qMRLab Documentation

Release 0.1

Ilana Leppert

Feb 11, 2022

qmrlab

Welcome to qMRLab, a software for quantitative MR image analysis. Please take a look at the following for an overview of what qMRLab can do for you:

CHAPTER 1

Key features

qMRLab is an open-source software for quantitative MR image analysis. The main goal is to provide the community with an intuitive tool for data fitting, plotting, simulation and protocol optimization for a myriad of different quantitative models. The modularity of the implementation makes it easy to add any additional modules and we encourage everyone to contribute their favorite recipe for qMR!

1.1 Data simulator

The simulation interface allows end users to easily simulate qMR data and evaluate how well these models perform under known parameters input, determine the most appropriate acquisition protocol and evaluate how fitting constraints impact the results.

1.2 Data fitting and visualization

The data fitting provides a simple interface to import real-world qMR data, fit them using the selected fitting procedure, and visualize the resulting parameter maps. More advanced users could also use the command line tools used in the background by the GUI to include data fitting in their analysis scripts.

Here's a short video to get you acquainted with the package:

CHAPTER 2

Methods available

2.1 FieldMaps

2.1.1 b1_afi map: Actual Flip-Angle Imaging for B1+ mapping

2.1.2 b1_dam map: Double-Angle Method for B1+ mapping

2.1.3 b0_dem map : Dual Echo Method for B0 mapping

2.2 T2_relaxometry

2.2.1 mono_t2: Compute a monoexponential T2 map

2.2.2 mwf : Myelin Water Fraction from Multi-Exponential T2w data

2.3 Processing

2.3.1 filter_map: Applies spatial filtering (2D or 3D)

2.4 T1_relaxometry

2.4.1 mp2rage: Compute a T1 map using MP2RAGE

2.4.2 vfa_t1: Compute a T1 map using Variable Flip Angle

2.4.3 mtv : Macromolecular Tissue Volume in brain

2.4.4 inversion_recovery: Compute a T1 map using Inversion Recovery data

2.5 Magnetization_transfer

2.5.1 qmt_bssfp : qMT using Balanced Steady State Free Precession acquisition

2.5.2 qmt_spgr: quantitative Magnetizatoion Transfer (qMT) using Spoiled Gradient Echo

2.5.3 mt_sat : Correction of Magnetization transfer for RF inhomogeneities and T1

2.5.4 mt_ratio : Magnetization transfer ratio (MTR)

2.5.5 qmt_sirfse: qMT using Inversion Recovery Fast Spin Echo acquisition

2.6 Diffusion

2.6.1 dti: Compute a tensor from diffusion data

2.6.2 amico: Accelerated Microstructure Imaging via Convex Optimization

2.6.3 noddi: Neurite Orientation Dispersion and Density Imaging

2.6.4 charmed: Composite Hindered and Restricted Model for Diffusion

2.7 QSM

2.7.1 qsm_sb: Fast quantitative susceptibility mapping

2.8 UnderDevelopment

- Name your Model

2.9 Noise

2.9.1 denoising_mppca : 4d image denoising and noise map estimation

2.9.2 noise_level : Noise histogram fitting within a noise mask

CHAPTER 3

Protocols

This section addresses some frequently asked questions regarding acquisition protocols and how to set them in qMRLab.

3.1 mp2rage

You can visit our MP2RAGE blog post to find out about the basics of the `MP2RAGE` method.

3.1.1 `NumberOfShots`

This section of the `Protocol` panel expects two `NumberOfShots` values: `Pre` and `Post`.

In the original implementation, the `NumberOfShots` is referred as `nZSlices`, and `before/after` designates the number of segments before and after the center of k-space.

To calculate these values, you need to know the values of `Slices Per Slab` **(NSlices)** and `Slice Partial Fourier` **(PF)** parameters. To calculate `Pre` and `Post`:

Pre Formula:

```
Pre  = NSlices(PF - 0.5)
```

Post Formula:

```
Post = NSlices/2
```

3.1.2 `Repetition Times`

Inversion (Inv) `RepetitionTime` (in `DICOM`) between two inversion pulses in seconds. In the (Siemens) `PDF` it is described as `TR` under the **Contrast - Common** section. In **BIDS**, this corresponds to the `RepetitionTimePreparation` metadata field. Typically between 3 - 6 seconds.

Excitation (Inv) The repetition time between two excitation pulses in each GRE readout block. In the (Siemens) `PDF` it is described as `Echo spacing` (usually) under the **Sequence - Part 1** section. In **BIDS**, this corresponds to the `RepetitionTimeExcitation` metadata field. Typically around 0.006 - 0.008 seconds.

> **Warning:** Some parameters are not included in the `DICOM` header by Siemens, such as the `RepetitionTimeExcitation`. Nonetheless, can be accessed in the protocols exported as `PDF`.

3.1.3 Inversion Times

Inversion times belonging to each GRE readout block. In the (Siemens) protocol `PDF`, these values can be found under the **Contrast - Common** section with the names of `TI 1` and `TI 2`.

3.1.4 Flip Angles

Excitation flip angles in each GRE readout block. In the (Siemens) protocol `PDF`, these values can be found under the **Contrast - Common** section with the names of `Flip angle 1` and `Flip angle 2`.

CHAPTER 4

User preferences

You can customize certain qMRLab features according to your needs by modifying the `usr/preferences.json` file.

4.1 1. Default method selection

Note: Takes effect only in MATLAB with GUI.

The `GUIDefault` field determines which qMRLab module will be shown first in the method selection dropdown by default. For example:

```
"GUIDefault": {
    "Method": "vfa_t1"
}
```

4.2 2. Parallelization settings

4.2.1 `FitParallelWheneverPossible` selection

Note: Takes effect only in MATLAB if the parallel computing toolbox is available. The number of workers depends on the computational resources, as well as the MATLAB preferences.

This field configures qMRLab to execute fitting using multiple CPU cores when the `Fit Data` button is clicked in the GUI or when the `FitBIDS` command is issued.

```
"FitParallelWheneverPossible": true,
```

If you would like to use parallelization only for certain (voxelwise) methods, you can use *ParFitData* function. For details:

```
help ParFitData
```

4.2.2 `ParFitData` configurations

Configures the default behavior of *ParFitData* function.

Note: Takes effect only in MATLAB if the parallel computing toolbox is available. Only the voxelwise models (e.g., `inversion_recovery`) are affected.

```
"ParFitData": {
              "AutosaveEnabled": 1,
              "AutosaveInterval": 5,
              "Granularity": 3,
              "RemoveTmpOnSuccess": 1
    }
```

Each option is explained below.

- **`AutosaveEnabled` [on 1, off 0, default 1]**
  - Save partial results when a batch is done processing or when the `AutosaveInterval` is reached.
- **`AutosaveInterval` [min 1, default 5]**
  - If `AutosaveEnabled` is set to 1, this option determines the duration (in minutes) at which the outputs will be saved. Each CPU worker times its own process.
- **`Granularity` [min 2, max 5, default 3]**
  - Determines how many data chunks will be created to parallelize the fitting.
  - `nChunks = nCores X Granularity.`
- **`RemoveTmpOnSuccess` [min 2, max 5, default 3]**
  - Determines whether to remove `ParFitTempResults` folder after the execution finished successfully. For further details about `ParFitData`:

```
help ParFitData
```

4.3 3. Unit configurations

qMRLab aims at collecting qMRI implementations from different labs under one umbrella. We keep the implementations as close as possible to their original form and avoid enforcing a particular scaling.

By configuring `preferences.json`, it is possible to unify the units across all qMRLab models. This high-level configuration framework gives users the flexibility to work with units they prefer and keep implementations closer to they were originally developed. The units are handled for 3 main cases:

- **Input protocol units:** Defines the units for the acquisition parameters (e.g., `RepetitionTime` in seconds or milliseconds).

- **Output map units:** Defines the units for the quantitative maps calculated by qMRLab (e.g., `T1 map` in seconds or milliseconds).
- **Input map units:** Some quantitative maps are provided as inputs to certain models. For example, a `B1+ map` can be provided for `vfa_t1`, or a `T1 map` is expected by the `mvf`. This category defines in which units does qMRLab expect a qMRI map as an input.

Keywords for defining units in qMRLab

- **Time**
  - `microsecond`
  - `millisecond`
  - `second`
  - `minute`
  - `hour`
- **Rate**
  - `reciprocal_microsecond`
  - `reciprocal_millisecond`
  - `reciprocal_second`
- **Fraction**
  - `fraction_decimal` (from 0 to 1)
  - `fraction_percent` (from 0 to 100)
- **B1**
  - `relative_scaling_factor_decimal` (-inf, 1=ideal, +inf)
  - `relative_scaling_factor_percent` (-inf, 100=ideal, +inf)
- **B0**
  - `offset_factor_hertz` (-inf, 0Hz=ideal, +inf]
- **Susceptibility**
  - `part_per_million`
- **Angle**
  - `radian`
  - `degree`
- **Diffusivity**
  - `square_meter_per_second`
  - `square_micrometer_per_millisecond`
- **Arbitrary**
  - `arbitrary` (not scaled)
- **Categorical**
  - `categorical` (not scaled)

- **Length**
  - millimeter
  - micrometer
- **Tensor**
  - tensor

Warning: To configure units in the `preferences.json`, the units **MUST** be set using the unit keywords above. For example, to set `Time` unit to ms, the correct keyword is `millisecond`, not `ms` or `milliseconds`.

Unit changes take effect in both MATLAB and Octave. To see the unit changes in the GUI, you need to restart qMRLab after modifying the `preferences.json` file. The following dropdowns explain unit selections available in the `usr/preferences.json` file.

`ForAllUnitsUseBIDS`

If this setting is `true` then:

1. Output maps will be generated in BIDS units (for example, `second` for relaxometry outputs, or `fraction_percent` for fraction maps)
2. Protocol values will be expected to be in BIDS units (for example,‘‘second‘‘ for `RepetitionTime`)
3. If a method inputs a fieldmap or quantitative map, they are expected to follow BIDS units (for example, `relative_scaling_factor_percent` for B1+ maps)

```
"ForAllUnitsUseBIDS": true,
```

Warning: Setting this to `true` will override `UnifyOutputMapUnits`, `UnifyInputProtocolUnits` and `ChangeProvidedInputMapUnits`.

`UnifyOutputMapUnits`

If `"Enabled": true`, then the units defined for each unit category will determine output units of the quantitative maps generated by qMRLab.

```
"UnifyOutputMapUnits": {
   "Enabled": true,
   "Time": "second",
   ...
     }
```

For example, if you set `"Time": "millisecond",` as shown above, all the time-relevant output maps (e.g., `T1map`, `T2map`) will be saved in milliseconds.

For available units and unit categories, please see the list above.

Warning: Overriden by `ForAllUnitsUseBIDS (true)` user setting or `setenv('ISBIDS','1')` environment variable.

`UnifyInputProtocolUnits`

If `"Enabled": true`, then the units defined for each unit category will determine in which units are the inputs protocols expected.

```
"UnifyInputProtocolUnits ": {
   "Enabled": true,
   "Time": "second",
   ...
   }
```

For example, if you set `"Time": "millisecond",` as shown above, all the time-relevant protocols (e.g., `EchoTime`, `InversionTime`) will be expected in milliseconds.

Activating this setting will change the protocol field labels in GUI and the units shown in demo scripts generated by the `qMRGenBatch` function.

For available units and unit categories, please see the list above.

Warning: Overriden by `ForAllUnitsUseBIDS (true)` user setting or `setenv('ISBIDS','1')` environment variable.

`ChangeProvidedInputMapUnits`

If `"Enabled": true`, then the units defined for each unit category will determine in which units are the input maps expected (e.g. `B1+map` or `R1map`).

```
"ChangeProvidedInputMapUnits": {
   "Enabled": false,
   "Time": "second",
   "B1": "relative_scaling_factor_decimal",
   ...
   }
```

For example, if you set `B1` as shown above, qMRLab will assume that the B1+ maps you provide are normalized such that `1` indicates `actual` = `nominal` flip angle. Values smaller than `1` will scale down the actual Flip Angle, and vice versa. Or, a `T1map` that is input to a qMRLab model (e.g., *mvf*) will be expected in the unit of seconds.

For available units and unit categories, please see the list above.

Warning: Overriden by `ForAllUnitsUseBIDS (true)` user setting or `setenv('ISBIDS','1')` environment variable.

Note: qMRLab will use the units provided by original implementations when all the following settings are dis-

abled: `UnifyOutputMapUnits`, `UnifyInputProtocolUnits`, `ChangeProvidedInputMapUnits` and `ForAllUnitsUseBIDS`.

In this case, different models may operate in different units. For example, for `inversion_recovery` it is `milliseconds`, whereas for `vfa_t1` it is seconds.

CHAPTER 5

How to install

5.1 qMRLab in MATLAB

- Minimum required version: *R2014b*
- **Required MATLAB products:**
  - Image Processing Toolbox
  - Optimization Toolbox
- **Available interfaces:**
  - Graphical User interface (GUI)
  - Command Line Interface (CLI)

If you have a MATLAB license, you can easily start using qMRLab:

1. **Download the latest stable release here.**
   - Scroll down to the page, download links are under the *Assets* tab.
2. Extract the downloaded *.zip* or *.tar.gz* file to a directory of your choice.

