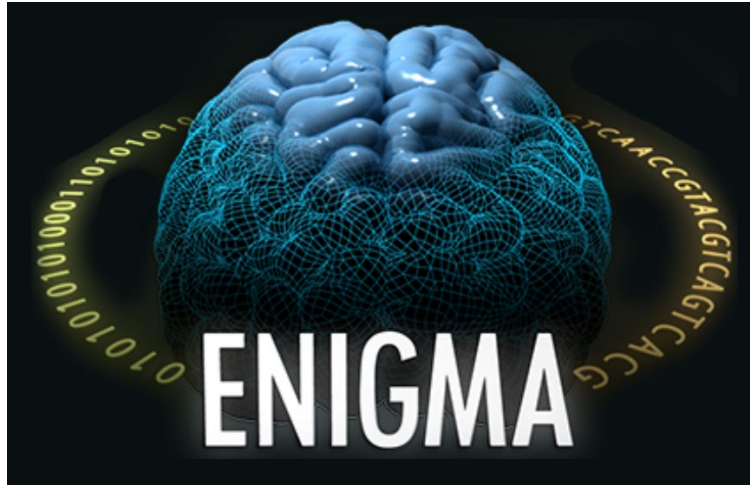


ENIGMA MRS

Osprey Standard Operating Procedures



Georg Oeltzschner
Johns Hopkins University
v1.0
February 4, 2025

Table of Contents

1	Introduction	3
1.1	Goals	3
1.2	Eligible datasets	3
2	Installation & Setup	3
2.1	System requirements	5
2.2	Setup instructions	5
2.3	Testing the LCModel binary	6
2.3.1	Compiling your own LCModel binary	6
3	Preparing your data	8
3.1	BIDS folder organization	9
3.2	BIDS file naming conventions	12
3.2.1	Metabolite data	12
3.2.2	Water reference data	15
3.2.3	Structural (T1-weighted) images	15
4	Setting up the Osprey analysis	17
4.1	Osprey job file template	17
4.1.1	Data handling and modeling options	17
4.2	Metabolite data paths (files)	17
4.3	Water reference data paths (files_ref)	18
4.4	Structural (T1-weighted) image paths (files_nii)	19
4.5	LCModel basis set	21
4.6	LCModel control file	21
5	Running the analysis	22
5.1	Command line vs. executable	22
5.1.1	Running Osprey from the MATLAB window	22
5.1.2	Running the compiled Osprey executable	22
5.2	OspreyJob	22
5.3	OspreyLoad	22
5.4	OspreyProcess	22
5.5	OspreyFit	23
5.6	OspreyCoreg	23
5.7	OspreySeg	23
5.8	OspreyQuantify	23
6	Collecting the output data	23
6.1	Inspecting your data	24
6.2	Selecting the output files for transfer	24
6.2.1	Quality control metrics	25
6.2.2	Quantification results	25
6.2.3	Demographics	26
6.2.4	MRSinMRS checklist	25
6.2.5	Voxel co-registration & segmentation results & masks	26
6.3	Packaging up and sharing	Error! Bookmark not defined.

1 Introduction

Magnetic Resonance Spectroscopy (MRS) is a sensitive non-invasive technique to quantitatively measure brain metabolism using readily available clinical MRI scanners. Thousands of papers have been published that have successfully used this technique as a biomarker of pathology/injury and outcome across a broad range of clinical applications such as providing a virtual biopsy of brain tumors, prognosis of outcome after brain injury, early diagnosis of Alzheimer's disease, and can be the only means of non-invasively monitoring the altered metabolism as the result of genetic disorders, to name a few (Öz et al. 2014). However, the wider use of MR spectroscopy across CNS pathologies is hampered by the lack of standardization and adequate normative data (Choi and Kreis 2021). Unlike imaging methods such as structural MRI or diffusion tensor imaging, there has been little work to harmonize MRS methods, data, and analysis methods despite the great potential for quantitative, and informative biomarkers of disease.

ENIGMA (Enhanced NeuroImaging Genetics through MetaAnalysis) is a network that brings together researchers in imaging fields to share ideas, algorithms, data, and information. The ENIGMA MRS group previously published a technical introduction to the use of MRS in pediatric and adult traumatic brain injury, outlining several considerations and recommendations to improve studies within this injury group (Bartnik-Olson et al. 2021) and more recently has published on considerations of data harmonization across multiple sites, with considerations of prospective study designs as well as retrospective data aggregation (Harris et al. 2023). The MRS Group represents major academic sites including Harvard Medical School, Loma Linda University, Johns Hopkins University, University of Calgary, University of Florida, University of Dresden, Charité Berlin, and others. Through the combination of these sites, the ENIGMA MRS study group has a unique collection of what is likely the world's largest collection of MR spectroscopy data. In addition, each of the sites have collected hundreds of spectroscopy cases of normative data across ages.

1.1 Goals

It is well known that there are many technical factors that can influence MRS data quantification (e.g., acquisition sequence, field strength, analysis approach). However, rather than avoiding these sources of heterogeneity, our goal is to test a single analysis pipeline and account for this heterogeneity in the statistical analysis to develop a normative SVS database across the human lifespan.

1.2 Eligible datasets

All single-voxel spectroscopy (SVS) data, regardless of manufacturer, field strength, or sequence type, are eligible for inclusion. The Osprey data processing pipeline supports many different sequences and vendor-specific raw data formats. Initial analyses will focus on normative (healthy control) data from typically developing, healthy subjects where the study exclusion criteria included the absence of a previous medical diagnosis of brain injury, stroke, neuropsychiatric disorders.

1.3 Contact

If you have questions about the project, please contact the ENIGMA-MRS group leaders:

- Dr. Alexander Lin aplin@bwh.harvard.edu

- Dr. Brenda Bartnik-Olson bbartnik@llu.edu
- Dr. Ashley Harris ashley.harris2@ucalgary.ca.

For technical questions about the Osprey analysis pipeline, please submit a GitHub issue at the Osprey repository (<https://github.com/schorschinho/osprey/issues>) or contact

- Dr. Georg Oeltzschner goeltzs1@jh.edu

1.4 Contributors

This document has been drafted and edited by the following individuals:

Brenda Bartnik-Olson, Alexander P. Lin, Ashley D. Harris, Helge J. Zöllner, Georg Oeltzschner

2 Installation & Setup

2.1 System requirements

Osprey can generally be run on any modern personal computer with MATLAB. If you don't have a MATLAB license (or do not have access to the required toolboxes set out in Section 1.2), please read Section 1.4 for instructions on using the compiled Osprey executable.

For this project, Osprey is going to internally call a compiled LCMModel binary to carry out the modeling process. We have pre-compiled LCMModel binaries for several systems that should work out of the box, but please test on your system as described in Section 1.3. We recommend running the analysis on one of the following operating systems:

- macOS on Intel or M1 processors (versions), but not M2 processors
- Windows 10 or 11
- Ubuntu

If you encounter a problem during setup or analysis, please open a new GitHub issue in the Osprey repository.

2.2 Setup instructions

First, remove all toolboxes from your MATLAB path that might interfere with Osprey and SPM12, e.g., Gannet, FID-A, or FieldTrip.

Osprey currently requires MATLAB 2020b (or later) with the “Optimization” and “Statistics and Machine Learning” toolboxes installed.

Download **Osprey** version 2.9.0 (released November 26, 2024) from its [GitHub release repository](#).

Download **SPM12** [from the UCL website](#). If you run an Apple Silicon processor (M1 and later), please download the [SPM development version from GitHub](#) instead.

