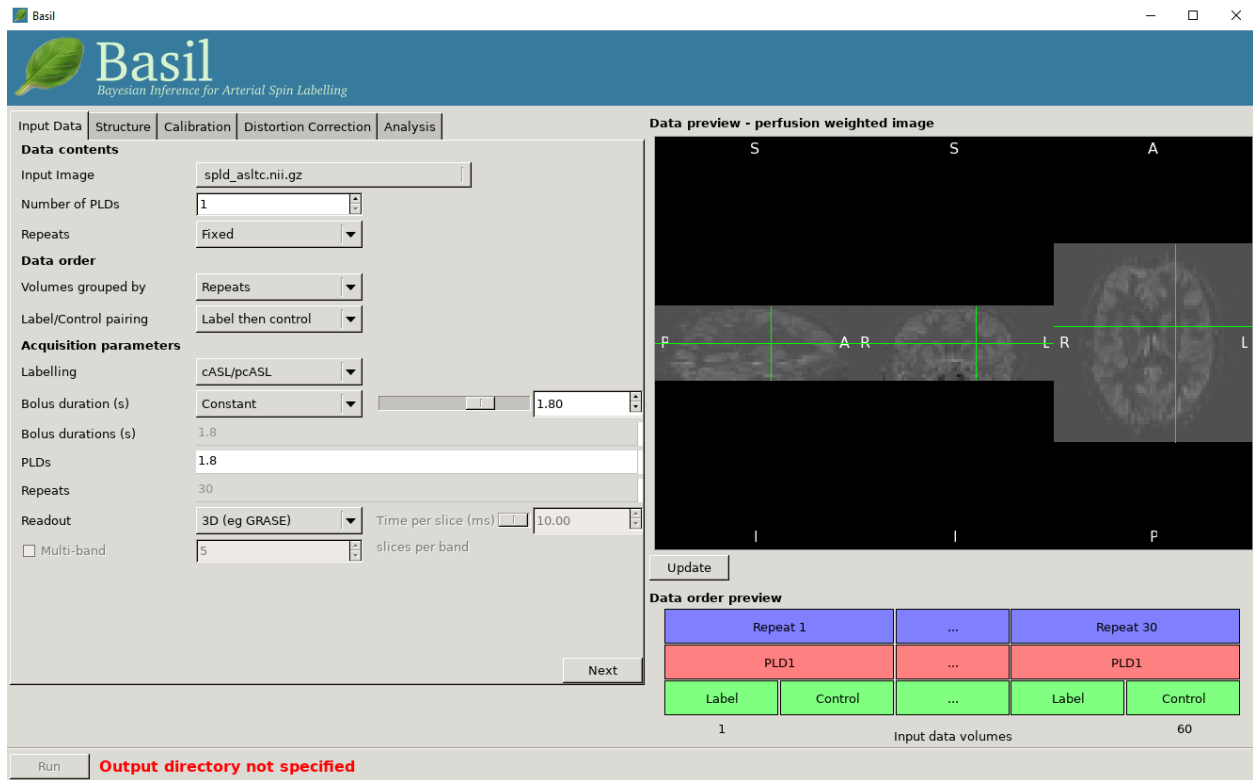
Acquisition parameters

- *Labeling*: Choose the labelling scheme employed, either `pcASL/cASL` or `pASL` (note for analysis purposes there is no difference between `cASL` and `pcASL`).
- *PLD / TIs*: Set the post-labeling delay(s) (or inversion time(s)) for the acquisition. For a multi-PLD/TI acquisition there will be a value to be set for each PLD/TI in the data.
- *Bolus duration (s)*: The duration of the labeled bolus in seconds.

[Advanced] If your data is multi-PLD you can additionally specify different bolus durations for the different PLD, by changing from *single* to *multiple*. Using this option you can have any combination of PLD and bolus duration to match those used in the acquisition

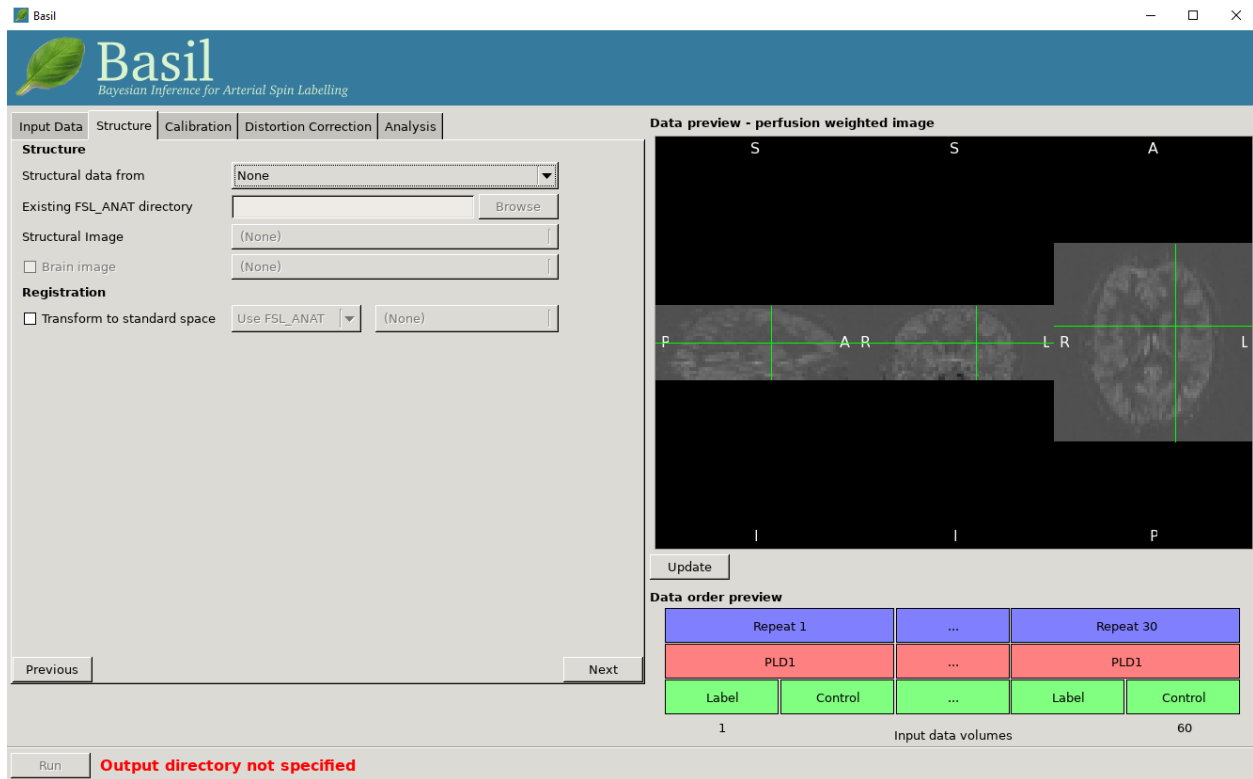
- Readout: Specify if a full 3D or a 2D multi-slice acquisition was used. For 2D you should then set the time taken to acquire each slice (in milliseconds) as this determines the true PLD/TI for each slice.
- Multi-band: Include this option where the readout was a multi-band (simultaneous multi-slice) 2D acquisition, in which case you need to specify the number of slices in each band, to correctly set the PLD/TI for each slice.

Data preview - perfusion weighted image



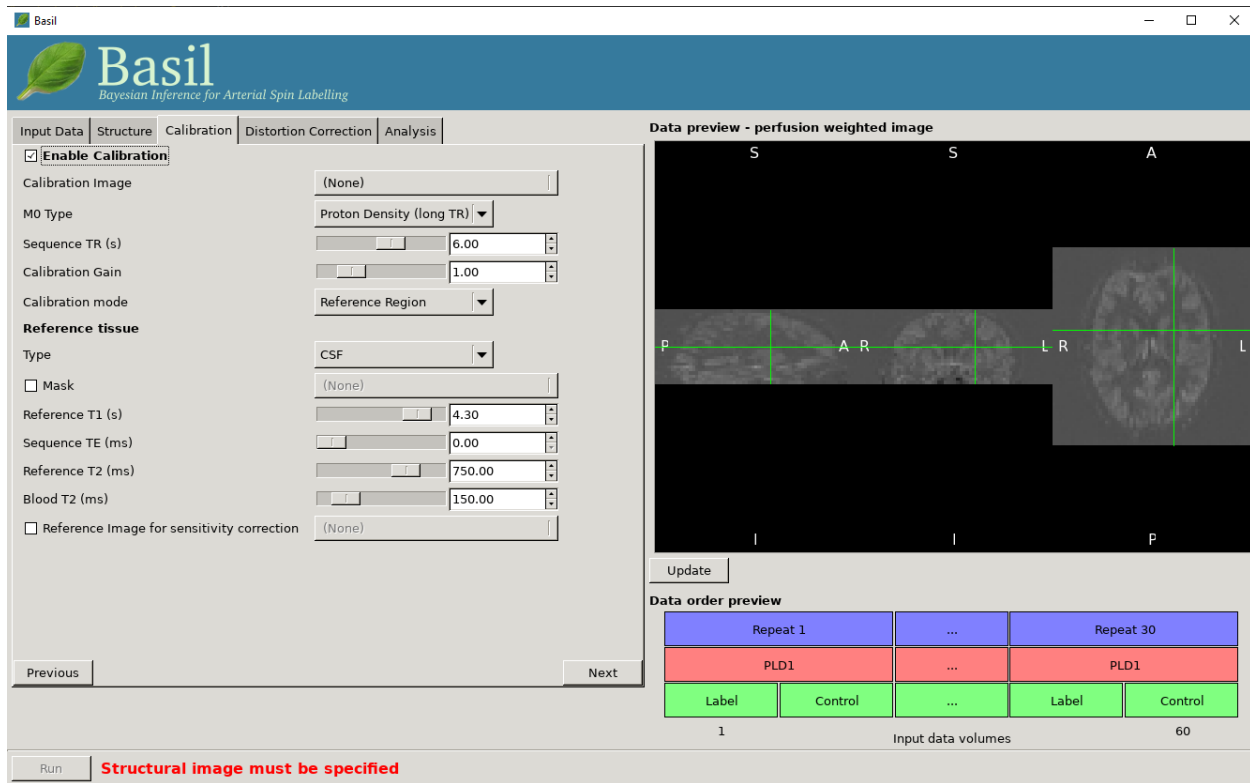
On the right of the window is a preview pane. Once you have set the *Data contents* and *Data order* parameters you can press *preview* and the GUI will generate a perfuion-weighted image based on the information you have provided. This is a good sanity check to ensure that you have set these parameters correctly - if at this stage you dont see something that looks like a perfusion-weighted image - i.e., something in which the grey matter structure of the brain is evident - then you probably haven't set the ordering correctly. You can scroll through the slices in the image using your mouses' scroll wheel (or equivalent). Note that for multi-PLD data this preview averages all the different PLD.

4.2 Structural



- *Structural Data From* - The recommended option is Existing FSL_ANAT output where you have already run FSL_ANAT on the structural image. Select the FSL_ANAT output directory below. Alternatively you can supply a structural image and run FSL_ANAT on it as part of the analysis, however this will add quite a lot of runtime so it's better to do it once and then just re-use the output. You can also avoid using FSL_ANAT at all and just supply your own structural image. If you have a brain segmentation you can supply it (otherwise BET will be run), and you can supply an existing transformation to standard (MNI) space (either a linear FLIRT matrix transformation or a nonlinear FNIRT warp).

4.3 Calibration



Enable Calibration

Note you specifically have to choose the option to enable calibration. It is possible to perform ASL analysis without calibration, but only then possible to get relative perfusion images.

- *Calibration image*: An image to be used for calibration. This should have the same resolution as the ASL data, ideally have the same readout, and be proton density weighted.
- *M0 type*: Specify what sort of calibration image you are supplying. In most cases this will be *Proton Density*, unless you have data with a pre-saturation in which case choose *saturation recovery*.
- *Sequence TR (s)*: The repetition time of the sequence used to collect the calibration image (this may not be the same as the ASL data itself). This is used to correct for non-equilibrium effects on the PD image.
- *Calibration gain*: If there is a difference in acquisition gain between the calibration image and the ASL data it can be set here. This might be the case where background suppression has been applied for the main ASL data.
- *Calibration mode*: Choose either to compute and apply the calibration factor (equilibrium magnetization of arterial blood) *voxelwise*, or from within a *Reference Region*. The latter relies upon a structural image being provided (unless you supply your own mask for this below), the former is the approach assumed by the white paper.

