**mpm-for-rodents installation and run on HPC**

**date: 12/14/23**

**CONTENTS**

**1) Download**

**2) Open HPC-frontend node via MobaXterm**

**3) Install ANTx2-toolbox**

**4) Install mpm-for-rodents-wrapper functions**

**5) Update antx2-toolbox and mpm\_rodent-toolbox**

**6) Download template**

**7) Add ANTx-paths**

**8) Go to study-folder**

**9) Create an ANTx project-file**

**10) Load the project-file 'proj.m'**

**11) Check whether the project-file loaded**

**12) Import BRUKER-data**

**13) Visualize files and folders**

**14) Selection of animals (not needed now, but later)**

**15) Modify the ANTx-project file**

**16) Set path of mpm\_rodent-toolbox**

**17) Create 'mpm'-dir & copy necessary files**

**18) Modify 'mpm\_NIFTIparameters.xlsx'**

**19) Modify 'mpm\_config.m'**

**20) Run all processing steps (interactive session)**

**21) SLURM SBATCH & non-interactive session**

**22) SLURM SBATCH & parallel processing**

**Get pre-orientations from command-line (easiest way)**

**MISC-1: Other command snips**

**1) Download**

**download MPM-installer**

Copy below link in web browser

<https://raw.githubusercontent.com/ChariteExpMri/mpm_rodent/master/updatempm.m>

save file as 'install\_mpm.m' (via ctrl+s) to path where the mpm-wrapper functions should be installed

My path is 'X:\ressources' (windows-path to HPC-storage )

**download ANTx2-installer**

Copy below link in web browser

<https://raw.githubusercontent.com/ChariteExpMri/antx2/master/installfromgithub.m>

save file as 'installfromgithub.m' (via ctrl+s) to path where the ANTx2-tbx should be installed

My path is 'X:\ressources' (windows-path to HPC-storage )

**download hMRI-toolbox-0.2.4**

Please download the version **hMRI-toolbox-0.2.4** (this was the only version which was tested):

<https://github.com/hMRI-group/hMRI-toolbox/releases/tag/v0.2.4>

For more information of hMRI-toolbox see: <https://www.cbs.mpg.de/abteilungen/neurophysik/software/hmri-toolbox>

-Unzip folder and store it on your local path

**download a template from googledrive**

🡪 see information below

**2) Open HPC-frontend node via MobaXterm**

**OPEN INTERACTIVE JOB on HPC (optional)**

srun --time 24:00:00 --mem=64G --ntasks=8 --pty bash -i

-Please check the parameters, here for time 24h, 64 Gb Ram, 8 cores

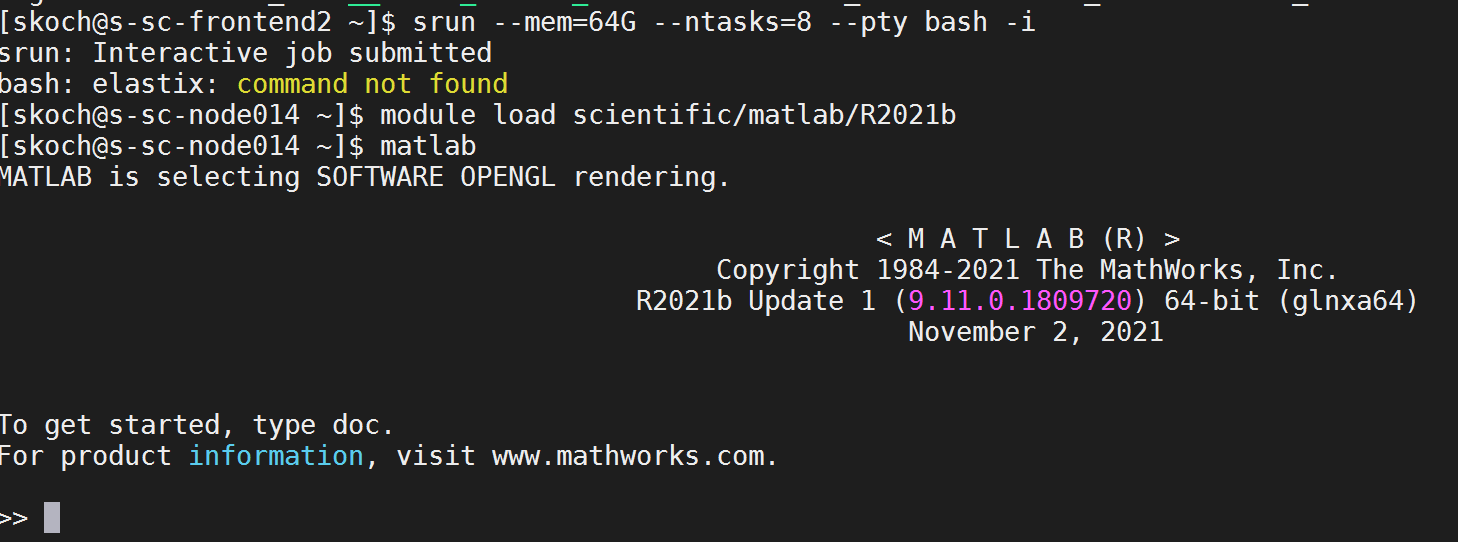
**LOAD MATLAB-MODULE AND START MATLAB (optional)**

module load scientific/matlab/R2021b

matlab

-Note: The access to Matlab might be different on your machine!

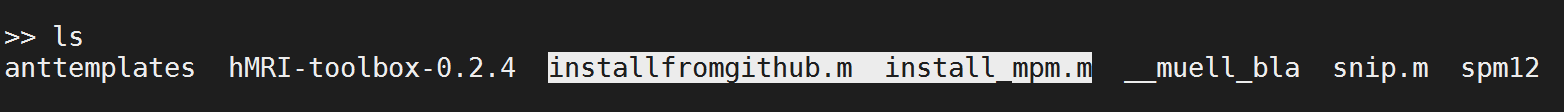
The below fig shows the Matlab-console from HPC-interactive session



**3) Install ANTx2-toolbox**

-go to path where file **'installfromgithub.m'** was stored. Note, this path is used for installation of ANTx2

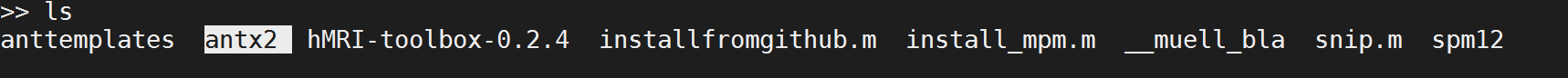
cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/



Type the following to install antx2:

installfromgithub('install','path',pwd)

When successful, the antx2-folder is created:



-for further information how to install ANTx2 see: <https://chariteexpmri.github.io/antxdoc/docs/tutorial_noGraphic_support.pdf>