Make sure to regularly check for updates, as we frequently commit new features, bug fixes, and improved functions.

If you want to use the Osprey Graphical User Interface (GUI), please download the following toolboxes from the MATLAB File Exchange:

- [GUI Layout Toolbox](#) (David Sampson)
- [Widget Toolbox](#) (Robin Jackey)
- [Widgets Toolbox - Compatibility Support](#) (Robin Jackey)

Download the toolboxes in the MATLAB toolbox format (.mltbx). You can double-click to install. MATLAB will automatically add the toolboxes to its path.

It is recommended that you extract the Osprey and SPM12 code into your default MATLAB path, which you can determine by opening MATLAB and typing `userpath` at the prompt.

Before you start the analysis, you must add the entire Osprey folder (**with** subfolders) and the SPM12 folder (**without** subfolders) to your MATLAB path with the following MATLAB prompt commands

(replace ‘userpath’ with the appropriate path if you have stored them in a different directory than the default MATLAB path):

```
addpath(genpath(fullfile(userpath, 'osprey')));
addpath(fullfile(userpath, 'spm12'));
```

2.3 Testing the LCModel binary

While the Osprey processing software itself is written in MATLAB and should run anywhere MATLAB can run, for this project, it will call upon the external fitting tool ‘LCModel’ to carry out the spectral modeling. LCModel is a binary executable that needs to be compiled for every operating system and processor architecture.

Osprey comes pre-packaged with pre-compiled binaries for several systems (Linux, MacOS M1 Pro, MacOS Intel, Windows 10 64 bit). Before setting up your analysis, please test whether the LCModel executable runs on the system that you intend to run the analysis on:

- Download or clone the LCModel binary repository at <https://github.com/schorschinho/LCModel>
- Identify the executable in the **binaries** folder that most closely matches your system.
- Copy that executable into the **test_lcm** folder.
- Open a terminal/command window and navigate into the **test_lcm** folder.
- Run the test dataset analysis with one of the following commands:
 - `./lcmmodel < control.file` (Mac, Linux)
 - `lcmmodel < control.file` (Windows)
- Then, compare the Postscript output file (**out.ps**) with the gold-standard output (**out_ref_build.ps**). Apart from the date and the version (6.3-N vs. 6.3-R), the output (Figure 1) should be identical.

Conc.	%SD	/Cr+PCr	Metabolite
7.20E-08	166%	3.9E-02	Ala
5.64E-07	36%	0.309	Asp
1.20E-06	13%	0.659	Cr
2.92E-07	45%	0.160	GABA
6.78E-08	150%	3.7E-02	Glc
6.18E-07	31%	0.339	Gln
4.70E-07	12%	0.257	GSH
2.14E-06	8%	1.171	Glu
4.56E-07	4%	0.250	GPC
1.66E-06	5%	0.911	Ins
1.07E-07	116%	5.8E-02	Lac
1.91E-06	5%	1.047	NAA
4.74E-07	16%	0.260	NAAg
0.000	999%	0.000	PCh
6.22E-07	26%	0.341	PCr
1.48E-07	16%	8.1E-02	sIns
0.000	999%	0.000	Tau
0.000	999%	0.000	-CrCH2
4.56E-07	4%	0.250	GPC+PCh
2.38E-06	3%	1.306	NAA+NAAg
1.83E-06	3%	1.000	Cr+PCr
2.76E-06	8%	1.510	Glu+Gln
1.71E-07	172%	9.4E-02	Lip13a
0.000	999%	0.000	Lip13b
0.000	999%	0.000	Lip09
1.83E-06	12%	1.005	MM09
2.15E-08	389%	1.2E-02	Lip20
4.22E-06	9%	2.314	MM20
6.24E-07	30%	0.342	MM12
2.06E-06	21%	1.128	MM14
1.49E-06	21%	0.818	MM17
1.71E-07	172%	9.4E-02	Lip13a+Lip13b
2.85E-06	17%	1.563	MM14+Lip13a+L
1.83E-06	12%	1.005	MM09+Lip09
4.25E-06	9%	2.326	MM20+Lip20
NO DIAGNOSTICS			
MISCELLANEOUS OUTPUT			
FWHM = 0.084 ppm S/N = 21			
Data shift = 0.008 ppm			
Ph: 9 deg 2.2 deg/ppm			

Figure 1. Gold standard LCModel output for the test dataset.

2.3.1 Compiling your own LCModel binary

If you receive an error message at Step 6, it likely means that the executable is not a suitable match for your system architecture or that some vital internal libraries are missing (although we believe that we have everything included).

In that case, you will need to compile the LCModel executable yourself. This sounds more daunting than it is – it should entail nothing more than downloading and installing a GFortran-compatible compiler for your system and then executing 1-2 lines of compiler code on the command line. Please refer to the instructions at <https://github.com/schorschinho/LCModel> for details. For support, please open an issue in that GitHub repository.

If you compile a new binary for a system architecture that we do not have in the repository, please submit a pull request or otherwise share with us so we can expand the executable library.

Once you have a new LCModel binary compiled, you need to modify your job file and let Osprey know where to find this binary (see Section 4.1.1.4 *Path to LCModel executable*).

3 Preparing your data

We **strictly require** that you organize your raw data according to the folder and file naming structure outlined below. This may require some re-organization effort from you, as you probably store and organize your data in a lab-specific fashion according to your own conventions. The purpose of this strict structure is two-fold:

- Automate and harmonize the Osprey analysis to the highest possible degree while minimizing the risk of human error.
- Standardize the input, output, and metadata to the highest possible degree to minimize the need for additional data wrangling effort during statistical analysis.

We have therefore adopted the folder and file hierarchy and naming conventions [proposed by the BIDS \(Brain Imaging Data Structure\) initiative](#). Very briefly, there are four main levels of the folder hierarchy:

```
project/  
└─ subject  
   └─ session  
      └─ acquisition
```

Except the top-level **project** folder, all sub-folders have a specific structure to their name, representing different subjects with running indices, different sessions with running indices, and unique identifiers for different imaging modalities. Here's an example for the first session (**ses-01**) of the first subject (**sub-001**):

```
myProject/  
└─ sub-01  
   └─ ses-01  
      └─ anat  
         └─ mrs
```

Here, **anat** and **mrs** are the BIDS folder naming conventions for the modalities “anatomical scans” and “MRS scans” (for a complete list of data types, please refer to <https://bids-specification.readthedocs.io/en/bep022/common-principles.html>). All scans of the respective modality will be stored inside these folders and differentiated by their filenames – no subfolders for regions, within-session repetitions, etc. For details about naming files, see Sections 2.2.1, 2.2.2, and 2.2.3.

We realize that different sites will contribute different types of data, i.e., some will have structural images and water references, while others won't. We have branched out the workflows to set up the analysis job file (Section 3), to run the analysis (Section 4), and to select the output files (Section 5) according to the data you have available.

(Note: In general, Osprey does not make a lot of assumptions about your folder structure, since you can specify the exact location for each file in a job file prior to each analysis. For your other projects, you can organize your data however you want, although BIDS is still highly recommended).

3.1 Converting raw data to NIfTI

We **strongly recommend** (but do not require) converting your raw MRS data to the new NIfTI-MRS format. This simplifies the creation of the Osprey job file enormously.

3.1.1 Installing and running spec2nii

The open-source tool spec2nii (developed by Will Clarke) supports the most commonly used data formats and sequences of the main vendors. Follow the installation instructions at https://open.win.ox.ac.uk/pages/fsl/fsl_mrs/install.html (spec2nii requires at least Python 3.8).

