BrkRaw: A comprehensive tool for accessing raw Bruker Biospin MRI data

Sung-Ho Lee[1, 2, 3], **Woomi Ban**[1, 3], **and Yen-Yu Ian Shih**[1, 2, 3]

1 Center for Animal MRI, University of North Carolina at Chapel Hill **2** Department of Neurology, University of North Carolina at Chapel Hill **3** Biomedical Research Imaging Center(BRIC), University of North Carolina at Chapel Hill

DOI: 10.21105/joss.02277

Software

- Review
- Repository
- Archive

Editor: Richard Gowers

Reviewers:

- @mih
- @mwegrzyn
- @PeerHerholz

Submitted: 20 May 2020
Published: 30 May 2020

Summary

Accessing and analyzing Magnetic Resonance Imaging (MRI) data required conversion from a hardware-specific format to one compatible with available analysis software. The Digital Imaging and Communications in Medicine (DICOM) is considered as an international standard for handling MRI data in the clinical field due to its flexibility to cover the broad range of information from various imaging modalities and patient information. On the other hand, the complex data structure and the inclusion of unnecessary metadata in DICOM decrease analysis efficiency on various imaging research when the patient-related metadata is not essential for the study. For this reason, image data formats with a more efficient structure capable of encompassing spatial and temporal in a single file, such as NifTi-1, ANALYZE, or NRRD file formats, are becoming the new standard of choice in many imaging research studies. Converting the raw data into DICOM format is still necessary for clinical research, since it is important to preserve the patient-related metadata coupled with the image data. However, the preclinical MRI research using laboratory animals or objects does not require the DICOM conversion step and is rather inefficient.

Researchers have developed conversion software for Bruker Biospin MRI (the leading preclinical MRI system), for converting raw data directly into a NIfTI or ANALYZE format (Chavarrias, 2017; Ferraris et al., 2017; Rorden, 2018; Brett, 2002) for improving efficiency for data processing.

Although the converters significantly improve analysis workflow by direct converting into software friendly file format, the software introduced so far has not provided the feature to access the data immediately without conversion. Here we present the 'BrkRaw' python module, a more comprehensive preclinical tool for accessing and utilizing raw Bruker Biospin MRI data. The module has been built up upon robust low-level Python Application Programming Interface (API), allowing direct raw data access without conversion to provide the advanced and easy-to-use features for data analysis.

The current version of BrkRaw is composed of four components, the low-level Python API, the high-level Python API, the command-line tools, and the graphical user interface (GUI).

The low-level Python API provides a robust JCAMP-DX loader to convert parameter files embedded in raw data into a Python object and the raw Bruker data loader that converting a whole folder containing parameter and binary files of a single imaging study session into a Python object. We also designed the zip file to be loaded without extraction, considering the accessibility of the archived data.

For the high-level Python API, we focused on providing useful functions to reduce extra effort on converting data into a usable form. It preserves the coordinate system of the image according to the subject position and orientation on the scanner coordinate system (Figure1),

which is known to be a challenge in the Bruker Biospin MRI system. (Ferraris et al., 2017, p. @Naveau:2019). The high-level Python API offers to load image data as the two most popular object types in the Python eco-system for image processing (nibabel(Brett et al., 2020) and SimpleITK(Lowekamp, Chen, Ibáñez, & Blezek, 2013)), as well as to convert multiple parameters into a structured JSON-type object using a custom JSON-based syntax.

The BrkRaw modules provide two command-line tools, which mainly utilizing low- and high-level Python API for automating routine operation for data management, as well as data conversion. The 'brk-backup' used the direct accessibility of the archived file (zip compressed) to enable the validation of archived data compared to corresponding raw data without extraction. As a result, the command offers to automate the inspection of raw and archived data, archiving of missing or updated data and removing the broken or duplicated data (Figure2).

The 'brkraw' command, on the other hand, offers a useful function to check data information (Figure3) and to convert data into NifTi-1 format. In addition to this, the command provides automate conversion and organization of large datasets into a ready-to-share data structure, the Brain Imaging Data Structure (BIDS), a standard data structure for neuroimaging research proposed by the open science community for pursuing reproducible science (Gorgolewski et al., 2016) (Figure4).