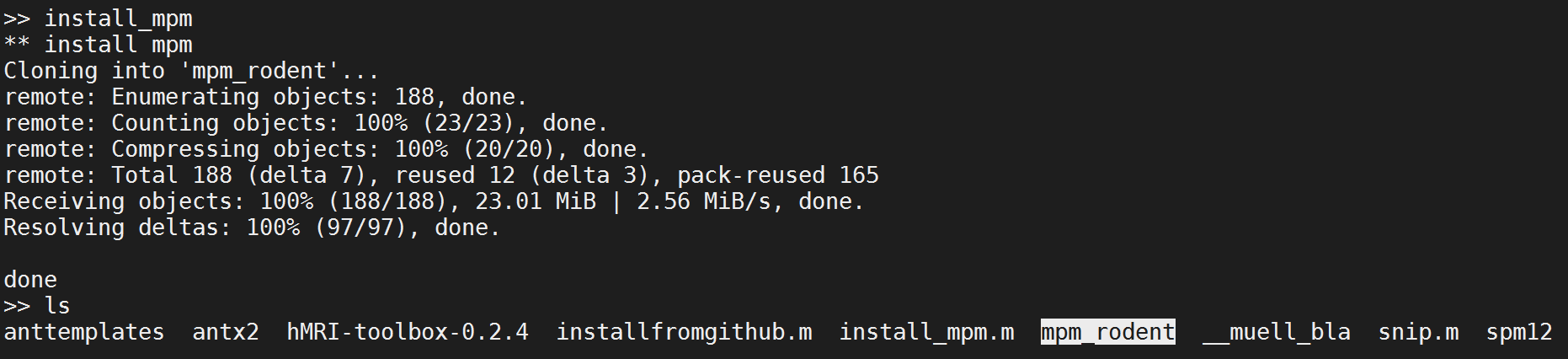
**4) Install mpm-for-rodents-wrapper functions**

-go to path where file '**install\_mpm.m'** was stored. Note this path is used for installation of mpm-wrapper-functions

Type the following to install mpm-wrapper-functions:

install\_mpm

When successful, the mpm\_rodent-folder is created:



**5) Update antx2-toolbox and mpm\_rodent-toolbox**

To update antx2 and the mpm\_rodent-toolbox execute the following commands:

% \*\*\* UPTATE ANTx2 \*\*\*

cd antx2

updateantx(3)

cd ..

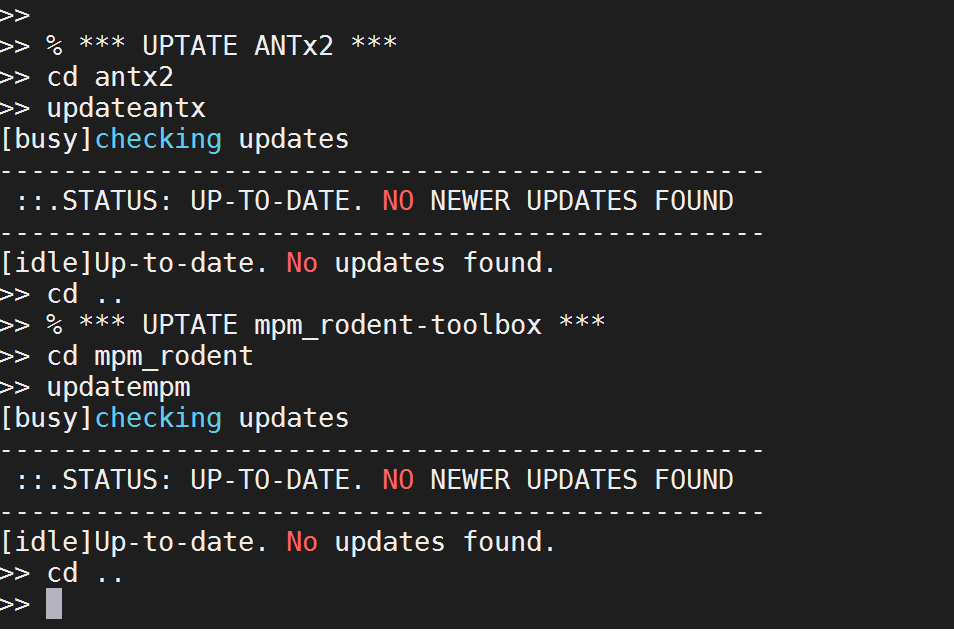
% \*\*\* UPTATE mpm\_rodent-toolbox \*\*\*

cd mpm\_rodent

updatempm(3)

cd ..

When done you should see the following:



**6) Download template**

**Go to** [**https://drive.google.com/drive/u/0/folders/1q5XOOVLvUYLqYsQJLqNRF7OK8fNwYhI9**](https://drive.google.com/drive/u/0/folders/1q5XOOVLvUYLqYsQJLqNRF7OK8fNwYhI9)

And download the template '**mouse\_Allen2017HikishimaLR'**

Create a new folder '**anttemplates**' at the same hierarchical level as the 'antx2'-folder

Unzip the 'mouse\_Allen2017HikishimaLR.zip' and store it without subfolders in the anttemplates-folder

i.e.:

good: 'myPATH/ressources/**anttemplates/mouse\_Allen2017HikishimaLR**'

bad: 'myPATH/ressources/**anttemplates/mouse\_Allen2017HikishimaLR/mouse\_Allen2017HikishimaLR**'

My anttemplates-folder looks as follows:

'/sc-projects/sc-proj-agtiermrt/Daten-2/ressources/**anttemplates'**│

├───**mouse\_Allen2017HikishimaLR**

│ ANO.nii

│ ANO.xlsx

│ AVGT.nii

│ AVGThemi.nii

│ AVGTmask.nii

│ parameter.m

│ readme.txt

│ \_b1grey.nii

│ \_b2white.nii

│ \_b3csf.nii

**7) Add ANTx-paths**

Go to the ANTx-path and link all necessary paths using '**antlink**'-command:

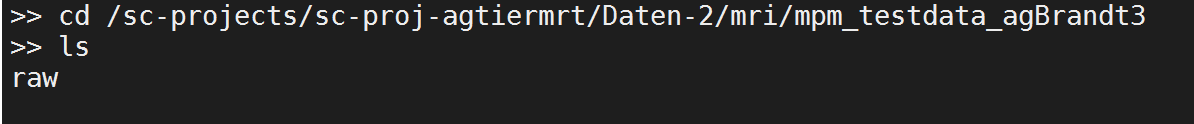
cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/antx2/

antlink

**8) Go to study-folder**

Create an empty study folder (here ' **mpm\_testdata\_agBrandt3**'). The study-folder is the folder where the processing of several animals of a study is performed.

**cd /sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3**



You can see that the project-folder contains a single folder 'raw', which contains the Bruker-raw data.