The basic spec2nii command usage looks like

```
spec2nii [FORMAT] [OPTIONS] [FILENAME]
```

Here, FORMAT is a string describing the format you're converting from (rda, twix, dicom, etc.), OPTIONS specifies several options for conversion, and FILENAME represents the filename of the original raw data file you intend to convert from.

We recommend using the -j option to automatically generate the NIfTI-MRS JSON sidecar metadata file.

-f FILEOUT and -o OUTDIR are optional (if not provided, spec2nii defaults to using the name and directory of the input file).

3.1.2 Converting Philips SPAR/SDAT

For conventional short-TE data, you should use the following command:

```
spec2nii philips [-j] [-f FILEOUT] [-o OUTDIR] sdatfilename sparfilename

-j                creates the JSON sidecar file
-f FILEOUT        creates the output file with the name FILEOUT
-o OUTDIR         creates the output file in the directory OUTDIR
sdatfilename      path to the input sdat file
sparfilename      path to the input spar file
```

For example, the following script creates the files test001.nii.gz and test001.json in the same directory as test001.spar and test001.sdat.

```
spec2nii philips -j test001.sdat test001.spar
```

Metabolite (_raw_act.sdat) and water reference (_raw_ref.sdat) data need to be converted separately. Append the suffixes `svs` and `ref` to the output file names as described in sections 3.3.1 and 3.3.2.

3.1.3 Converting Siemens RDA

For conventional short-TE data, you should use the following command:

```
spec2nii rda [-j] [-f FILEOUT] [-o OUTDIR] rdafilename

-j                creates the JSON sidecar file
-f FILEOUT        creates the output file with the name FILEOUT
-o OUTDIR         creates the output file in the directory OUTDIR
rdafilename       path to the input rda file
```

For example, the following script creates the files test001.nii.gz and test001.json in the same directory as test001.rda.

```
spec2nii rda -j test001.rda
```

Metabolite and water reference data need to be converted separately. Append the suffixes `svs` and `ref` to the output file names as described in sections 3.3.1 and 3.3.2.

3.1.4 Converting Siemens TWIX

The content of TWIX data varies considerably based on the sequence implementation and Siemens software version. First, inspect your data by typing

```
spec2nii twix -v twixfilename

-v          View contents of twix file, no files converted
twixfilename path to the input twix file
```

This shows a list of contained MDH flags and multi-RAID files. You usually only want the actual data (but not noise scans), which should be contained in the RAID file called 'image' (typically the second file). If the above command returns something like (note the `image` in the last line):

```
pymapVBVD version 0.5.7
Software version: VD
Contents of file test001.dat:
Multiraid file, 2 files found.
Selecting file 2. Use -m option to change.
The file contains these evalinfo flags with dimensions and sizes as follows:
image      :      Col, Cha, Ave, Phs      [2080  32  64  2]
```

Then, you can extract the data with

```
spec2nii twix -e image -j twixfilename

-e image    converts the data contained in `image`
-j          creates the JSON sidecar file
twixfilename path to the input twix file
```

For example, the following script creates the files test001.nii.gz and test001.json in the same directory as test001.dat.

```
spec2nii twix -e image -j test001.dat
```

Metabolite and water reference data need to be converted separately. Append the suffixes `svs` and `ref` to the output file names as described in sections 3.3.1 and 3.3.2.

Note that some newer sequences (for example from CMRR) may have the water reference data included in the same TWIX file as the metabolite data. In that case, they are likely to be stored as separate RAID files, and will need to be extracted separately (please contact us if you struggle with that).

3.1.5 Converting GE P-files

For conventional short-TE data, you should use the following command:

```
spec2nii ge [-j] [-f FILEOUT] [-o OUTDIR] pfilename
-j          creates the JSON sidecar file
-f FILEOUT  creates the output file with the name FILEOUT
-o OUTDIR   creates the output file in the directory OUTDIR
pfilename   path to the input p file
```

For example, the following script creates the files test001.nii.gz, test001.json, test001_ref.nii.gz, and test001_ref.json in the same directory as test001.7.

```
spec2nii ge -j test001.7
```

Note that metabolite and water reference data need do not need to be extracted or converted separately. However, make sure that you correctly append the suffixes `svs` and `ref` to the output file names as described in sections 3.3.1 and 3.3.2.

3.1.6 Converting Siemens DICOM

For conventional short-TE data, you should use the following command:

```
spec2nii dicom [-j] [-f FILEOUT] [-o OUTDIR] dcmfileorfolder
-j          creates the JSON sidecar file
-f FILEOUT  creates the output file with the name FILEOUT
-o OUTDIR   creates the output file in the directory OUTDIR
dcmfileorfolder path to the input Siemens DICOM file or folder
```

You can provide a single DCM/IMA file or an entire directory of DCM/IMA files.

For example, the following script creates the files test001.nii.gz and test001.json in the same directory as test001.dcm.

```
spec2nii dicom -j test001.dcm
```

Metabolite and water reference data need to be converted separately. Append the suffixes `svs` and `ref` to the output file names as described in sections 3.3.1 and 3.3.2.

3.1.7 Converting Philips DICOM

For conventional short-TE data, you should use the following command:

```
spec2nii philips_dcm [-j] [-f FILEOUT] [-o OUTDIR] dcmfileorfolder
-j          creates the JSON sidecar file
-f FILEOUT  creates the output file with the name FILEOUT
-o OUTDIR   creates the output file in the directory OUTDIR
dcmfileorfolder path to the input Philips DICOM file or folder
```

You can provide a single DCM file or an entire directory of DCM files.

For example, the following script creates the files test001.nii.gz and test001.json in the same directory as test001.dcm.

```
spec2nii philips_dcm -j test001.dcm
```

Metabolite and water reference data need to be converted separately. Append the suffixes `svs` and `ref` to the output file names as described in sections 3.3.1 and 3.3.2.

3.2 BIDS folder organization

You can use the MATLAB function “**GenerateBIDSStructure**” in the ‘utilities’ folder of Osprey to generate a new, generic, empty BIDS folder structure. **GenerateBIDSStructure** is called by running

```
GenerateBIDSStructure(nSubs, nSes, dataTypes)
```

at the MATLAB command prompt, where **nSubs** is the number of subjects, **nSes** is the number of repeat sessions, and **dataTypes** is a cell array of modalities that you want to generate, e.g. `{ 'anat', 'mrs' }`.

If you only have one session, you may omit the session label altogether. Consequently, if you specify **nSes = 1**, **GenerateBIDSStructure** does not create a session layer.

3.3 BIDS file naming conventions

In BIDS, each dataset is uniquely identifiable by its filename, which is assembled from multiple identifiers (subject, session, modality, contrast, run, etc.). There are no duplicate filenames across the entire project.

3.3.1 Metabolite data

Each metabolite acquisition shall be named and sorted in the **mrs** folder for the respective subject and session according to several key-value pairs, which are referred to as entities in BIDS-speak. You can find out more about them at <https://bids-specification.readthedocs.io/en/bep022/appendices/entities.html>.

```
sub-<label>/
  [ses-<label>/]
    mrs/
      sub-<label>[_ses-<label>][_acq-<label>][_voi-<label>][_run-<index>][_echo-<index>]_svs.json
      sub-<label>[_ses-<label>][_acq-<label>][_voi-<label>][_run-<index>][_echo-<index>]_svs.nii.gz
```

Subject (sub): Use the correct running subject ID, e.g., **sub-001**.

Session (ses): Use the correct running session ID, e.g., **ses-01**. If you only have one session, you may omit the session label altogether.