Lastly, via 'brkraw gui' command, the GUI offers improved accessibility for previewing the image and parameters without conversion. The converting button will convert a previewing image into NifTi-1 format, so allow the user to visually check the image before conversion (Figure 5).

This module has been actively utilized in the Center for Animal MRI (CAMRI) at the University of North Carolina at Chapel Hill for several on-going preclinical functional MRI studies, including sequence development and data management. We expect this tool to reduce the burden of handling, and management of raw Bruker MRI data, thereby benefitting other animal imaging researchers. In the future we will develop additional Python-based tools for acute quality control and real-time fMRI data analysis.

Figures

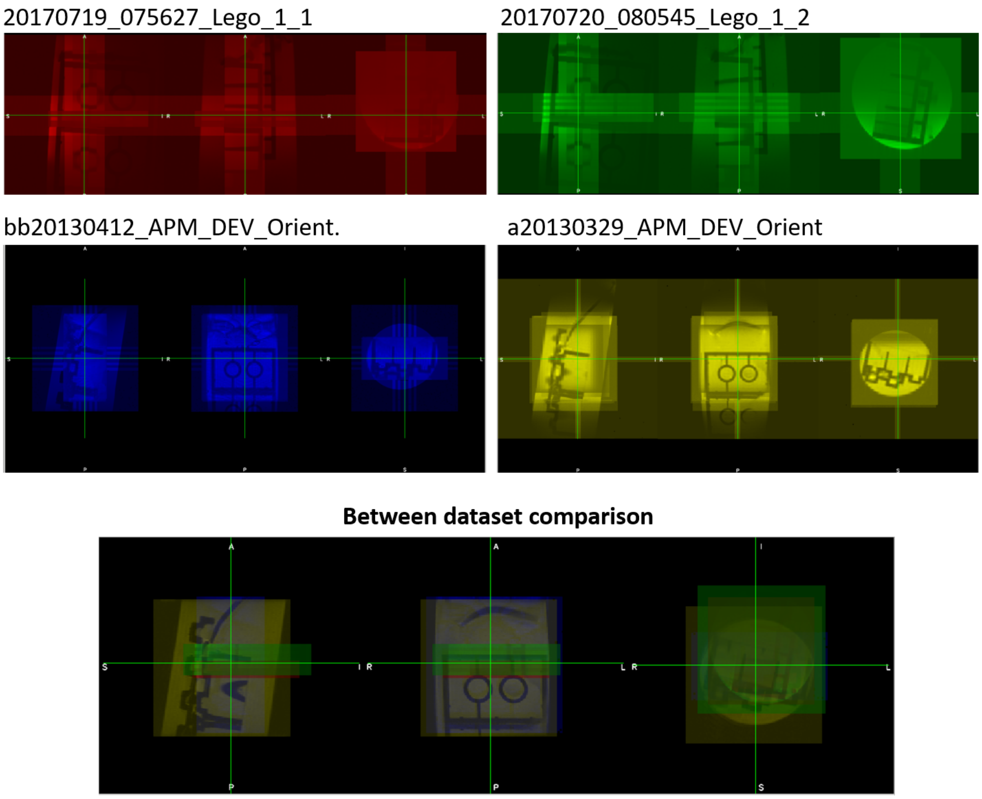
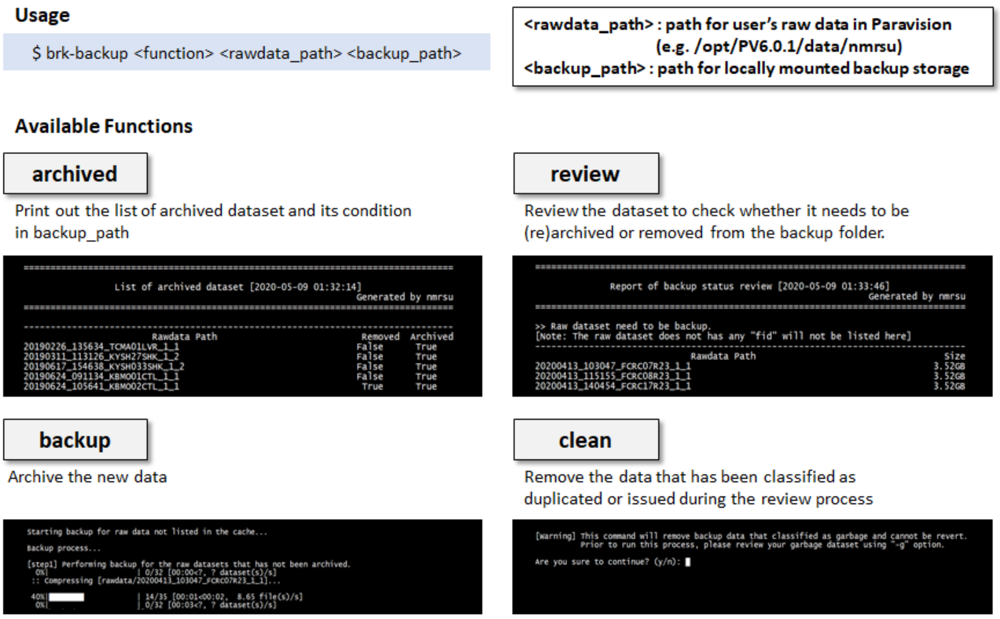