**9) Create an ANTx project-file**

Execute the following command

**makeproject('projectname',fullfile(pwd,'proj.m'), 'voxsize',[.07 .07 .07],'wa\_refpath','/sc-projects/sc-proj-agtiermrt/Daten-2/ressources/anttemplates/mouse\_Allen2017HikishimaLR','wa\_species','mouse');**

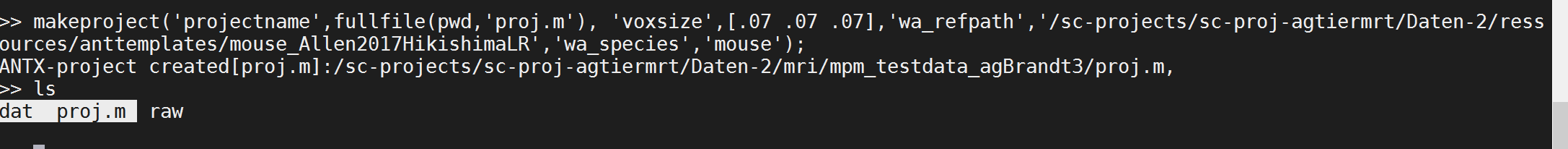
-This command creates a project-file '**proj.m**' with defined target voxel size of 0.07 x 0.07 x 0.07 mm, the **animal template** is '**mouse\_Allen2017HikishimaLR**', and the species is **'mouse'**.

NOTE: A suitable template can be downloaded from google-drive : <https://drive.google.com/drive/folders/1q5XOOVLvUYLqYsQJLqNRF7OK8fNwYhI9>

The template should be unzipped and stored where it could be reached (do not save the downloaded template in the current project-folder!).

**Creating a project-file has to be done only once!**

Aside the project-file '**proj.m**' an empty 'dat'-folder is created



**10) Load the project-file 'proj.m'**

Here we load the created project-file **'proj.m':**

loadconfig(fullfile(pwd,'proj.m'));

**11) Check whether the project-file loaded**

global an;

an

The global variable 'an' (struct) contains the basic information for this study. In the CMD-window you should the following:

project: 'NEW PROJECT'

datpath: '/sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3/dat'

voxsize: [0.0700 0.0700 0.0700]

wa: [1x1 struct]

**12) Import BRUKER-data**

Before doing this step, I just created the 'raw'-folder within the study's folder and copied the datasets of two animals into the '**raw'**-folder. First we just read the file-information of the Bruker raw-data from the 'raw'- folder (fullfile(pwd,'raw')) which is located in the current study folder.

The file-information will be stored in the resulting w-struct.

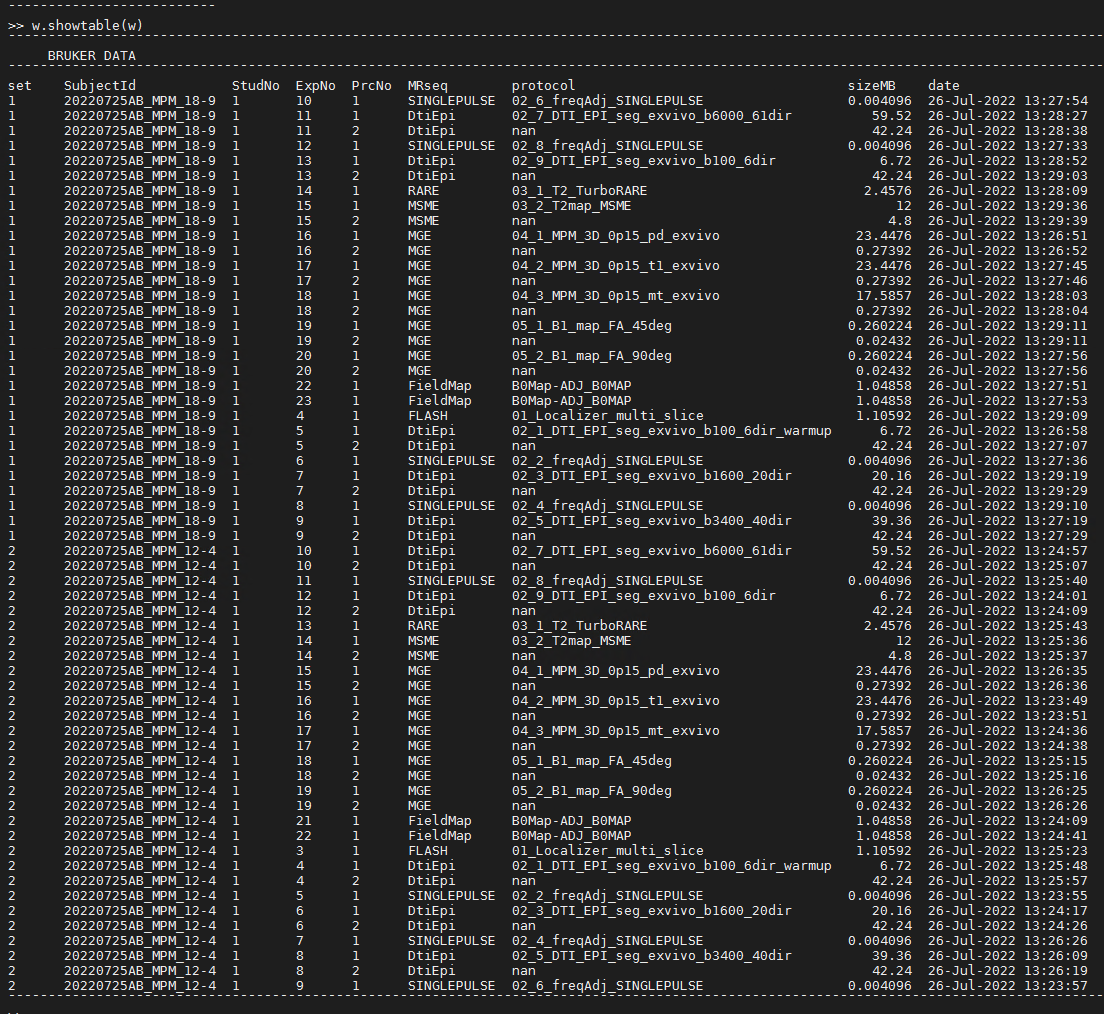
w=xbruker2nifti(fullfile(pwd,'raw'),0,[],[],'gui',0,'show',1); % first read all data and show it

The w-struct contains the file-information table 'd' with header 'hd'. This table is listed in the CMD-window when running this command. You could save & reload the struct and import some data later on (advantage: loading time is reduced). To show the table again just type:

w.showtable(w) ; %to show the table in CMD-window

If graphic is supported you could also visualize the table in an extra window via w.showtable2(w) .