Reference tissue

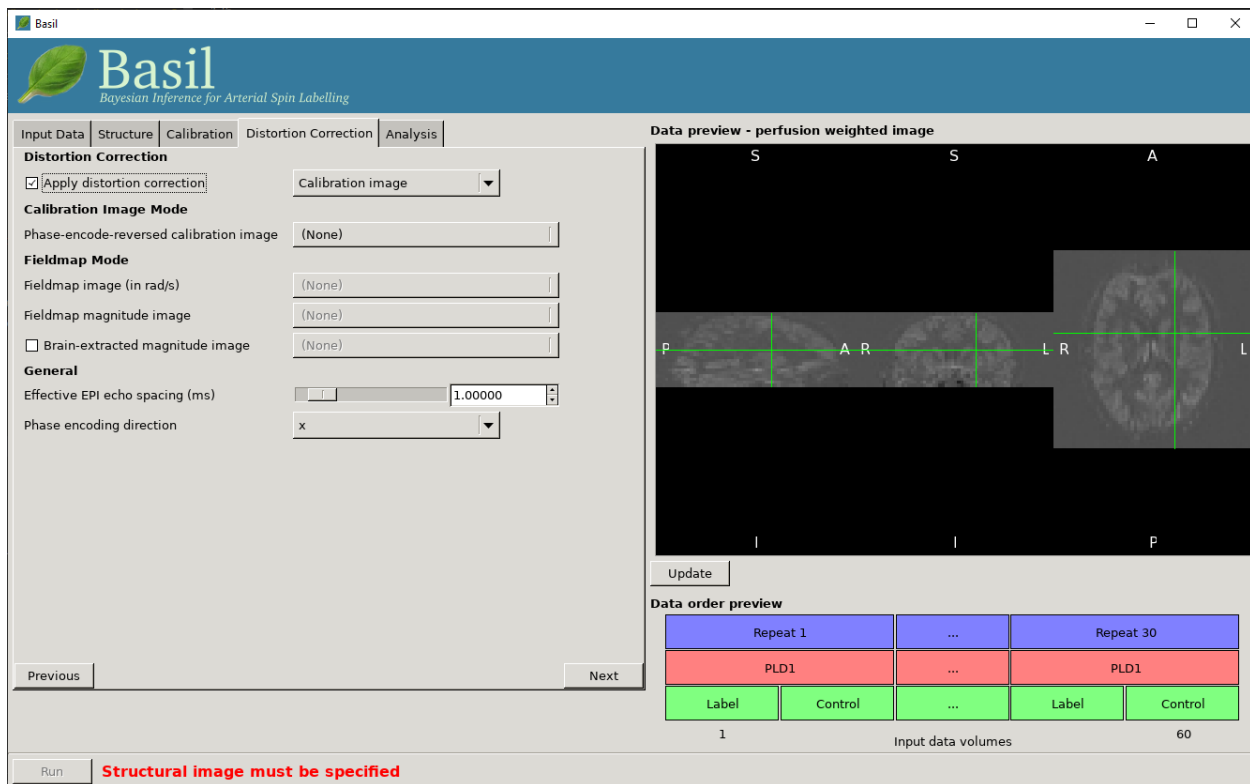
These options apply to the *Reference Region* mode of calibration.

- *Type*: The reference tissue type to use, one of: *CSF/white matter/grey matter/none*. For the first three options a mask will be generated automatically from the structural image (if you have specified one), otherwise (or alternatively) you should specify your own mask in the *Reference Tissue Mask* box (this should be in the same

space as the ASL data). Generally CSF or white matter are good choices, avoid grey matter due to partial volume effects.

- *Sequence TE*: This corrects for T2 differences between the reference tissue and the brain tissue (using a blood T2 value) based on the TE of the data (which is assumed to be the same as any calibration image).
- *Reference T1*: T1 of the reference tissue.
- *Reference T2/Blood T2*: T2 values, these are only relevant if you specify the TE of your sequence. T2 of the reference defaults to a CSF value. These should be replaced by T2* values if appropriate.

4.4 Distortion Correction



- *Apply distortion correction*: Select to apply correction for readout distortions in the ASL data using a suitable set of reference images. You will need either a *calibration image* with a different phase encoding direction to the main calibration image, or a *fieldmap*.

Calibration image mode

- *Phase encode reversed calibration image*: An image that matches the calibration image in all acquisition parameters, except that the phase-encode direction is reversed.
- *Effective EPI echo spacing*: Set this value from the sequence in seconds (typical values are of the order of 0.01 ms).
- *Phase encode direction*: The phase encode direction of the calibration image (i.e. the image entered on the *Calibration* tab) and must match that of the main ASL data.

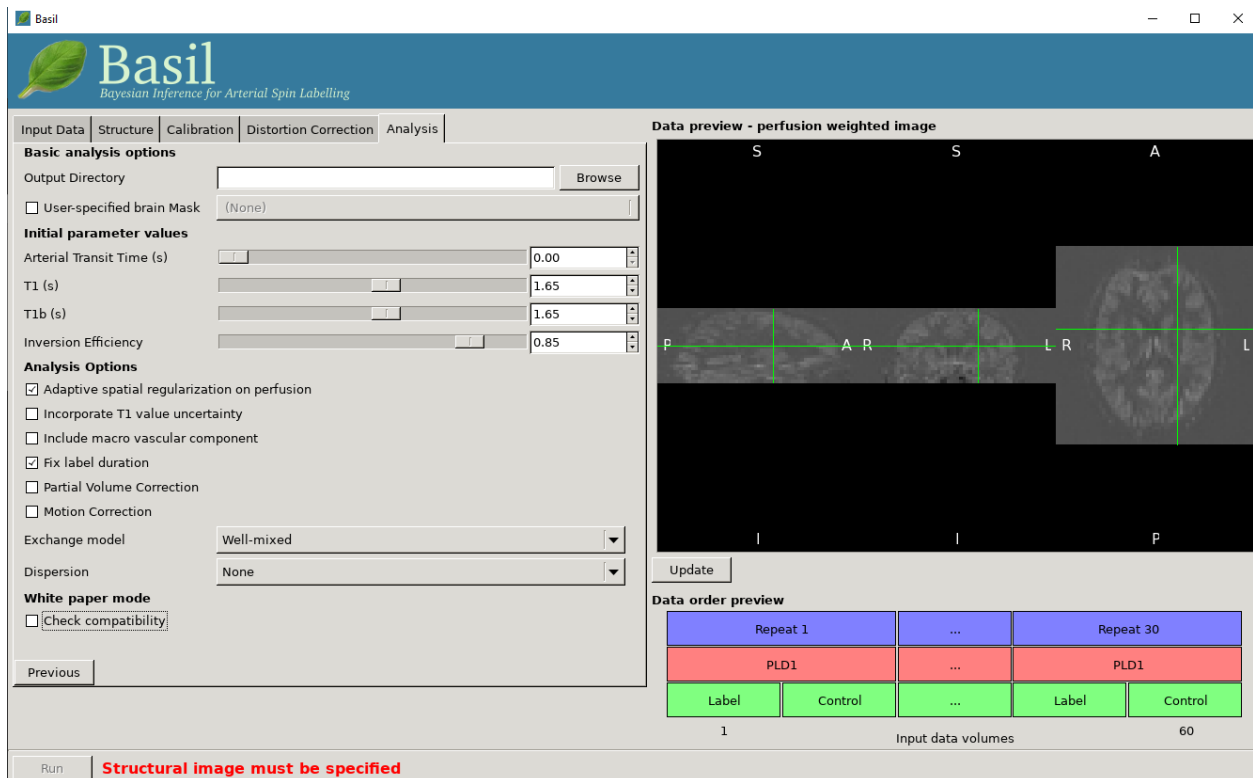
Note that in this mode `oxford_asl` uses **TOPUP** to do the distortion correction. For this, the Effective EPI echo spacing is converted to total readout time by multiplication by the number of slices (minus one) in the encode direction.

Fieldmap mode

- *Effective EPI echo spacing*: Set this value from the ASL sequence in seconds (sometimes called the dwell time), typical values are of the order of 0.01 ms.
- *Phase encode direction*: The phase encode direction of the ASL images (entered on the *Calibration* tab).
- *Fieldmap image*: A fieldmap image (need not necessarily matched to the ASL nor structural image resolution) in rad/s - be careful about the units, as this is not consistent between neuroimaging tools.
- *Fieldmap magnitude image*: A magntiude image to go with the fieldmap, this is used for registration of the fieldmap.
- *Brain extracted fieldmap magntiude image*: Brain extracted version of above image.

For more information on fieldmapping see the documentation associated with [FUGUE](#) (note the fieldmap correction in the BASIL GUI is akin to using FEAT for fieldmap correction and uses `epi_reg`, albeit in a way specifically setup for ASL data). A common choioce for phase encoding is Anterior Posterior (AP), which would normally tranlsate into 'y'. However, whether this is '+y' or '-y' isn't necessarily preditable and you might need to try both and see which one visibly makes the distortion worse (and then use the other!)

4.5 Analysis



Basic analysis options

- *Output directory*: where to put the results.
- *User-specified brain mask*: Normally, BASIL will try to create a brain mask for you using the available data, however you may specify your own mask here (in the same space as the ASL data).

Initial parameter values

This section sets parameter values for the kinetic model. In some cases these are treated as initial and/or prior (mean) values, but are also estimated in the analysis itself.

- *Arterial Transit Time*: The assumed value for the ATT. For multi-PLD ASL, ATT is estimated from the data and this value is used as prior information, for single delay data this value will be treated as fixed. The default 0.7 appears to be reasonable for pASL, but longer values ~1.3s have been found to be more suitable for pcASL data, these are the defaults used by the GUI. (in *white paper mode* this value is set to 0)
- *T1/T1b*: T1 values for tissue and blood. Defaults are 1.65 seconds for blood and 1.3 seconds for tissue (based on 3T field strength). (In *white paper mode* both T1 values are set to 1.65 seconds).
- *Inversion efficiency*: A fixed value for the inversion efficiency applied in the calculation of absolute perfusion. The default values (0.85 for pcASL and 0.98 for pASL) are taken from the white paper.

Analysis Options

- *Adaptive spatial regularisation on perfusion*: applies a spatial prior to the perfusion image during estimation, thus making use of neighbourhood information. This is a highly recommended option, and is on by default.
- *Incorporate T1 uncertainty*: Permits voxelwise variability in the T1 values, this will primarily be reflected in the variance images for the estimated parameters, don't expect accurate T1 maps from conventional ASL data.
- *Include macro vascular component*: Corrects for arterial or macrovascular contamination, and is suitable where the data multi-PLD (even where flow suppression has been applied).
- *Fix label duration*: Takes the value for the label duration from the *Input Data* tab as fixed, turn off to estimate this from the data (the value on the data tab will be used as prior information in that case). You are most likely to want to deselect the option for pASL data, particularly where QUIPSSII/Q2TIPS has not been used to fix the label duration.
- *Partial Volume Correction*: Correct for the different contributions from grey and white matter, and CSF to the perfusion image. This will produce separate grey and white matter perfusion maps.
- *Motion Correction*: Uses `mcfliirt` to perform motion correction of the ASL data (and the calibration image).
- *Exchange/Dispersion model*: These are advanced options that affect the modelling of the blood/tissue compartments and the dispersion of the labelled blood bolus during transit.

White paper mode

This option allows you to check if the analysis you have set up is compatible with the recommendations in the consensus paper (Alsop et al, 2014). This paper described a simple kinetic model for ASL analysis primarily designed for single-PLD data. While BASIL is capable of more complex modelling, particularly for multi-PLD data, it may be useful for comparison to be able to perform an analysis that uses the white paper recommendations. Selecting this option allows the user to view and correct their options for this purpose.