Acquisition (acq): Use to specify the technique used, e.g., **acq-press** or **acq-slaser**. See <https://bids-specification.readthedocs.io/en/bep022/modality-specific-files/magnetic-resonance-spectroscopy.html#mrs-sequences> for a list of suitable labels.

VOI (voi): Use to specify the anatomical location in which the voxel was placed, e.g., **voi-pcc** or **voi-dlpfc**. There is no prescribed list of labels for the VOI string.

Run (run): Use to specify identical repetitions of a scan, for example, for within-scan test-retest acquisitions. Use running indices in order of acquisition, if possible, e.g., **run-1**, **run-2**, etc. If you only have one run, you may omit the run label altogether.

Echo (echo): Use to specify repetitions of a scan, but with different echo times, for example, if you acquire PRESS data at TE = 30 ms and TE = 144 ms. Use running indices, not the echo time itself, e.g., **echo-1**, **echo-2**, etc.

Suffix: Note the suffix **svs** to denote the metabolite data. This is the only difference in file naming to the water reference data (see next section). For example, a correctly named file for a single-session of short-TE (30 ms) PRESS (no-test-retest) in the PCC would look like

```
sub-001/  
  mrs/  
    sub-001_acq-press_voi-pcc_svs.json  
    sub-001_acq-press_voi-pcc_svs.nii[.gz]
```

Since there is only one session and only one run of this sequence within this session, the session and run labels can be omitted. Likewise, we do not acquire at multiple echo times, so the echo label can be omitted.

For a second example, two sessions of short-TE (30 ms) and long-TE (144 ms) sLASER (with test-retest) in the ACC and PCC would look like

```
sub-001/
  ses-01/
    mrs/
      sub-001_ses-01_acq-slaser_voi-acc_run-1_echo-1_svs.json      % session 1, acc, test, TE 30
      sub-001_ses-01_acq-slaser_voi-acc_run-1_echo-1_svs.nii      % session 1, acc, test, TE 30
      sub-001_ses-01_acq-slaser_voi-acc_run-1_echo-2_svs.json      % session 1, acc, test, TE 144
      sub-001_ses-01_acq-slaser_voi-acc_run-1_echo-2_svs.nii      % session 1, acc, test, TE 144
      sub-001_ses-01_acq-slaser_voi-acc_run-2_echo-1_svs.json      % session 1, acc, re-test, TE 30
      sub-001_ses-01_acq-slaser_voi-acc_run-2_echo-1_svs.nii      % session 1, acc, re-test, TE 30
      sub-001_ses-01_acq-slaser_voi-acc_run-2_echo-2_svs.json      % session 1, acc, re-test, TE 144
      sub-001_ses-01_acq-slaser_voi-acc_run-2_echo-2_svs.nii      % session 1, acc, re-test, TE 144
      sub-001_ses-01_acq-slaser_voi-pcc_run-1_echo-1_svs.json      % session 1, pcc, test, TE 30
      sub-001_ses-01_acq-slaser_voi-pcc_run-1_echo-1_svs.nii      % session 1, pcc, test, TE 30
      sub-001_ses-01_acq-slaser_voi-pcc_run-1_echo-2_svs.json      % session 1, pcc, test, TE 144
      sub-001_ses-01_acq-slaser_voi-pcc_run-1_echo-2_svs.nii      % session 1, pcc, test, TE 144
      sub-001_ses-01_acq-slaser_voi-pcc_run-2_echo-1_svs.json      % session 1, pcc, re-test, TE 30
      sub-001_ses-01_acq-slaser_voi-pcc_run-2_echo-1_svs.nii      % session 1, pcc, re-test, TE 30
      sub-001_ses-01_acq-slaser_voi-pcc_run-2_echo-2_svs.json      % session 1, pcc, re-test, TE 144
      sub-001_ses-01_acq-slaser_voi-pcc_run-2_echo-2_svs.nii      % session 1, pcc, re-test, TE 144
sub-001/
  ses-02/
    mrs/
      sub-001_ses-02_acq-slaser_voi-acc_run-1_echo-1_svs.json      % session 2, acc, test, TE 30
      sub-001_ses-02_acq-slaser_voi-acc_run-1_echo-1_svs.nii      % session 2, acc, test, TE 30
      sub-001_ses-02_acq-slaser_voi-acc_run-1_echo-2_svs.json      % session 2, acc, test, TE 144
      sub-001_ses-02_acq-slaser_voi-acc_run-1_echo-2_svs.nii      % session 2, acc, test, TE 144
      sub-001_ses-02_acq-slaser_voi-acc_run-2_echo-1_svs.json      % session 2, acc, re-test, TE 30
      sub-001_ses-02_acq-slaser_voi-acc_run-2_echo-1_svs.nii      % session 2, acc, re-test, TE 30
      sub-001_ses-02_acq-slaser_voi-acc_run-2_echo-2_svs.json      % session 2, acc, re-test, TE 144
      sub-001_ses-02_acq-slaser_voi-acc_run-2_echo-2_svs.nii      % session 2, acc, re-test, TE 144
      sub-001_ses-02_acq-slaser_voi-pcc_run-1_echo-1_svs.json      % session 2, pcc, test, TE 30
      sub-001_ses-02_acq-slaser_voi-pcc_run-1_echo-1_svs.nii      % session 2, pcc, test, TE 30
      sub-001_ses-02_acq-slaser_voi-pcc_run-1_echo-2_svs.json      % session 2, pcc, test, TE 144
      sub-001_ses-02_acq-slaser_voi-pcc_run-1_echo-2_svs.nii      % session 2, pcc, test, TE 144
      sub-001_ses-02_acq-slaser_voi-pcc_run-2_echo-1_svs.json      % session 2, pcc, re-test, TE 30
      sub-001_ses-02_acq-slaser_voi-pcc_run-2_echo-1_svs.nii      % session 2, pcc, re-test, TE 30
      sub-001_ses-02_acq-slaser_voi-pcc_run-2_echo-2_svs.json      % session 2, pcc, re-test, TE 144
      sub-001_ses-02_acq-slaser_voi-pcc_run-2_echo-2_svs.nii      % session 2, pcc, re-test, TE 144
```

3.3.2 Water reference data

Osprey supports two types of water data: (i) *lineshape reference data*: acquired at the same TE as the metabolite acquisition; (ii) *concentration reference data*: acquired at a shorter TE than the metabolite acquisition. The *lineshape reference data* are used to correct eddy currents and to perform tissue-and-relaxation-corrected quantification. If *concentration reference data* are additionally provided, they are used to perform the quantification, since they will have lower T_2 weighting at medium to long echo times.

Of course, lineshape and concentration reference are often the same scan, for example, for a TE = 30 ms PRESS scan, the respective TE = 30 ms water scan will serve as both lineshape and concentration reference.

In BIDS, the only difference between the file names of the metabolite scan and the respective water scan will be in the **suffix**. While metabolite scans are denoted with the suffix **svs**, water scans will have the suffix **ref**.