Here, the raw-data folder contains two data-sets (i.e. the data from two animals):



Here we want to import the following files:

**03\_1\_T2\_TurboRARE**

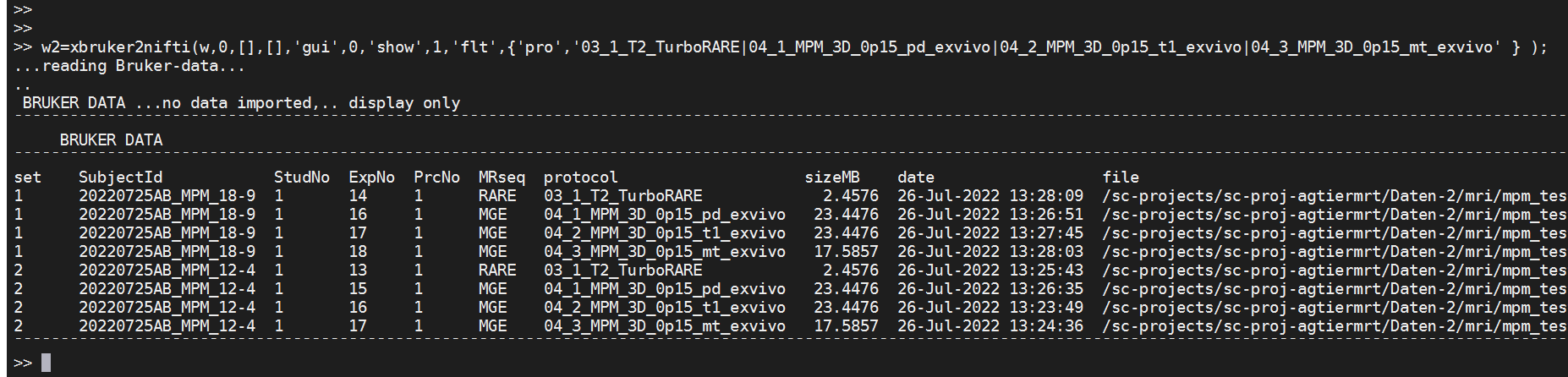
**04\_1\_MPM\_3D\_0p15\_pd\_exvivo**

**04\_2\_MPM\_3D\_0p15\_t1\_exvivo**

**04\_3\_MPM\_3D\_0p15\_mt\_exvivo**

To **visualize** the filtered table run the following command (note that the w-struct is used as 1st argument):

w2=xbruker2nifti(w,0,[],[],'gui',0,'show',1,'flt',{'pro','**03\_1\_T2\_TurboRARE|04\_1\_MPM\_3D\_0p15\_pd\_exvivo|04\_2\_MPM\_3D\_0p15\_t1\_exvivo|04\_3\_MPM\_3D\_0p15\_mt\_exvivo**' } );

****

Now let's import these data. For this just set the 'show'-parameter to 0:

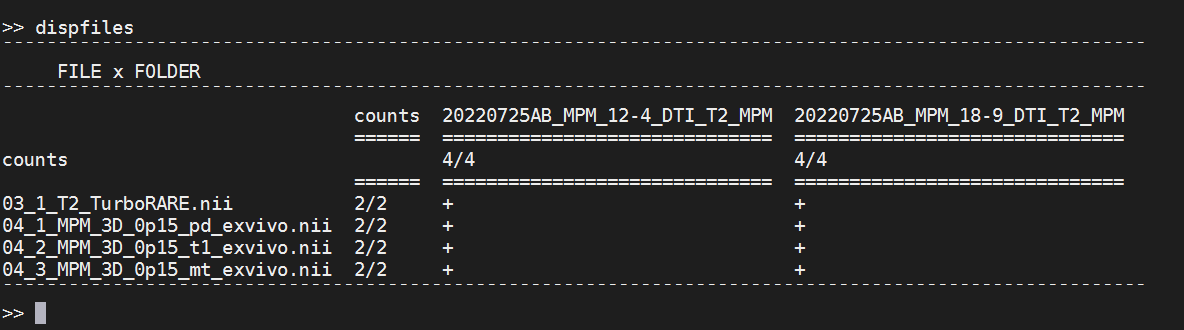
w2=xbruker2nifti(w,0,[],[],'gui',0,'show',**0**,'flt',{'pro','**03\_1\_T2\_TurboRARE|04\_1\_MPM\_3D\_0p15\_pd\_exvivo|04\_2\_MPM\_3D\_0p15\_t1\_exvivo|04\_3\_MPM\_3D\_0p15\_mt\_exvivo**' } );

**13) Visualize files and folders**

-type **dispfiles** to see a file-by-folder matrix in the command-window (see dispfiles for further help):

dispfiles

The following is displayed in the CMD-window:



Here we see that the study's '**dat'**-folder now contains two animal-folders ('**20220725AB\_MPM\_12-4\_DTI\_T2\_MPM**' and '**20220725AB\_MPM\_18-9\_DTI\_T2\_MPM**'). Each Folder contains the imported files **('03\_1\_T2\_TurboRARE.nii', '04\_1\_MPM\_3D\_0p15\_pd\_exvivo.nii', '04\_2\_MPM\_3D\_0p15\_t1\_exvivo.nii' and '04\_3\_MPM\_3D\_0p15\_mt\_exvivo.nii')**

**14) Selection of animals (not needed now, but later)**

This is a way to simple obtain the animal-directories

For example: to obtain **all** animal-folders from the 'dat'-folder:

mdirs=antcb('getallsubjects')

Note that the variable '**mdirs**' contains the full paths names of the selected animals:

>> mdirs=antcb('getallsubjects')

mdirs =

2x1 cell array

{'/sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3/dat/20220725AB\_MPM\_12-4\_DTI\_T2\_MPM'}

{'/sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3/dat/20220725AB\_MPM\_18-9\_DTI\_T2\_MPM'}

>>Alternative selection of all animals:

mdirs =antcb('selectdirs',[1:2 ]) ;% obtain path of the first two animals

mdirs =antcb('selectdirs',[1 ]) ;% obtain path of the 1st animal

mdirs =antcb('selectdirs',[2 ]) ;% obtain path of the 2nd animal

mdirs=antcb('selectdirs','all') ; % obtain path of all animals

**15) Modify the ANTx-project-file**

We have to modify the project-file **'proj.m'** because:

1) The 1st volume of **'04\_2\_MPM\_3D\_0p15\_t1\_exvivo.nii'** is used for registration to standard space (i.e. to the template 'mouse\_Allen2017HikishimaLR'). You can see that the orientation is different compared to the template ('AVGT.nii')

|  |  |
| --- | --- |
| The 1st volume of **'04\_2\_MPM\_3D\_0p15\_t1\_exvivo.nii'** | **'AVGT.nii' of**  **'mouse\_Allen2017HikishimaLR'** |

I.e. a proper initial re-Orientation is need:

