

## Table of contents

1. Overview
2. Contact
3. Disclaimer
4. Installation notes
5. Running the software
6. Basic operation



## 1. Overview

*T1mappLL* is a software utility and graphical user interface to reconstruct T1 maps from a Look-Locker inversion recovery experiment acquired with an MR Solutions preclinical MRI system.

## 2. Contact

*T1mappLL* and this manual are written by:

Gustav Strijkers  
Amsterdam UMC  
Biomedical Engineering and Physics  
Amsterdam, the Netherlands  
[g.j.striekers@amsterdamumc.nl](mailto:g.j.striekers@amsterdamumc.nl)

## 3. Disclaimer

The software has been extensively tested using mouse and rat data acquired with an MR Solutions preclinical 7.0T/24cm system.

However, this does not warrant the functions contained in the program will meet your requirements or that the operation of the program will be uninterrupted or error-free.

In case of questions or issues, please contact Gustav Strijkers.



## 4. Installation notes

### Software download

Matlab source code and a Windows standalone version (using the free Matlab runtime engine) can be downloaded from github:

<https://github.com/Moby1971?tab=repositories>

### Installation of the Windows standalone version

`MyAppInstaller_web.exe`

Will install the Matlab runtime engine and the *T1mappLL* program.

### Bart toolbox download

The advanced reconstruction options in *T1mappLL* require the Bart reconstruction toolbox, which can be downloaded from:

<https://mrirecon.github.io/bart/>



## Bart toolbox installation in OSX with MacPorts

- (1) Install Xcode from the Mac App Store

```
$ xcode-select --install
```

- (2) Install MacPorts (<http://www.macports.org/>)

It is recommended to install a newer version of gcc from MacPorts

- (3) Install packages

```
$ sudo port install fftw-3-single
$ sudo port install gcc10
$ sudo port install libpng
$ sudo port install openblas
```

- (4) Make

Replace 'gcc-mp-6' with 'gcc-mp-10' in 'makefile'

```
$ make
```

## Bart toolbox installation in OSX with HomeBrew (for Mac with Apple Silicon)

- (1) Install Xcode from the Mac App Store

```
$ xcode-select --install
```

- (2) Install HomeBrew

```
$ /bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)"
```

- (3) Install packages

```
$ brew install --cask gcc-arm-embedded
$ brew install libpng
$ brew install fftw
$ brew install openblas
```

- (3) Make

- Use the provided makefile for Apple Silicon

```
$ make
```



## **Bart toolbox installation in Windows 10**

### **(1) Install the Windows subsystem for Linux**

Start a windows powershell as administrator and run the following command:

```
Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux
```

A system restart will be needed.

For more information, see:

<https://docs.microsoft.com/en-us/windows/wsl/install-win10>

### **(2) Download Ubuntu Linux**

Open PowerShell

To see a list of available Linux distributions for download, enter:

```
wsl -l -o
```

Install a Linux distribution:

```
wsl --install -d <distribution name>
```

You will be asked to create a user account the first time you start a Linux command prompt.

### **(3) Upgrade the Linux distribution**

```
$ sudo apt-get update
```

```
$ sudo apt-get dist-upgrade
```

### **(4) Install Bart prerequisites**

```
$ sudo apt-get install make gcc gcc-10 libfftw3-dev liblapacke-dev  
libpng-dev libopenblas-dev gfortran
```

### **(5) Download Bart**

```
$ wget https://github.com/mrirecon/bart/archive/v0.7.00.tar.gz
```

For WSL1 version 0.4.02 is recommended. Newer versions seem to compile but some functions which are required produce errors.

Recommended: Upgrade to WSL2 (see e.g. <https://pureinfotech.com/install-windows-subsystem-linux-2-windows-10/> on how to upgrade).

### **(6) Build Bart**

```
$ tar xvfz v0.7.00.tar.gz
```

```
$ cd bart-0.7.00
```

```
$ make
```

```
$ sudo make PREFIX=/usr/local install
```



## 5. Running the software

### Running in Matlab 2022a

The T1mappLL software can be started from its root directory from the command line.

```
>> T1mappLL
```

Notes:

Additional licenses may be required.