If you have only *lineshape reference data* (TE-matched to the metabolite scan), simply name it

```
sub-001/
  ses-01/
    mrs/
      sub-001_ses-01_acq-slaser_voi-acc_run-1_echo-1_svs.json      % session 1, acc, test, TE 30, metabs
      sub-001_ses-01_acq-slaser_voi-acc_run-1_echo-1_svs.nii      % session 1, acc, test, TE 30, metabs
      sub-001_ses-01_acq-slaser_voi-acc_run-1_echo-1_mrsref.json   % session 1, acc, test, TE 30, lineshape ref
      sub-001_ses-01_acq-slaser_voi-acc_run-1_echo-1_mrsref.nii   % session 1, acc, test, TE 30, lineshape ref
```

If you have lineshape **and** water data, append **ecc** or **conc** to the acquisition label:

```
sub-001/
  ses-01/
    mrs/
      sub-001_ses-01_acq-slaser_voi-acc_svs.json                  % session 1, acc, test, TE 144, metabs
      sub-001_ses-01_acq-slaser_voi-acc_svs.nii                  % session 1, acc, test, TE 144, metabs
      sub-001_ses-01_acq-slaserecc_voi-acc_mrsref.json            % session 1, acc, test, TE 144, lineshape ref
      sub-001_ses-01_acq-slaserecc_voi-acc_mrsref.nii            % session 1, acc, test, TE 144, lineshape ref
      sub-001_ses-01_acq-slaserconc_voi-acc_mrsref.json          % session 1, acc, test, TE 30, water ref
      sub-001_ses-01_acq-slaserconc_voi-acc_mrsref.nii          % session 1, acc, test, TE 30, water ref
```

3.3.3 Structural (T_1 -weighted) images

Osprey supports automated co-registering and segmenting structural images to determine GM, WM, CSF fractions inside the voxel. To achieve this, the structural image must be the one that the MRS voxel was planned on, or otherwise be in the same coordinate system. The image **must** be available in NIfTI format (.nii or .nii.gz). If your data is presently in DICOM format, please convert to NIfTI/JSON using the command line tool *dcm2niix* (available for free from <https://github.com/rordenlab/dcm2niix>).

Structural data are provided in the **anat** folder with usually fewer labels:

```
sub-001/  
  ses-01/  
    anat/  
      sub-001_ses-01_T1w.json  
      sub-001_ses-01_T1w.nii
```

If you are unsure how you should organize your data or name your files, please contact us.

4 Setting up the Osprey analysis

Every Osprey analysis requires the user to provide a job file. The Osprey job file contains paths to MRS data files and structural images, defines processing and modeling options, and determines whether (and where) output files are being saved. The job file system ensures that all processing, modeling, and quantification steps are performed in an operator-independent, reproducible way.

The LCModel control file specifies the settings that the pre-compiled LCModel binary will use during spectral fitting. The LCModel basis set file contains the simulated metabolite signals that it will use.

We provide you with suitable template job files, control files and basis set files for the analysis of short-TE data. The script attempts to auto-populate the names of the MRS data files and structural images, assuming the convention laid out in Section 2. Depending on what types of data your dataset includes, you may have to add, remove, or edit certain lines in the job file. This is described below.

4.1 Osprey job file template

Most of the settings in the Osprey job file template `jobTemplateENIGMA.m` will be the same for all short-TE analyses. Generally, please do not modify settings on your own. A few, however, may need to be tweaked according to your specific dataset:

4.1.1 Data handling and modeling options

4.1.1.1 Eddy-current correction (`opts.ECC.raw`)

Set `opts.ECC.raw = 0` if your source data are Philips SPAR/SDAT (no spectral editing) and your acquisition exam card has the setting “Apply Spectral Correction” on the “Postproc” tab set to “Yes”. These have already been eddy-current-corrected on the scanner.

Leave `opts.ECC.raw = 1` if your source data are TWIX, RDA, DATA/LIST, or P files, or if you are using SPAR/SDAT data acquired with an exam card where the “Apply Spectral Correction” setting has been set to “No”.

4.1.1.2 Path to LCModel basis set file (`opts.fit.basisSetFile`)

We have pre-filled correct lines (all starting with `opts.fit.basisSetFile`) for each sequence that we have simulated basis sets for. Please uncomment the line that best describes your data type, then comment out all other lines. If you do not find anything suitable, please contact us.

4.1.1.3 Path to LCModel control file template (`opts.fit.controlFile`)

We have pre-populated this field to point to the correct control file template to use for this project. Please do not use your own control file or comment this line out.

4.1.1.4 Path to LCModel executable (`opts.fit.customLCModelBinary`)

This field is, by default, not used. If the pre-compiled LCModel binaries that come with Osprey do not work on your system (see Section 2.3 *Testing the LCModel binary*), you can compile the LCModel executable yourself and insert the path to it here. (Recommended location: `osprey/libraries/LCModel/custom`).

4.2 Metabolite data paths (**files**)

The **files** variable is a cell array of full paths to the raw MRS data files containing your metabolite (water-suppressed) data.

This is an example of a valid definition:

```
% Specify metabolite data
% (MANDATORY)

files = {'C:\data\sub-01\ses-01\mrs\sub-01_ses-01_acq-press_svs.nii', ...
        'C:\data\sub-02\ses-01\mrs\sub-02_ses-01_acq-press_svs.nii', ...
        'C:\data\sub-03\ses-01\mrs\sub-03_ses-01_acq-press_svs.nii', ...
        'C:\data\sub-04\ses-01\mrs\sub-04_ses-01_acq-press_svs.nii', ...
        'C:\data\sub-05\ses-01\mrs\sub-05_ses-01_acq-press_svs.nii', ...
        'C:\data\sub-06\ses-01\mrs\sub-06_ses-01_acq-press_svs.nii', ...
        'C:\data\sub-07\ses-01\mrs\sub-07_ses-01_acq-press_svs.nii', ...
        'C:\data\sub-08\ses-01\mrs\sub-08_ses-01_acq-press_svs.nii', ...
        };
```

You can also define the **files** array programmatically, for example by loop over your BIDS folder structure. Since the exact structure and file names used will vary between sites, we do not provide a template for programmatic definition (but are happy to help).

Note that you cannot process different data types (e.g., Philips SDAT or Siemens TWIX) or data from different protocols (e.g., PRESS with different TEs; PRESS and MEGA-PRESS) in the same job file. Data from different protocols must be processed with separate job files.

4.3 Lineshape reference data paths (**files_ref**)

The **files_ref** variable is a cell array of full paths to the raw MRS data files containing your lineshape (water-unsuppressed) reference data which will be used for eddy-current correction (and, if you do not provide separate `files_w` data, for water-scaled quantification).

The **files_ref** cell array needs to have exactly as many entries as the `files` array, and the data need to be provided in the exact same order.

This is an example of a valid definition:

```
% Specify water reference data for eddy-current correction
% (same sequence as metabolite data!)
% (OPTIONAL)

files_ref = {'C:\data\sub-01\ses-01\mrs\sub-01_ses-01_acq-pressecc_mrsref.nii', ...
            'C:\data\sub-02\ses-01\mrs\sub-02_ses-01_acq-pressecc_mrsref.nii', ...
            'C:\data\sub-03\ses-01\mrs\sub-03_ses-01_acq-pressecc_mrsref.nii', ...
            'C:\data\sub-04\ses-01\mrs\sub-04_ses-01_acq-pressecc_mrsref.nii', ...
            'C:\data\sub-05\ses-01\mrs\sub-05_ses-01_acq-pressecc_mrsref.nii', ...
            'C:\data\sub-06\ses-01\mrs\sub-06_ses-01_acq-pressecc_mrsref.nii', ...
            'C:\data\sub-07\ses-01\mrs\sub-07_ses-01_acq-pressecc_mrsref.nii', ...
            'C:\data\sub-08\ses-01\mrs\sub-08_ses-01_acq-pressecc_mrsref.nii', ...
            };
```

Comment out the lines defining **files_ref** if you do not have water reference data available.