3. Open MATLAB point your current directory to the *qMRLab* folder.
4. Ensure that you can see *qMRLab.m* in the list of files of your current directory.
5. Execute the following command in your *Command Window* to initialize qMRLab's default environment:

```
startup
```

6.1. If you would like to use **GUI**, execute the following command in your *Command Window*:

```
qMRLab
```

- To see the beginners guide for **GUI** please refer to Beginners example with GUI

6.2. If you would like to use **CLI**, ensure that the environment is ready to use by executing the following command in your *Command Window*:

```
qMRLabVer
```

- To see the beginners guide to **CLI** please refer to Beginners example with batch

5.2 qMRLab in Octave

- Minimum required version: *4.2.0*
- **Available interfaces:**
  - Command Line Interface (CLI)
- If you don't have Octave installed, you can find the instructions here.

GNU Octave is the free clone software for MATLAB. You can use nearly all the qMRLab methods in Octave via **CLI**:

1. **Download the latest stable release here.**
   - Scroll down to the page, download links are under the *Assets* tab.
2. Extract the downloaded *.zip* or *.tar.gz* file to a directory of your choice.
3. To initialize qMRLab for **CLI** use in Octave, open your terminal:

```
cd ../directory/where/you/extracted/qMRLab
octave
startup
```

- **If you are running qMRLab in Octave for the first time, qMRLab will attempt to install the following packages (if not alre**
  - *struct*
  - *optim*
  - *io*
  - *statistics*
  - *image*

4. Ensure that qMRLab is ready to use by executing the following command in your *Command Window*:

```
qMRLabVer
```

- To see the beginners guide to **CLI** please refer to Beginners example with batch

5.3 qMRLab in Docker

Beginning from the release *v2.3.0*, qMRLab offers different flavors of Docker images:

- **qmrlab/mcrgui**
  - Use qMRLab **GUI** without MATLAB license! Instructions are available elsewhere.
- **qmrlab/octjn**
  - Use qMRLab in **Jupyter Notebooks**! This image comes with **SoS Kernel** and some cool visualization libraries in **Python**, allowing you to combine qMRI processing with qMRLab in Octave and interactive visualization, all in the same notebook. Instructions are available elsewhere.
- **qmrlab/minimal**
  - Barebones qMRLab in Octave (i.e *qmrlab/octjn* without *jupyter*). Instructions are available elsewhere.

Docker image tags are coherent with qMRLab release versions. For example, if you would like to get Docker image with qMRLab *v2.4.0* installed:

```
docker pull qmrlab/octjn:v2.4.0
```

Docker images are built and published automatically by qMRLab's Azure Release Pipelines.

5.4 How to cite?

If you use qMRLab in you work, please cite:

Karakuzu, A., Boudreau, M., Duval, T., Boshkovski, T., Leppert, I. R., Cabana, J. F., ... & Stikov, N. (2020). qMRLab: Quantitative MRI analysis, under one umbrella. Journal of Open Source Software, 5(53), 2343. doi: 10.21105/joss.02343

Please also cite the reference for the particular module you are using (specified in each model's page).

CHAPTER 6

Known issues

6.1 OSX

Due to security updates in OSX release 10.15 (macOS Catalina), the *Gatekeeper* may flag **.mex** files in qMRLab as unsafe and quarantine them. In this case, you won't be able to start qMRLab and the following error will appear:

> **Warning: filename.mexmaci64 cannot be opened because the developer cannot be verified.** macOS cannot verify that this app is free from malware.

To lift the quarantine and to add Gatekeeper exceptions for the qMRLab's *mex* files, start a terminal and run the following:

```
sudo xattr -r -d com.apple.quarantine /full/path/to/qMRLab
sudo find /full/path/to/qMRLab -name \*.mexmaci64 -exec spctl --add {} \;
```

Please open a new issue if this solution did not work for you.

6.2 Apple Silicon (M1) compatibility

- **MATLAB**
  - According to the official Mathworks announcement, MATLAB is fully compatible with M1 CPU from R2020b onward. On the other hand, we successfully run qMRLab in R2018b (MacBook Pro, 13", Early 2020, M1).
- **Octave**
  - The latest Octave release 6.2.0 works with M1 CPU.

CHAPTER 7

Beginner's example with GUI

This will guide you step by step in processing a sample dataset with the user interface.

7.1 1. Open matlab

7.2 2. Run startup

Go to qMRLab installation folder (note: replace "PATH_QMRLAB" with the actual path)

```
cd PATH_QMRLAB
```

Setup your path (note: you only need to do it once)

```
startup
```

7.3 3. Launch the GUI

Type this is matlab to launch the main GUI:

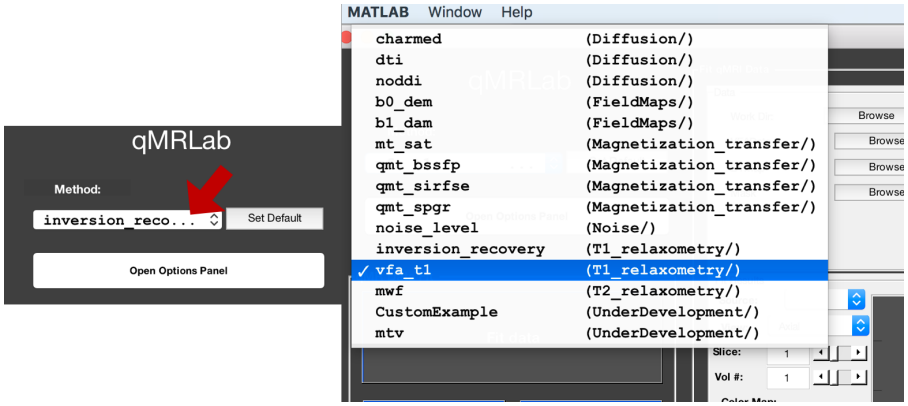
```
qMRLab
```

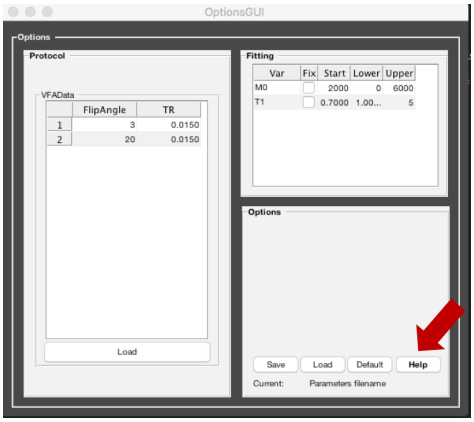
7.4 4. Model selection

On the top left-hand side pull-down menu, select the method we are going to use, in this case, *vfa_t1 (T1_relaxometry/)* :

At any time, you may use the *Help* button in the *Options* panel to get a description of the model:

For the list of available models, please check *Methods available*

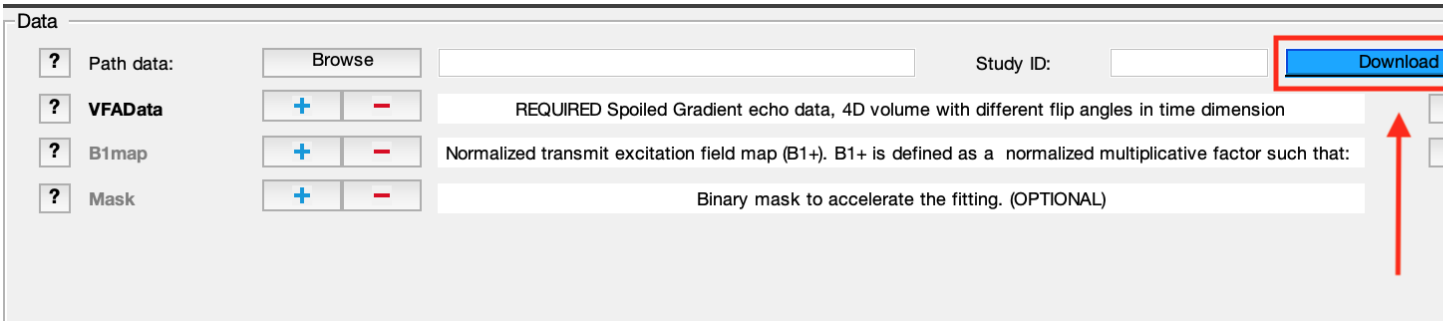




7.5 5. Download example data

In this case we will be working with Variable Flip Angle Data to compute a T1map. The main input data is stored as a 4D volume, where the 4th dimension is different flip angles. For example, in this test dataset, 1 slice at 2 different flip angles were acquired: volume 1 was FA=3degrees and volumes 2 was FA=20degrees, such that *VFAData.nii.gz* is 128x128x1x2. The other optional inputs are a *Mask.nii.gz* and a *B1Map.nii.gz*.

Click *Download Data* (blue) button located at the upper right corner of the data panel:



You will be prompted for a directory where the example dataset will be saved. After the dataset has been downloaded, qMRLab will automatically set *Path data* to the download directory and load the input files.

Note: **Please note that** such auto-loading takes effect only if the name of the images (e.g. VFAData.nii.gz) in the Path data directory are identical to that of the data fields (e.g., VFAData) listed in the data panel. There are no filename

assumptions for the user data. Any file name is acceptable as long as the data format and the data dimensions are in agreement with the method's inputs.

Warning: The default 'Protocol' parameters are intended for the example datasets. Users are required to update these values with regards to their data.

For a more detailed description about input data formats, please refer to *4.1 Data format*

7.6 6. View the data

You can look at your data by clicking *View* next to the name of the file:

You can browse through slices or volumes of the data files by using arrows and mouse controls. Please refer to *3. Data Viewer*.

7.7 7. Set up the protocol

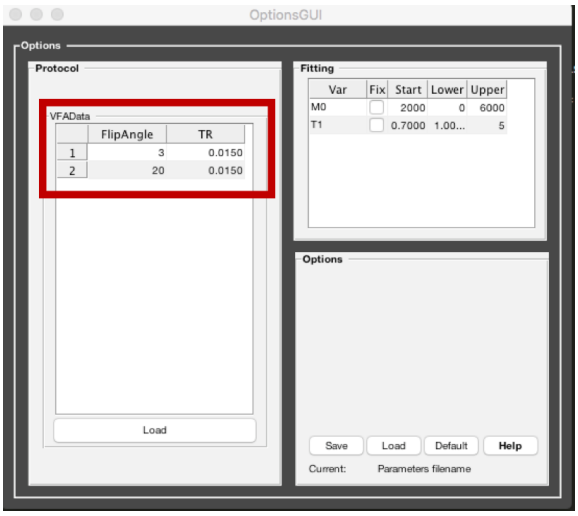
For this dataset, the protocol will be set up by default with the flip angles and TRs:

For your own acquisition, you will have to use an external txt file to load the parameters, please refer to *6.1 Protocol*.

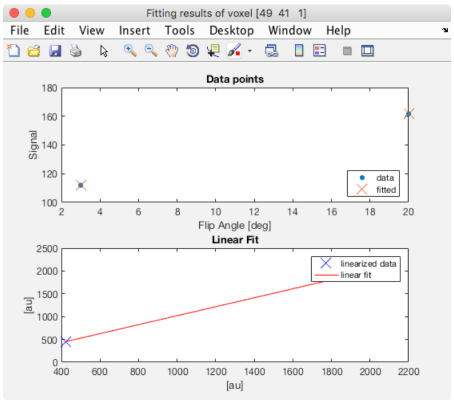
7.8 8. View the data fit in 1 voxel

Before fitting the whole volume, it's a good idea to take a look at your data and how it fits the model. Here, we can visualize the fit in 1 voxel at a time. In the *Cursor* section, press *Select*. Then select a voxel in the image and the press *View data fit*:

A new window will pop-up with the results of the fit in that voxel:

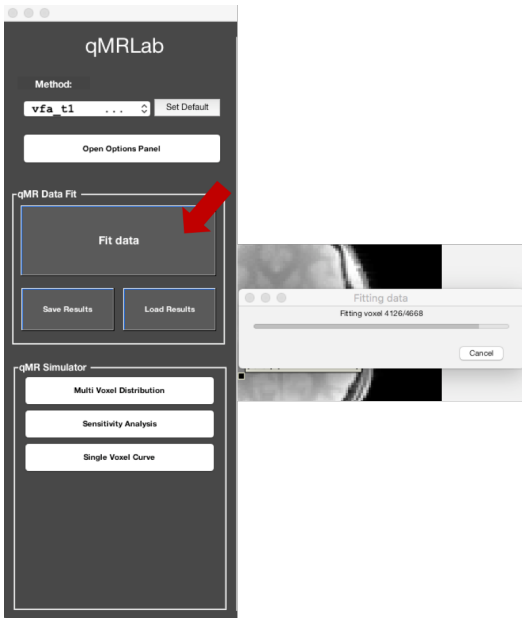

OptionsGUI
Options
Protocol
VFAData
FlipAngle TR
1 3 0.0150
2 20 0.0150
Load
Fitting
Var Fix Start Lower Upper
M0 2000 0 6000
T1 0.7000 1.00... 5
Options
Save Load Default Help
Current: Parameters filename


Source: VFA...
View: Axial
Slice: 1
Vol #: 1
Color Map:
gray
min: 0.213349
max: 206.753
Auto
Open viewer
Save figure
Sanity Check
Histogram
View data fit
Cursor
Pan
Zoom
Select
[X,Y]: [49 88]
Index: 111.9
[R,G,B]: [0.5373 0.5373 0.5373]
1
2
3
60 80 100 120 140

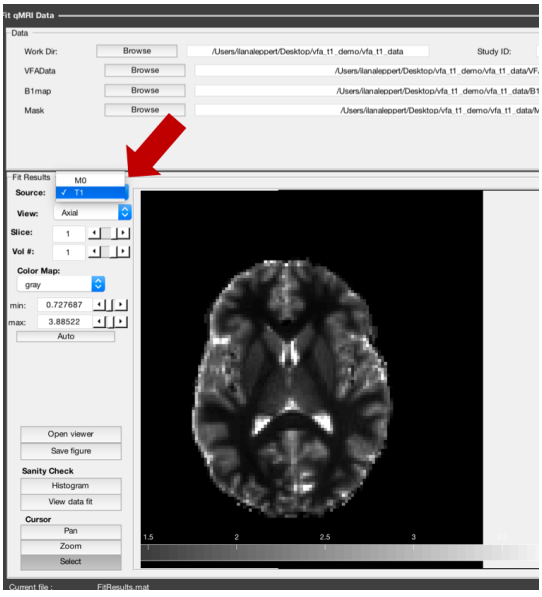

Fitting results of voxel [49 41 1]
File Edit View Insert Tools Desktop Window Help
Data points
Signal
Flip Angle [deg]
data
fitted
Linear Fit
linearized data
linear fit
[au]
[au]

7.9 9. Fit the whole dataset

We can now fit the whole volume by pressing the large *Fit Data* button.