White paper mode

☒ Check compatibility

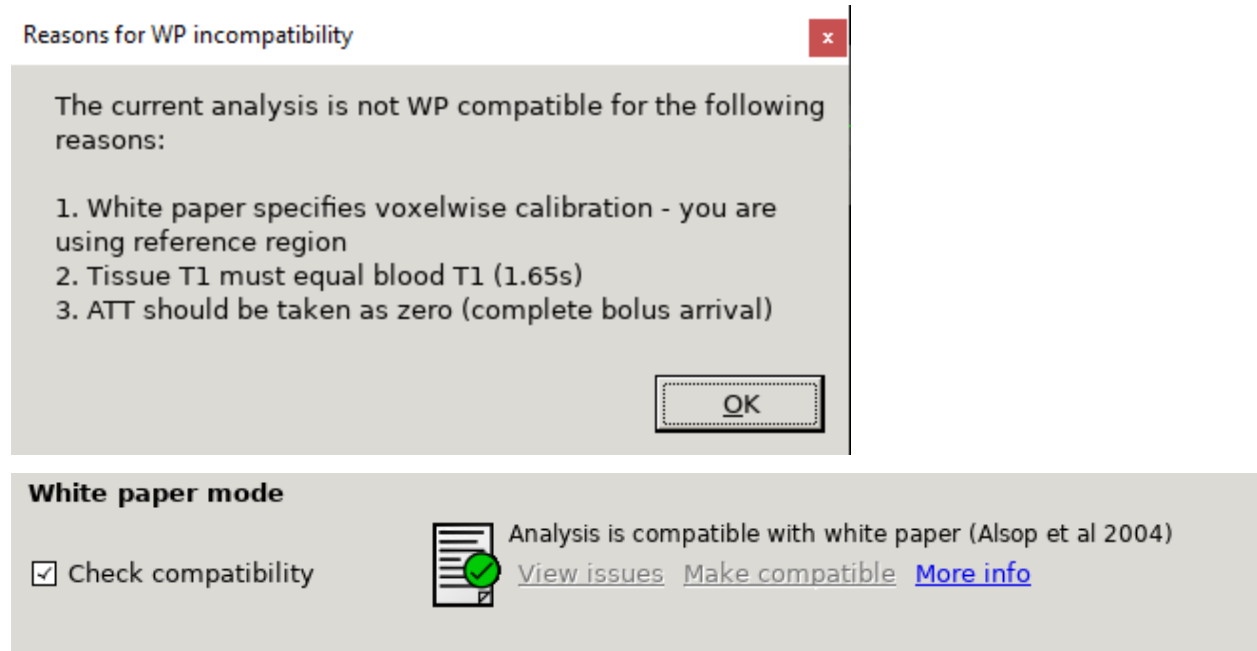


Analysis is not compatible with white paper (Alsop et al 2004)

[View issues](#)

[Make compatible](#)

[More info](#)



4.6 Output

The outputs from the GUI are a perfusion image called `perfusion.nii.gz`, which provides blood flow in relative (scanner) units, and an arrival time image called `arrival.nii.gz` for multi-PLD ASL. If a calibration image has been provided then a further image, `perfusion_calib.nii.gz`, is also produced, which is a flow map in absolute units (ml/100g/min). Results in native ASL, structural and standard space will appear in the output directory within separate subdirectories. Where applicable transformation between spaces will also be saved, along with summary measures of perfusion in the `native_space` subdirectory if the structural information is available to calculate these.

If calibration was performed then a separate subdirectory will be created and will contain text file called `M0b.txt` that records the estimated M0 value from arterial blood if the reference region mode was used, otherwise an image will be supplied instead. For reference region calibration, if a manual reference tissue mask was not supplied then the automatically generated one will also be saved in as `refmask.nii.gz`, you should inspect this to ensure that it is a reasonable mapping of the tissue you are using for the reference region (normally CSF in the ventricles).

5.1 Overview

`oxford_asl` is an automated command line utility that can process ASL data to produce a calibrated map of resting state tissue perfusion. It also includes a range of other useful analysis methods including amongst others:

- motion correction
- registration to a structural image (and thereby a template space)
- partial volume correction
- distortion correction
- ROI analysis

If you have ASL data to analyse, `oxford_asl` is most likely the tool you will want to use, unless you want a graphical user interface. In practice, the GUI in BASIL is largely a means to construct the right call to `oxford_asl`.

5.2 What you will need

As a minimum to use `oxford_asl` all you need are some ASL data (label and control pairs or already subtracted). In practice you will also most probably want:

- *a calibration image*: normally a proton-density-weighted image (or a close match) acquired with the same readout parameters as the main ASL data. Only once you have a calibration image can you get perfusion in absolute units.
- *a structural image*: it is helpful to have a structural image to pass to `oxford_asl` and if your data includes this we strongly suggest you do use it with `oxford_asl`. By preference, we strongly suggest you process your structural image with `fsl_anat` before passing those results to `oxford_asl`. This is a good way to get all of the useful information that `oxford_asl` can use, and you can scrutinise this analysis first to check you are happy with it before starting your ASL analysis.

- *multi-delay ASL*: the methods in `oxford_asl` are perfectly applicable to the widely used single delay/PLD ASL acquisition. But, they offer particular advantages if you have multi-delay/PLD data.

5.3 Things to note

To produce the most robust analysis possible `oxford_asl` includes a number of things in the overall analysis pipeline that you might want to be aware of:

- *spatial regularisation*: this feature is now enabled by default for all analyses and applies to the estimated perfusion image. We do not recommend smoothing your data prior to passing to `oxford_asl`. If you really want to, only do ‘sub-voxel’ level of smoothing.
- *masking*: `oxford_asl` will attempt to produce a brain mask in which perfusion quantification will be performed. This is normally derived from any structural images with which it is provided (highly recommended), via registration. Therefore, if the registration is poor there will be an impact on the quality of the mask. Where no structural information is provided, the mask will be derived from the ASL data via brain extraction, this can be somewhat variable depending upon your data. It is thus **always** worth examining the mask created. `oxford_asl` provides the option to input your own mask where you are not satisfied with the one automatically generated or you need a specific mask for your study.
- *registration*: `oxford_asl` performs the final registration using the perfusion image and the BBR cost function. We have found this to be reliable, as long as the perfusion image is of sufficient quality. In practice, an initial registration is done earlier in the pipeline using the raw ASL images and this is used in the mask generation step. You should **always** inspect the quality of the final registered images.
- *multiple repeats*: ASL data typically contains many repeats of the same measurement to increase the overall signal-to-noise ratio of the data. You should provide this data to `oxford_asl`, and not average over all the repeats beforehand (unlike earlier versions of the tool). `oxford_asl` now includes a pipeline where it initially analyses the data having done averaging over the repeats, followed by a subsequent analysis with all the data - to achieve both good robustness and accuracy. If your data has already had the repeats averaged, it is still perfectly reasonable to do analysis with `oxford_asl`, if you have very few measurements in the data to pass to `oxford_asl` you might want to use the special ‘noise prior’ option, since this sets information needed for spatial regularisation.
- *Advanced analyses*: Partial volume correction, or analysis of the data into separate epochs, are available as advanced supplementary analyses in `oxford_asl`. If you choose these options `oxford_asl` will *always* run a conventional analysis first, this is used to initialise the subsequent analyses. This also means that you can get both conventional and advanced results in a single run of `oxford_asl`.
- *Multi-stage analysis*: By default `oxford_asl` will analyse the data in multiple-stages where appropriate in an attempt to get as accurate and robust a result as possible. The main example of this is a preliminary analysis with the data having been averaged over multiple-repeats (see above). But, this also applies to the registration (see above). This does mean that you might find some differences in the results than if you did an analysis of the data yourself using a combination of other command line tools.

5.4 Typical Usage

Typing `oxford_asl` with no options will give the basic usage information, further options are revealed by typing `oxford_asl --more`.

A typical processing run would usually look something like this:

```
oxford_asl -i [asl_data] -o [output_dir] <data parameters> <analysis options> -c [M0_calib] <calibration parameters> -fslanat [fsl_anat_output_dir]
```

This command would analyse the ASL data, including calculation of perfusion in absolute (ml/100g/min) units using the calibration data, and register the results to the structural image, as well as producing perfusion maps in MNI152 standard space. In general, the use of an `fsl_anat` analysis of a structural image with `oxford_asl` is recommended, but it is not required: perfusion can be calculated in the native space without the structural information.

5.5 Output

The outputs from Oxford_ASL are a resting state perfusion image called `perfusion.nii.gz`, which provides blood flow in relative (scanner) units, and an arrival time image called `arrival.nii.gz`. If a calibration image has been supplied then a further image `perfusion_calib.nii.gz` is also produced, which is a flow map in absolute units (ml/100g/min).

If calibration was performed then in the `calib` subdirectory you will find: - In *reference region (single)* mode: a text file called `M0b.txt` will be created that saves the estimated M0 value from arterial blood. If a CSF mask was not supplied then the automatically generated one will also be saved in the output directory as `csf_mask.nii.gz`. - In *voxelwise* mode (automatic when no structural image is provided): a `M0.nii.gz` image will be produced.

Various subdirectories are created:

- `native_space` in which perfusion and arrival time images in the native resolution of the ASL data are saved.
- `struct_space` provides results in the same space as the structural image (if supplied).
- `std_space` provides results in MNI152 standard space (if an `fsl_anat` results directory has been provided).

If you find the registration to be unsatisfactory, a new registration can be performed without having to repeat the main analysis using the results in `native_space`.

Calibrated output

If calibration is enabled, calibrated perfusion outputs are available with the suffix `_calib`. These contain quantified perfusion in ml/100g/min

Whole-brain averages

Within the `native_space` subdirectory, several whole-brain average values are defined:

- `<output>_gm_mean`: These are averages values in pure GM which by default is defined as voxels with more than 80% GM partial volume. This

threshold can be modified using the `--gm-thresh` option - `<output>_wm_mean`: These are averages values in pure WM which by default is defined as voxels with more than 90% WM partial volume. This threshold can be modified using the `--wm-thresh` option - `<output>_cortical_gm_mean`: These are average values in cortical GM which is defined as 'pure GM' voxels (see above) that are included in the Harvard-Oxford atlas Left/Right cortical mask (i.e. excluding subcortical GM). - `<output>_cerebral_wm_mean`: These are average values in cerebral WM which is defined as 'pure WM' voxels (see above) that are included in the Harvard-Oxford atlas Left/Right cortical mask (i.e. excluding subcortical WM).

****Normalized output ****

In all spaces, normalized output is produced, regardless of whether calibrated output is also being generated. Normalized output is generated by dividing the relative perfusion values by one of the whole brain averages defined above.

- `perfusion_norm` - This is perfusion normalized by whole brain mean pure GM (`perfusion_gm_mean`)
- `perfusion_wm_norm` - When partial volume correction is enabled, this is WM perfusion normalized by whole brain mean pure WM (`perfusion_wm_mean`)

5.6 Detailed usage information

This section contains a more in-depth look at some of the options available in `oxford_asl`

Main options

-m <mask>	a brain mask in the native space of the ASL data. This will be generated automatically by <code>oxford_asl</code> , this option is for the cases where you need your own mask.
--spatial	use spatial regularisation. This option is enable by default and is highly recommended. Use <code>--spatial=off</code> to disable.
--wp	Do analysis in 'White Paper Mode'. This analysis will conform to the assumptions made in the white paper about the underlying kinetic model and T1 values. Note, it still uses the Bayesian kinetic inference method in BASIL (thus spatial regularisation can be applied etc) and not the formula in the 'White Paper'.
--mc	Apply motion correction (using <code>mcfliirt</code>). This will also correct for motion between calibration image and main ASL data using an approach that minimises the interpolation applied to the main ASL data.

Acquisition specific

There are a number of acquisition sepecific parameters that you should set to describe your data to `oxford_asl`. Note, it is highly unlikely that the defaults for all of these parameters will be correct for your data - in particular you should pay attention to the follwing options.

--iaf=<diff,tc,ct>	Input ASL format: specifies if the data has already been label-control subtracted (<code>diff</code> , default), or is in the form of label(tag)-control pairs (<code>tc</code> or <code>ct</code> depending upon if label/tag is first).
--ibf=<rpt,tis>	Input block format. Specifically for multi-delay (multi-PLD) ASL data to identify whther the individual delays/PLDs are groups togther or by repeats of the same sequence of PLDs.
--casl	Data were acquired using cASL or pcASL labelling (pASL labeling is assumed by default).
--tis=<csv>	<p>The list of <i>inflow times</i> (TIs), a comma separated list of values should be provided (that matches the order in the data).</p> <p>Note, the inflow time is the PLD plus bolus duration for pcASL (and cASL), it equals the inversion time for pASL. If the data contains multiple repeats of the same set of TIs then it is only necessary to list the unique TIs.</p> <p>When using the <code>--tis=</code> you can specify a full list of all TIs/PLDs in the data (i.e., as many entries as there are label-control pairs). Or, if you have a number of TIs/PLDs repeated multiple times you can just list the unique TIs in order and <code>oxford_asl</code> will automatically replicate that list to mathc the number of repeated measurements in the data. If you have a variable number of repeats at each TI/PLD then either list all TIs or use the <code>--rpts=<csv></code> option (see below).</p>
--bolus=<value>	use this to specify the duration of the ASL labeling bolus used in the sequence (in seconds). For pcASL/cASL this will be the value fixed by the sequence, for pASL this will be taken as the initial value for bolus duration estimation (unless the <code>--fixbolus</code>) option is specified.
--bolus=<csv>	alternatively supply a list of bolus duration for each TI/PLD in the data (the length of the list should match that provided to <code>--tis=</code>).

- slicedt=<value>** For multi-slice (2D) acquisitions where superior slices are acquired later than those below, this option does not apply to 3D readouts. This provides the increase in time (in seconds) after labeling for a superior slice relative to the one directly below. It is assumed that the TIs provided refer to the lowest slice in the dataset.

There are further acquisition specific parameters that you might need to invoke depending upon your data, although the defaults here are more likely to apply.