4.4 Short-TE water data paths (**files_w**)

The **files_w** variable is a cell array of full paths to the raw MRS data files containing your short-TE water (water-unsuppressed) reference data. These are an optional input that can be used to derive water-scaled concentration estimates (while lineshape reference data are only used for eddy-current correction). Using short-TE water as the concentration reference standard reduces T_2 -weighting of the water reference signal (and associated correction errors) compared to longer-TE water data.

The **files_w** cell array needs to have exactly as many entries as the 'files' array, and the data need to be provided in the exact same order.

This is an example of a valid definition:

```
% Specify water data for quantification (e.g. short-TE water scan)
% (OPTIONAL)

files_w = {'C:\data\sub-01\ses-01\mrs\sub-01_ses-01_acq-press-conc_mrsref.nii', ...
           'C:\data\sub-02\ses-01\mrs\sub-02_ses-01_acq-press-conc_mrsref.nii', ...
           'C:\data\sub-03\ses-01\mrs\sub-03_ses-01_acq-press-conc_mrsref.nii', ...
           'C:\data\sub-04\ses-01\mrs\sub-04_ses-01_acq-press-conc_mrsref.nii', ...
           'C:\data\sub-05\ses-01\mrs\sub-05_ses-01_acq-press-conc_mrsref.nii', ...
           'C:\data\sub-06\ses-01\mrs\sub-06_ses-01_acq-press-conc_mrsref.nii', ...
           'C:\data\sub-07\ses-01\mrs\sub-07_ses-01_acq-press-conc_mrsref.nii', ...
           'C:\data\sub-08\ses-01\mrs\sub-08_ses-01_acq-press-conc_mrsref.nii', ...
           };
```

Comment out the lines defining **files_w** if you do not have specific short-TE water reference data for quantification available.

4.5 Structural (T1-weighted) image paths (**files_nii**)

The **files_nii** variable is defined as a cell array of full paths to T_1 -weighted structural images used for co-registration and segmentation purposes.

The **files_nii** cell array needs to have exactly as many entries as the 'files' array, and the data need to be provided in the exact same order. If you are analyzing two MRS datasets from the same subject, you will need to provide the same structural T1 entry *twice* at the respective positions in the **files_nii** array.

This is an example of a valid definition:

```
% Specify T1-weighted structural imaging data
% (OPTIONAL)

files_nii = {'C:\data\sub-01\ses-01\anat\sub-01_ses-01_T1w.nii', ...
            'C:\data\sub-02\ses-01\anat\sub-02_ses-01_T1w.nii', ...
            'C:\data\sub-03\ses-01\anat\sub-03_ses-01_T1w.nii', ...
            'C:\data\sub-04\ses-01\anat\sub-04_ses-01_T1w.nii', ...
            'C:\data\sub-05\ses-01\anat\sub-05_ses-01_T1w.nii', ...
            };
```

```
'C:\data\sub-06\ses-01\anat\sub-06_ses-01_T1w.nii', ...  
'C:\data\sub-07\ses-01\anat\sub-07_ses-01_T1w.nii', ...  
'C:\data\sub-08\ses-01\anat\sub-08_ses-01_T1w.nii', ...  
};
```

Comment out the lines defining **files_nii** if you do not have structural images available.

4.6 LCModel basis set

Please use the LCModel basis set file that is provided by us, even if you already have one that you are using for your own studies. If your sequence is highly customized and you simulated the basis set yourself, please contact us.

4.7 LCModel control file

Please use the LCModel control file template **controlTemplateENIGMA.control** that is provided by us, even if you already have one that you are using for your own studies.

Please do not modify the control file template; in particular, do not change the list of metabolites to be omitted from the analysis, the baseline knot spacing, soft constraints, eddy-current correction or water-scaling settings.

5 Running the analysis

5.1 Command line vs. executable

There are two ways to execute the analysis that should yield identical results. The first one is to run a sequence of commands from the MATLAB command window (Section 4.1.1), which should work on any computer with a valid MATLAB license and the necessary toolboxes (Section 1.1). The second one uses a pre-compiled MATLAB executable (Section 4.1.2), does not require a MATLAB license or toolboxes and should be deployable on most systems.

5.1.1 Running Osprey from the MATLAB window

5.1.1.1 MATLAB Search Path

Only include Osprey and spm12 on your current MATLAB path. Make sure that no copies of the FID-A toolbox, the Gannet toolbox, or the Fieldtrip toolbox are added to your MATLAB path. FID-A and Fieldtrip share functions with Osprey that have the same name, but different functionality. Including them on the MATLAB path during an Osprey analysis usually leads to shadowing conflicts and errors.

To print out a list of the folders that are included in the MATLAB search path, type ``path`` at the MATLAB prompt. You can also click “Set Path” in the “Environment” ribbon on the “Home” tab at the top of the MATLAB screen.

5.1.2 Running the compiled Osprey executable

Helge

5.2 OspreyJob

Run the command `MRSCont = OspreyJob('jobENIGMA.m');`

OspreyJob will extract all necessary information from your specific job file and populate a MATLAB struct called `MRSCont`. This struct will serve as the data container for all subsequent analysis steps.

5.3 OspreyLoad

Run the command `MRSCont = OspreyLoad(MRSCont);`

OspreyLoad applies various pre-processing steps to the raw data, like frequency/phase alignment, eddy-current correction, etc. The steps it takes depend on the ‘rawness’ of your input data format. Depending on the complexity of the pre-processing route and the quality of the data, this can take between approximately 10 seconds and 1-2 minutes per spectrum.

5.4 OspreyProcess

Run the command `MRSCont = OspreyProcess(MRSCont);`

OspreyProcess applies various pre-processing steps to the raw data, like frequency/phase alignment, eddy-current correction, etc. The steps it takes depend on the ‘rawness’ of your input data format. Depending on the complexity of the pre-processing route and the quality of the data, this can take between approximately 10 seconds and 1-2 minutes per spectrum.

5.5 OspreyFit

Run the command **MRSCont = OspreyFit(MRSCont) ;**

OspreyFit converts the processed data into LCModel format, calls the external LCModel executable to carry out the fitting, and imports the quantitative outputs and visualizations. This should take approximately 5 seconds per spectrum, depending on the data quality.

5.6 OspreyCoreg

Run the command **MRSCont = OspreyCoreg(MRSCont) ;**

OspreyCoreg parses the voxel geometry information and creates a binary voxel mask. This should only take approximately 10 seconds per voxel.

You can only run the voxel co-registration routine if you have structural images available and specified in the job file (see Section 2.2.3). If you do not have structural images, skip to Section 4.8.

5.7 OspreySeg

Run the command **MRSCont = OspreySeg(MRSCont) ;**

OspreySeg calls the SPM12 segmentation functions and calculates the fractions of grey matter, white matter, and cerebrospinal fluid in the MRS voxel. If you have previously segmented the images, this step will only take a few seconds. However, if Osprey has to initiate the segmentation first, it can take up to several minutes per structural image.

You can only run the segmentation routine if you have structural images available and specified in the job file (see Section 2.2.3). If you do not have structural images, skip to Section 4.8.

5.8 OspreyQuantify

Run the command **MRSCont = OspreyQuantify(MRSCont) ;**

OspreyQuantify calculates several quantitative metabolite estimates, such as tCr ratios and water-scaled estimates with different degrees of sophistication. It also creates the result tables that you will collect in the next step. Please refer to the Osprey manuscript for details of the quantification procedures. This should take less than a second per dataset.