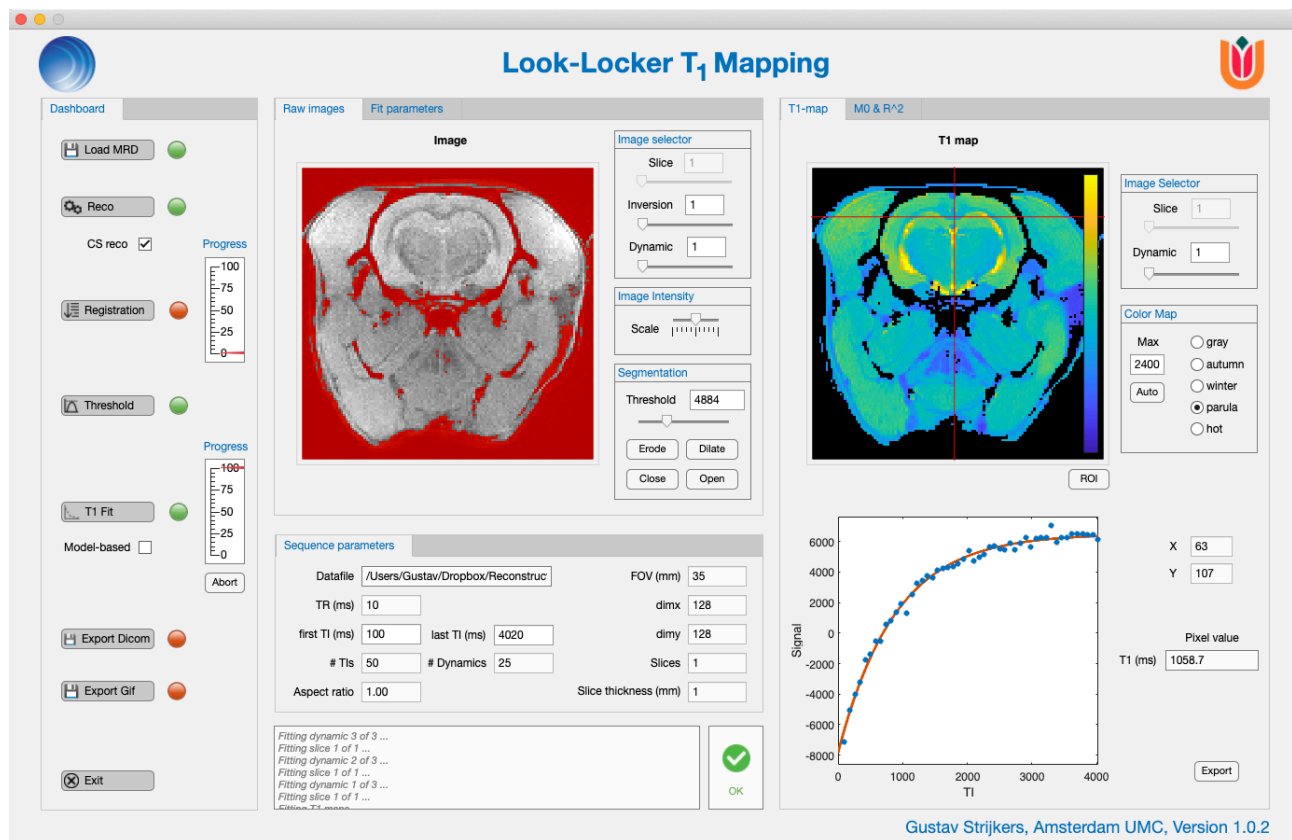
```
>> license 'inuse'
distrib_computing_toolbox
image_toolbox
matlab
optimization_toolbox
signal_toolbox
```

### Running the Windows standalone

The Windows standalone version can be run from the start menu or the desktop icon.

## 6. Basic operation

The T1mappLL program operates from a single window with 5 panels.