For this, we have to set the '**orientType**' in the project-file to **[11]**

**x.wa.orientType = [11] ; % index from the reorientation Table**

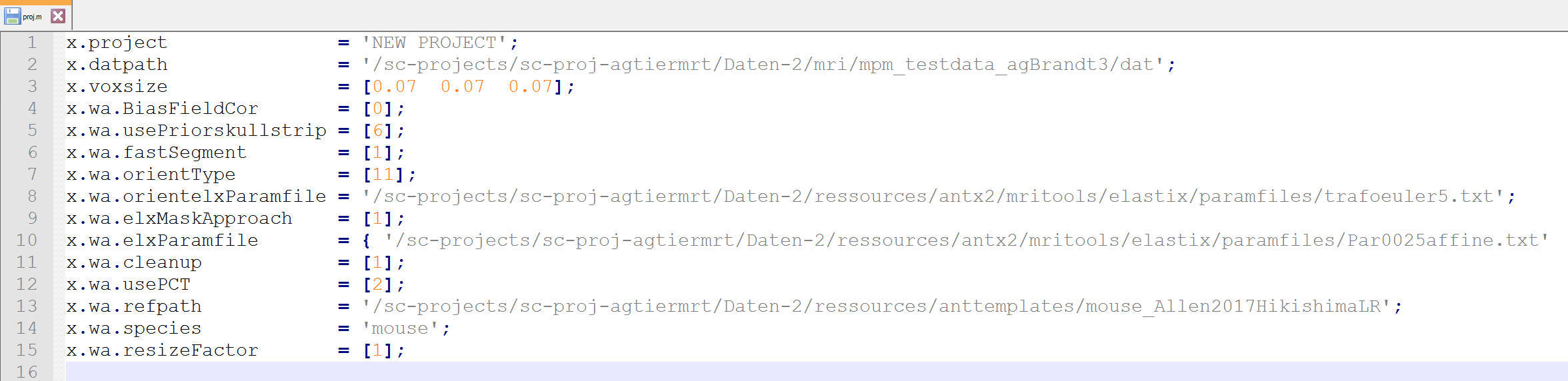
To obtain the pre-orientation see section **' Get pre-orientation from command-line (easiest way)'** below.

Also we see that the brain is skullstripped and embedded in Fomblin (exVivo-approach). Therefore we have to change the parameter **'usePriorskullstrip' to [6]** (otherwise use [1])

**x.wa.usePriorskullstrip = [6] ;% use a priori skullstripping (used for automatic registration)**

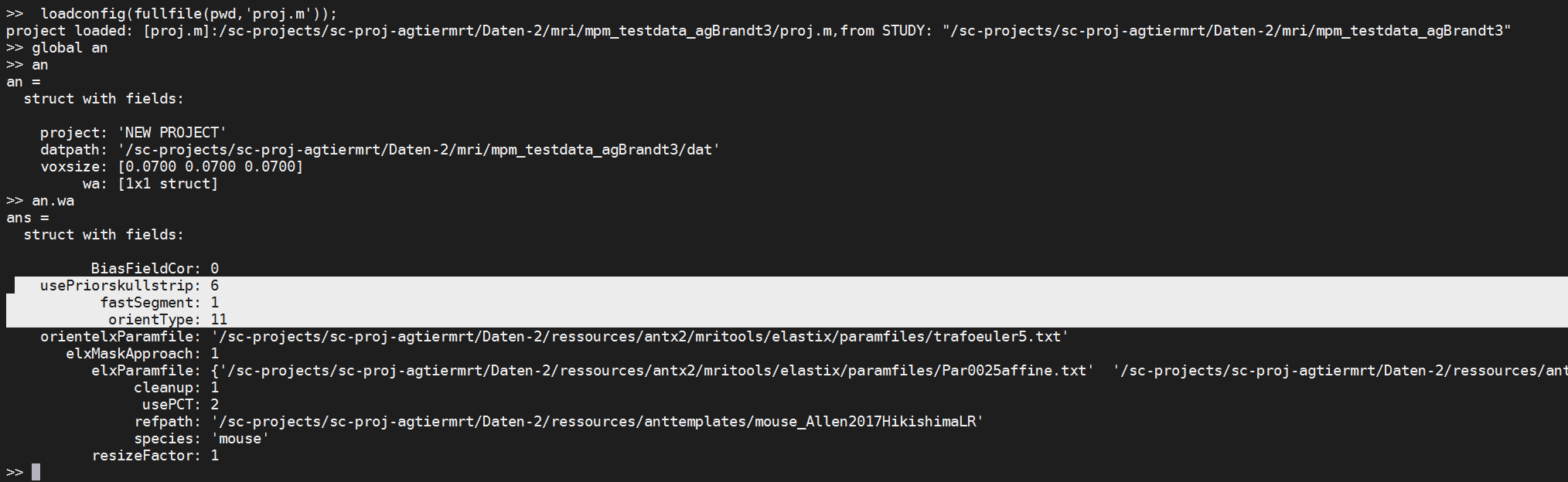
You have to set these two parameters manually in the proj.m-file

When done, the proj.m-file looks as follows:



We have to re-load the project-file **'proj.m':**

loadconfig(fullfile(pwd,'proj.m'));



**16) Set path of mpm\_rodent-toolbox**

Execute the following commands

pa\_wd=pwd; %get study-path (current path)

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/mpm\_rodent; % go to path of mpm-wrapper-functions

mpmlink ; %link path of mpm-wrapper-functions

cd(pa\_wd); %go back to study-path

**17) Create 'mpm'-dir & copy necessary files**

Now we need to create a subfolder **'mpm'** in the study's folder and copies of three files from the mpm-rodent-tbx:

**'mpm\_NIFTIparameters.xlsx'**

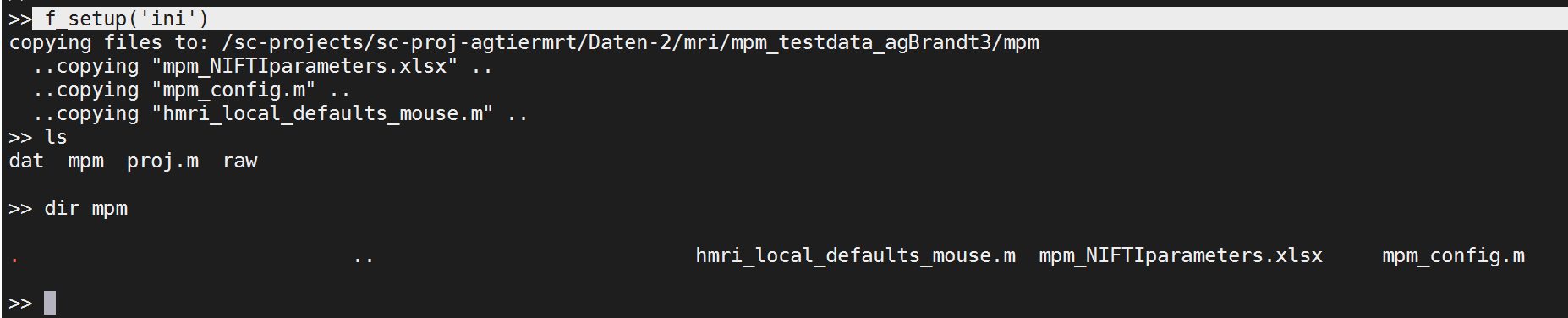
**'mpm\_config.m'**

**'hmri\_local\_defaults\_mouse.m'**

To create the 'mpm'-folder in the study-directory and copy these three files execute the following command. (**Important**: the current folder must be the study-folder):

f\_setup('ini')