5.9 OspreyOverview

Run the command **MRSCont = OspreyOverview(MRSCont) ;**

OspreyOverview creates useful figures and statistics at the group level (across all processed datasets), including the standardized MRSinMRS reporting checklist (see section 6.2.4).

6 Collecting the output data

This section explains how you can inspect the results of the Osprey analysis pipeline. It also details which output files you need to package into an archive (.zip) to send back to the ENIGMA MRS analysis core.

6.1 Inspecting your data

You can call `MRSCont = OspreyGUI (MRSCont)` ; to visually inspect all steps of the analysis.

You can select the dataset you wish to be displayed by clicking on the entry in the dataset list in the bottom left corner, and you can easily scroll through the different datasets using the \uparrow and \downarrow keys on your keyboard.

By clicking the metabolites, reference and water tabs at the bottom of the window, you can display the water-suppressed, lineshape reference, and short-TE water data (if applicable).

Raw tab: This shows the coil-combined, but un-aligned and un-averaged data.

Process tab: The top left panel shows the individual averages prior to frequency-and-phase alignment. The bottom left panel shows them after frequency-and-phase alignment. The top right panel shows a scatter plot of the maximum of the 3.02 ppm Cr/PCr signal over the course of the acquisition for both pre- and post-alignment. Finally, the bottom right panel shows the aligned, averaged and referenced spectrum after eddy-current correction and residual water removal. In the box above the spectrum display, you will find some data quality metrics, such as the signal-to-noise ratio (SNR), linewidth (FWHM), and average frequency drift pre- and post-alignment.

Fit tab: Shows the complete fit overlaid on top of the data, along with the baseline, and the fit residual at the top. Below, the contributions from each basis functions are plotted, with the estimated lineshape convolution, linebroadening and frequency shift parameters applied. For better display, the phase parameters have been applied to the spectrum, so that the basis functions appear without phase. The left panel shows the raw water-scaled metabolite estimates, i.e. the signal amplitudes normalized by the water amplitude. In the box above the spectrum display, you will find fit parameters and information, such as the number of basis functions used, phase estimates, and scaling factors.

Coreg/Seg tab: You can see an overlay of the voxel mask on top of the anatomical image in three directions. You can also see you can now see four images showing the voxel mask next to the contributions from grey matter, white matter, and CSF, along with the fractional volume estimates for each tissue class. These values are used during quantification to account for tissue-specific effects of relaxation, tissue water content, and metabolite content. The *Nii Viewer* button in the top left corner of the window opens the current participant's structural image in an external NIfTI viewer for closer inspection.

Quantify tab: This tab contains a bare-bone table of the quantitative results for each basis function and each quantification routine for the selected dataset. These numbers are saved in .TSV format tables in the QuantifyResults sub-directory of the output folder (see Section 6.2). At this stage, all analysis steps have been completed.

6.2 Selecting the output files for transfer

Navigate to the **derivatives** output directory generated by Osprey that was specified in the job file.

6.2.1 Quality control metrics

Your analysis will always generate the following files containing SNR, FWHM, etc. Transfer all of them.

`QM_processed_spectra.tsv`

`QM_processed_spectra.json`

6.2.2 MRSinMRS checklist

Your analysis will also always (after running `OspreyOverview`) generate the following file containing a standardized reporting checklist (Lin et al. 2021) in the popular Markdown format. Transfer this file as well.

`SummaryMRSinMRS.md`

6.2.3 Quantification results

Navigate to the `QuantifyResults` folder generated by Osprey in the `derivatives` output directory specified in the job file.

6.2.3.1 *tCr ratios*

Your analysis will always generate the following files. Transfer all of them.

`A_amplMets_Voxel_1_Basis_1.tsv`

`A_amplMets_Voxel_1_Basis_1.json`

`A_CRLB_Voxel_1_Basis_1.tsv`

`A_CRLB_Voxel_1_Basis_1.json`

`A_amplMets_Voxel_1_Basis_1.tsv`

`A_amplMets_Voxel_1_Basis_1.json`

6.2.3.2 *Raw water-scaled values*

If you provided water reference data, but no structural images, your analysis will also generate the following files. Transfer all of them.

`A_h2oarea_Voxel_1_Basis_1.tsv`

`A_h2oarea_Voxel_1_Basis_1.json`

`A_rawWaterScaled_Voxel_1_Basis_1.tsv`

`A_rawWaterScaled_Voxel_1_Basis_1.json`

6.2.3.3 *Tissue-corrected water-scaled values*

If you provided water reference and structural imaging data, your analysis will also generate the following files. Transfer all of them.

`A_CSFWaterScaled_Voxel_1_Basis_1.tsv`

`A_CSFWaterScaled_Voxel_1_Basis_1.json`

`A_TissCorrWaterScaled_Voxel_1_Basis_1.tsv`

A_TissCorrWaterScaled_Voxel_1_Basis_1.json

6.2.4 Voxel co-registration & segmentation results & masks

Navigate to the **SegMaps** folder generated by Osprey in the **derivatives** output directory specified in the job file. Transfer the following files:

TissueFractions_Voxel_1.tsv

TissueFractions_Voxel_1.json

Navigate to the **VoxelMasks** folder generated by Osprey in the **derivatives** output directory specified in the job file. Transfer all of the following files:

***_space-scanner_mask.nii.gz**

***_space-spm152_mask.nii.gz**

***_space-scanner_mask_VoxelOverlap.nii.gz**

Here, the **scanner_mask** files are the binary voxel masks for each dataset in the same coordinate space as the respective T₁, the **spm152_mask** files are the same masks normalized to the SPM152 standard space, and the single **scanner_mask_VoxelOverlap** file will include a mask containing the Sorensen-Dice overlap coefficients between all datasets. The ***** will be subject and session identifiers from the metabolite data file names.

6.2.5 Demographics

Fill in the template **participants.tsv** file that we have provided. The following information will be required:

Age of the participant at time of measurement (in years)

Group of the participant (levels: “control” or other clinical groups)

Sex (assigned at birth) of the participant (levels: “male” or “female”)

Gender of the participant (levels: see below, “n/a” if not assessed)

Gender identity refers to a person’s internal sense of being a woman, man, both, neither or elsewhere on the gender spectrum. We recognize there are many definitions to gender, and we are not able to include all here. Therefore, please choose the selection that best represented the chosen gender identity:

Man

Woman

Transgender/non-binary

Other

Initial data collection will focus on normative (healthy control) data from typically developing, healthy subjects where the study exclusion criteria included the absence of a previous medical diagnosis of brain injury, stroke, neuropsychiatric disorders.

You can provide additional demographic or clinical information in the participants files – simply add columns in **participants.tsv** and the respective annotation in **participants.json**.

See <https://bids-specification.readthedocs.io/en/stable/modality-agnostic-files.html#participants-file> for details.

6.3 Packaging & transferring

Copy all selected files to a new directory. Rename the directory according to “*SiteName_DataSetName_VersionNumber_Date*”.

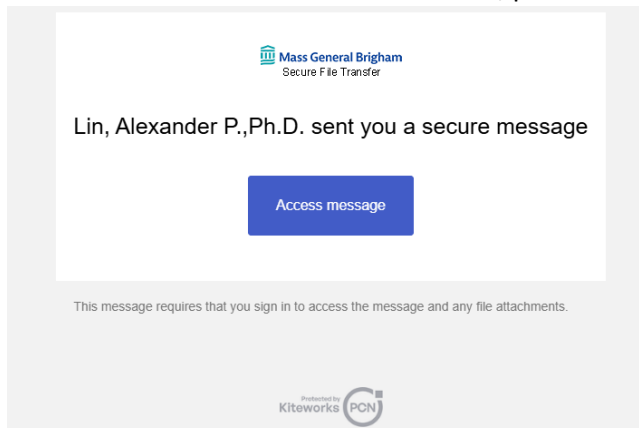
- *SiteName* should be a short, unique string identifying your site.
- *DataSetName* should be a short, unique string identifying the dataset.
- *VersionNumber* should be a running index (v1, v2, etc.) of analyses you have run.
- *Date* of upload in DD-MM-YYYY format (e.g., 29-01-2025)

Finally, pack the entire directory into a ZIP file with the same name.