A wait bar will appear while the data is being processed and will automatically when done. From the pull-down menu to the left of the image, it's possible to select the output you would like to view. For example, the T1map:



For more information and to explore other functionality such as the simulations, please visit Graphical User Interface Usage.

CHAPTER 8

Beginner's example with CLI

This will guide you through the steps to create a script that can be used to process sample data. The generated example script scan then be used as a guideline to analyze your own data.

8.1 1. Open MATLAB and setup path

Open MATLAB, then go to qMRLab installation folder (note: replace "PATH_QMRLAB" with the actual path)

```
cd PATH_QMRLAB
```

Setup your path (note: you only need to do it once)

```
startup
```

8.2 2. Generate a batch example

To get familiar with the command-line usage, you can automatically generate an example (with sample data) for your model of choice. For example, for *inversion_recovery* type:

```
model=inversion_recovery; % create an instance of the model
qMRgenBatch(model)
```

When prompted, select the folder where you want to download the data and save the batch example file.

8.3 3. Run the batch

You can run the example directly:

```
inversion_recovery_batch
```

or take a look at the sections independently:

```
edit inversion_recovery_batch
```

Please refer to *inversion_recovery: Compute a T1 map using Inversion Recovery data* to see a batch example and the expected output.

Please refer to Command-Line Usage for a more detailed description of the available functions

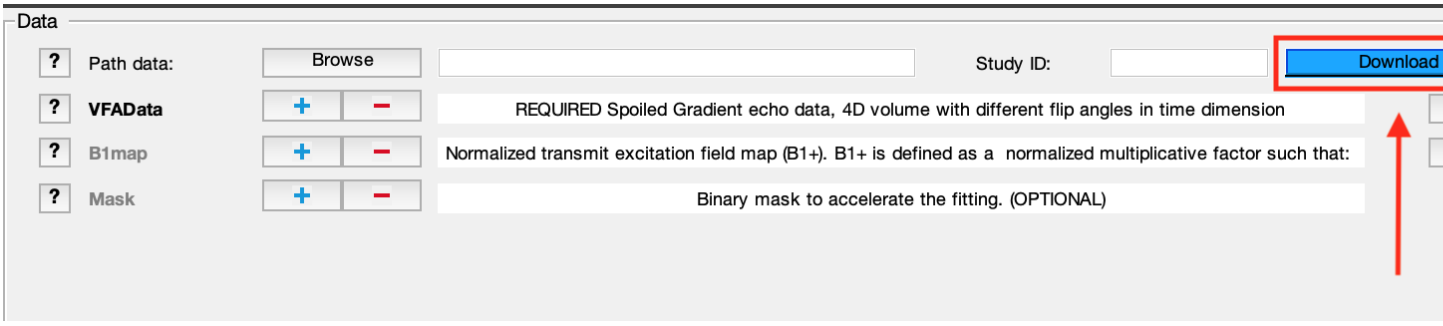
CHAPTER 9

Example datasets

Every qMRLab model comes with an example dataset stored in our public OSF repository. These datasets can be directly downloaded from the OSF, or using qMRLab's interfaces.

9.1 Download example datasets via GUI

After selecting a model from the dropdown menu, you can use *Download Data* (blue) button located at the upper right corner of the data panel to download the respective dataset:



You will be prompted for a directory where the example dataset will be saved. After the dataset has been downloaded, qMRLab will automatically set *Path data* to the download directory and load the input files.

Note: **Please note that** such auto-loading takes effect only if the name of the images (e.g. VFAData.nii.gz) in the Path data directory are identical to that of the data fields (e.g., VFAData) listed in the data panel. There are no filename

assumptions for the user data. Any file name is acceptable as long as the data format and the data dimensions are in agreement with the method's inputs.

Warning: The default 'Protocol' parameters are intended for the example datasets. Users are required to update these values with regards to their data.

Step by step instructions to get started with GUI are available at *Beginner's example with GUI*.

9.2 Download example datasets via CLI

qMRlab can automatically generate an m-script (*model_name_batch.m*) or a Jupyter Notebook (*model_name_batch.ipynb*)for every model and download corresponding datasets for fitting by *qMRGenBatch* and *qMRGenJNB* functions, respectively.

Step by step instructions to get started with CLI are available at *Beginner's example with CLI*.

Note: You can execute Jupyter Notebooks online by clicking the Binder badge located at the top of the respective method's documentation. page. See an example for variable flip angle T1 mapping.

If you would like to download the example dataset without generating scripts

```
Model = model_name;
path = 'directory/to/download/the/data/';
downloadData(Model, path);
```

CHAPTER 10

Interfaces

10.1 Graphical User Interface (GUI)

This section describes the various features and functionality of the user interface.

10.1.1 1. Startup

Launch matlab from the qMRLab folder. Then load the default setup by typing:

```
startup
```

Then open the GUI by typing:

```
qMRLab
```

10.1.2 2. Layout

When you first launch qMRLab, you will be presented with a blank interface. The interface consists of three columns, or panels. On the left, you have the *Menu* panel, in the center you have the *Main panel*, and on the right, in a separate floating window, you have the *Options* panel.

2.1 Menu Panel

The `Menu panel` is where you can choose the task you want to perform. It is divided in three sections: `Method`, `qMR Data Fit` and `qMR Data Simulator`.

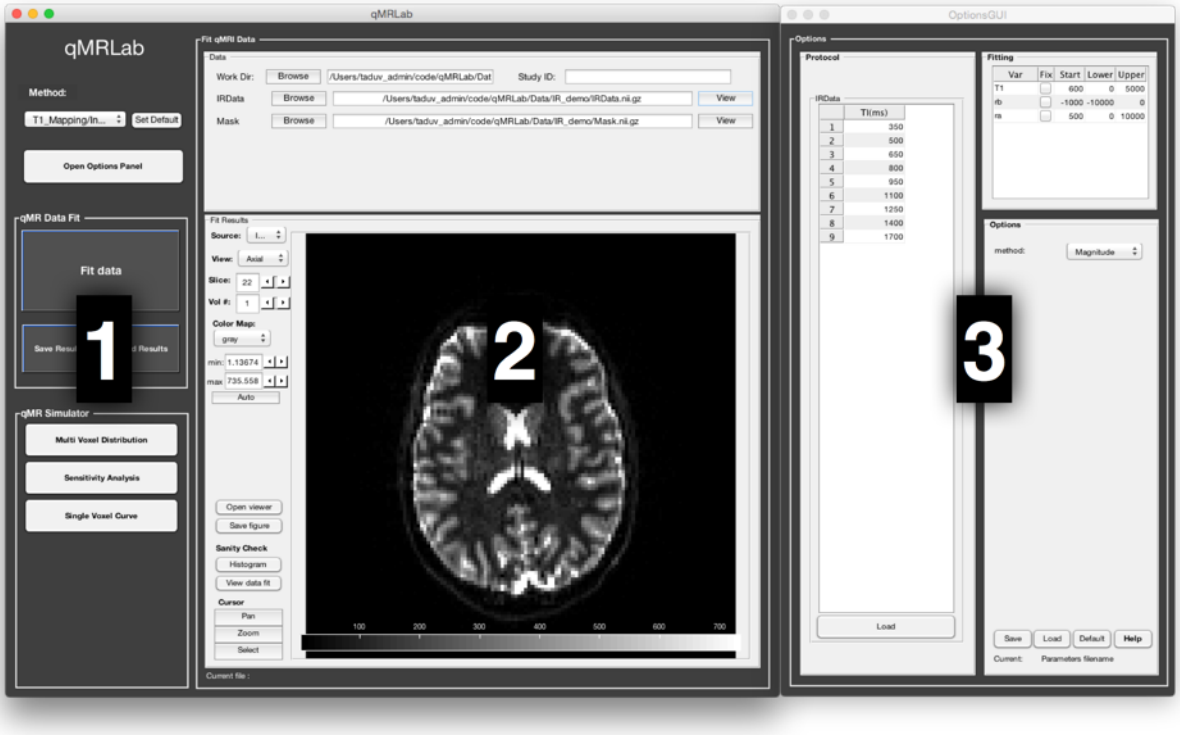


Fig. 1: Default interface with 3 panels: (1) `Menu`, (2) `Main` and (3) floating `Options`

2.1.1 Method

At the top, you will find a drop-down menu where you can choose the MR acquisition method that you want to be working with. An updated list of the available methods is here: *Methods available*

Note that the `Main` and `Options` panels will update to the appropriate window according to your selection. If you plan to be working mainly with a particular method, select it from the drop-down menu first, and the click on the `Set Default` button next to it. Next time you open qMRLab, your preferred method will be selected by default.

Clicking on the `Open Options Panel` below the method drop-down menu will open the *Options* panel window and set its position on the right side of the `Main` panel. This is useful to bring back the `Options` panel window to the front if it's hidden behind another window, to reset its position if you have resized the windows, or to reopen it in case you closed it.

2.1.2 qMR Data Fit

Click on the big `Fit Data` button only when you have selected your data files, set up your protocol and fitting options and are ready to begin the fitting process, which, depending on the size of your data and the method, can take from a few minutes to a couple of hours. The `Save Results` button will prompt you to save a .mat file with the results of your data fit. `Load Results` will load previously saved results and display them. Refer to this section *4. Data Fitting* for more information.

2.1.3 qMR Data Simulator

The buttons of this menu allow you to choose between different data simulation mode. All Methods that involve a data fitting procedure present at least the following simulations: `Single Voxel Curve`, `Sensitivity Analysis` and `Multi Voxel Distribution`. Clicking on any one of these buttons will bring the corresponding interface in a floating window. When any of these interfaces are opened, clicking on the `update` button will launch the simulation using the current options (defined in the `Options` panel). The `Save Results` button will prompt you to save a .mat file with the current simulation results. `Load Results` will load previously saved simulation results and display them in the appropriate panel. Refer to this section *5. Simulation* for more information.

Single Voxel Curve

The `Single Voxel Curve` panel is a simple interface to simulate MR data from a single voxel, using the defined MR parameters and protocol. It is the fastest way to evaluate various acquisition protocols, the performance of the model and fitting options. Refer to section *5.1. Single Voxel Curve* for more information.

Sensitivity Analysis

The `Sensitivity Analysis` simulation allows you to systematically vary one MR parameter, over a defined range and number of points, while keeping the others fixed. For each simulated data point, noise is added with a given SNR, and the fit is run multiple times while adding Gaussian noise. This allows you to evaluate the variance of the fit at each point. When the simulation is done, a plot shows any variable input parameters as the independent variable, as well as the mean values and variance of any fitted parameters. Refer to section *5.2. Sensitivity Analysis* for more information.

Multi Voxel Distribution

The `Multi Voxel Distribution` is a tool to simulate any number of voxels, where any parameters combination are allowed to be varied simultaneously. You can choose how many voxels to simulate and which parameters are to be normally distributed, with its mean value and variance. The results can be displayed in a number of ways such as distribution histograms, scatter plots of input vs fitted parameters, error histograms, etc. Refer to section *5.3. Multi Voxel Distribution* for more information.

2.2 Main Panel

The `Main` panel is where you can load your data files for fitting and for viewing the resulting parameters maps. This panel changes correspondingly to the Method selected in the *Menu* panel.

2.3 Options Panel

This is where you can set up all the parameters that are related to the simulation, the fitting and the protocol. The `Options` panel is displayed in a separate window than the `Menu` panel or `Main` panel. This is because each qMR acquisition method has its own particular options, and this window needs to be changed correspondingly. It can also be closed at any time, if it is not currently needed, to provide for a simpler interface. The *Options* panel consists of three sub-panels: `Protocol`, `Fitting`, and `Options`. At the bottom of all these sub-panels you will find buttons to `Reset` the changes you made, `Save` the current settings as a .mat file, `Load` a .mat file of settings, or go back to the `Default` settings. Refer to section *6. Options Panel* for more information.