### Panel 1: Dashboard

This panel contains the task buttons and parameters that control the reconstruction process. The dashboard tasks need to be completed from top to bottom. A green light next to the task indicates that the task has been completed. Red indicates not completed yet.

### Panel 2: Raw images / Fit parameters

This panel has 2 tabs, raw images and fit parameters.

### Panel 3: Scan parameters

This panel displays the relevant acquisition parameters.

### Panel 4: T1-map / M0 & R<sup>2</sup>


During and after fitting is completed this panel shows the fitted T1-maps. M0-maps and R<sup>2</sup> goodness of fit maps are displayed in the second tab.

### Panel 5: Messages

Displays program status and messages.



## Step 1: Loading data

Press  to import load the MRD raw data file.

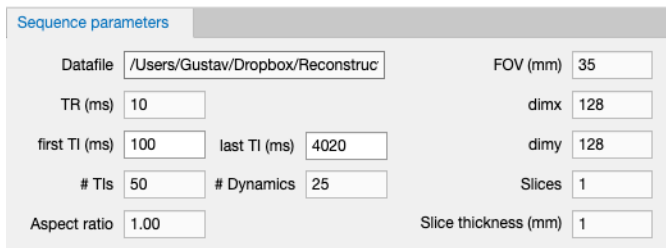
T1mappLL can read multi-slice and multi-dynamic (repetitions) inversion-recovery (FLASH) data.

The MRD file might need re-ordering using the *seg2echo* tool provided by MR Solutions.

Usage: *seg2echo* <mrd file> <output file> <lines per segment>

For multi receiver coil data, select 1 of the MRD raw data files.


Relevant acquisition parameters will be shown in panel 3.



Sequence parameters

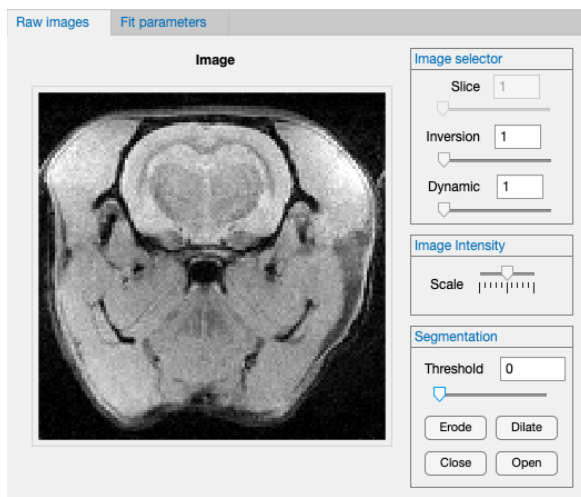
Datafile	/Users/Gustav/Dropbox/Reconstruc		FOV (mm)	35	
TR (ms)	10		dimx	128	
first TI (ms)	100	last TI (ms)	4020	dimy	128
# TIs	50	# Dynamics	25	Slices	1
Aspect ratio	1.00		Slice thickness (mm)	1	

## Step 2: Image reconstruction

Press  to reconstruct the images from the k-space data.

The standard reconstruction is a 2D FFT.

When the CS reco checkbox is checked and if the BART toolbox is available a compressed sensing reconstruction of the data will be performed with l1-norm in the spatial dimension and total variation regularization in the dynamic dimension.



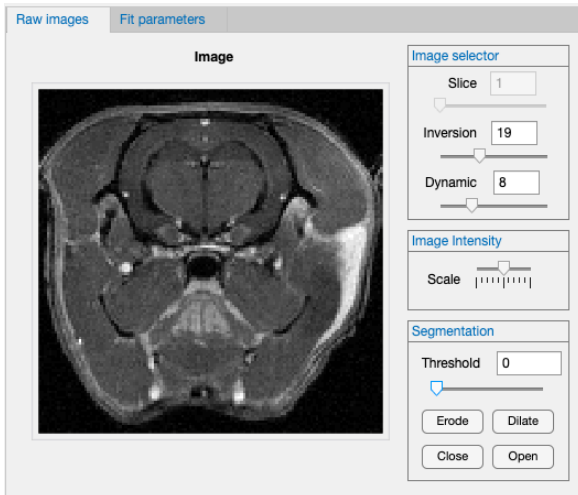
Images with different inversion and dynamic times can be inspected with the sliders.