- bat=<value>** A value for Arterial Transit Time (ATT), here called Bolus Arrival Time (BAT). For single delay/PLD ASL this is the value used in the perfusion calculation (and it is set to 0 in 'White Paper Mode'). For multi-delay/PLD ASL this value will be used to initialise the estimation of ATT from the data. Typically, the ATT is longer in pcASL compared to pASL. The defaults are 0.7 s for pASL and 1.3 s for pcASL based on typical experience.
- t1=<value>** The T1 value of tissue, 1.3 s by default (assuming acquisition at 3T).
- t1b=<value>** The T1 value of arterial blood, 1.65 s by default (assuming acquisition at 3T).
- sliceband=<number>** Number of slices per band in a multi-band acquisition.
- rpts=<csv>** Number of repeated measurements for each TI/PLD in the TIs list (**--tis=<csv>**), for use where the number of repeated measurements varies at each TI.

Structural image

The inclusion of a structural image is optional but highly recommended, as various useful pieces of information can be extracted when this image is used as part of `oxford_asl`, and partial volume correction can be done. Generally, we recommend the use of `fsl_anat` to process the structural image prior to use with `oxford_asl`.

- fslanat=<directory>** An `fsl_anat` results directory from the structural image (Note that ideally brain extraction and segmentation will have been performed, `oxford_asl` will also use the bias field correction if present).
- s <image>** High resolution structural image (assumed to be T1 weighted or similar). An alternative to `--fslanat`, if neither is not provided then results will be provided in native space only. Also requires the provision of a brain extracted version of the image with `--sbrain`.
- sbrain=<image>** Brain extracted (e.g., using `bet`) version of the structural image.
- fastsrc=<image_stub>** The results of a `fast` segmentation of the structural image. This option is an alternative to `--fslanat` for entering partial volume estimates (and bias field), in the same space as the structural image, into `oxford_asl`. It presumes the images will be presented with the same naming syntax as a `fast` output, but any alternative source of partial volume estimates could be used.
- senscorr** Instruct `oxford_asl` to use the bias field map from `fsl_anat` or `fast` for coil sensitivity correction where this hasn't been done on the scanner or there isn't a separate correction available.
- region-analysis** Generate additional regional analysis of the perfusion map by registration of the image to standard space and comparison with regions in

the Harvard-Oxford standard atlas.

Calibration

Most commonly you will have a calibration image that is some form of (approximately) proton-density-weighted image and thus will use the `-c` option.

- c <M0_calib_image>** specifies the M0 calibration image that is used to get flow values in absolute units. This should be an image with any repeated measurements stacked in the 4th (time) dimension.
- tr=<value>** the repetition time for the calibration image.
- alpha=<value>** the inversion efficiency of the labeling process, the defaults are likely to apply for most ASL data: 0.98 (pASL) or 0.85 (pcASL/cASL)
- cmethod=<single,voxel>** Specifies whether the calibration is done via a single M0 value calculated from the CSF in the ventricles (*single*) or using a voxelwise approach where M0 is calculated in every voxel (*voxel*).
- The voxelwise method is the simplest and follows the procedure in the ‘White Paper’, adding a correction for partial volume effects around the edge of the brain. This is used whenever a structural image is not supplied. The single method, using CSF for calibration, automatically generates a ventricle mask in ASL space from the segmentation of the structural image. You should inspect this mask to ensure it has been successful (in the `calib` subdirectory of the results). This procedure can sometimes fail, in which case you can supply your own mask using the `--csf` option. More advanced calibration can be performed using `asl_calib`.
- M0=<value>** A single precomputed value for the value of equilibrium magnetization in arterial blood. Useful when you have already performed calibration, e.g. using `asl_calib`.

There are further advanced/extended options for calibration:

- csf=<image>** Image in the same space as the structural that is a mask of voxels containing CSF to be used in calibration. This is a further option of the calibration step and allows the CSF mask to be manually specified if the automated procedure fails.
- cgain=<value>** If the calibration image has been acquired with a different gain to the ASL data this can be specified here. For example, when using background suppression the raw ASL signal will be much smaller than the (non background suppressed) calibration image so a higher gain might be employed in the acquisition.
- t1csf=<value>** Supply a value for the T1 of CSF to be used in the calibration process. Default values are used by `asl_calib` based on a 3T field strength (these can be checked by calling `asl_calib` at the command line).
- te=<value>** Set the echo time (in milliseconds) for the readout so that T2 (or T2*) effects are taken into account in the calibration. If this is not supplied then TE = 0 ms is assumed, i.e. T2/T2* effects are negligible. Default values are assumed by `asl_calib` for T2/T2* values, you might wish to treat these with caution as these are estimates based on the literature.
- t2star** Tells `oxford_asl` to correct for T2* rather than T2 effects. This simply tells `asl_calib` to use the default values for T2* in place of T2 in the calculations.
- t2csf=<value>** Supply a value for the T2 (in milliseconds) of CSF to be used in the calibration process, only relevant if you supply the TE value. Default values are used by `asl_calib` based on a 3T field strength (these can be checked by calling `asl_calib` at the command line).
- t2bl=<value>** Supply a value for the T2 of blood to be used in the calibration process, only relevant if you supply the TE value. Default values are used by `asl_calib` based on a 3T field strength (these can be checked by calling `asl_calib` at the command line).

Registration

There are some extended options (to be used alongside a structural image) for the purposes of registration.

- asl2struc=<mat>** an existing ASL to structural image transformation matrix, skips the registration process.
- r <image>** low resolution structural image used as an extra step in the registration to improve resulting transformation.
- regfrom=<image>** An alternative image to use as the basis of registration. This should be the same resolution as the ASL data and aligned to it.

Kinetic Analysis

- artoff** Turn off correction for signal arising from ASL signal still within the (macro) vasculature, this might be appropriate if the acquisition employed flow suppression. This is enabled by default for single-delay/PLD ASL.
- fixbolus** Turn off the automatic estimation of bolus duration, this might be appropriate if the bolus duration is well defined by the acquisition sequence and is on by default for cASL and pcASL. It might be appropriate to use this with pASL where the bolus duration has been fixed using QUIPSSII or Q2TIPS.
- fixbat** Force basil not to infer the ATT (BAT), this is on by default for single-delay/PLD ASL.
- batsd** The standard deviation for the ATT (BAT) prior distribution (default 0.316 seconds for single-PLD, 1.0 second for multi-PLD). See BASIL command line user guide for more information.
- infert1** Incorporate uncertainty in the T1 values into the analysis. Strictly this includes the T1 values in the inference process, but don't expect accurate T1 maps from ASL data.
- noiseprior** Use the in-built informative prior for noise estimation. This is particularly useful where you only have a small number of repeats/volumes in the main ASL data (e.g., if your data has already been averaged before you get it). This provides information to *basil* about the typical noise present in ASL data and helps with the application of appropriate spatial regularisation.
- noisesd** The standard deviation of the noise as described by the noise prior, overrides the values set internally and needs to be of the form of the standard deviation of the noise relative to the magnitude of the ASL data (only for very advanced use).

Distortion Correction

Distortion correction for (EPI) ASL images follows the methodology used in BOLD EPI distortion correction.

Using a separately acquired fieldmap (structural image is required), this can in principle be in any image space (not necessarily already aligned with the ASL or structural image), the syntax follows *epi_reg*:

- fmap=<image>** fieldmap image (in rad/s)
- fmapmag=<image>** fieldmap magnitude image - wholehead extracted
- fmapmagbrain=<image>** fieldmap magnitude image - brain extracted
- echospcing=<value>** effective EPI echo spacing (sometimes called dwell time) - in seconds
- pedir=<dir>** phase encoding direction, dir = x/y/z/-x/-y/-z
- nofmapreg** do not perform registration of fmap to T1 (use if fmap already in T1-space)

Further information on fieldmaps can be found under the *fsl_prepare_fieldmap* documentation on the FSL webpages.

Using phase-encode-reversed calibration image (a la `topup`):

--cblip phase-encode-reversed (blipped) calibration image
--echospacing=<value> Effective EPI echo spacing (sometimes called dwell time) - in seconds
--pedir=<dir> phase encoding direction, dir = x/y/z/-x/-y/-z

For `topup` the effective EPI echo spacing is converted to total readout time by multiplication by the number of slices (minus one) in the encode direction. Earlier versions of `oxford_asl` (pre v3.9.22) interpreted the `--echospacing` parameter as total readout time when supplied with a phase-encode-reversed calibration image.

Partial volume correction

Correction for the effect of partial voluming of grey and white matter, and CSF can be performed using `oxford_asl` to get maps of 'pure' grey (and white) matter perfusion. When partial volume correction is performed a separate subdirectory (`pvcorr`) within the main results subdirectories will appear with the corrected perfusion images in: in this directory the `perfusion.nii.gz` image is for grey matter, `perfusion_wm.nii.gz` contains white matter estimates. Note that, the non-corrected analysis is always run prior to partial volume correction and thus you will also get a conventional perfusion image.

--pvcorr : Do partial volume correction
 PV estimates will be taken from:

- `fsl_anat dir (--fslanat)`, if supplied
- existing fast segmentation (`--fastsrc`), if supplied
- FAST segmenation of structural (if using `-s` and `-sbet`)
- User supplied PV estimates (`-pvgm`, `-pvwm`)

--pvgm : Partial volume estimates for GM

--pvwm : Partial volume estimates for WM

Epoch analysis

The data can also be analysed as separate epochs based on the different measurements (volumes) within the ASL data. This can be a useful way of examining changes in perfusion over the duration of the acquisition, although shorter epochs will contain fewer measurements and thus be more noisy. Epoch analysis is always preceded by a conventional analysis of the full data and thus the conventional perfusion image will also be generated from the full dataset.

--elen Length of each epoch in TIs.
--eol Overlap of each epoch in TIs (default is 0).

5.7 Region analysis

Region analysis involves the generation of summary statistics for perfusion and arterial transit time within defined brain regions, either from standard atlases or from ROI images supplied by the user.

Basic region analysis with `oxford_asl`

If the `--region-analysis` option is specified an additional directory `native_space/region_analysis` will be created containing three files:

- `region_analysis.csv` - This file contains region analysis statistics for all voxels within the brain mask

- `region_analysis_gm.csv` - This file contains region analysis statistics for grey matter
- `region_analysis_wm.csv` - This file contains region analysis statistics for white matter

Region analysis is performed by using the registration from the structural image to standard space from an `fsl_anat` run. Hence `--fslanat` must be used in order to run region analysis.

The output files are in comma-separated format, suitable for loading into most spreadsheet or data processing applications. Within each region the following information is presented:

- `Nvoxels` - The number of voxels identified as being within this region
- `Mean, Std, Median, IQR` - Standard summary statistics for the perfusion values within this region
- `Precision-weighted mean` - The mean perfusion weighted by voxelwise precision ($1/\text{std.dev}$) estimates. This measure takes into account the confidence of the inference in the value returned for each voxel and is a standard measure used in meta-analysis to combine results of varying levels of confidence.
- `I2` - A measure of heterogeneity for the voxels within the region expressed as a percentage. A high value of `I2` suggests that there is significant variation in perfusion *within* the region that is not attributable to the inferred uncertainty in the estimates. For a definition of `I2` and an overview of its use in meta-analyses, see <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC192859/>

Definition of grey/white matter voxels

The definition of the included data for GM/WM output files varies according to whether or not you have included partial volume correction in your `oxford_asl` run.

If you have **not** used partial volume correction then GM voxels are derived from the structural segmentation and by default includes voxels with at least 80% GM. This threshold can be modified using the `--gm-thresh` option. WM voxels are those with at least 90% WM, and again this can be modified using `--wm-thresh`. The intention here is to restrict the statistics to those voxels which are near-enough ‘pure’ GM/WM.

If you **do** have partial volume correction, then `oxford_asl` will have generated separate perfusion maps for GM and WM which (in principle) only contains the perfusion contribution from these components. We use these single-tissue perfusion maps to generate the GM/WM statistics. However a base threshold of 10% is used to remove voxels that contain very little of the selected tissue type, e.g. the GM stats will ignore voxels with less than 10% GM. This is because the GM perfusion estimates in such voxels will have very high uncertainty and could bias the statistics.

Standard regions

By default, statistics are generated for a standard set of regions as follows:

GM and WM segmentation maps are used to define ‘pure’ GM and WM ROIs thresholded at 80% and 90% respectively. Note that these regions are included in all data files regardless of whether partial volume correction was performed, and are independent of the separation of voxels into GM and WM described above.

A second set of GM and WM ROIs are included based on 10%+ thresholding - i.e. regions including *some* of the corresponding tissue type.

A further set of standard regions are taken from the Harvard-Oxford cortical and subcortical atlases. Standard space regions are transformed to native ASL space and voxels with probability fraction > 0.5 are considered to lie within a region. At least 10 voxels must be found in order for statistics to be presented.

Using user-specified ROIs

In many cases users will want to provide their own ROIs to generate statistics in. This is supported via the `--region-analysis-atlas` option. This option can contain one or more image files (comma separated) each of which contains a 3D label image in MNI space. Each voxel contains an integer label and each unique integer > 1 defines a region in which to generate statistics.