Figure 1. The overlapped converted images that shows successful conversion of Bruker2Nifti_QA dataset



Usage

```
$ brk-backup <function> <rawdata_path> <backup_path>
```

<rawdata_path> : path for user's raw data in Paravision (e.g. /opt/PV6.0.1/data/nmrsu)
<backup_path> : path for locally mounted backup storage

Available Functions

archived

Print out the list of archived dataset and its condition in backup_path

```
==============================================================================

                 List of archived dataset [2020-05-09 01:32:14]
                                                         Generated by nmrsu
==============================================================================

------------------------------------------------------------------------------
                     Rawdata Path                          Removed  Archived
20190226_135634_TCMA01LVR_1_1                              False     True
20190311_113126_KYSH27SHK_1_2                              False     True
20190617_154638_KYSH033SHK_1_2                             False     True
20190624_091134_KBMO01CTL_1_1                              False     True
20190624_105641_KBMO02CTL_1_1                               True     True
```

review

Review the dataset to check whether it needs to be (re)archived or removed from the backup folder.

```
==============================================================================

              Report of backup status review [2020-05-09 01:33:46]
                                                         Generated by nmrsu
==============================================================================

>> Raw dataset need to be backup.
[Note: The raw dataset does not has any "fid" will not be listed here]
------------------------------------------------------------------------------
                          Rawdata Path                                  Size
20200413_103047_FCRC07R23_1_1                                         3.52GB
20200413_115155_FCRC08R23_1_1                                         3.52GB
20200413_140454_FCRC17R23_1_1                                         3.52GB
```

backup

Archive the new data

```
Starting backup for raw data not listed in the cache...

Backup process...

[step1] Performing backup for the raw datasets that has not been archived.
  0%|                     | 0/32 [00:00<?, ? dataset(s)/s]
 :: Compressing [rawdata/20200413_103047_FCRC07R23_1_1]...

 40%|████████              | 14/35 [00:01<00:02,  8.65 file(s)/s]
  0%|                     | 0/32 [00:03<?, ? dataset(s)/s]
```