### Step 3: Image registration

**NOTE:** this step usually can be omitted.

Use the sliders and/or edit-fields to inspect the images as function of slice (Slice), inversion time (Inversion), or repetition (Dynamic). The image scale can be adjusted with the Scale slider.



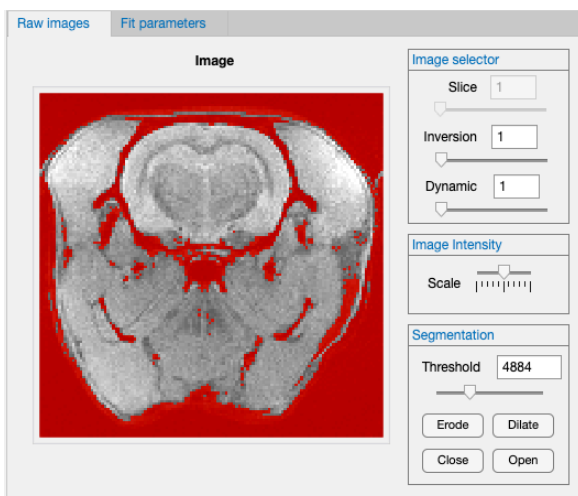
In case the images as function of the inversion times or dynamics are not perfectly registered, press  **Registration** to perform an affine registration which includes translation, rotation, and scale.

### Step 4: Segmenting the images

To prevent unnecessary fitting of the background pixels, the images need to be segmented first.

Press  **Threshold** to perform an automatic thresholding segmentation.

The result will look something like the figure below in which the red pixels indicate the regions that will be skipped during T1 fitting.



In case automatic thresholding is not satisfying, the threshold can be manually adjusted per slice. The segmentation can be optimized by erode, dilate, close, and open morphological operations.




## Step 5: Fitting T1 maps

Press the Fit parameters tabs to inspect some fit settings.

The first and last slice that will be fitted can be selected.

Min R-square is the minimally accepted  $R^2$  value that is accepted after fitting. If the result of a T1 fit in a pixel results in an  $R^2$  value less than Min R-square, then the T1 and M0 values in this pixel will be omitted and set to zero. This only holds for normal least-squares fitting.

The bottom part of this tab shows a list of all the inversion numbers. The inversion times that are selected will be included in the fitting process. This option can be used to exclude certain TIs.

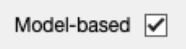
Press  to start the T1 fitting.