We can see that the mpm-folder with the three files was created:



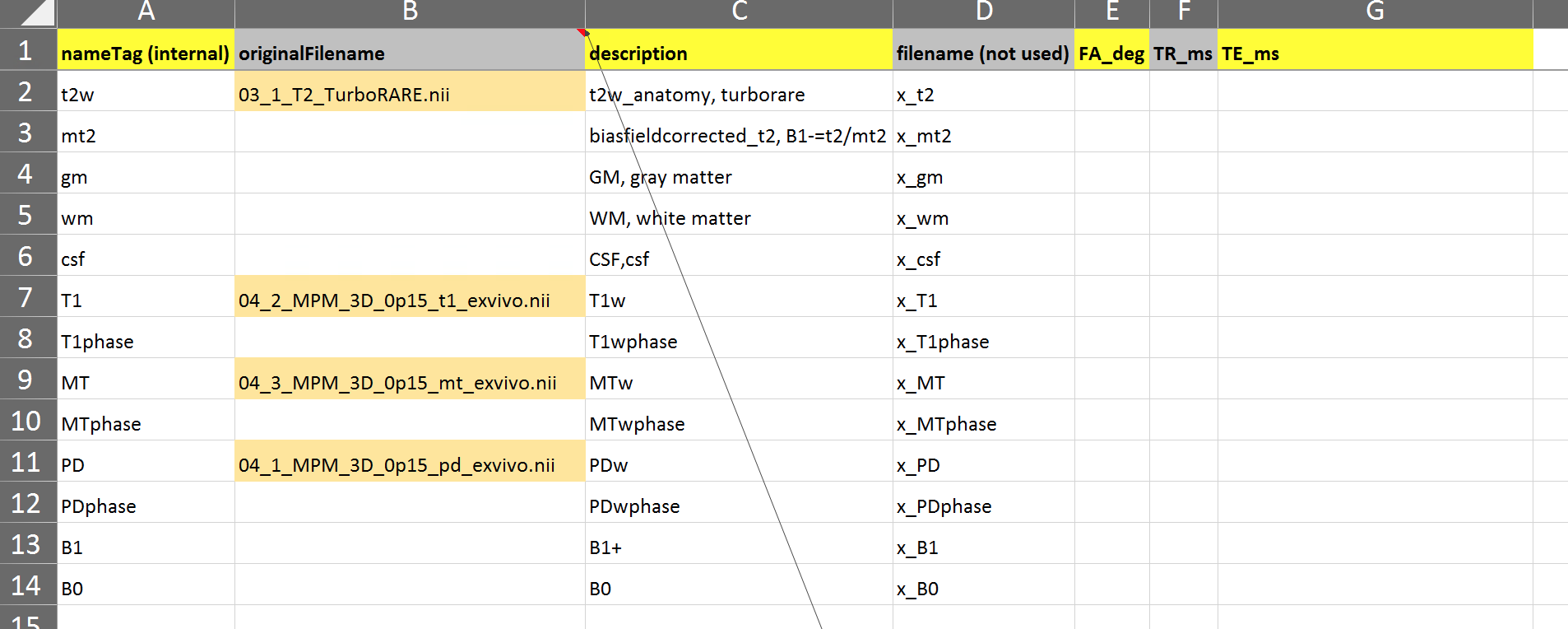
**18) Modify 'mpm\_NIFTIparameters.xlsx'**

Now, we have to modify the Excel-file '**mpm\_NIFTIparameters.xlsx**' manually.

Note that this file is located in the study's mpm-folder.

Specifically, we have to add the input-NIFTI-filenames (**'03\_1\_T2\_TurboRARE.nii', '04\_1\_MPM\_3D\_0p15\_pd\_exvivo.nii', '04\_2\_MPM\_3D\_0p15\_t1\_exvivo.nii' and '04\_3\_MPM\_3D\_0p15\_mt\_exvivo.nii'**) in the proper row of column-2.

(See fig below). When done, save excel file.

****

**19) Modify 'mpm\_config.m'**

We next have to modify the file 'mpm\_config.m' (Note that this file is located in the study's mpm-folder):

1) We have to set the path of the hmri-toolbox in the variable **'mpm.MPM\_path'**

Here, I set the path as follows:

mpm.MPM\_path ='/sc-projects/sc-proj-agtiermrt/Daten-2/ressources/hMRI-toolbox-0.2.4'

2) Here we use the SPM-version from the ANTx2-toolbox. Therefore, the variable **'mpm.SPM\_path'** must me empty (''):

mpm.SPM\_path ='' ;%SPM-toolbox path

3) The field **'mpm.t2w\_preorient'** has to be set. This variable defines the difference in orientation from the turborare-image (t2w) to the **'04\_2\_MPM\_3D\_0p15\_t1\_exvivo.nii'-image** (T1). Later, the turborare-image will be registered to the T1/PD/MT-images. For this, a proper reorientation is needed. Here the reorientation is **[1.5708 0 1.5708]**

Thus the following is set:

mpm.t2w\_preorient=[1.5708 0 1.5708]; % preorientation (rotations) of turborare (t2) to match orientation with PD/T1/MT-images

To obtain the pre-orientation see section **' Get pre-orientation from command-line (easiest way)'** below.



When done, save file.

**20) Run all processing steps (interactive session)**

**Execute the following commands to run all processing steps**

Run all processing steps (interactive mode)

mpm('steps'); %display processing steps,functions and numeric indicators

cd /sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3

loadconfig(fullfile(pwd,'proj.m')); %load ANTX-project of current study

v=struct();

v.mpm\_configfile = fullfile(pwd,'mpm','mpm\_config.m' ); % mpm-configfile

v.antx\_configfile = fullfile(pwd,'proj.m' ); % antx-projectfile(configfile)

v.pstep = 'all' ;%preprocessing steps,'all' or use indices([1:5]; see mpm('steps')

v.hstep = 'all' ;%hmri-processing steps,'all' or use indices([1:5]; see mpm('steps');

v.mdirs=antcb('getallsubjects') ; % path of animals-DIRs to process

%or hardcode animals to process

%v.mdirs={?

%'/sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3/dat/20220725AB\_MPM\_12-4\_DTI\_T2\_MPM'

%'/sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3/dat/20220725AB\_MPM\_18-9\_DTI\_T2\_MPM'

%}

mpm('nogui',v) ; % run proccessing steps without GUI

**21) SLURM SBATCH & non-interactive session**

Using a bash-starterfile (\*.sh) and an additional matlab-script we can run all steps via the SLURM **sbatch** command.

