Madym usage instructions

# Overview

This document provides usage instructions for the Matlab function that act as “wrappers” for the Madym C++ toolkit. Currently there are 3 functions that can be called:

1. run\_madym: Applies DCE analysis to whole image volumes (saved on disk in Analyze format), saving output parameter maps as Analyze images
2. run\_madym\_lite: Applies DCE analysis directly to Matlab arrays, returning the output directly to Matlab
3. run\_madym\_T1: Applies T1 VFA mapping, and can be run either on Analyze images or directly on Matlab arrays, returning T1 and M0 estimates directly to Matlab, and, if the input were Analyze images, saving T1 and M0 maps as Analyze images

More details on each function are given below. Note this document focuses on the Matlab functions - for a full description of how the underlying C++ tools are structured, please see the separate documentation in the cxx GIT repository.

# Installation

There are separate instructions for installing the Matlab tools on your computer. Please follow these first.

# DCE models

# run\_madyn

The function run\_madym calls the C++ program madym, which applies DCE analysis to a dataset of dynamic images, stored in Analyze format (eg each image has a .img, .hdr and optionally a .xtr file). Madym has multiple modes of operation: ultimately it fits one of a selection of tracer kinetic models to contrast-agent concentration time-series. The concentration time-series are either extracted, one per voxel, directly from a series of concentration image volumes (eg that have previously been computed from raw signal volumes) or will be computed from raw signal time-series that are extracted, again, one-per-voxel, from a series of signal images. If converting concentration from signal input, the function must also supply baseline T1 values - these can either be input as a T1 image volume, or can be computed from a series of variable flip-angle images. All images must be of the same dimensions. Parameter maps of the same dimension are generated by the function and saved. The number and names of the maps depends on the selected model (see DCE models section). In addition,

# run\_madyn\_lite

Stuff

# run\_madyn\_T1

Stuff

# User interactive installation

At the Matlab command-line, run **install\_madym**. This will automatically install the Madym tools from the shared drive, and will prompt you for the following input:

1. The root directory of the shared drive, selected via a file explorer/finder window
2. Your operating system: MacOS, Windows or Ubuntu
3. A folder on your machine where you would like to install the Madym tools, selected via a file explorer/finder window (you can use the new folder option in this if you would like to create a folder that doesn’t exist)

The install function will then attempt to copy the Madym tools from the shared drive to the folder you have selected. It will cache this location so you do not need to remember it whenever you run the Madym tools, they will be found automatically. If there are any problems with the installation they will be reported in the command window.

If the installation has been successful, the function will then run basic tests for the Madym tools **run\_madym**, **run\_madym\_lite** and **run\_madym\_T1**. These tests will generate output in the command window and produce figures with sample fits to some synthetic DCE and variable flip-angle data. Please inspect this output to check it is as expected. If there are any problems ask Mike Berks. If not, everything is done. Instructions on how to use the Madym tools are given in their help documentation (eg type doc **run\_madym**) and in the separate user instructions document.

# Advanced installation

In addition to the interactive installation, the install function can be called with inputs to directly set the share path, operating system, local root, in addition to specifying a specific version of the Madym tools, and customising the level of testing applied. There are less checks applied to these inputs, so you should only use this version if you are confident of what you are doing. The input options are described below:

* qbi\_share\_path (str): path to the root of the shared drive (eg ‘Q:’ on Windows, or ‘/Volumes/QBI’ on a Mac)
* operating\_system (str): must be one of Windows, MacOS or Ubuntu
* madym\_root (str): path to folder on your computer where Madym will be installed. If empty, and you already have a local madym root defined (eg from a previous installation) it will use your existing one, otherwise you will be prompted to choose a new one. If you supply a different path to your existing madym root, you will be asked whether you want to override the old one.
* version (str) – if empty, will use the latest available version. However, you can set this to specify loading a particular version. Version strings are in the form vX.Y.Z. If the version you enter is not available on the share drive, this will generate an error.
* run\_tests (int) – the default is 1, which runs the basic test. Setting this option to 0 switches off testing (not recommended unless you have good reason). Setting to >1 runs any advanced tests implemented in run\_madym\_tests. Currently there aren’t any (beyond the basic tests), however there should be soon (eg testing various combinations of the many optional inputs you can set in Madym).

# Existing installations

If you have an existing installation of the Madym tools, then 1 of 2 things will happen.

1. If you install to the same Madym root as before (recommended), the installation function will move the existing tools in to a folder tagged with their version name. For example, if your madym root it code/madym\_tools and you had version v2.2.0 installed, then install the latest version v2.2.1, the existing executable files will be moved into a folder code/madym/v2.2.0, and the new executables will be copied into code/madym
2. If you choose to install into a new location, your old executables will not be changed. However, your madym root will be updated to the new location you have chosen, so the Matlab wrappers will not ‘see’ the old versions. To check which location is being called by the Matlab wrappers, run **local\_madym\_root** at the command line. To check the current version being, run **local\_madym\_version**. You can compare this against the latest available version on the shared drive by running **latest\_madym\_version** or **check\_madym\_updates**.