2.3.1 Protocol

Here you define the `acquisition protocol` that you wish to use for simulation, or in the case of data fitting, the protocol you used to acquire the data. See *6.1 Protocol* for more information.

2.3.2 Fitting

This is where you set up your `fitting options`. The fit parameters table lists all the variables that are available for fitting in the current method, a tick box to select which variables are to be held fixed, a starting value and lower/upper bounds. Note that some methods do not have fitting procedures, this table is empty in this case. Depending on the method, additional options may be available. See *6.2. Fitting* for more information.

2.3.3 Options

This is where you set up all the options related to the simulations. Depending on the qMR method, different sets of options are available (e.g. the fitting procedure, assumptions of the model, etc). More info in *6.3. Options*.

10.1.3 3. Data Viewer

The viewer allows you to navigate through your (up tp) `5D` dataset easily using arrows:

Use mouse controls to display your volume:

Middle (shift+) Click and drag Zoom in/out

Left Click and drag Contrast/Brightness

Right (ctrl+) Click and drag Pan

Scroll wheel Change slice

The viewer provides ROI tools to create and modify a multi-label mask that is overlaid on the image. The `Mask` in the file browser is loaded automatically, you can delete it using right click on the button `label 1`. The mask can be hidden/shown using the checkbox on the top (or use spacebar)... make sure the mask is toggle on when you draw it! The mask can have 5 different labels (1-5). Select the label on which you want to operate. One voxel can be attributed only one label (no overlaps between labels). The locker button prevents any modification and overwrite to labels that are not selected. ROI tools such as square or polygon can be converted to mask (right click on an ROI object) and conversely (mask2poly button). Statistics on each label (e.g. volume, mean intensity) can be obtained with mouse over the label number.

Brush tool controls:

Middle click and drag Change brush diameter

Right click erase

double click (smart brush only) Toggle between bright or dark segmentation

Polygon tool controls:

Click on a line add a button

Double click on a point toggle between a line and a curve

Middle click on a point delete this intermediary point

10.1.4 4. Data Fitting

qMRLab provides a convenient interface to fit your data and visualize the parameters maps. To ensure that the results are successful, you'll need to define the appropriate protocol, as it was used for data acquisition, and to format your qMR data in the way qMRLab expects it to be.

4.1 Data format

Currently supported file types are .mat and .nii files. Your files should respect the following:

- For .mat files, the name of the file can be anything, but the array it contains should be named appropriately. The list of input names is case sensitive and is specified at the top of the `Data` dialog box. For example, for a qMT SPGR experiment, `MTdata` (for the actual MT data array) or *R1map* / *B1map* / *B0map* / *Mask* respectively for a R1 / B1 / B0 or Mask file.

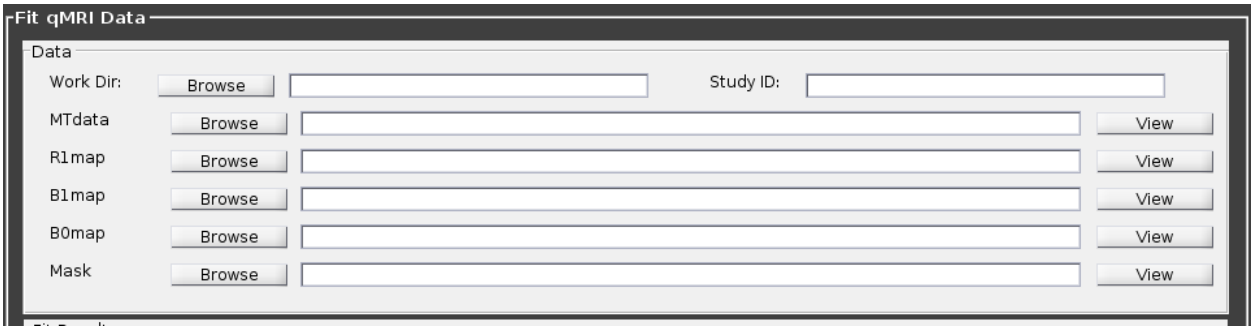


Fig. 2: Example of list of inputs for qMT SPGR experiment

- Each model will expect a different format of inputs, but in general, for single slice (2D) imaging, the main data is a 3D array with size [nx, ny, ndata], where nx/ny is the number of voxels in the x/y direction, and ndata is the number of data points for each voxel. For volume imaging (3D), data is a 4D array with size [nx, ny, nz, ndata], where nx/ny/ndata are as above, and nz is the number of voxels (or slices) in the z direction (e.g. in this example *MTdata* would have several datapoints per voxel).
- Other files (e.g. in this example *R1map / B1map / B0map / Mask*) are formatted as [nx, ny, nz].

For a more detailed description of the format required for each input, type this in the maltab window:

```
help Modelname
```

where `Modelname` is name of the available models (e.g. in this example *help SPGR*). Alternatively, in the `Options` panel, you can press on the `Help` button

4.2 Fitting Procedure

4.2.1 Single voxel

This is useful when you want to preview the fit of a single voxel (note this option is only available on voxelwise computations)

1. Select the acquisition method of your data using the 'Method' drop-down menu in the `Menu` panel.
2. In the `Menu` panel, in the `Fit qMR data` panel, enter your study ID in the `Study ID` box (optional).
3. Load your data by clicking the browse button beside the `Data:` line, or enter the full file path to it in the textbox.
4. You can view any of these maps by clicking its `View` button.

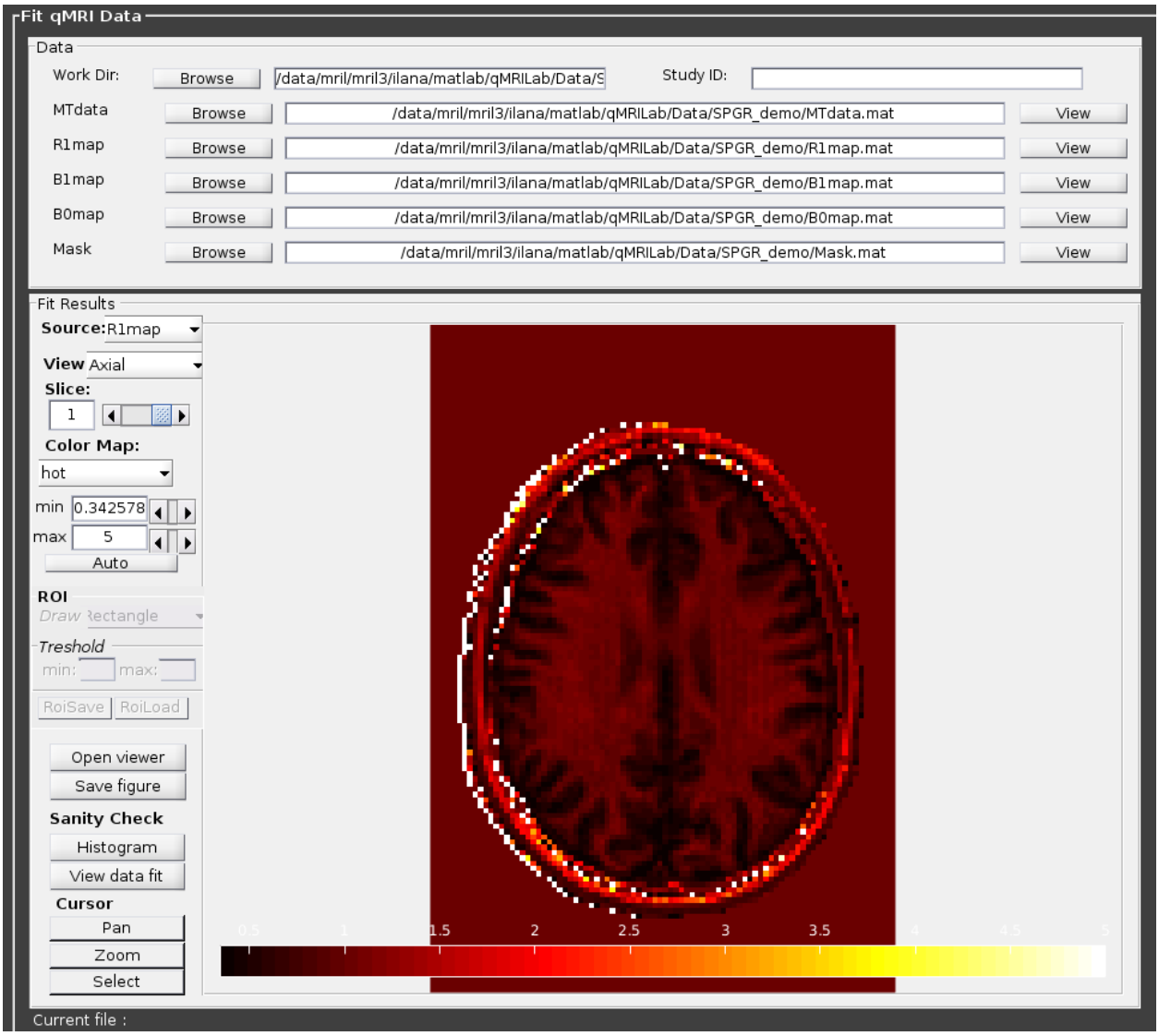


Fig. 3: Example of viewing option, in this case R1map used for qMT SPGR

5. Define or load the appropriate protocol in the *Options* panel (see *6.1 Protocol* for details).
6. Define your fitting options in the Options panel (see *6.2. Fitting* for details).
7. You can preview the fitted curve for a selected voxel by using the `View Data Fit` button. Make sure a dataset is loaded by clicking `View` beside the data file field
8. Click `Select` button in the `Cursor` section to activate voxel selection mode, select a voxel to preview and click `View Data Fit`. You can now see the fitted curve and the parameters computed for that voxel.

Fig. 4: Example of fitting in 1 voxel, which was selected with the cursor on the image.

4.2.2 Whole dataset

Follow steps 1-6 above, then

7. In the *Menu* panel, click on `Fit data` to start the fitting process.
8. Once the fitting is done, a temporary file will be saved in the 'FitResults' subfolder of the current working directory. You can save the current fit results elsewhere by clicking `Save Results` in the 'qMR data fit' section of the *Menu* panel.
9. Use the controls in the `Fit Results` section to visualize the results (ssee *4.3 Viewing the fit results* for details).

4.3 Viewing the fit results

Once you have finished fitting your qMR data, or when you load previously saved fit results by clicking `Load Results` in the `qMR data fit` section of the `Menu` panel, the maps will be displayed in the `Fit Results` section. Use controls on the left side of the figure to navigate the maps:

- *Source*: select the parameter map to display
- *View*: select the side from which to view the data (available only on 3D maps)
- *Slice*: navigate through the z direction of the current view (available only on 3D maps)
- *Color Map*: choose the color scheme to use from a set of pre-defined Malab colormaps
- *Range*: Control the colormap min/max values. Clicking 'Auto' will set the min/max values using the min/max of the currently displayed image. Top textbox/slider allows you to manually set the Min value, while bottom textbox/slider are for the Max value
- *Open viewer*: open the current data in an external viewer to display simultaneous axial/sagittal/coronal views.
- *Save figure*: save the current figure
- *Histogram*: open a new window with an histogram of the voxels in the currently selected slice (note that zooming in on a particular section while still produce an histogram of the full slice)
- *View data fit*: display raw data + fitted curve of the currently selected voxel (use 'Cursor' button to activate voxel selection mode).
- *Pan*: change the cursor mode to 'Pan'. Click and hold inside the figure to move around. Double click inside the figure to reset view. Clicking again the Pan button will turn off pan mode.
- *Zoom*: change the cursor mode to 'Zoom'. Click and hold inside the figure to draw a region to zoom in on. Double click inside the figure to reset view. Clicking again the Zoom button will turn off zoom mode.
- *Select*: change the cursor mode to 'Data Cursor'. Click on a voxel to display info (X/Y is the position of the voxel, index is the value of the map at this point, RGB is the mapped color code).

10.1.5 5. Simulation

5.1. Single Voxel Curve

The Single Voxel Curve simulation interface allows you to simulate qMR data for the defined parameters and protocol. Once the simulation is done, you can also rapidly test the effect of changing fitting options without having to run the

simulation again. It is the fastest way to evaluate various acquisition protocols and the performance of the model and fitting options. A plot of the fitted curve over the actual data will be displayed, and the resulting fitted parameters are compared to the input parameters.