**1) Matlab-starterScript**

First, we need a matlab-script, here named 'runHPC\_main.m', which is located in the study's folder

The file '**runHPC\_main.m**' looks as follows:

disp('\*\*\*STARTERSCRIPT: MPM on HPC \*\*\*')

disp(['path' pwd]);

%% ======================================

%% \*\*\* LINK ANTx-PATHs \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%% ======================================

disp('LINK ANTx-PATHs');

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/antx2/ %go to ANTx-path

antlink ; %link path of antx2

disp(['PATH-ANTx2:' which('ant.m') ]);

%% ======================================

%% \*\*\* LINK mpm-wrapper-functions-PATHs \*\*\*

%% ======================================

disp('LINK mpm-wrapper-functions-PATHs');

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/mpm\_rodent; % go to path of mpm-wrapper-functions

mpmlink ; %link path of mpm-wrapper-functions

disp(['PATH-mpmWF:' which('mpm.m') ]);

if 1

timex=tic ;% capture START-TIME

cd /sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3 % go to current study

loadconfig(fullfile(pwd,'proj.m')); %load ANTX-project of current study

v=struct();

v.mpm\_configfile = fullfile(pwd,'mpm','mpm\_config.m' ); % mpm-configfile

v.antx\_configfile = fullfile(pwd,'proj.m' ); % antx-projectfile(configfile)

v.pstep = 'all' ;%preprocessing steps,'all' or use indices([1:5]; see mpm('steps')

v.hstep = 'all' ;%hmri-processing steps,'all' or use indices([1:5]; see mpm('steps');

v.mdirs=antcb('getallsubjects') ; % path of animals-DIRs to process

%or hardcode animals to process

%v.mdirs={â€¦

%'/sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3/dat/20220725AB\_MPM\_12-4\_DTI\_T2\_MPM'

%'/sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3/dat/20220725AB\_MPM\_18-9\_DTI\_T2\_MPM'

%}

mpm('nogui',v) ; % run proccessing steps without GUI

disp(sprintf('FINISHED: ET=%2.2fh',toc(timex)/(60\*60))); %DISPLAY ELAPSED TIME

end

exit; %iMPORTANT, otherwise HPC would run until time-out

**2) starter-shellscript**

We also need a shellscript (here named **'batch\_MPM\_agbrandt.sh'**) which is located in the HPC user's workspace. The shellscript does the following jobs: set HPC parameters, go to the study's folder and run the '**runHPC\_main.m**' file via Matlab

#!/bin/bash

#SBATCH --job-name=mpm #Specify job name

#SBATCH --output=mpm.o #FileName of output with %A(jobID) and %a(array-index);(alternative: .o%j) output=mpm.o%A\_%a

#SBATCH --error=mpm.e #FileName of error with %A(jobID) and %a(array-index);alternative: .e%j) error=mpm.e%A\_%a

#SBATCH --partition=compute # Specify partition name

#SBATCH --nodes=1 # Specify number of nodes

#SBATCH --cpus-per-task=50 # Specify number of CPUs (cores) per task

#SBATCH --time=48:00:00 # Set a limit on the total run time; example: 22:00:00(22hours) or 7-00(7days)

#SBATCH --array=1 # Specify array elements (indices), i.e. indices of of parallel processed dataSets

module load scientific/matlab/R2021b

#matlab

#matlab -nodisplay -r "./my\_simulation ; exit"

#source: https://arc.umich.edu/software/matlab/#:~:text=To%20run%20a%20MATLAB%20script,m%20from%20the%20current%20directory.&text=Note%20that%20the%20MATLAB%20script,an%20exit%20command%20in%20it.

cd /sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3

matlab -r runHPC\_main

**3) run via sbatch-command**

We can finally run the shellscript via the sbatch command from the HPC-front-Node

sbatch **batch\_MPM\_agbrandt.sh**

****

To check the progress/error/output you can use the following commands

squeue –me 🡪 check whether job is pending/running or finished

cat mpm.o 🡪 to check output

cat mpm.e 🡪 to check errors

**22) SLURM SBATCH & parallel processing**

We can slightly change the bash-starterfile and the Matlab file from the previous section and let the animals run in parallel. Using a bash-starterfile (\*.sh) and an additional matlab-function we can run all steps via the SLURM **sbatch** command. Instead of a Matlab-starterScript we use a **Matlab**-**function** and use the SLURM-array-ID as first input-argument. The SLURM-array-ID works as 'parallelizer' and is used to assigns the respective animal.

**1) Matlab-starterFunction (parallel processing)**

First, we need a matlab-Function(**'runHPC\_main\_parallel.m'**), which is located in the study's folder.

Important: The 1st input argument (animalNo) is the SLURM-array-ID (transferred from the shellscript). AnimalNo is a numeric value and can be 1 or 2 because we have two animals (this is defined in the shellscript: see line with '#SBATCH --array=1-2').

The file **'runHPC\_main\_parallel.m'** looks as follows:

function runHPC\_main\_parallel(animalNo)

disp('\*\*\*STARTERSCRIPT: MPM on HPC \*\*\*')

disp(['path' pwd]);

%% ======================================

%% \*\*\* LINK ANTx-PATHs \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

%% ======================================

disp('LINK ANTx-PATHs');

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/antx2/ %go to ANTx-path

antlink ; %link path of antx2

disp(['PATH-ANTx2:' which('ant.m') ]);

%% ======================================

%% \*\*\* LINK mpm-wrapper-functions-PATHs \*\*\*

%% ======================================

disp('LINK mpm-wrapper-functions-PATHs');

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/mpm\_rodent; % go to path of mpm-wrapper-functions

mpmlink ; %link path of mpm-wrapper-functions

disp(['PATH-mpmWF:' which('mpm.m') ]);

if 1

timex=tic ;% capture START-TIME

cd /sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3 % go to current study

loadconfig(fullfile(pwd,'proj.m')); %load ANTX-project of current study

v=struct();

v.mpm\_configfile = fullfile(pwd,'mpm','mpm\_config.m' ); % mpm-configfile

v.antx\_configfile = fullfile(pwd,'proj.m' ); % antx-projectfile(configfile)

v.pstep = 'all' ;%preprocessing steps,'all' or use indices([1:5]; see mpm('steps')

v.hstep = 'all' ;%hmri-processing steps,'all' or use indices([1:5]; see mpm('steps');

mdirs=antcb('getallsubjects') ; % path of animals-DIRs to process

v.mdirs=mdirs(animalNo); %select animal via SLURM-job-array-index

disp(['executing: running animal: ' char(v.mdirs) ]);

mpm('nogui',v) ; % run proccessing steps without GUI

disp(sprintf('FINISHED: ET=%2.2fh',toc(timex)/(60\*60))); %DISPLAY ELAPSED TIME

end

exit; %iMPORTANT, otherwise HPC would run until time-out

**2) starter-shellscript (parallel processing)**

We also need a shellscript (here named '**batch\_MPM\_parallel\_agbrandt.sh'**) which is located in the HPC user's workspace. The shellscript does the following jobs: set HPC parameters, go to the study's folder and run the **'runHPC\_main\_parallel.m'** file via Matlab.