To make the output more readable you can specify the names of the regions for each atlas using the `--region-analysis-atlas-labels` option. Again this should be one or more file names (comma separated)

and each file contains a list of text labels, one per line. The number of labels should be equal to the number of regions defined in the corresponding atlas image.

Other components of the BASIL toolset

6.1 BASIL (command)

6.1.1 BASIL (command) User Guide

Calling BASIL

A typical call to BASIL would look like:

```
basil -i asldiffdata.nii.gz -m mask.nii.gz -o basilout --spatial -@ basil_params.txt
```

You should always supply a parameter file using the `-@` option (see parameter file)

We highly recommend the use of the `--spatial` option for automated spatial regularisation of your data.

Basic options

BASIL can be called from the command line with the following information:

-i <file>	Input file containing label-control differenced data.
-m <file>	Brain mask for the data.
-o <dir>	Name of directory into which results are to be written (default is a subdirectory called basil within the input directory).
--optfile <file>	Model and sequence parameters (to be passed to FABBER) (was previously <code>-@</code>).
--spatial	Apply a spatial regularising prior to the estimated perfusion image, this is preferable to spatial smoothing of the data before analysis.

Model options

BASIL provides a number of options to access more advanced parts of the kinetic curve models, you should consult the literature to determine whether you want to explore these model options (the Further Reading and Literature sections for BASIL should be a good start):

--infertau	Infer the bolus duration from the data. This option is particularly suitable for pASL data in which the bolus duration is undefined (e.g. when not using QUIPSSII or Q2TIPS)
--inferart	Include an arterial compartment in the model. This option will infer the arterial blood volume (aCBV) and arterial blood arrival time. This option is suitable for data with multiple post-labelling delays, especially where short PLD (or TIs) are used.
--inferpc	Include a non-exchanging compartment in the model. This option might be used to model pre-capillary vessels assuming minimal water exchange. This parameter adds a pre-capillary transit time parameter.
--infert1	Infer the values of T1 and T1b from the data. This option is primarily to account for uncertainty in the T1 values in the inference. ASL data does not have sufficient sensitivity to T1 to estimate it accurately.

Advanced Options

BASIL also has a few more advanced options:

--t1im	For loading a image of T1 (tissue) values to be used in the kinetic modelling (same resolution as the ASL data).
--fast=<value>	Use a value of 2 to perform the analysis in a single step, mostly for use with --spatial. A value of 1 reduces the number of iterations performed in each step.
--pvgm	Perform partial volume correction with the supplied partial volume estimates for grey and white matter (these should be the same resolution as the ASL data).
--pvwm	White matter partial volume estimates (to go with --pvgm).
--init	Initialise BASIL with the results of a previous run - this option expects the <code>finalMVN.nii.gz</code> image from a previous <code>basil</code> run (model specification must match - use with caution).

Kinetic model choice

BASIL also offers a number of variants on the kinetic model used:

--disp=<option>	Model of the label dispersion <ul style="list-style-type: none"> <code>--disp=none</code> No dispersion. <code>--disp=gamma</code> Dispersion modelled according to a gamma dispersion kernel (vascular transport function). <code>--disp=gauss</code> Dispersion modelled according to a Gaussian dispersion kernel. <code>--disp=sgauss</code> Dispersion modelled according to a spatially (rather than temporally) derived Gaussian dispersion function.
--exch=<option>	Model of the exchange of labeled water in the capillary bed (selects the residue function). <ul style="list-style-type: none"> <code>--exch=mix</code> Well-mixed single compartment <code>--exch=simple</code> Simple single compartment with T1 of blood (mimics the assumptions made in the white paper)

- `--exch=2cpt` A two compartment exchange model following Parkes & Tofts (by default the slow solution is used).
- `--exch=spa` An implementation of the single pass approximation from St. Lawrence.

BASIL assumes by default a single well-mixed tissue compartment (`--exch=mix`) and no dispersion of the bolus of labeled blood water (`--disp=none`).

BASIL parameter file

BASIL requires a text file in which you specify model parameters, plus information about the collected data.

A generic BASIL options file for single-PLD pcASL (at 3T) might look like (preceding a line with # indicates it is a comment and will be ignored):

```
# Sequence/scanner parameters
--casl
--t1=1.3
--t1b=1.65
--tau=1.8
# tau specified label/bolus duration

# Data information
--repeats=10 --pld=1.8
```

An generic file for multi-TI pASL might look like:

```
# Sequence/scanner parameters
--t1=1.3
--t1b=1.65
--tau=0.7

# Data information
--repeats=10 --ti1=0.25 --ti2=0.5 --ti3=0.75 --ti4=1.0 --ti5=1.25 --ti6=1.5 --ti7=1.
↪75 --ti8=2.0
```

Model parameters

By default BASIL assumes that your data is pulsed ASL (pASL), if you are using continuous (cASL) or pseudo continuous (pcASL) labelling then you should set the cASL option:

--casl Use the cASL version of the model (NOTE: the default ATT value is likely to be poorly suited to pcASL/cASL data, see below).

For the model you can set the appropriate values of T1 (and T1b) as well as the duration of the label as set by your sequence, if these are not specified in the parameter file then the default values are used:

--t1=<value> The value of T1 (default 1.3 seconds).

--t1b=<value> The value of T1b (default 1.65 seconds).

--t1wm=<value> The T1 value of white matter (default 1.1 seconds) - only for partial volume correction.

You can set an appropriate Arterial Transit Time (sometimes called Bolus Arrival Time) value. This will be used as the mean of the prior distribution for the ATT parameter during inference, i.e., the default value for ATT which will be updated based on the data.

--bat=<value> The value of ATT (aka Bolus Arrival Time) (default 0.7 seconds).

NOTE: in `oxford_asl` the default ATT is automatically changed from 0.7 seconds to 1.3 seconds for cASL/pcASL. This does not happen in `basil`, you need to do this using the `--bat` option.

--batsd=<value> The value of the standard deviation for the ATT prior distribution (default 0.316 seconds).

The default value is appropriate if you are treating ATT as a confound. If you are interested in estimating ATT from multi-PLD/TI ASL you may wish to use `--batsd=1`, the default value chosen by `oxford_asl`.

Some models variants will have their own specific options, see Kinetic Model.

Data Parameters

Alongside model information the parameter file also contains information about the data, including the post-label delay(s) for pcASL or the inversion times for pASL and how many repeats of each are contained in the file. You should specify each PLD/TI individually in the order that they appear in the data.

Post Label delay(s)

--pld=<value> The time (in seconds) for the PLD in single-PLD cASL/pcASL.

--pld1=<value>, --pld2=<value>, --pld-n=<value> The time (in seconds) of the n th PLD in multi-PLD cASL/pcASL.

Inversion time(s)

--ti1=<value>, --ti2=<value>, --ti-n=<value> The time (in seconds) of the n th TI for multi-TI pASL.

Label duration(s)

--tau=<value> Label bolus duration (default is infinite).

--tau1=<value>, --tau2=<value>, --tau-n=<value> Label duration for the n th PLD measurement. Used where pcASL has been applied with different label durations.

A fixed bolus duration is set in any cASL/pcASL implementation. For pASL a fixed bolus duration is often implemented using QUIPSS2 for example. If the bolus length is not fixed, e.g. FAIR then BASIL can estimate the bolus duration from multi-TI data if you use the `infertau` option when calling BASIL.

Slice timing

--slicedt The time (in seconds) between acquisition of different slices in a 2D multi-slice readout. This is used to adjust the PLD for more superior slices (this assumes that the most inferior slice is acquired first with a PLD/TI that matches the value supplied via `--pld` or `--ti`).

Look-locker readout (for multi-PLD/TI)

--FA=<value> The flip angle in a Look-Locker readout scheme.

Flow suppression (multiple phases)

--crush1=<value>, --crush2=<value>, --crush-n=<value> Specification of the flow suppressing crusher direction for the n th PLD/TI. Any one of `xyz`, `-xyz`, `x-yz`, `-x-yz`.

Time or Hadamard encoding

BASIL is directly compatible with time/hadamard encoding where ‘decoding’ has been performed. In that case the multi-PLD data can be used exactly like any other multi-PLD pcASL with suitable setting of the PLDs and label duration.

BASIL can also directly estimate perfusion from ‘raw’, i.e. not decoded, data. Although this is currently limited to specific cases - largely ones that use the same duration for each of the encoded blocks. To use this option the input

data is the raw data as acquired and you tell BASIL the number of cycles to expect, you should specify the appropriate **single** PLD and label duration values.

--hadamard=<value> Labeling has been performed using hadamard encoding with the number of cycles specified, and the data has not been ‘decoded’ prior to being input to BASIL.

For this analysis it is necessary to also infer the static tissue component (that would otherwise have been removed during decoding). Thus the following options need to be added to the basil options file: `--incstattiss`
`--inferstattiss`

--fullhad When the full Hadamard matrix is needed. This is for the case where the hadamard encoding included the first ‘column’ of all control boluses. (If this doesn’t mean anything to you, the chances are that it isn’t relevant).

Repeated measurements

--repeats=<n> The number of repeats of each PLD or TI in the data (default is 1).

BASIL processes data where there are multiple measurements at the same PLD/TI, as indicated by the `--repeats` option: in which case it is assumed that the data comes with the individual time points in the 4th dimension, with **repeats at each PLD/TI coming in blocks (groups)**. Suitable manipulation of the data can be done using `asl_file`.

For example: the data contains 8 readings taken at 4 TIs (0.5, 1, 1.5, 2 seconds), repeated twice. It should be presented to BASIL with each TI grouped together

i.e. TI1 TI1 TI2 TI2 TI3 TI3 TI4 TI4

Hence the parameter file would contain:

```
--ti1=0.5 --ti2=1 --ti3=1.5 --ti4=2 --repeats=2
```

NOTE that the number of TIs specified multiplied by the number of repeats should equal the number of time points in the 4D input data set.

It is possible to deal with more complicated data by specifying an individual `--ti[n]=` for every time point in the data, for the above example we could equally input it to BASIL as:

```
--ti1=0.5 --ti2=0.5 --ti3=1 --ti4=1 --ti5=1.5 --ti6=1.5 --ti7=2 --ti8=2
```

Results (outputs)

Within the output directory a number of subdirectories will be created containing the results from each step these comprise:

- `info.txt` Text file containing information from BASIL about what was done in this step.
- `paramnames.txt` A list of names of the parameters inferred, these will correspond with the names of the results files.
- `mean_{paramname}.nii.gz` The parameter estimate image for paramname.
- `var_{paramname}.nii.gz` The estimate variance image for parameter paramname.
- `zstat_{paramname}.nii.gz` A pseudo z-statistic image for paramname, uses variance information to give a measure of the confidence with which that parameter deviates from 0.
- `finalMVN.nii.gz` All the parameter estimates and variances (including noise parameters) in one file. This can be interrogated with `mvntool` and can be used to initialise a further run of BASIL.
- `logfile` The logfile from FABBER.
- `FreeEnergy.nii.gz` Images of the free energy from FABBER, see references for more information.

Depending upon the model options chosen there will be a range of parameters for which results will be provided. The multi-step nature of basil means that more parameters are likely to be found in the later steps, as models of increasingly complexity are fit as the step number is increased.

Typical parameter names from BASIL are:

- `ftiss` (relative) tissue perfusion.
- `delttiss` arterial transit time (transit time or bolus arrival time to the tissue component).
- `fblood` (relative) arterial cerebral blood volume, the scaling parameter of the arterial/macrovascular component.
- `deltblood` bolus arrival time (to arterial component).
- `fwm` (relative) white matter perfusion.
- `deltwm` arterial transit time for white matter.

Noise Model (Advanced option)

BASIL assumes that you wish to use a standard white noise model to analyse resting-ASL data. This model assumes that the noise in each voxel can be described by a single noise magnitude, this is sufficient in practice for most ASL data. If you are feeling adventurous (or have good reason) you may instruct BASIL to use different noise magnitudes for different sections of the input data, e.g. a different value at each inversion time.

This is done in the parameter file using the `--noise-pattern=` option, which is used as follows: Taking the example of data with 4 TIs each repeated 5 times, to get a different noise magnitude at each inversion time use:

```
--noise-pattern=11111222223333344444
```

i.e. the first 5 entries correspond to the first TI and these should use the first noise magnitude, the next 5 entries are the next TI and next noise magnitude etc. The numerbs here are purely labels and do not relate to the actual magnitude of the noise, which will be estimed by `basil` from the data.

NOTE: if you have more than 9 TIs then for the 10th TI and onward letters should be used in place of numbers starting with a, i.e. for 12 TIs and 2 repeats:

```
--noise-pattern=112233445566778899aabbcc
```

NOTE: if you have only a small number of repeats (like these examples) then this more complex noise modelling is probably not a good idea.

6.1.2 BASIL (command) Examples

The following provides some examples of the usage of the BASIL command line tool.

Single PLD pcASL

This example uses the single-PLD pcASL data from the Oxford Neuroimaging Primer: Introduction to Perfusion Quantification using Arterial Spin Labelling MRI. This can be found [here](#).