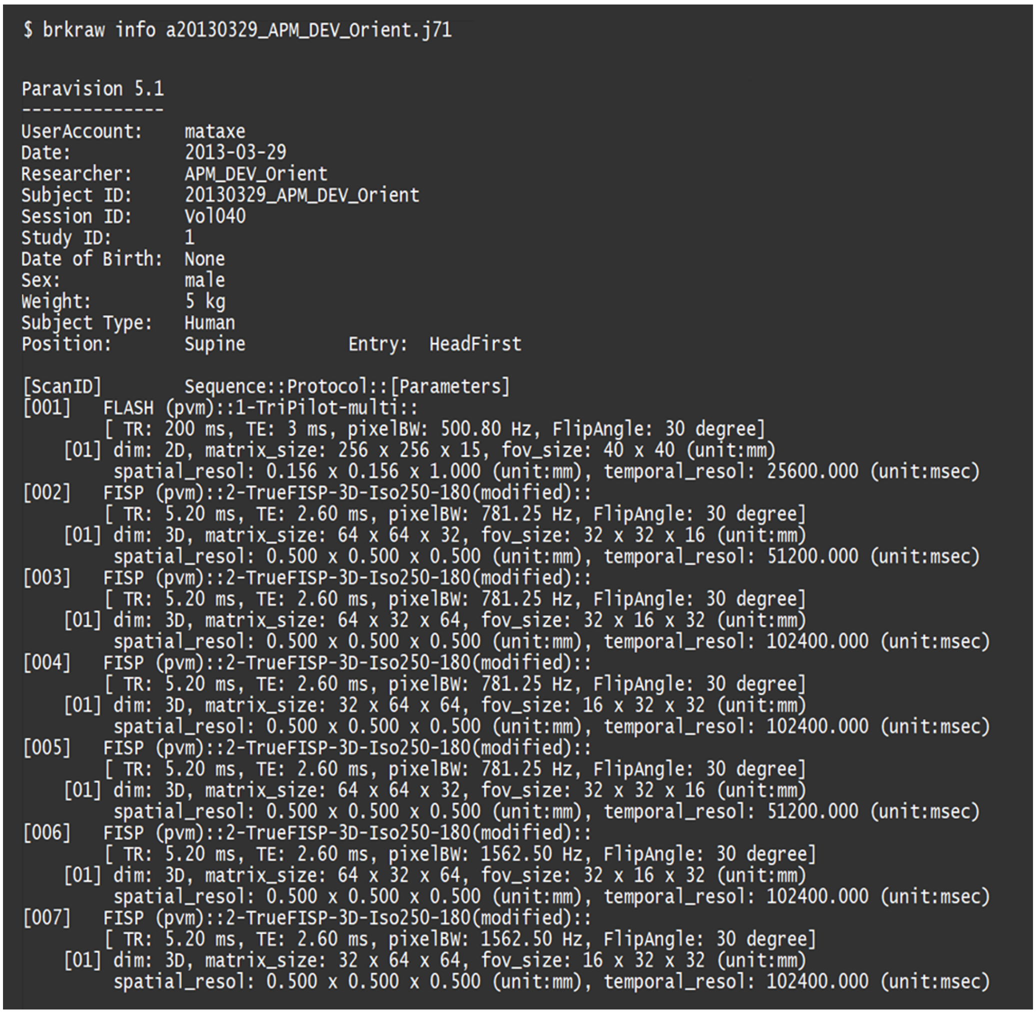
clean

Remove the data that has been classified as duplicated or issued during the review process

```
[warning] This command will remove backup data that classified as garbage and cannot be revert.
          Prior to run this process, please review your garbage dataset using "-g" option.

Are you sure to continue? (y/n): █
```