The difference to the previous session is that that the animals will be processed at the same time.

Important lines are:

|  |  |
| --- | --- |
| output=mpmPP.o%A\_%a  error=mpmPP.e%A\_%a | The output/error-Filename is created via:  "mpmPP"+"o"/"e"+"jobID"+" arrayIndex"  For example:  The output-LOG-files of the two animals will be:  'mpmPP.o1662013\_1' & 'mpmPP.o1662013\_2'  The error-LOG-files of the two animals will be:  'mpmPP.e1662013\_1' & 'mpmPP.e1662013\_2' |
| #SBATCH --array=1-2 | Here we specify the number of array elements—in our case dataSets  Because we have two animals the command is: #SBATCH --array=1-2 |
| matlab -r "runHPC\_main\_parallel($SLURM\_ARRAY\_TASK\_ID)" | The function 'runHPC\_main\_parallel' is started via Matlab and the variable $SLURM\_ARRAY\_TASK\_ID is used as 1st input-argument  -$SLURM\_ARRAY\_TASK\_ID is a numeric value and can be only 1 or 2 because of: "SBATCH --array=1-2" "  -quotes are required to keep the parentheses from being interpreted |

The file '**batch\_MPM\_parallel\_agbrandt.sh'** looks as follows:

#!/bin/bash

#SBATCH --job-name=mpmPP #Specify job name

#SBATCH --output=mpmPP.o%A\_%a #FileName of output with %A(jobID) and %a(array-index);(alternative: .o%j) output=mpm.o%A\_%a

#SBATCH --error=mpmPP.e%A\_%a #FileName of error with %A(jobID) and %a(array-index);alternative: .e%j) error=mpm.e%A\_%a

#SBATCH --partition=compute # Specify partition name

#SBATCH --nodes=1 # Specify number of nodes

#SBATCH --cpus-per-task=50 # Specify number of CPUs (cores) per task

#SBATCH --time=48:00:00 # Set a limit on the total run time; example: 22:00:00(22hours) or 7-00(7days)

#SBATCH --array=1-2 # Specify array elements (indices), i.e. indices of of parallel processed dataSets

module load scientific/matlab/R2021b

cd /sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3

matlab -r "runHPC\_main\_parallel($SLURM\_ARRAY\_TASK\_ID)" # quotes are required to keep the parentheses from being interpreted by the shell

**3) run sbatch-command (parallel processing)**

We can finally run the shellscript via the sbatch command from the HPC-front-Node

sbatch **batch\_MPM\_parallel\_agbrandt.sh**

**MISC-1: Other command snips**

**Start interactive session & start Matlab, link path of antx2 and mpm-rodent, go to studyfolder,**

srun --time 24:00:00 --mem=64G --ntasks=8 --pty bash -i

module load scientific/matlab/R2021b

matlab

cd /sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3

pa\_wd=pwd ; % get study-path (current path)

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/antx2/ %go to ANTx-path

antlink ; %link path of antx2

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/mpm\_rodent; % go to path of mpm-wrapper-functions

mpmlink ; %link path of mpm-wrapper-functions

cd(pa\_wd); % go back to study-path

**%------UPDATE ANTx and mpm-rodent TOOLBOXES**

pa\_wd=pwd ; % get study-path (current path)

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/antx2/ %go to ANTx-path

updateantx(3)

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/mpm\_rodent; % go to path of mpm-wrapper-functions

updatempm(3)

cd(pa\_wd); % go back to study-path

**%------start: interactive session, add Paths, load study's projectfile**

srun --time 24:00:00 --mem=64G --ntasks=8 --pty bash -i

module load scientific/matlab/R2021b

matlab

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/antx2/ %go to ANTx-path

antlink ; %link path of antx2

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/mpm\_rodent; % go to path of mpm-wrapper-functions

mpmlink ; %link path of mpm-wrapper-functions

cd /sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3

loadconfig(fullfile(pwd,'proj.m')); %load ANTX-project of current study

**%------start: interactive session, UPDATE ANTx and mpm-rodent TOOLBOXES**

srun --time 24:00:00 --mem=64G --ntasks=8 --pty bash -i

module load scientific/matlab/R2021b

matlab

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/antx2/ %go to ANTx-path

updateantx(3)

cd /sc-projects/sc-proj-agtiermrt/Daten-2/ressources/mpm\_rodent; % go to path of mpm-wrapper-functions

updatempm(3)

**%------obtain HTML-docs for pre-orintation**

cd /sc-projects/sc-proj-agtiermrt/Daten-2/mri/mpm\_testdata\_agBrandt3

loadconfig(fullfile(pwd,'proj.m')); %load ANTX-project of current study

**Get pre-orientations from command-line (easiest way)**

Before doing this step: updateTBX, link TBX-paths (ANTx & mpm); load ANTx-project , import Bruker-raw-data, fill fileNames in 'mpm\_NIFTIparameters.xlsx'

Note that the orientation of the T1/MT/PD-images differs from the standard-space (AVGT.nii/template-space). Moreover, the tubrborare-image (t2w) has also another orientation compared to T1/MT/PD-images. Unfortunately, this pre-alignment/pre-orientation task has to be done manually, but only once (per study).

The below commands create two HTML-files with different orientations of

(1) T1/MT/PD to standard-space (1st command)

(2) turborare (T2w) to T1/MT/PD:

mpm(**'orientSS'**,'mpm\_configfile',fullfile(pwd,'mpm','mpm\_config.m'));

mpm(**'orientT2w'**,'mpm\_configfile',fullfile(pwd,'mpm','mpm\_config.m'));

The HTML-files will be saved in the study's 'checks'-folder. Next, open this html-files in your browser!

|  |  |
| --- | --- |
| **A1) Change pre-orientation to standard-space**  1) When selection the file '###-AVGT.html' you see the following: | 2) select the best-matching orientation:  Here**: rotTable-Index [11]: which is '3.1416 0 1.5708'**    3) in the **ANT-projectfile** change the variable **x.wa.orientType** as follows:  **x.wa.orientType =** [11];  or  **x.wa.orientType = '3.1416 0 1.5708'**  4) **reload ANT-projectfile**  type: **antcb('reload')** |

|  |  |
| --- | --- |
| **A2) Change pre-orientation from t2w to T1/PD/MT**  1) When selection the file 'Reorient\_03\_1\_T2\_TurboRARE--04\_2\_MPM\_3D\_0p15\_t1\_exvivo.html' you see the following: | 2) select the best-matching orientation:  Here**: rotTable-Index [14]: which is '1.5708 0 1.5708'**    3) open the [**mpm\_config.m**]-file located in the mpm-folder and change the variable **mpm.t2w\_preorient** as follows:  **mpm.t2w\_preorient=[1.5708 0 1.5708];**  \*use here three **numeric** values [rotX rotY rotZ]  **4) reload mpm-toolbox**  type: **mpm('reload')** |