Fitting will be done using a least-squares algorithm according to the equation

$$\text{Signal} = (A - B \exp(-TI/T_1^*))$$

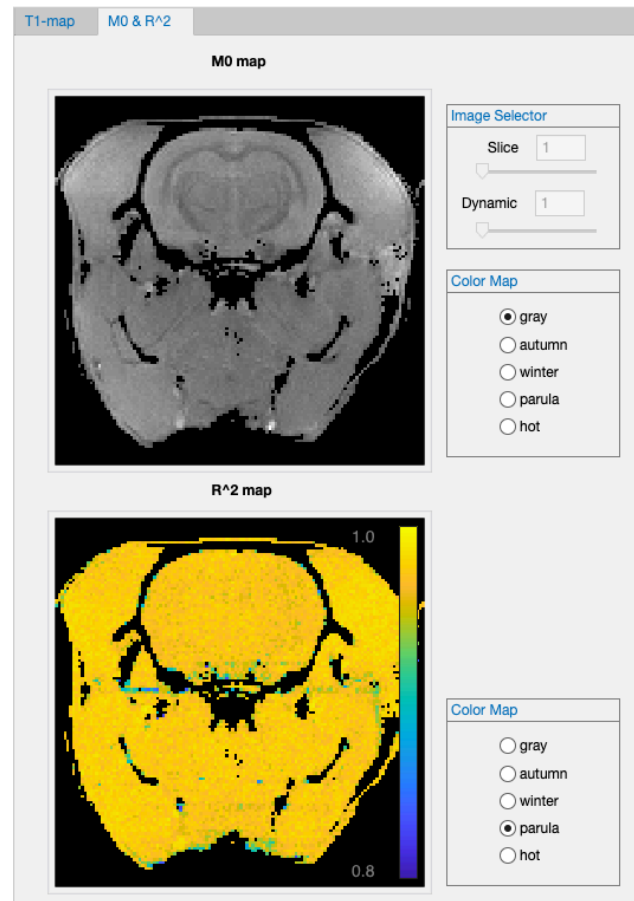
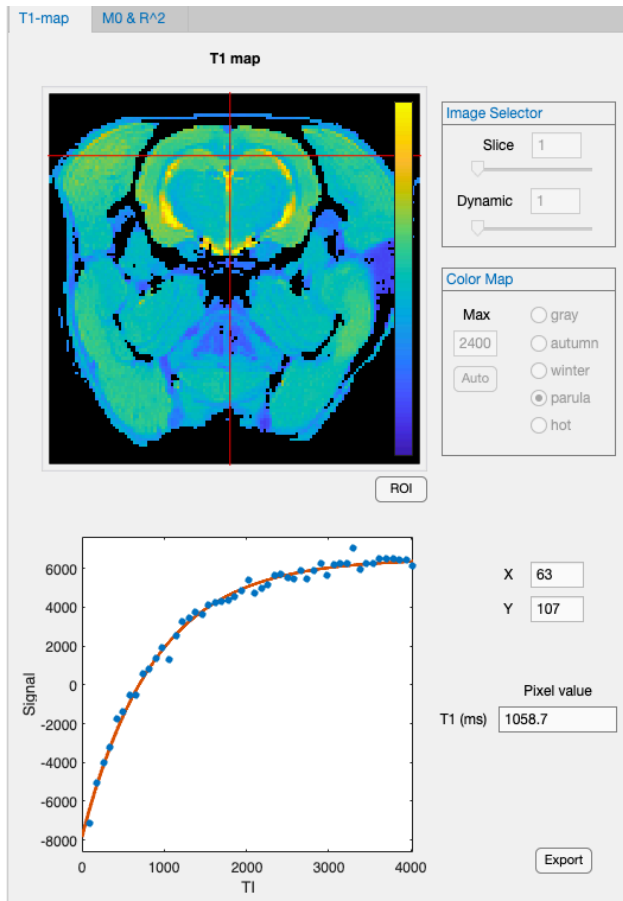
$$T_1 = (B/A - 1) T_1^*$$

## Model-based fitting

Thick the model-based checkbox  to perform a model-based reconstruction of the data using the Bart toolbox.

## Step 6: Fit results

The T1 maps will be shown in the T1-map tab. An M0 map and R<sup>2</sup> map (for least-squares fitting only) are shown in the second tab. A different color scheme can be chosen for the different maps.