1. Select the acquisition method of your qMR data using the 'Method' drop-down menu in the *Menu* panel.
2. In the *Menu* panel, click on **Single Voxel Curve** to display the interface in the *Main* panel.
3. Using the *Options* panel, define or load the protocol you wish to use (see section 5.1).
4. Using the *Options* panel, define or load your initial fitting options (see section 5.2).
5. Using the `Options` panel, define or load your simulation parameters (see section 5.3).
6. In the `Menu` panel, click on the big `Simulate data` button. A progress bar will appear to show the progression of the simulation. Clicking `Cancel` in the progress bar window will stop the current simulation.
7. Once the simulation is done, the results are displayed in the `Main` panel.
8. If you want to see the effect of changing fitting options, use the *Options* panel to make your changes. Then, in the `Main` panel inside the 'Simulation Fit Results' panel, click on `Update Fit`. Clicking this button without changing fitting options will also generate a new noisy data distribution and recalculate the fitted curve.
9. Once the fitting is done, a temporary file (SimCurveTempResults) will be saved in the 'SimResults' subfolder of the current active method (e.g. *qMTLab/SPGR/SimResults/*). You can save the current simulation results by clicking `Save Results` in the `qMR Data Simulator fit` section of the *Menu* panel. You can later load it using the `Load Results` button.

5.2. Sensitivity Analysis

The Sensitivity Analysis simulation allows you to systematically vary one parameter, over a defined range and number of points, while keeping the others fixed. For each simulated data point, noise is added with a given SNR, and the fit is run multiple times while adding gaussian noise. This allows you to evaluate the variance of the fit at each point. When the simulation is done, a plot shows any variable input parameters as the independent variable, as well as the mean values and variance of any fitted parameters.

1. Select the acquisition method of your data using the 'Method' drop-down menu in the *Menu* panel.
2. In the *Menu* panel, click on **Sensitivity Analysis** to display the interface in the *Main* panel.
3. Using the *Options* panel, define or load the protocol you wish to use (see section 5.1).
4. Using the *Options* panel, define or load your fitting options (see section 5.2).
5. Using the *Options* panel, define or load your simulation parameters (see section 5.3). The parameters defined here are used as the fixed parameters values as one parameter at a time is systematically varied during the simulation process.
6. In the *Main* panel, use the 'Parameters variation' table to define your analysis settings. Select the parameters that are to be varied by setting a mark in the appropriate checkbox, set the minimum and maximum values for this parameter under the column 'Min' and 'Max', and the size of the incrementing step under 'Step'. Set the number of times you want to add noise and fit for each data point by entering an integer value in the '# of runs' box. These settings can be saved, retrieved or reset to their initial settings using the 'Save', 'Load' and 'Reset' buttons respectively.
7. In the *Menu panel*, click on the big **Simulate data** button. A progress bar will appear to show the progression of the simulation. Clicking **Cancel** in the progress bar window will stop the current simulation.
8. Once the simulation is done, the results are displayed in the 'Plot Results' section in the Main panel. Using the 'x axis' and 'y axis' dropdown menu, you can change the independent/dependant parameters respectively. The parameters that have been varied will be available under the 'x axis' menu, while all the model parameters will be available under the 'y axis' menu.

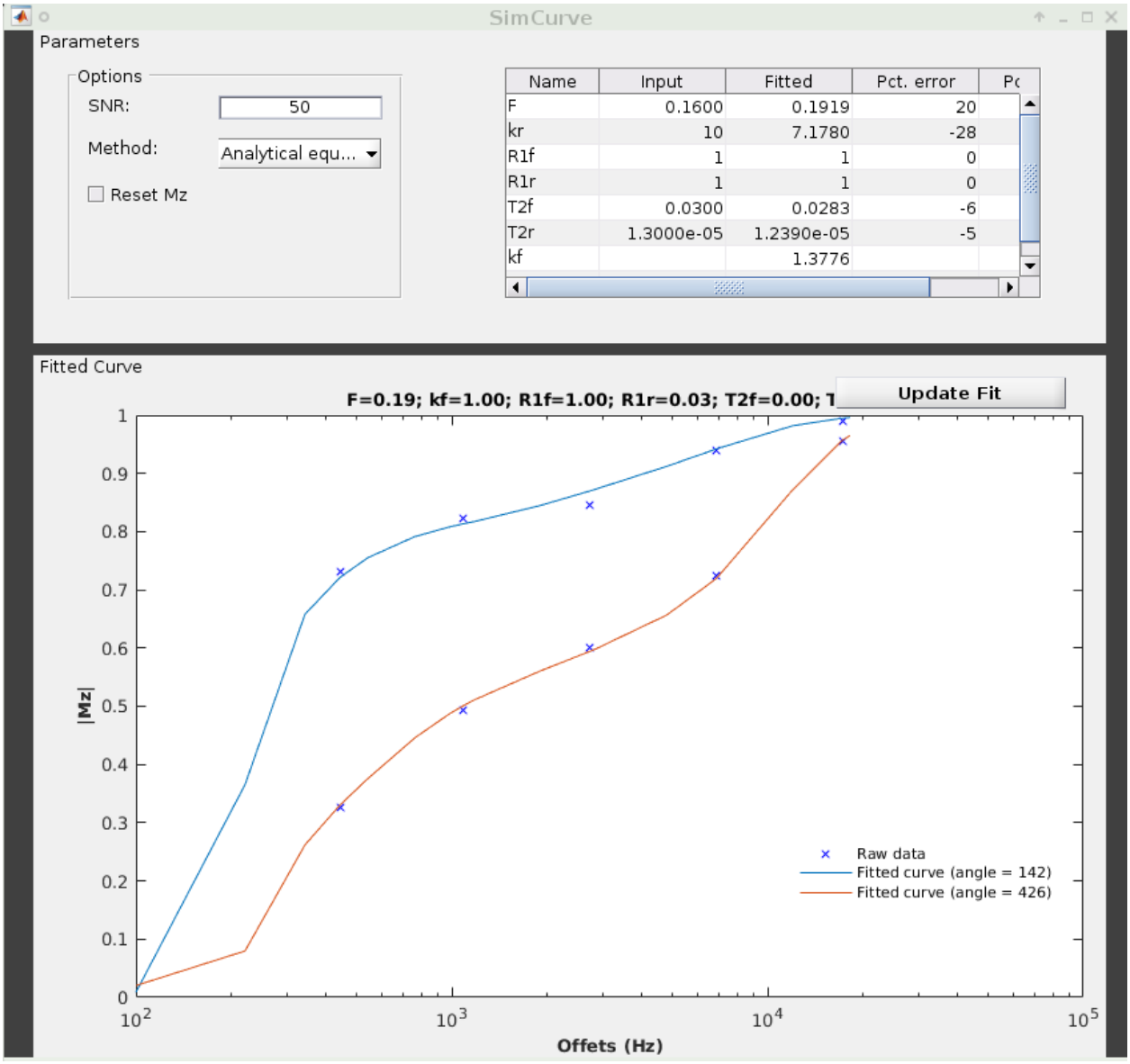


Fig. 5: Example result of simulation in 1 voxel. Remember to set options in 'Options' panel

9. A temporary file (SimVaryTempResults) will be saved in the 'SimResults' subfolder of the current active method (*e.g. qMTLab/SPGR/SimResults/*). You can save the current simulation results by clicking **Save Results** in the 'qMT Data Simulator fit' section of the *Menu panel*. You can later load it using the **Load Results** button.

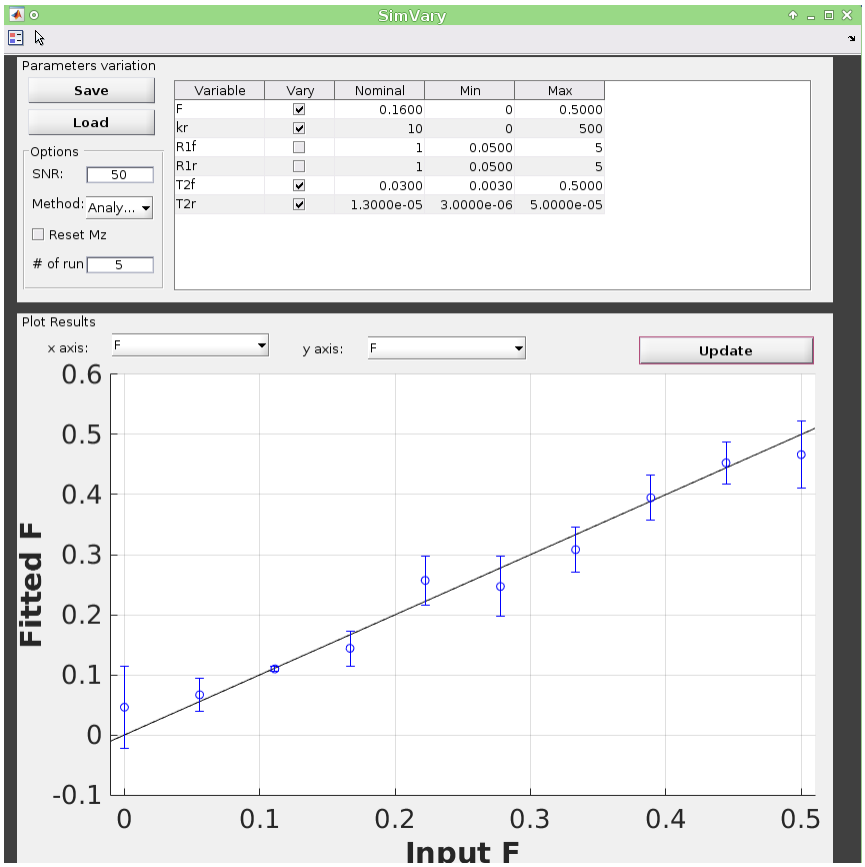


Fig. 6: Example of sensitivity analysis of the F parameter for qMT

5.3. Multi Voxel Distribution

The Multi Voxel Distribution is a tool to simulate any number of voxels, where any combination of parameters are allowed to be varied simultaneously. You can choose how many voxels to simulate and which parameters are to be normally distributed, with its mean value and variance. The results can be displayed in a number of ways such as distribution histograms, scatter plots of input vs fitted parameters, error histograms, etc.

1. Select the acquisition method of your qMR data using the 'Method' drop-down menu in the *Menu* panel.
2. In the *Menu* panel, click on **Multi Voxel Distribution** to display the interface in the *Main* panel.
3. Using the *Options* panel, define or load the protocol you wish to use (see section 5.1).
4. Using the *Options* panel, define or load your fitting options (see section 5.2).
5. Using the *Options* panel, define or load your simulation parameters (see section 5.3). The parameters defined here are used as the fixed parameters values for parameters that are not selected to be varied.
6. In the *Main* panel, use the 'Parameters distribution' table to define your distribution settings. Select the parameters that are to be varied by setting a mark in the appropriate checkbox, set the mean and standard deviation values for this parameter under the column 'Mean' and 'Std' respectively. Set the number of voxels you want to

simulate by entering an integer value in the '# of voxels' box. These settings can be saved, retrieved or reset to their initial settings using the **Save**, **Load** and **Reset** buttons respectively.

7. Click on **Get Parameters** in the 'Parameters distribution' section to generate a set of normally distributed parameters using the current settings. You can look at the distribution in the 'Plot Results' section, by choosing 'Input parameters' under the 'Plot type' dropdown menu. Select the parameters you want to look at with the 'x axis' dropdown menu. You can generate a new set of random values by clicking on the **Get Parameters** button again.
8. In the *Menu* panel, click on the big **Simulate data** button. A progress bar will appear to show the progression of the simulation. Clicking **Cancel** in the progress bar window will stop the current simulation.
9. Once the simulation is done, the results are displayed in the 'Plot Results' section in the Main panel. Using the 'Plot type' dropdown menu, choose what plot you want to view. Plot types are defined below.
10. A temporary file (SimRndTempResults) will be saved in the 'SimResults' subfolder of the current active method (e.g. qMTLab/SPGR/SimResults/). You can save the current simulation results by clicking 'Save Results' in the 'qMT Data Simulator fit' section of the Menu panel. You can later load it using the 'Load Results' button.

Plot types

Different plot types are available to analyze your simulation results. Depending on the plot type, available selections under 'x axis' and 'y axis' dropdown menus will change accordingly.

- *Input parameters*: Histogram of initial input parameters distribution.
- *Fit results*: Histogram of fitted parameters distribution.
- *Input vs. Fit*: Scatter plot of input parameter value vs fitted value.
- *Error*: Histogram of the error distribution. Error is defined as: Fit-Input
- *Pct error*: Histogram of the percentage error distribution. Percentage error is defined as: 100×(Fit-Input)/Input
- *MPE*: Bar graph of the mean percentage error, defined as: 100/n ((Fit-Input)/Input), where n is the number of simulated voxels.
- *RMSE*: Bar graph of the root mean squared error, defined as: (1/n(Fit-Input)^2), where n is the number of simulated voxels.
- *NRMSE*: Bar graph of the normalized root mean squared error, defined as RMSE/(max(Input)-min(Input)) , where max(Input) is the maximum value in the input parameter distribution, and min(Input) is the minimum value.

10.1.6 6. Options Panel

Each qMR acquisition method has its own particular options for simulation, protocol and fitting. These options can be modified by using the *Options* panel. The *Options* panel consists of three sub-panels of options: 'Protocol', 'Fitting' and 'Options'. At the bottom of all these sub-panels you will find buttons to **Reset** the changes you made, **Save** the current settings as a .mat file, **Load** a .mat file of settings, or go back to the **Default** settings. The **Help** button will open the help for the particular model.