Firstly, we need a mask in which to do the analysis. We will form a very simple mask using BET on the raw ASL data (more robust options are typically used in `oxford_asl` based on the structural image if available).:

```
bet asltc.nii.gz asltc_brain -m
```


Secondly, we need to do label-control subtraction prior to basil analysis, we can use `asl_file` for this (for more information see the `asl_file` examples):

```
asl_file --data=asltc.nii.gz --ntis=1 --iaf=tc --diff --out=diffdata
```

For analysis we need a `basil_options.txt` file. For this data the following file specifies all the information we need, taking the default values of T1 as adequate.:

```
#BASIL options file
--casl
--bolus=1.8
--pld=1.8
--repeats=30
```

Finally, we can perform the analysis.:

```
basil -i diffdata.nii.gz -o basilout -m asltc_brain_mask -@
basil_options.txt --spatial
```

We have used the recommended `spatial` option in this case. The terminal should display an output similar to:

```
Creating output directory: basilout
STEP 1: VB - Tissue
-----
Welcome to FABBER v3.9.2-36-g4ba2db9
-----
Logfile started: basilout/step1/logfile
100%

Final logfile: basilout/step1/logfile
STEP 2: Spatial VB Tissue - init with STEP 1
-----
Welcome to FABBER v3.9.2-36-g4ba2db9
-----
Logfile started: basilout/step2/logfile
100%

Final logfile: basilout+/step2/logfile
End.
```

Note that BASIL has run in two steps, the second step is 'spatial' and was initialised by the first step. In both steps only a tissue component was inferred.

The contents of the `basilout` directory should look something like:

```
logfile      params.txt      step1      step2
```

Both `step1` and `step2` contain a similar set of files, the difference in this case being whether the spatial method was applied or not. In general, the contents of the highest numbered step subdirectory is the result you will want.

The `step2` subdirectory should have contents similar to:

```
finalMVN.nii.gz      mean_delttiss.nii.gz      noise_stdevs.nii.gz      std_ftiss.nii.gz_
→ zstat_ftiss.nii.gz
info.txt             mean_ftiss.nii.gz         paramnames.txt           uname.txt
logfile              noise_means.nii.gz       std_delttiss.nii.gz
zstat_delttiss.nii.gz
```

Of most interest in this case is `mean_fitss.nii.gz` which is the (relative) tissue perfusion image. If you compare this to the same file in the `step1` subdirectory you will be able to see the difference that the spatial prior makes in this case.

Of note are:

- `std_fitss.nii.gz` an estimate of the standard deviation on the tissue perfusion estimates.
- `mean_delttiss.nii.gz` since this is single-PLD data this image looks like the mask we supplied with a value of 1.3, which is the default value for the ATT parameter for pcASL. Since we cannot estimate ATT from single-PLD pcASL data, BASIL has applied the default value in the model inference for us. If we had wanted a different value we could have set that using the `--bat=<value>` option in the `basil_options.txt` file.

Multi PLD pcASL

This example uses the multi-PLD pcASL data from the Oxford Neuroimaging Primer: Introduction to Perfusion Quantification using Arterial Spin Labelling MRI. This can be found [here](#).

As in the single-PLD example we need a brain mask:

```
bet asltc.nii.gz asltc_brain -m
```

We need to do label-control subtraction:

```
asl_file --data=asltc.nii.gz --ntis=6 --iaf=tc --diff --out=diffdata
```

Note that this data has six PLD, hence `--ntis=6`. Also note that helpfully the call to `asl_file` calculates the number of repeats for us:

```
Number of voxels is:98304
Number of repeats in data is:8
Done.
```

We need a `basil_options.txt` file that includes a specification of the PLD in the data:

```
#BASIL options file
--casl
--bolus=1.8
--pld1=0.25 --pld2=0.5 --pld3=0.75 --pld4=1.0 --pld5=1.25 --pld6=1.5
--repeats=8
```

We are ready to call basil:

```
basil -i diffdata.nii.gz -o basilout -m asltc_brain_mask -@
basil_options.txt --spatial
```

Which produces an output that is essentially identical to that for the single-PLD case (as we are doing the same analysis here, just on different data). Note that if you are running this example after the single-PLD case you will get `basilout+` as your output directory, `basil` preserves any existing directories with the same name as the output directory specified.

As with the single-PLD example we can examine the perfusion image from the highest numbered step: `mean_ftiss` in subdirectory `step2`. We can now also examine the ATT map `mean_delttiss`.

Being multi-PLD data, we might consider a more advanced analysis. For example, we could add an arterial (or macrovascular) component to the model:

```
basil -i diffdata.nii.gz -o basilout -m asltc_brain_mask -@
basil_options.txt --spatial --inferart
```

This gives a three step analysis:

```
Creating output directory: basilout+
STEP 1: VB - Tissue
-----
Welcome to FABBER v3.9.2-36-g4ba2db9
-----
Logfile started: basilout+/step1/logfile
100%

Final logfile: basilout+/step1/logfile
STEP 2: VB - Tissue Arterial - init with STEP 1
-----
Welcome to FABBER v3.9.2-36-g4ba2db9
-----
Logfile started: basilout+/step2/logfile
100%

Final logfile: basilout+/step2/logfile
STEP 3: Spatial VB Tissue Arterial - init with STEP 2
-----
Welcome to FABBER v3.9.2-36-g4ba2db9
-----
Logfile started: basilout+/step3/logfile
100%

Final logfile: basilout+/step3/logfile
End.
```

The arterial component was added in step two, with the spatial prior applied in the third and final step.

Now in the output directory are three subdirectories for the different steps. In the both `step2` and `step3` you will find, alongside the files present in the previous analysis, files related to the arterial cerebral blood volume parameter, named `fblood`.

The BASIL command line tool performs kinetic model inversion on ASL label-control difference data. It uses a common Bayesian inference method regardless of whether the data contains a single or multiple post labelling delays. BASIL includes a flexibly defined kinetic model appropriate for ASL kinetics that can be applied in humans and also other species - for more information see the model section.

To run BASIL on resting-state ASL data you will need:

- ASL difference data (single or multiple post labeling delays)
 - differencing of label and control images should have been done already, see `asl_file`.
- Details of the sequence, i.e. post-labelling delay(s), bolus duration, etc.

6.1.3 Multi-step inference

BASIL runs in multiple steps increasing the model complexity at each stage, this ensures a more robust final result by ensuring good convergence upon the global solution in all voxels. In general we recommend including the spatial prior/regularisation option that BASIL offers, this is run as a final step.

A rough overview of the process would be:

- STEP 1: Bayesian inference - Inference for CBF and arrival time (and optionally label duration)
- STEP 2+: Bayesian inference - further parameters of the model can be inferred from the data.
- STEP N: Bayesian inference with spatial prior - a final run for all the parameters including a spatial prior on the perfusion parameter, initialised by the prior step.

In `oxford_asl` the data analysis using `basil` is often run twice: firstly on the data where the different repeats at the various PLD have been averaged, and then on the full data using the output of the first run to initialise the second. This is for similar reasons of robustness and encouraging good convergence as the multi-step process outlined above. This is not a default behaviour of the `basil` command line tool, but can be achieved using the `--init` option.

6.1.4 Kinetic model

The Kinetic curve model for resting state ASL is built into FABBER and called by BASIL. The model implemented follows the ‘standard’ or ‘Buxton’ model, for more information see:

Buxton, R. B., L. R. Frank, et al., ‘A general kinetic model for quantitative perfusion imaging with arterial spin labeling’, *Magnetic Resonance in Medicine* 40(3): 383-396, 1998.

As per this paper, BASIL assumes by default a single well-mixed tissue compartment and no dispersion of the bolus of labeled blood water. Different T1 values are assumed for blood and tissue water, but it is possible to set these to be identical to match the simple model assumed by the quantification formula in the ‘white paper’.

BASIL also implements a range of alternative arterial input functions - to model dispersion - and residue functions - to model restricted water exchange.

Dispersion and Arterial Input Functions

BASIL includes a number of Vascular Transport Function (Dispersion Kernel) models of dispersion. These include modelling the VTF as either a Gamma or Gaussian function. For more information see:

Chappell, M. A., Woolrich, M. W., Kazan, S., Jezzard, P., Payne, S. J., & MacIntosh, B. J. (2013). Modeling dispersion in arterial spin labeling: validation using dynamic angiographic measurements. *Magnetic Resonance in Medicine*, 69(2), 563–570. <http://doi.org/10.1002/mrm.24260>

Hrabe, J., & Lewis, D. (2004). Two analytical solutions for a model of pulsed arterial spin labeling with randomized blood arrival times. *Journal of Magnetic Resonance*, 167(1), 49–55.

Exchange and residue functions

As well as the single well-mixed tissue compartment model in which the residue function just accounts for T1 decay (and a small venous outflow), BASIL includes two-compartment exchange models as described in the following papers:

Parkes, L., & Tofts, P. (2002). Improved accuracy of human cerebral blood perfusion measurements using arterial spin labeling: Accounting for capillary water permeability. *Magnetic Resonance in Medicine*, 48(1), 27–41.

St Lawrence, K., Frank, J., & McLaughlin, A. (2000). Effect of restricted water exchange on cerebral blood flow values calculated with arterial spin tagging: A theoretical investigation. *Magnetic Resonance in Medicine*, 44(3), 440–449.

6.2 asl_calib - calibration of ASL perfusion

6.2.1 Overview

ASL tag-control difference data can be used to quantify perfusion. However, the values obtained are not by default provided in conventional units. To get absolute CBF quantification it is also necessary to estimate the equilibrium

magnetization (M_0) of arterial blood.

A popular option is to correct for the equilibrium magnetization of arterial blood using a proton density (PD) weighted image (using a relatively long TR) and dividing the ASL perfusion image (the output of `basil`) by the PD image voxel-by-voxel. Alternatively, the M_0 value for arterial blood can be estimated indirectly from a measurement in a reference ‘tissue’, such as the CSF or white matter, either:

- **LongTR:** From a separate calibration image that uses the same acquisition as the ASL data, but contains no inversion (i.e. a ‘control’ image) and no background suppression. Ideally the images would be acquired with a very long TR. However, it is possible to account for shorter TR values, for example matching that of ASL sequence, with an estimate of the T1 of the reference ‘tissue’.
- **SatRecov:** From the saturation recovery of the control images in the ASL data sequence, if a presaturation has been applied in the imaging region.

`asl_calib` performs the necessary steps to obtain the M_0 of arterial blood value from such a calibration images. It can also:

- **LongTR method:** produce a spatial sensitivity estimate for the coil used for acquisition, if another calibration image is supplied that was acquired using some other coil (assumed to have a flat spatial sensitivity) as a reference (e.g. the body coil).
- **SatRecov method:** produce an estimated T1 of tissue image for use in kinetic curve model fitting.

6.2.2 User Guide

Using M_0 and sensitivity images to calculate absolute CBF

`asl_calib` can be instructed to save the M_0 value and the sensitivity image (if calculated) for subsequent use to calculate absolute CBF. Given an estimated perfusion image, e.g. from `basil`, absolute CBF in ml/100g/min can be obtained using `fslmaths`:

With M_0 only:

```
fslmaths [perfusion_image] -div cat [M0_text_file] -mul 6000 [absolute_CBF_output_
↪image]
```

With M_0 and sensitivity image:

```
fslmaths [perfusion_image] -div cat [M0_text_file] -div [sensitivity_image] -mul 6000_
↪[absolute_CBF_output_image]
```

For these calculations the CBF image should still be in the native resolution of the ASL data. The first option (with M_0 only) will work with perfusion images that have been converted to another resolution, e.g. standard space.

`asl_calib` usage

Typing the `asl_calib` with no options will give the basic usage information, the following is a more detailed version:

- c **<calib_data>** Calibration data in Nifti file format with the individual images stacked in the time dimension.
- s **<structural_image>** Structural image used for determining reference ‘tissue’ mask (not required if reference ‘tissue’ mask is supplied, see below).
- t **<asl to structural transformation matrix>** Transformation matrix for ASL images to structural image space, e.g. from `asl_reg`, (not required if reference ‘tissue’ mask is supplied, see below).

- mode <mode>** Specify what form the calibration data takes, options are: longtr, satrecov. See below for mode specific options.
- tissref <Reference_tissue_type>** The 'tissue' type to use as a reference, see below, options are: csf, wm, gm, none.
- te <TE_value>** TE of the calibration sequence in seconds, default is 0 s.
- i <perfusion_image>** A perfusion image for calibration. This should be still at the native resolution of the ASL data.

Output options

- o <absolute_CBF_image_name>** File to which absolute CBF image should be saved, if input image has been supplied with **-i**.
- Mo <M0_value_save_file>** The estimated M0 value of arterial blood will be saved as text to a file of this name. This can then be used to convert a perfusion image into absolute values.