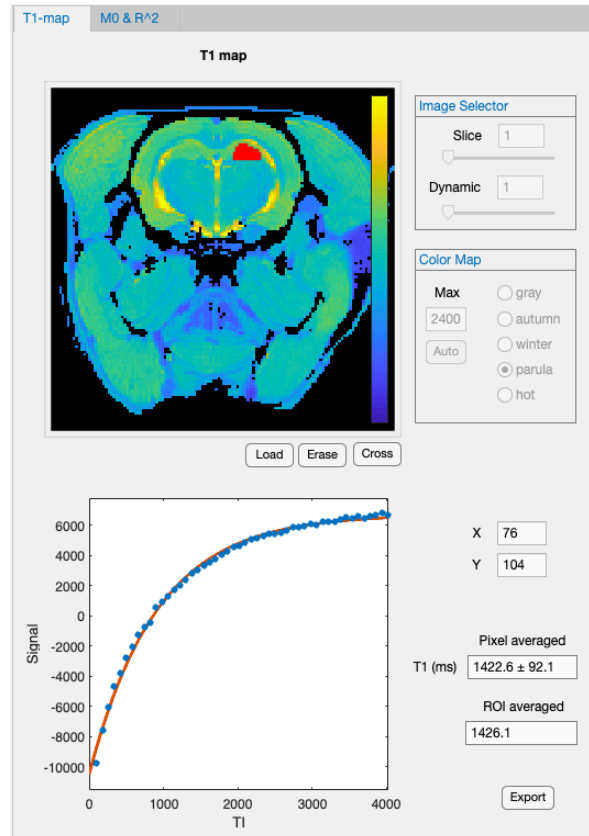
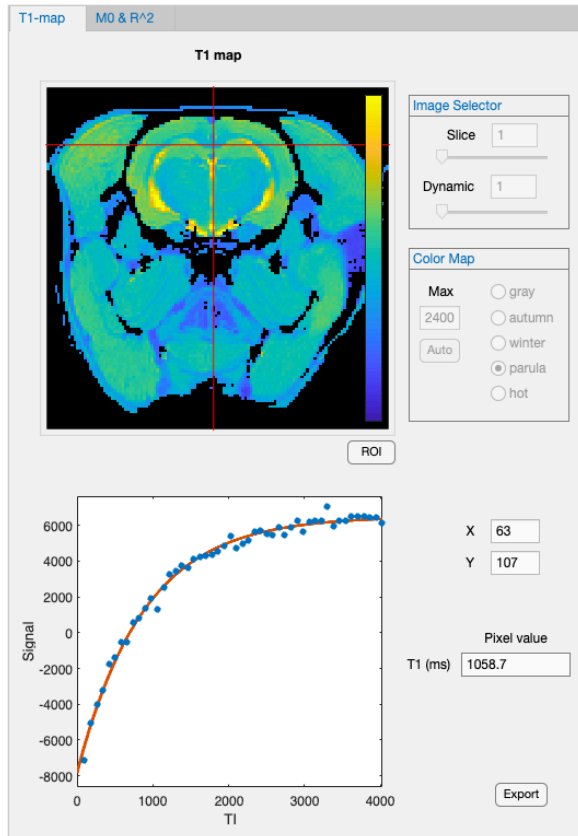




## Step 7: Extracting T1 values from pixels and ROIs

When you click on the T1 map with the mouse cursor a cross-hair will be shown. The Signal as function of T1 (symbols) and the fit result (red lines) of the cross-hair center pixel, as well as the fitted T1 value of that pixel will be shown.

A red symbol indicates that this T1 was omitted in the fitting process, according to the list in the Fit parameters tab (see step 5).



You can toggle between the cross-hair and region-of-interest (ROI) values by pressing the **ROI** button. A single ROI can be 'painted' on the T1 map in one or more slices by mouse clicking on the image.

For the ROI, two T1 values are calculated:

- Pixel averaged: The average of the T1 values in the pixels.  
 $T1_{ROI} = 1 / \text{Sum}(R1_{\text{pixel}})$
- ROI averaged: The T1 equation is fitted to the average signal intensities in the ROI, producing a single T1 value for that ROI.

The ROI can be removed with **Erase**.

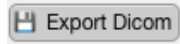
Alternatively you can load a NIFTI ROI file by pressing **Load**. The NIFTI ROI file should have the same matrix dimensions as the T1 map. Such a ROI can be made for example with ITKsnap (<http://www.itksnap.org>) using the generated DICOM images.



## Step 8: Exporting the T1 maps

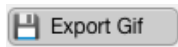
There are two ways to export the T1, M0, and R<sup>2</sup> maps.

### (1) **Export Dicom**



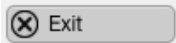
Exports the data in Dicom format for further processing in 3rd party software. The program searches for the Dicom information. If this information is not found, tags will be generated by the program itself. In the latter case the correct image position and orientation information are lost.

### (2) **Export Gif**



Exports the data in Gif format.

## Step 9: Exit

Press  to shut down the program.