6.1 Protocol

The 'Protocol' panel is where you define all options relating to the acquisition sequence. These options are specific for each method. For all methods, you will find (at the top of the protocol panel) input text fields corresponding to the independent variables. You will need to load a previously saved text (.txt) file with the required options and format

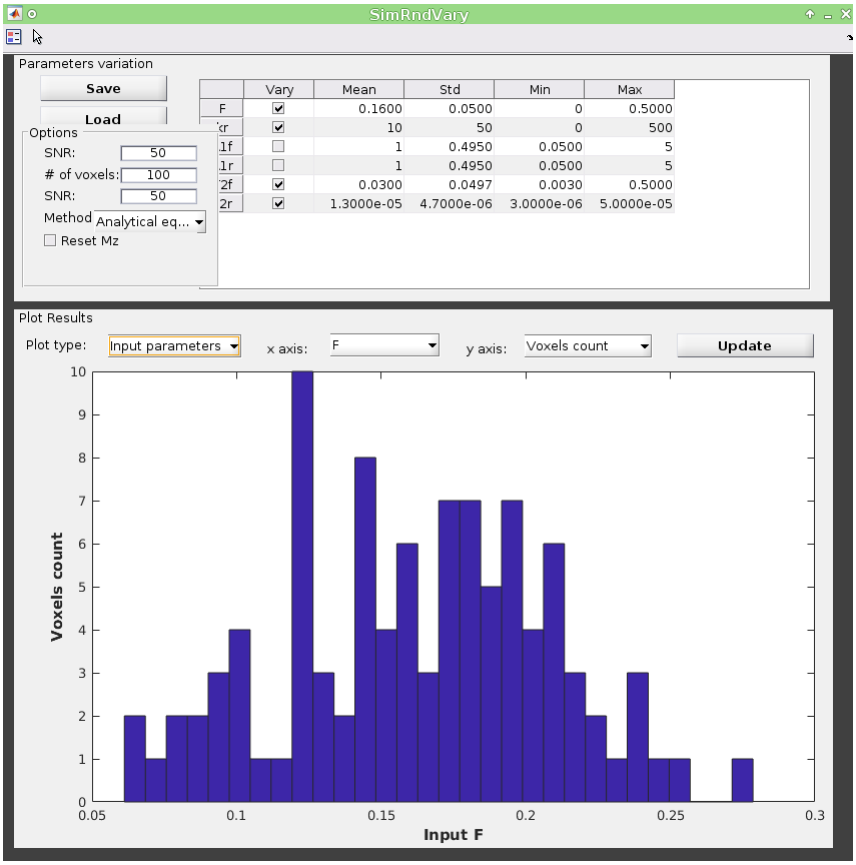


Fig. 7: Example of MultiVoxel Distribution (option: Input parameters)

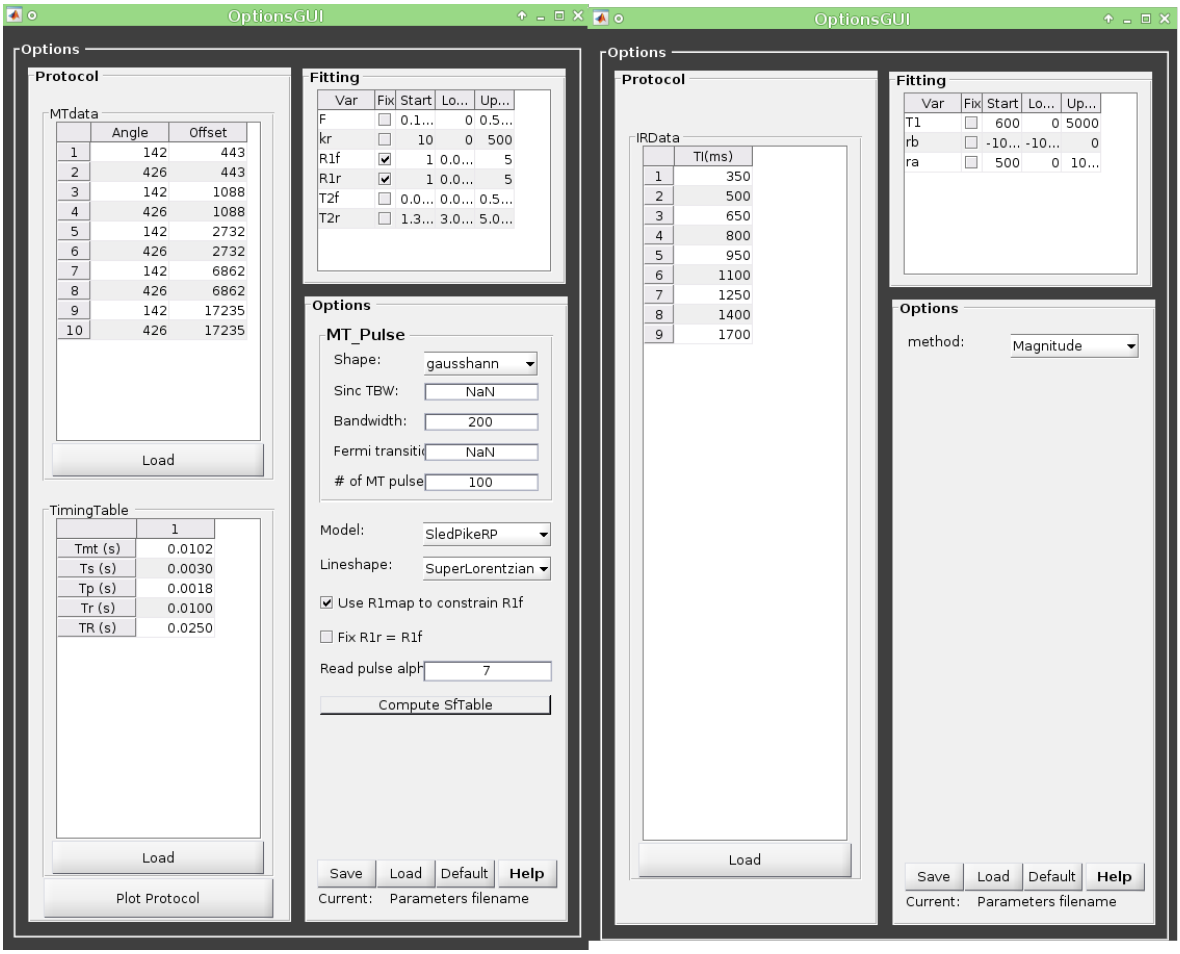


Fig. 8: Example of *Options* panel for qMT SPGR and Inversion Recovery

by using the **Load** button. Press the **Help** button in this panel to see the format expected by each of the models in the *Protocol* section

For example, the *vfa_t1* model is expecting

Protocol

VFAData Array [nbFA x 2]: [FA1 TR1; FA2 TR2;...] flip angle [degrees] TR [s]

Which means a text file where each row is a different flip angle, 1st column is the flip angle in degrees, 2nd is the TR in sec, e.g.:

```
3 0.015
20 0.015
```

6.2. Fitting

The 'Fitting' panel is where you determine the upper, lower and starting points of your parameters. You can also select which parameters should be kept fixed for the fitting.

6.3. Options

The 'Options' panel is where you specify the properties of the model and the fitting. For example, the assumptions/type of model (e.g. for SPGR, the SledPikeRP or Yarnykh model), type of images (magnitude or magnitude/phase for Inversion Recovery), etc.

10.1.7 7. Tutorial

See the video here:

10.2 Command Line Interface (CLI)

10.2.1 General structure

The models are objects, each with a set of properties and generic functions. Before interacting with a model, it must first be instantiated, e.g.

```
model=charmed
```

10.2.2 General commands

- **qMRInfo(model)** : Print the help for the 'model' object
- **qMRusage(model)** : Print the methods of 'model' and examples of how to interact with them
- **qMRgenBatch(model)** : Generate a batch example script for 'model' (will automatically download test data)

10.2.3 Model structure

All models have the following properties:

- **MRIinputs** : Names of input data
- **voxelwise** : Whether the fit is voxelwise [1] or a matrix operation [0]
- **xnames** : Names of output data
- **Prot** : structure containing the protocol parameters
- **buttons** :
- **Options** : options specific to the model (e.g. linear or non-linear fir for VFA-T1)

And functions:

- **equation**:

```
Compute MR signal
USAGE:
  Smodel = Model.equation(x)
INPUT:
  x: [struct] OR [vector] containing Model output parameters
```

- **fit**:

```
Fit experimental data
USAGE:
  FitResults = Model.fit(data)
INPUT:
  data: [struct] containing input data IN ORDER as in MRinuts
NOTE: data are 1D. For 4D datasets use FitData(data,Model)
```

- **plotModel**:

```
Plot model equation (and fitting)
USAGE:
       Model.plotModel(obj, x)
       Model.plotModel(obj, x, data)
INPUT:
  x: [struct] OR [vector] containing Model output parameters
  data: [struct] containing input data in ORDER as in MRinuts
```

Most models have these additional functions:

- **Sim_Sensitivity_Analysis**:

```
Simulates sensitivity to fitted parameters:
   (1) vary fitting parameters from lower (lb) to upper (ub) bound in 10 steps
   (2) run Sim_Single_Voxel_Curve Nofruns times
   (3) Compute mean and std across runs
USAGE:
  SimVaryResults = Model.Sim_Sensitivity_Analysis(OptTable, Opt);
INPUT:
```

(continues on next page)

(continued from previous page)

```
  OptTable: [struct] nominal value and range for each parameter.
     st: [vector] nominal values for output parameters
     fx: [binary vector] do not vary this parameter?
     lb: [vector] vary from lb...
     ub: [vector] up to ub
  Opt: [struct] Options of the simulation
```

- **Sim_Single_Voxel_Curve**:

```
Simulates Single Voxel curves:
   (1) use equation to generate synthetic MRI data
   (2) add rician noise
   (3) fit and plot curve
USAGE:
  FitResults = Model.Sim_Single_Voxel_Curve(x)
  FitResults = Model.Sim_Single_Voxel_Curve(x, Opt,display)
INPUT:
  x: [struct] OR [vector] containing fit results
  display: [binary] 1=display, 0=nodisplay
```

Please type the following to see the specific usage of the model you are interested in