Extended Options

- m <CSF_mask>** Provide a 'tissue' reference mask, e.g. hand drawn, instead of relying upon automated mask creation. If a mask is supplied the structural image and ASL to structural transformation are no longer required.
- bmask <brain_mask>** A mask of the brain in (ASL native space), this will be used for sensitivity estimation (LongTR method) or T1 estimation (SatRecov method). If not supplied a brain mask will be generated automatically from the calibration data if it is needed, this option allows the same mask from other processing steps to be employed for consistency.
- t2star** Tells `asl_calib` to do T2* correction rather than T2 correction. This option simply alters which set of default T2(*) values are used.
- t1r <T1_reference_tissue>** T1 (in seconds) for the reference tissue, the defaults for the different **--tissref** options are (based on 3T): csf 3.4, gm, 1.3, wm 1.0.
- t2r <T2_reference_tissue>** T2(*) (in milliseconds) for the reference tissue, the defaults for the different **--tissref** options are (based on 3T) T2/T2*: csf 750/500, gm, 100/50, wm 50/20. These defaults are general estimates based on the literature and should be used with care.
- t2b <T2_blood>** T2 (in milliseconds) for blood, the default is 150/50 (T2/T2*). The defaults are a general estimate based on the literature and should be used with care.

Mode specific options

LongTR

- tr <TR_value>** TR of the calibration sequence in seconds, default is 3.2 s.
- cagin <calibration_gain>** The relative gain of the ASL data to that of the calibration image, default 1. This allows for the case where the ASL data has been acquired with a higher gain than the calibration images, for example where background suppression was used allowing for a higher gain to be set for the ASL data.
- cref <calibration_reference_image>** A further image acquired using the same parameters as the main calibration file, but with a different coil to be used as a reference to calculate the sensitivity of the coil used for the main ASL data.

- osen <sensitivity_image_out_file>** Specify where the sensitivity file can be saved, if a reference image has been supplied with **--cref**. This can be used later to correct an estimated CBF image for coil sensitivity.
- isen <sensitivity_image>** provide a sensitivity image (that matches the calibration image) to be used in calculations.

SatRecov

- tis <List_of_tis>** Comma separated list of inversion times in the data (in seconds), e.g. **--tis 0.2,0.4,0.6**.
- fa <Flip_angle>** Flip angle in degrees for Look-Locker readouts, do not set if not using Look-Locker.
- lfa <Low_flip_angle>** Low flip angle for Look-Locker readouts in which an extra set of TIs were acquired with a lower flip angle. This is used to estimate the correction for true flip angle at every voxel. It is assumed that the low flip angle data is the final phase (set of TIs) in the calibration data.
- nphases <number_of_phases>** The number of phases (sets of TIs) at the higher flip angle.

‘Tissue’ reference type

`asl_calib` will let you choose what ‘tissue’ you want to use as the reference. `M0` is calculated within a mask of this ‘tissue’, as the mean over all the voxels within the mask. This option tells `asl_calib` which ‘tissue’ from the automatic segmentation as well as what T1 and T2(*) values should be used.

By default `asl_calib` uses CSF as the reference because it is relatively easy to segment and a mask can be defined containing a reasonable number of voxels that do not suffer substantial partial volume effects. The automated masking is optimized to extract CSF from the ventricles and this is probably the best reference to use. However, ventricular CSF is likely to be in the region of lowest coil sensitivity for multi-channel coils, and the longer T1 value of CSF can lead to bias when the TR is comparatively short (< 5 seconds). White matter is a reasonable alternative as partial volume effects can be minimized to a good degree. Grey matter is generally not a good option for that reason.

Automatic reference ‘tissue’ mask

`asl_calib` attempts to automatically generate the reference ‘tissue’ mask from the structural image, unless you supply your own custom mask with the `-m` option. It does this using `FAST`, thus the normal caveats for segmentation when using that program apply, for example the structural image must already have been brain extracted.

Having a really perfect mask is not vital, since the `M0` calculation is performed over all the voxels within the mask. However, the mask needs to at least be sensible, hence it is a very good idea to check the mask created at the end. If `asl_calib` detects that after segmentation, transformation into ASL native space and thresholding, that there are no voxels in the mask it will halt and tell you that the automated method has failed.

6.3 `asl_file` - preprocessing of ASL data

6.3.1 Overview

`asl_file` is a command line tool designed for the convenient manipulation of ASL data within FSL. It is part of the BASIL toolset (and is used extensively within the `oxford_asl` and `Asl_gui` tools).

ASL data has the relatively unique feature compared to other MRI datasets that it is comprised of an interleaved series of label and control images. Common manipulations of ASL data require either the pairwise subtraction of the volumes and/or the extraction of control (or label) images, this can be quite tedious with existing image file manipulation tools.

This gets even more complicated with multi inflow-time ASL that contains label-control pairs at a range of inflow-times. `asl_file` was thus designed to know about the common structures of ASL data and permits direct operations without the need to separate out the individual volumes.

`asl_file` also includes some more advanced features including:

- the ability to generate epochs of ASL data so that perfusion variations during acquisition can be investigated.
- simple partial volume correction using the Linear Regression method (this is an alternative to the spatial method implemented in BASIL itself).
- a simple routine to correct for partial volume effects at the edge of the brain in the M0/calibration image that causes overestimation artifacts in voxelwise calibration.

6.3.2 User Guide

A list of command line options can be found for `asl_file` by typing:

```
asl_file -h
```

The basic usage is:

```
asl_file --data=<asldata> --ntis=<number_if_inflow_times> [options]
```

This specifies the ASL data source file and the number of inflow-times present in the data. Note that inflow-time here is a generic reference to the time between labeling and imaging: it refers to a post-labeling delay in pcASL or inversion time in pASL. For `asl_file` it is only necessary to specify the number of inflow-times in the file and not the individual parameters (i.e. the delay time) associated with them.

Various options exists to specify the output required and, where needed, the structure of the data in the source and output files, as well as any operations that need to be done to the data.

Input options

By default `asl_file` assumes that the input data is ‘difference data’, i.e. label-control subtraction has already been performed. It is often useful to use `asl_file` with data in the form of label-control pairs, this can be specified using the `--iaf` option.

--iaf=diff	Input data is differenced, pairwise label-control subtraction has already been performed.
--iaf=tc	Input data is label-control (tag-control) pairs, with the first volume being a tag (labelled) image.
--iaf=ct	Input data is label-control pairs, with the first volume being a control image.

`--iaf=tcb/ctb` Input data contains label-control volumes, but all of the label are grouped together separately from all of the control images within the data (in a ‘block’). This is fairyl unusla, as the label and control images are normally interleaved as is assumed by the other options.

Block format (multi inflow data)

Additionally, where the data contains repeated volumes at different inflow times (multi inflow-time) it may be necessary to specify the ‘block format’, i.e. whether repeated measurements at the same inflow time are grouped together or appear grouped by repeat number.

--ibf=tis	Volumes with the same inflow times have been grouped together. This is the inout format expected by the <code>basil</code> command line tool. For example: T11 T11 T11 T12 T12 T12 T13 T13 T13
------------------	---

--ibf=rpt Volumes are grouped as repeated of the same set of inflow-times. This is most common, since it reflects a usual acquisition strategy to iterate through the inflow-times repeatedly. For example: **T1 T12 T13 T1 T12 T13 T1 T12 T13**

Numbers of repeats in the data

For multi inflow-time data, `asl_file` will automatically calculate the number of repeats of each inflow-time present, assuming that there is the same number of each. For an acquisition where the number of repeats at each inflow time varies you can use the `--rpts=<list>` option to specify, as a comma separated list, the number of repeats of each inflow-time. Note this is only valid when using `--ibf=tis`.

Brain mask (optional)

--mask=<image> Optionally a mask in which the processing is to be performed can be supplied. For most operations a mask is not essential, although will result in faster processing.

Output Options

A filename for the output volume is specified by the `--out=<filename>` option. For multi inflow-time data the 'block format' for the output data can be specified (see above) using the `--obf` option, by default this matches the value of `--ibf`.

Pre-output Operations

A number of operations can be applied to the data prior to output (these cannot be combined).

Label-control (pairwise) subtraction

--diff Do a pairwise subtraction of odd volumes from even volumes, this will compute the difference data, if the input image contains label-control pairs. For N volumes in the input image there will be $N/2$ in the output image.

--surdiff Do a pairwise subtraction of every sequential pair of volumes. This also produces label-control difference data, but performs subtraction of even from odd volumes as well as odd from even volumes (correcting for the sign difference in label-control subtraction that would ensue). For N volumes in the input image there will be $N-1$ in the output image.

Splitting label-control pairs

--spairs Split pairs within the data, i.e. split label and control images. The output from each output option (see below) will create two (rather than one) files with `_odd` and `_even` appended to the name given to the output option. These contain the result of the output operation odd and even volumes respectively.

Partial volume correction

Partial volume correction using the Linear Regression method can be performed by specifying the following (requires `--mask`):

--pvgm=<image> An image of the partial volume of grey matter within every voxel within the mask (at the same resolution as the input data).

--pvwm=<image> An image of the partial volume of white matter within every voxel within the mask (at the same resolution as the input data).

--kernel=<value> the size of the kernel to be used, must be an odd number between 3 and 9, default 5. Note that `asl_file` uses a simple 2D (in plane) kernel.

Edge correction

To correct for edge effects in voxelwise calibration (a form of partial volume effect at the edge of the brain), `asl_file` can perform simple erosion and extrapolation on a calibration image if the option `--extrapolate` is specified. It is possible to control the neighbour size for extrapolation using the `--neighbour=<value>` option.

Output Operations

Apart from outputting the data subject to rearrangement of the internal structure using the output options or pre-output operations, further outputs can be generated after the other operations have been performed. You can supply multiple output options at the same time.

- mean=<filename>** Take the mean over the volumes, e.g. for the creation of the mean difference image from label-control pairs using the `--diff` option. For multi-delay data the mean is taken within each inflow-time, thus the final image will contain the same number of volumes as inflow-times specified by the `--ntis` option.
- split=<filenameroot>** For multi inflow-time data split the data into separate files for each inflow time.

Extracting epoch of data

- epoch=<filenameroot>** Outputs as separate images the mean within individual epochs of the data.

Parameters of the epochs are defined by

- elen=<value>** The length of each epoch in the specified epoch units.
- eol=<value>** The amount of overlap between epochs in the specified epoch units.
- eunit** The units to be used for the creation of the epochs. `--eunit=rpt` (default) Epochs are calculated with the unit of calculation being the number of repeats, this would always be appropriate for single inflow-time data. For multi inflow-time data the mean would be taken within each inflow-time in each epoch, thus each image would contain the same number of volumes as inflow-times specified by the `--ntis` option. `--eunit=tis` Specific to multi-inflow time data (and a very advanced option). This permits the creation of epochs from the raw time series such that each epoch will contain the specified number of volumes from the input data given by `--elen`, this could be a mixture of inflow-times (and repeats thereof) depending upon the ordering in the data.

6.4 QUASIL - quantification of QUASAR data

6.4.1 Overview

QUASIL is a special implementation of BASIL specifically designed to exploit the features of QUASAR ASL data. It uses the same two component (tissue plus macro vascular signal) model that is employed by BASIL, but it has been extended to use all the information provided by the various phases for flow suppression provided by the QUASAR sequence. QUASIL uses information from the full QUASAR dataset to produce CBF images in absolute units (using an implementation of `asl_calib`). QUASIL also provides the option to perform a ‘model-free’ analysis using a very similar methodology as presented in the original QUASAR paper.

More information on the model used can be found in:

Chappell, M. A., Woolrich, M. W., Petersen, E. T., Golay, X., & Payne, S. J. (2012). Comparing model-based and model-free analysis methods for QUASAR arterial spin labeling perfusion quantification. Magnetic resonance in medicine. doi:10.1002/mrm.24372

More information on the model-free method can be found in the original QUASAR paper:

Petersen, E., Lim, T., & Golay, X. (2006). Model-free arterial spin labeling quantification approach for perfusion MRI. Magnetic resonance in medicine , 55(2), 219–232. doi:10.1002/mrm.20784

6.4.2 User Guide

Usage

Since the acquisition of data using QUASAR is very well defined there are far fewer options to set with QUASIL than a typical BASIL analysis. NOTE that QUASIL expects the data without tag-control subtraction having been performed. A typical command line usage would be:

```
quasil -i <asl_data> -o <output_directory>
```

This would carry out a model-based analysis of the ASL data and provide voxelwise estimates of perfusion, arterial transit time and arterial blood volume (aBV). The calibration of the data to the equilibrium magnetization is also carried out as part of the processing so that the perfusion image is provided in absolute units (ml/100ml/min). Additionally, the perfusion image prior to calibration is also provided: `perfusion_raw`.

Typing `quasil` with no options will give basic usage information.