6.3.1 Transferring the data via Secure File Transfer

Accessing the Secure File Transfer website:

1. Send an email to aplin@bwh.harvard.edu from your primary email address with the Subject: Request ENIGMA access. In the body of the email, please include your name, site, address, and contact information.
2. You will receive an email as shown below, please click on “Access Message”



3. This will take you to transferkw.partners.org where you will set your password and enter a captcha code. Click on “Create Account”



Create account

Already a kiteworks user? [Sign in](#)

Email

test@email.com

Password

Confirm password

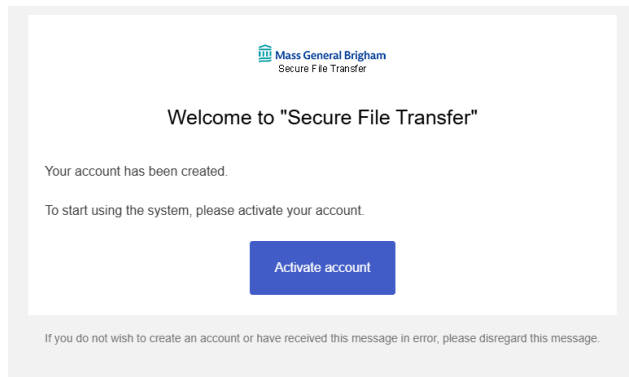
- ☐ 1 number
- ☐ 1 lowercase character
- ☐ 1 uppercase character
- ☐ 8 characters minimum

Next

 English 

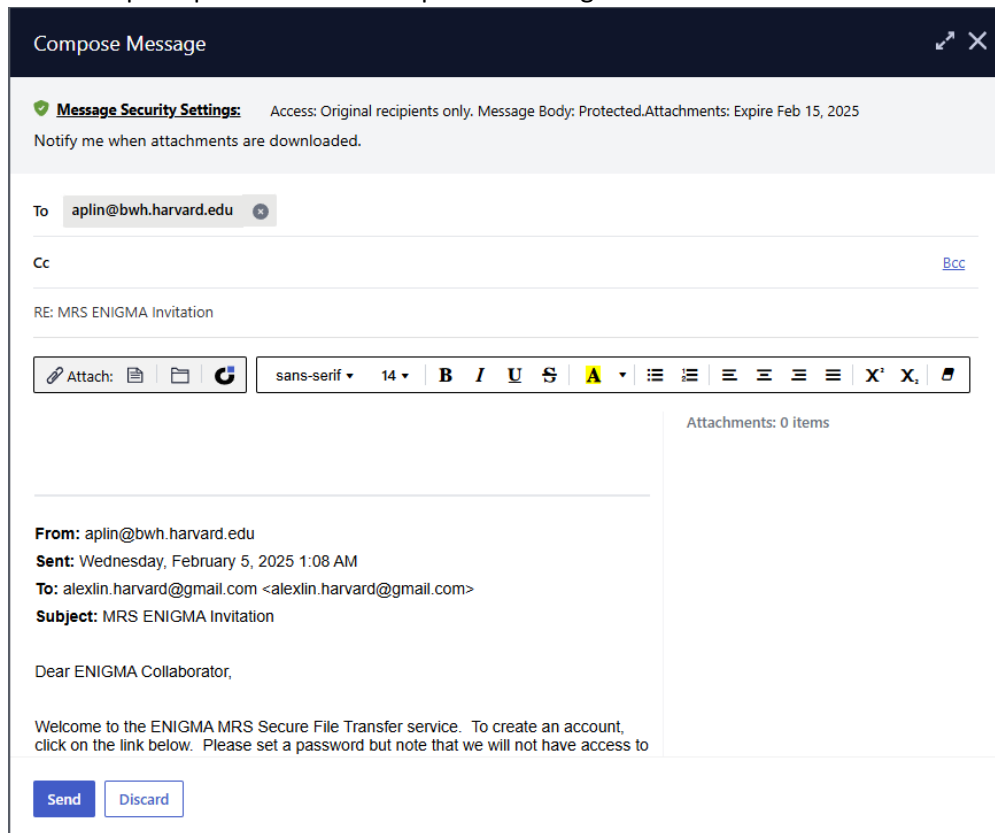
[Getting Started?](#)


4. You will receive an email from MGB Secure File Transfer <transfer@partners.org> . Click on "Activate account"



5. This will take you to the transferkw.partners.org site again where you can view the welcome message.
6. If you are ready to upload files, click on "Reply All". If you are not ready, continue to step 9 when you are ready.

7. This will open up a window to compose a message.



Here you will attach the files you want to upload as you would as an email attachment by clicking on the . Please compile all files you wish to send in a single compressed file as described above. Multiple files are, in principle, acceptable but the upload may be interrupted resulting in missing files.

8. After you upload the file, please wait until the file is fully uploaded (this may take time depending on the size of the file) then click Send.
9. To upload additional files, log into transferkw.partners.org with your email address and password. Click on Compose message and send to aplin@bwh.harvard.edu

7 References

- Bartnik-Olson, Brenda L., Jeffry R. Alger, Talin Babikian, Ashley D. Harris, Barbara Holshouser, Ivan I. Kirov, Andrew A. Maudsley, et al. 2021. "The Clinical Utility of Proton Magnetic Resonance Spectroscopy in Traumatic Brain Injury: Recommendations from the ENIGMA MRS Working Group." *Brain Imaging and Behavior* 15 (2): 504–25. <https://doi.org/10.1007/s11682-020-00330-6>.
- Choi, In-Young, and Roland Kreis. 2021. "Advanced Methodology for in Vivo Magnetic Resonance Spectroscopy." *NMR in Biomedicine* 34 (5): e4504. <https://doi.org/10.1002/nbm.4504>.

- Harris, Ashley D., Houshang Amiri, Mariana Bento, Ronald Cohen, Christopher R. K. Ching, Christina Cudalbu, Emily L. Dennis, et al. 2023. "Harmonization of Multi-Scanner in Vivo Magnetic Resonance Spectroscopy: ENIGMA Consortium Task Group Considerations." *Frontiers in Neurology* 13 (January). <https://doi.org/10.3389/fneur.2022.1045678>.
- Lin, Alexander, Ovidiu Andronesi, Wolfgang Bogner, In-Young Choi, Eduardo Coello, Cristina Cudalbu, Christoph Juchem, et al. 2021. "Minimum Reporting Standards for in Vivo Magnetic Resonance Spectroscopy (MRSinMRS): Experts' Consensus Recommendations." *NMR in Biomedicine* 34 (5): e4484. <https://doi.org/10.1002/nbm.4484>.
- Öz, Gülin, Jeffry R. Alger, Peter B. Barker, Robert Bartha, Alberto Bizzi, Chris Boesch, Patrick J. Bolan, et al. 2014. "Clinical Proton MR Spectroscopy in Central Nervous System Disorders." *Radiology* 270 (3): 658–79. <https://doi.org/10.1148/radiol.13130531>.