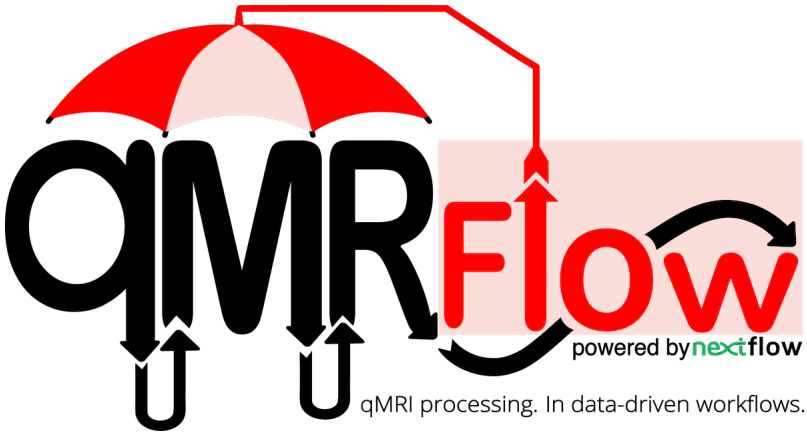
```
qMRusage(model)
```

Or the batch example associated with your model located here *Methods available*

CHAPTER 11

qMRFlow

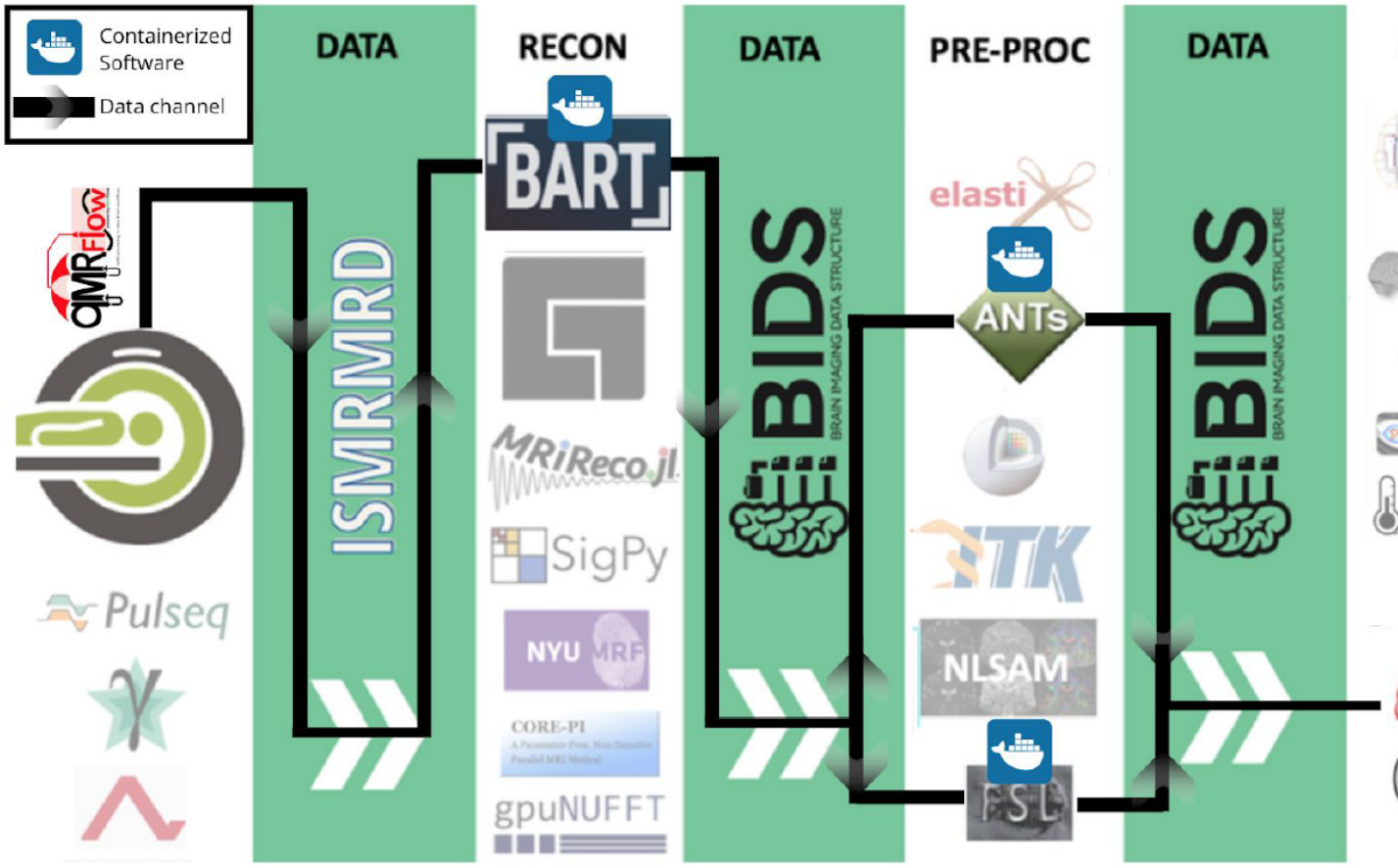
11.1 About



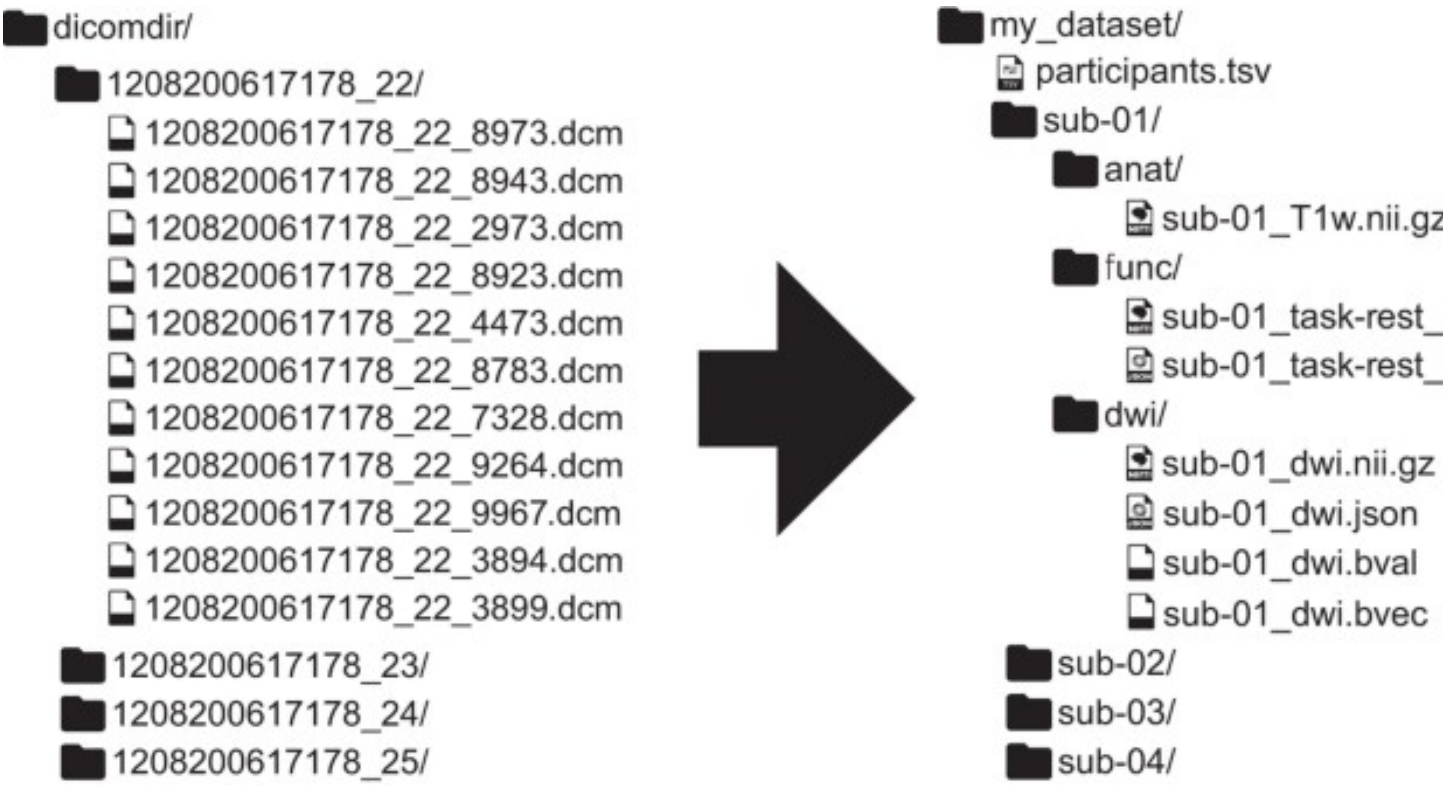
qMRFlow is a collection of container-mediated, data-driven, transparent and platform-agnostic qMRI workflows written in Nextflow.

11.2 Benefits

- qMRLab implements a myriad of qMRI methods, however does not provide pre-processing solutions such as volume alignment or skull stripping. qMRFlow enables the easy use of virtually any pre-processing software with qMRLab, as the process logic and descriptions are detached from workflow orchestration.

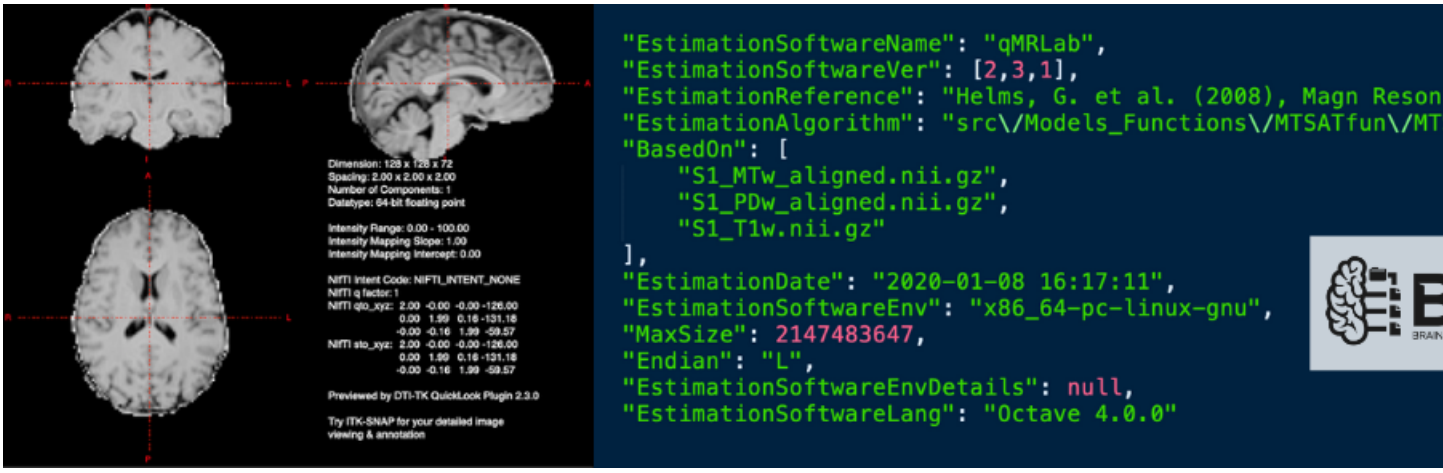


- qMRFlow pipelines run on BIDS formatted data, highly convenient for multi-subject processing:

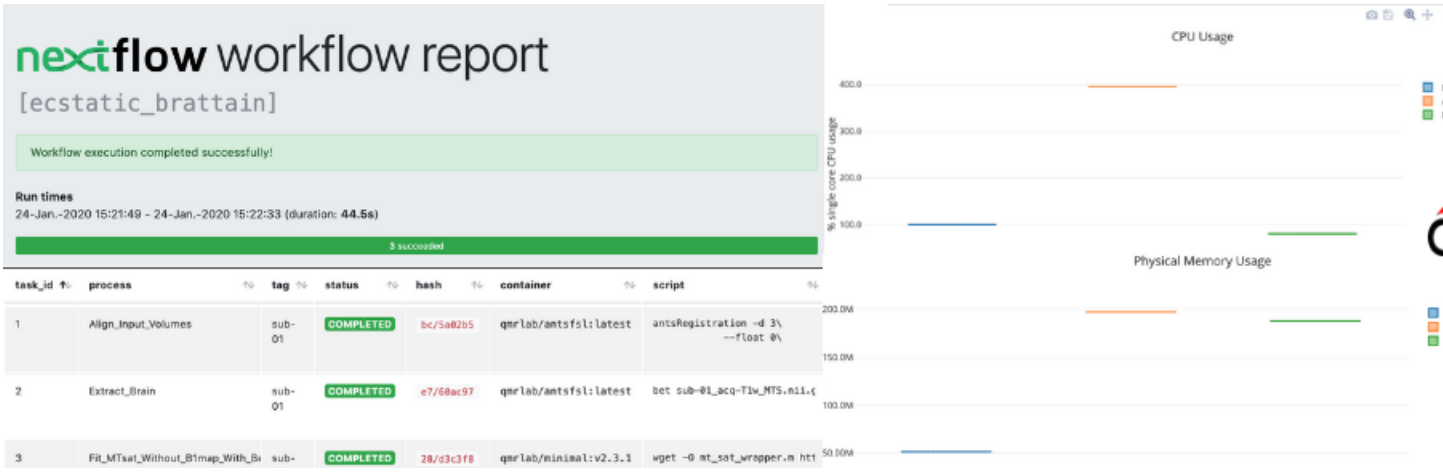


To see the latest BIDS developments on describing qMRI data, you can visit BEP001 GitHub repository.

- Quantitative maps created by qMRFlow are accompanied by sidecar json files containing provenance metadata about the executed qMRI process:



- Nextflow data-driven workflow engine provides comprehensive reports after a pipeline is run:



- In case that the workflow is interrupted for any reason, pipeline execution can be resumed from where it left off.

11.3 Use qMRFlow with Docker

If you have Docker installed on your computer, getting started with qMRFlow is a few steps away.

1. Install Nextflow as described in here.
2. Pull Docker images listed by a qMRFlow pipeline. For example, to run MTsat workflow in containers, following images must be pulled:

```
docker pull qmrlab/minimal:v2.3.1
docker pull qmrlab/antsfsl:latest
```

3. Run the pipeline as described in the *Usage* section of the desired qMRFlow pipeline documentation.

11.4 Use qMRFlow locally

We highly suggest using qMRFlow workflows in containers. However, if you don't have Docker installed, you can still use them by installing dependencies locally.

1. Ensure that all the dependencies listed by the *Local installation requirements* section of the desired qMRFlow pipeline documentation are met.
2. Install qMRLab
3. Run the pipeline as described in the *Usage* section of the desired qMRFlow pipeline documentation.

CHAPTER 12

Available pipelines

12.1 MagnetizationTransfer

12.1.1 MTsat: Magnetization transfer saturation

Usage

```
nextflow run mtsatflow_BIDS.nf [OPTIONAL_ARGUMENTS] (--root)
```

Description

`--root=/path/to/[root]` Root folder containing multiple subjects

Container requirements

If you have Docker installed, enabling docker option will make use of the following Docker images to execute processes:

- **qmrlab/minimal (https://hub.docker.com/repository/docker/qmrlab/minimal)** 594MB extends to 1.5GB Built at each qMRLab release. Minimum version requirement: v2.3.1
- **qmrlab/antsfsl (https://hub.docker.com/repository/docker/qmrlab/antsfsl)** 374MB extends to 1.2GB Dockerfile is available at qMRLab/qMRflow.

Local installation requirements

Unless the docker option is enabled in the *nextflow.config*, the following dependencies must be installed and added to the system path:

- ANTs registration (https://github.com/ANTsX/ANTs)

- FSL (https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/)
- Octave/MATLAB (https://www.gnu.org/software/octave/, https://www.mathworks.com/)
- qMRLab > v2.3.1 (https://qmrlab.org)
- git

Folder organization

```
[root]
├── sub-01
│   ├── anat
│   │   ├── sub-01_acq-MTon_MTS.nii.gz
│   │   ├── sub-01_acq-MTon_MTS.json
│   │   ├── sub-01_acq-MToff_MTS.nii.gz
│   │   ├── sub-01_acq-MToff_MTS.json
│   │   ├── sub-01_acq-T1w_MTS.nii.gz
│   │   └── sub-01_acq-T1w_MTS.json
│   └── fmap
│       └── sub-01_B1plusmap.nii.gz (optional)
└── sub-02
    ├── anat
    │   ├── sub-02_acq-MTon_MTS.nii.gz
    │   ├── sub-02_acq-MTon_MTS.json
    │   ├── sub-02_acq-MToff_MTS.nii.gz
    │   ├── sub-02_acq-MToff_MTS.json
    │   ├── sub-02_acq-T1w_MTS.nii.gz
    │   └── sub-02_acq-T1w_MTS.json
    └── fmap
        └── sub-02_B1plusmap.nii.gz (optional)
```

Optional arguments

--platform `["octave"/"matlab"]` Platform choice.