Main options

- i <image>** The QUASAR ASL data in Nifti file format. The data order should ‘as acquired’, i.e. as blocks of TIs measured in the different phases of flow suppression.
- o <directory>** Use this to place the result in a different directory to the current working directory.
- m <image>** Use this to provide a brain mask in which data analysis should take place. If this is not set, a mask will be generated automatically from the data.

Calibration

The calibration is carried out by `asl_calib` and uses the saturation recovery of the control images. M_0 of the tissue is estimated voxelwise from fitting a saturation recovery model and from this a voxelwise estimate of M_0 of the blood is derived and applied to the estimated perfusion images. More details are given in the references. An alternative is

to calculate the M0 of CSF within a CSF mask and from this estimate a single value of M0 of arterial blood, as is done by `oxford_asl` by default. This can be achieved using `asl_calib` and the resulting M0 value applied to the `perfusion_raw` image.

Sequence parameters

For the most part the QUASAR sequence is pretty much defined by the original publication and thus you are likely to use a version where all the parameters match the default assumed by `quasil`. However, there is the option to input values if your sequence is different from the standard one.

--slicedt	Increase in inversion time with slice (default is 0.035 s)
--fa	Flip angle used in the look-locker readout (default is 35 degrees)
--lfa	Flip angle used for the ‘low flip angle’ phase (default 11.7 degrees)
--tis	Comma separated list of inversion time values (default: 0.04,0.34,0.64,0.94,1.24,1.54,1.84,2.14,2.44,2.74,3.04,3.34,3.64)

Extended options

--t1b	use this to set the value of T1 of arterial blood (1.65 s by default).
--disp	Includes the effects of label dispersion in the model (using a gamma vascular transport function)
--infertau	Also estimate the label duration from the data (rather than assuming it be fixed by the sequence).
--corrcal	Include correction for partial volume effects present around the edges of the calibration image.
--mfree	Do a ‘model-free’ rather than model based analysis of the data.

Partial Volume Correction

It is possible to perform model-based partial volume correction with QUASAR ASL using the same methodology available in BASIL for other ASL variants.

--pvcrr	Do partial volume correction, requires structural image and its segmentation results.
--fslanat	An <code>fsl_anat</code> results directory to supply <code>quasil</code> with a structural image and the results of segmentation.
--t1wm	The value of T1 to be used for white matter (default 1.1 seconds)

6.5 TOAST - quantification of Turbo-QUASAR data

6.5.1 Overview

TOAST is a pipeline designed to quantify the hemodynamic parameters of Turbo-QUASAR data. Turbo-QUASAR is a PASL based technique that improves upon the SNR of QUASAR using multiple labelling pulses while preserving the application of crushing gradients and the Look-Locker readout multi-TI acquisition in QUASAR. Each labelling pulse creates a sub-bolus, and the duration of each sub-bolus is equal to the time between each acquisition under

optimal blood flow velocity. TOAST uses full Turbo-QUASAR data to produce perfusion in absolute units as well as arterial transit time and ABV images. TOAST also include the option to correct MT effects and estimate the duration of sub-boluses.

More information on the model used can be found in:

Zhao M Y, Václavů L, Petersen E T, Biemond B J, Sokolska M J, Suzuki Y, Thomas D L, Nederveen A J, Chappell M A, (2019). *Quantification of Cerebral Perfusion and Cerebrovascular Reserve Using Turbo-QUASAR Arterial Spin Labeling MRI*. *Magnetic resonance in medicine*. <https://doi.org/10.1002/mrm.27956>

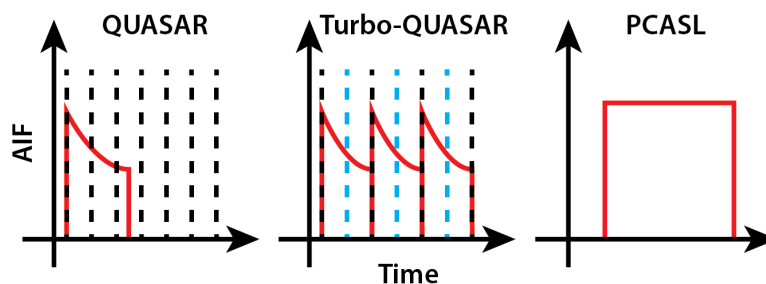
Chappell, M. A., Woolrich, M. W., Petersen, E. T., Golay, X., & Payne, S. J. (2012). *Comparing model-based and model-free analysis methods for QUASAR arterial spin labeling perfusion quantification*. *Magnetic resonance in medicine*. doi:10.1002/mrm.24372

E. T. Petersen, J. B. De Vis, C. a. T. van den Berg, and J. Hendrikse, "Turbo-QUASAR: a signal-to-noise optimal arterial spin labeling and sampling strategy," *Proc. Intl. Soc. Mag. Reson. Med.* 21 2146., vol. 60, no. 6, p. 2146, 2013.

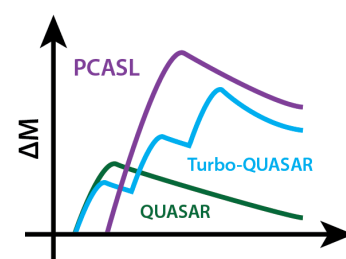
Turbo-QUASAR Description

As an improvement of the QUASAR method, Turbo-QUASAR was designed to overcome the low SNR of PASL when compared to PCASL by applying a series of labeling pulses to create a longer effective bolus duration. These labeling pulses are inserted between readout (also replacing the QUIPPS II pulse used in QUASAR) to maintain the magnetization level of the tracer for a longer period than conventional PASL techniques, thus increasing the overall SNR of the resulting ASL data. The figure below shows the different labeling techniques and the associated arterial input function for QUASAR, Turbo-QUASAR, and PCASL.

(A) QUASAR, Turbo-QUASAR, and PCASL Labeling Scheme

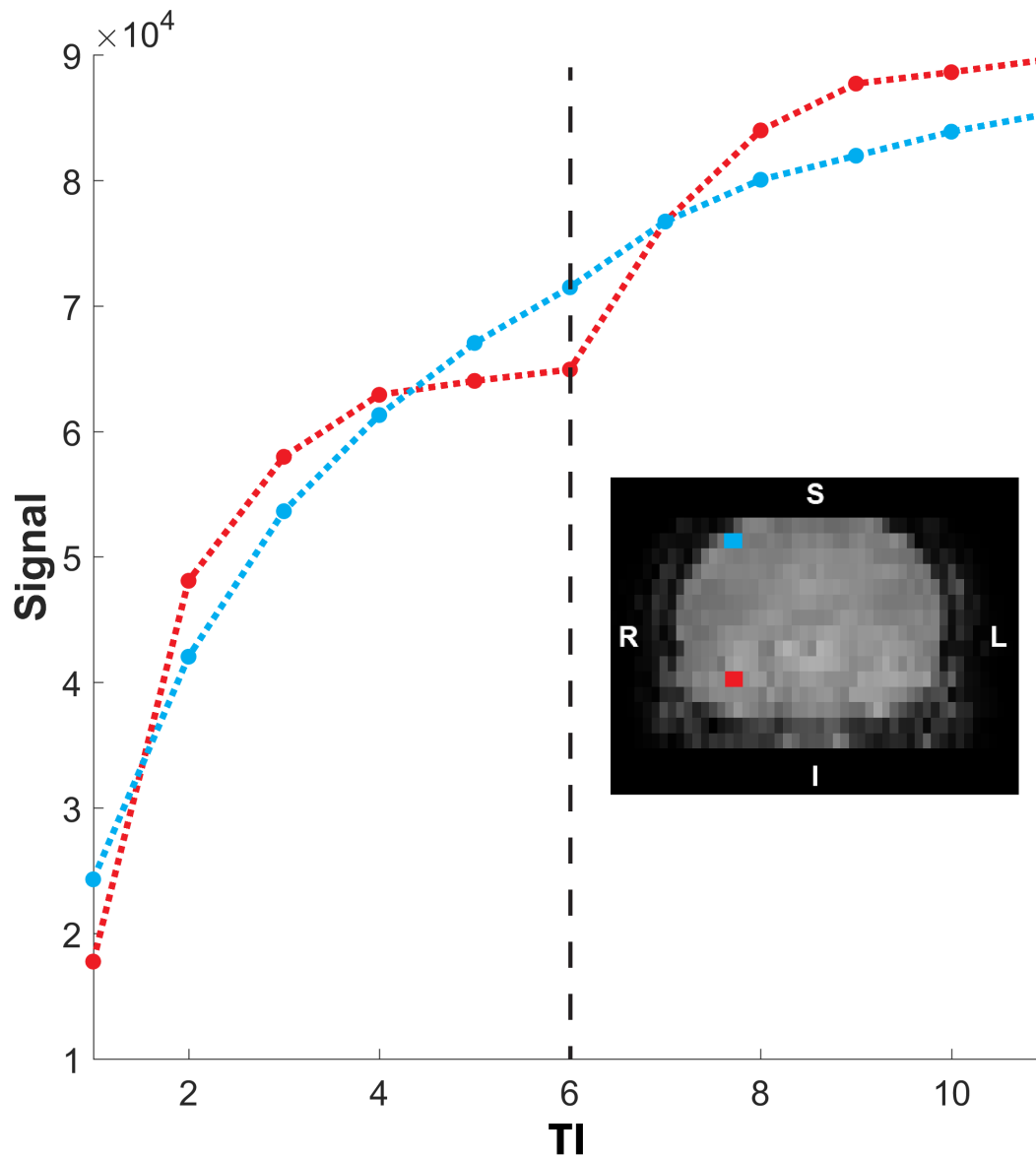


(B) Tissue Kinetic Curve



Magnetisation Transfer (MT) effects in Turbo-QUASAR

Due to the RF pulses in the label and control experiments of Turbo-QUASAR, MT effects could potentially affect the estimation accuracy of CBF. Turbo-QUASAR uses Look-Locker readout and the spins follows a saturation recovery. The figure below shows the saturation recovery curve for spins at different locations of the brain. The dash line indicates the time point that the labeling stopped. In essence, the MT effects have a stronger influence to spins that are closer to the labeling region. A correction method has been implemented in TOAST to correct the MT effects by modeling the saturation recovery signal in two parts: (1) under the influence of MT effects and (2) without the influence of MT effects. The correction is achieved by executing the `-infert1` option (default).



6.5.2 User Guide

Usage

TOAST expects the input data without label-control subtraction being performed. A typical command line usage would be:

```
toast -i <asl_data> -o <output_directory> --infert1
```

This would carry out a model-based analysis of Turbo-QUASAR data and provide voxelwise quantification of perfusion and arterial transit time. Note: the current implementation of TOAST does NOT support dispersion correction, model-free analysis, or partial volume correction.

Output files

File name	Description
ABV_absolute	Arterial blood volume (in decimals) after calibration
Arrival_time_blood	Arterial arrival time (in seconds) to macrovasculature
ATT	Arterial transit time (in seconds)
calib_M0t	M0 of tissue
CBF_absolute	CBF (in ml/100g/min) in absolute units after calibration
M0a_for_absolute_CBF	M0 of arterial blood used for calibration
mask	Mask used in the analysis
T1_tissue	T1 (in seconds) of tissue

Sequence parameters

A number of parameters are similar with QUASAR so users may wish to consult the user guide of QUASIL. Only the sequence parameters unique to Turbo-QUASAR are explained here.

--shift	Slice shifting factor to increase the effective temporal resolution. Default: 2
--break_1	Slice number of first acquisition point (start from 0). Default: 0
--break_2	Slice number of middle acquisition point (start from 0). Default: 7
--break_3	Slice number of last acquisition point (start from 0). Default: 14
--taupat	Specify the pattern of Turbo-QUASAR labeling pulses. Label: 1, no label: 0. Default: 1, 1, 1, 1, 1, 1.

Extended options

These options include the estimation of T1 of tissue, correcting the partial volume effects of the calibration data, and estimation of the arterial blood volume.

--infert1	Estimate voxelwise T1 of tissue
--calib	Include a calibration image
--tr	TR (in seconds) of the calibration image. Default: 5.0 seconds.
--struct	Include a structural image
--corrca	Include correction for partial volume effects present around the edges of the calibration image.
--inferart	Estimate voxelwise arterial blood volume (ABV or aCBV)

CHAPTER 7

References

If you employ BASIL in your research please reference the article below, plus any others that specifically relate to the analysis you have performed:

- 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.

If you employ spatial regularisation (priors) you should ideally reference this article too:

- 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.

If you fit the macrovascular (arterial) contribution you should reference this article too.

- 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.

If you employ the partial volume correction method then you should reference this article too.

- 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.

If you perform model-based analysis of QUASAR ASL data then you should reference this article too.

- Chappell, M. A., Woolrich, M. W., Petersen, E. T., Golay, X., & Payne, S. J. (2012). *Comparing model-based and model-free analysis methods for QUASAR arterial spin labeling perfusion quantification.* doi:10.1002/mrm.243