--qmrlab_dir `["/path/to/qMRLab" OR null]` Absolute path to the qMRLab's root directory. If docker is enabled, MUST be set to null (without double quotes). If docker is NOT enabled, then the absolute path to the qMRLab MUST be provided. Note that qMRLab version MUST be equal or greater than v2.3.1.

--octave_path `["/path/to/octave_exec" OR null]` Absolute path to Octave's executable. If docker is enabled, or, if you'd like to use Octave executable saved to your system path, MUST be set to null (without double quotes).

--matlab_path `["/path/to/matlab_exec" OR null]` Absolute path to MATLAB's executable. If you'd like to use MATLAB executable saved to your system path, MUST be set to null (without double quotes). Note that qMRLab requires MATLAB > R2014b. Docker image containing MCR compiled version of this application is NOT available yet. Therefore, container declarations for the processes starting with `Fit` prefix MUST be set to null (without double quotes).

--ants_dim `[2/3/4]` This option forces the image to be treated as a specified-dimensional image. If not specified, ANTs tries to infer the dimensionality.

--ants_metric `["MI"]` Confined to MI: Mutual information, for this particular pipeline.

--ants_metric_weight `[0-1]` If multimodal (i.e. changing contrast) use weight 1. This parameter is used to modulate the per stage weighting of the metrics.

--ants_metric_bins `[e.g. 32]` Number of bins.

--ants_metric_sampling `["Regular","Random:]` The point set can be on a regular lattice or a random lattice of points slightly perturbed to minimize aliasing artifacts.

--ants_metric_samplingprct `[0-100]` The fraction of points to select from the domain

--ants_transform

- `"Rigid"`
- `"Affine"`
- `"CompositeAffine"`
- `"Similarity"`
- `"Translation"`
- `"BSpline"`

--ants_convergence `[MxNxO,<convergenceThreshold=1e-6>, <convergenceWindowSize=10>]` Convergence is determined from the number of iterations per level and is determined by fitting a line to the normalized energy profile of the last N iterations (where N is specified by the window size) and determining the slope which is then compared with the convergence threshold.

--ants_shrink `[MxNxO]` Specify the shrink factor for the virtual domain (typically the fixed image) at each level.

--ants_smoothing `[MxNxO]` Specify the sigma of gaussian smoothing at each level. Units are given in terms of voxels ('vox') or physical spacing ('mm'). Example usage is '4x2x1mm' and '4x2x1vox' where no units implies voxel spacing.

--use_b1cor `[true/false]` Use and RF transmit field to correct for flip angle imperfections.

--b1cor_factor `[0-1]` Correction factor (empirical) for the transmit RF. Only corrects MTSAT, not T1. Default 0.4.

--use_bet Use FSL's BET for skull stripping.

--bet_recursive `[true/false]` This option runs more "robust" brain center estimation.

--bet_threshold `[0-1]` Fractional intensity threshold (0->1); default=0.45; smaller values give larger brain outline estimates.

Notes

- BIDS for quantitative MRI (BEP001) data is under development as of early 2020. You can visit the BEP001 GitHub repository.
- Example datasets:
  - BIDSified MTsat data https://osf.io/k4bs5/
- Files should be compressed Nifti files (.nii.gz)
- Timing parameters in the .json files MUST be in `seconds`.
- `Subject IDs` are used as the primary process `ID` and `tag` throughout the pipeline.

- We adhere to a strict `one-process one-container mapping`, where possible using off-the shelf qMRLab containers.
- All the OPTIONAL ARGUMENTS can be modified in the `nextflow.config` file. The same config file is consumed by `mtrflow_BIDS.nf`.
- You can take advantage of Nextflow's comprehensive `tracing and visualization` features while executing this pipeline: https://www.nextflow.io/docs/latest/tracing.html.
- For any requests, questions or contributions, please feel free to open an issue at qMRflow's GitHub repo at https://github.com/qMRLab/qMRflow.

Reference

Please cite the following if you use this module:

Karakuzu A. et al. 2019 The qMRLab workflow: From acquisition to publication., ISMRM 27th Annual Meeting and Exhibition, Montreal, Canada.

12.1.2 MTR: Magnetization transfer ratio

Usage

```
nextflow run mtrflow_BIDS.nf [OPTIONAL_ARGUMENTS] (--root)
```

Description

`--root=/path/to/[root]` Root folder containing multiple subjects

Container requirements

If you have Docker installed, enabling docker option will make use of the following Docker images to execute processes:

- **qmrlab/minimal (https://hub.docker.com/repository/docker/qmrlab/minimal)** 594MB extends to 1.5GB Built at each qMRLab release. Minimum version requirement: v2.3.1
- **qmrlab/antsfsl (https://hub.docker.com/repository/docker/qmrlab/antsfsl)** 374MB extends to 1.2GB Dockerfile is available at qMRLab/qMRflow.

Local installation requirements

Unless the docker option is enabled in the *nextflow.config*, the following dependencies must be installed and added to the system path:

- ANTs registration (https://github.com/ANTsX/ANTs)
- FSL (https://fsl.fmrib.ox.ac.uk/fsl/fslwiki/)
- Octave/MATLAB (https://www.gnu.org/software/octave/, https://www.mathworks.com/)
- qMRLab > v2.3.1 (https://qmrlab.org)
- git

Folder organization

```
[root]
├── sub-01
│   └── anat
│       ├── sub-01_acq-MTon_MTR.nii.gz
│       ├── sub-01_acq-MTon_MTR.json
│       ├── sub-01_acq-MToff_MTR.nii.gz
│       └── sub-01_acq-MToff_MTR.json
└── sub-02
    └── anat
        ├── sub-02_acq-MTon_MTR.nii.gz
        ├── sub-02_acq-MTon_MTR.json
        ├── sub-02_acq-MToff_MTR.nii.gz
        └── sub-02_acq-MToff_MTR.json
```

Note: This workflow can use a subset of the MTsat (*MTS*) data.

Optional arguments

--platform `["octave"/"matlab"]` Platform choice.

--qmrlab_dir `["/path/to/qMRLab" OR null]` Absolute path to the qMRLab's root directory. If docker is enabled, MUST be set to null (without double quotes). If docker is NOT enabled, then the absolute path to the qMRLab MUST be provided. Note that qMRLab version MUST be equal or greater than v2.3.1.

--octave_path `["/path/to/octave_exec" OR null]` Absolute path to Octave's executable. If docker is enabled, or, if you'd like to use Octave executable saved to your system path, MUST be set to null (without double quotes).

--matlab_path `["/path/to/matlab_exec" OR null]` Absolute path to MATLAB's executable. If you'd like to use MATLAB executable saved to your system path, MUST be set to null (without double quotes). Note that qMRLab requires MATLAB > R2014b. Docker image containing MCR compiled version of this application is NOT available yet. Therefore, container declarations for the processes starting with `Fit` prefix MUST be set to null (without double quotes).

--ants_dim `[2/3/4]` This option forces the image to be treated as a specified-dimensional image. If not specified, ANTs tries to infer the dimensionality.

--ants_metric `["MI"]` Confined to MI: Mutual information, for this particular pipeline.

--ants_metric_weight `[0-1]` If multimodal (i.e. changing contrast) use weight 1. This parameter is used to modulate the per stage weighting of the metrics.

--ants_metric_bins `[e.g. 32]` Number of bins.

--ants_metric_sampling `["Regular","Random:]` The point set can be on a regular lattice or a random lattice of points slightly perturbed to minimize aliasing artifacts.

--ants_metric_samplingprct `[0-100]` The fraction of points to select from the domain

--ants_transform

- `"Rigid"`
- `"Affine"`
- `"CompositeAffine"`

- "Similarity"
- "Translation"
- "BSpline"

--ants_convergence [MxNxO,<convergenceThreshold=1e-6>, <convergenceWindowSize=10>] Convergence is determined from the number of iterations per level and is determined by fitting a line to the normalized energy profile of the last N iterations (where N is specified by the window size) and determining the slope which is then compared with the convergence threshold.

--ants_shrink [MxNxO] Specify the shrink factor for the virtual domain (typically the fixed image) at each level.

--ants_smoothing [MxNxO] Specify the sigma of gaussian smoothing at each level. Units are given in terms of voxels ('vox') or physical spacing ('mm'). Example usage is '4x2x1mm' and '4x2x1vox' where no units implies voxel spacing.

--use_b1cor [true/false] Use and RF transmit field to correct for flip angle imperfections.

--b1cor_factor [0-1] Correction factor (empirical) for the transmit RF. Only corrects MTSAT, not T1. Default 0.4.

--use_bet Use FSL's BET for skull stripping.

--bet_recursive [true/false] This option runs more "robust" brain center estimation.

--bet_threshold [0-1] Fractional intensity threshold (0->1); default=0.45; smaller values give larger brain outline estimates.

Notes

- BIDS for quantitative MRI (BEP001) data is under development as of early 2020. You can visit the BEP001 GitHub repository.
- Example datasets:
  - BIDSified MTsat data https://osf.io/k4bs5/
- Files should be compressed Nifti files (.nii.gz)
- Timing parameters in the .json files MUST be in seconds.
- Subject IDs are used as the primary process ID and tag throughout the pipeline.
- We adhere to a strict one-process one-container mapping, where possible using off-the shelf qMRLab containers.
- All the OPTIONAL ARGUMENTS can be modified in the nextflow.config file. The same config file is consumed by mtrflow_BIDS.nf.
- You can take advantage of Nextflow's comprehensive tracing and visualization features while executing this pipeline: https://www.nextflow.io/docs/latest/tracing.html.
- For any requests, questions or contributions, please feel free to open an issue at qMRflow's GitHub repo at https://github.com/qMRLab/qMRflow.

Reference

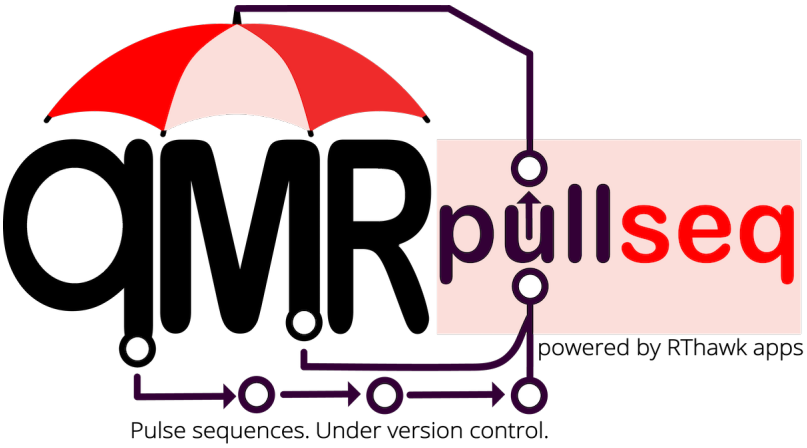
Please cite the following if you use this module:

Karakuzu A. et al. 2019 The qMRLab workflow: From acquisition to publication., ISMRM 27th Annual Meeting and Exhibition, Montreal, Canada.

CHAPTER 13

qMRPullseq

13.1 About



`qMRPullseq` is aimed to be collection of vendor-neutral & open-source pulse sequences developed in `RTHawk` for qMRI applications. As the name implies, you can pull a pulse sequence from GitHub and run it on your scanner. Currently, only certain `Siemens` and `GE` scanners support `RTHawk` platform.

Note: For more information regarding `RTHawk` please visit here.

Warning: `qMRPullseq` is in its early stage of development. In future releases, `qMRPullseq` will be improved to trigger qMRFlow pipelines to streamline the process of qMRI map generation.

13.2 VFA T1 mapping

A preliminary version of the RTHawk application is publicly available for variable flip angle T1 mapping using a 3D spoiled gradient-echo sequence.

Warning: This pulse sequence is still under development for accuracy improvements.

CHAPTER 14

Developer's Documentation

Thank you for considering to contribute to the standardization of quantitative Magnetic Resonance Imaging! Before starting, please see our contributions guideline.

We maintain a developer-friendly documentation on our wiki page. Below are some useful hyperlinks to get you started:

- How to add a new model to qMRLab?
- How to upload a sample data?
- Version control and model patching
- GUI development
- Automatic documentation generation
- Continuous Integration tests
- Our Docker organization
- Our Azure release pipeline

You can take a look at some recent changes in the CHANGELOG.

Note: Contribution is not all about making pull requests to the main codebase or reporting bugs. We maintain a blog for qMRI , and you are more than welcome to share your blog post there! You can simply open a new issue and let us know about your exciting ideas to share with the qMRI community.

CHAPTER 15

Indices and tables

- genindex
- modindex
- search