Figure 2. The main function of brk-backup command for data management

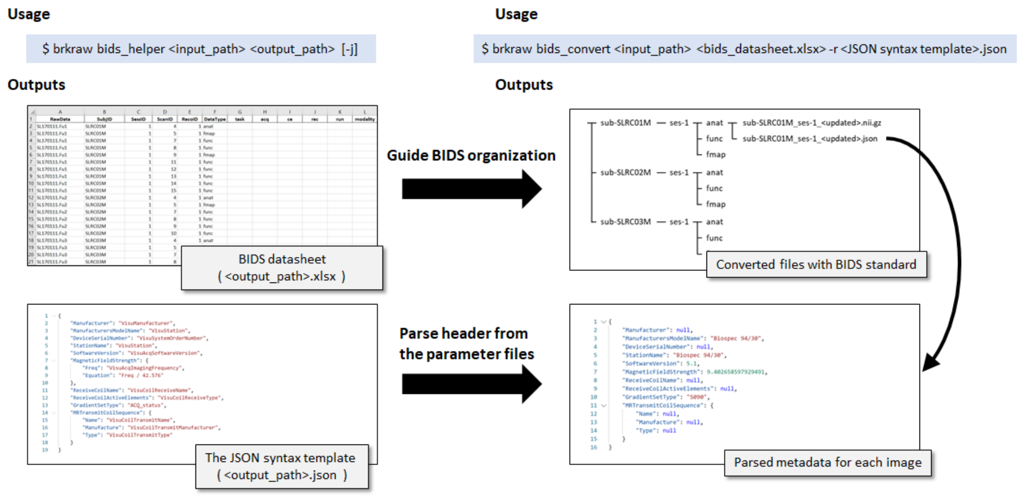


```
$ brkraw info a20130329_APM_DEV_Orient.j71

Paravision 5.1
--------------
UserAccount:    mataxe
Date:           2013-03-29
Researcher:     APM_DEV_Orient
Subject ID:     20130329_APM_DEV_Orient
Session ID:     Vol040
Study ID:       1
Date of Birth:  None
Sex:            male
Weight:         5 kg
Subject Type:   Human
Position:       Supine          Entry:  HeadFirst

[ScanID]        Sequence::Protocol::[Parameters]
[001]   FLASH (pvm)::1-TriPilot-multi::
        [ TR: 200 ms, TE: 3 ms, pixelBW: 500.80 Hz, FlipAngle: 30 degree]
    [01] dim: 2D, matrix_size: 256 x 256 x 15, fov_size: 40 x 40 (unit:mm)
         spatial_resol: 0.156 x 0.156 x 1.000 (unit:mm), temporal_resol: 25600.000 (unit:msec)
[002]   FISP (pvm)::2-TrueFISP-3D-Iso250-180(modified)::
        [ TR: 5.20 ms, TE: 2.60 ms, pixelBW: 781.25 Hz, FlipAngle: 30 degree]
    [01] dim: 3D, matrix_size: 64 x 64 x 32, fov_size: 32 x 32 x 16 (unit:mm)
         spatial_resol: 0.500 x 0.500 x 0.500 (unit:mm), temporal_resol: 51200.000 (unit:msec)
[003]   FISP (pvm)::2-TrueFISP-3D-Iso250-180(modified)::
        [ TR: 5.20 ms, TE: 2.60 ms, pixelBW: 781.25 Hz, FlipAngle: 30 degree]
    [01] dim: 3D, matrix_size: 64 x 32 x 64, fov_size: 32 x 16 x 32 (unit:mm)
         spatial_resol: 0.500 x 0.500 x 0.500 (unit:mm), temporal_resol: 102400.000 (unit:msec)
[004]   FISP (pvm)::2-TrueFISP-3D-Iso250-180(modified)::
        [ TR: 5.20 ms, TE: 2.60 ms, pixelBW: 781.25 Hz, FlipAngle: 30 degree]
    [01] dim: 3D, matrix_size: 32 x 64 x 64, fov_size: 16 x 32 x 32 (unit:mm)
         spatial_resol: 0.500 x 0.500 x 0.500 (unit:mm), temporal_resol: 102400.000 (unit:msec)
[005]   FISP (pvm)::2-TrueFISP-3D-Iso250-180(modified)::
        [ TR: 5.20 ms, TE: 2.60 ms, pixelBW: 781.25 Hz, FlipAngle: 30 degree]
    [01] dim: 3D, matrix_size: 64 x 64 x 32, fov_size: 32 x 32 x 16 (unit:mm)
         spatial_resol: 0.500 x 0.500 x 0.500 (unit:mm), temporal_resol: 51200.000 (unit:msec)
[006]   FISP (pvm)::2-TrueFISP-3D-Iso250-180(modified)::
        [ TR: 5.20 ms, TE: 2.60 ms, pixelBW: 1562.50 Hz, FlipAngle: 30 degree]
    [01] dim: 3D, matrix_size: 64 x 32 x 64, fov_size: 32 x 16 x 32 (unit:mm)
         spatial_resol: 0.500 x 0.500 x 0.500 (unit:mm), temporal_resol: 102400.000 (unit:msec)
[007]   FISP (pvm)::2-TrueFISP-3D-Iso250-180(modified)::
        [ TR: 5.20 ms, TE: 2.60 ms, pixelBW: 1562.50 Hz, FlipAngle: 30 degree]
    [01] dim: 3D, matrix_size: 32 x 64 x 64, fov_size: 16 x 32 x 32 (unit:mm)
         spatial_resol: 0.500 x 0.500 x 0.500 (unit:mm), temporal_resol: 102400.000 (unit:msec)
```

Figure 3. The example of brkraw command usage to print out data information



```
$ brkraw bids_helper <input_path> <output_path> [-j]
```

	A	B	C	D	E	F	G	H	I	J	K	L
1	RawData	SubjID	SessID	ScanID	RecoID	DataType	task	acq	ce	rec	run	modality
2	SL170111.Fu1	SLRC01M	1	4	1	anat						
3	SL170111.Fu1	SLRC01M	1	5	1	fmap						
4	SL170111.Fu1	SLRC01M	1	7	1	func						
5	SL170111.Fu1	SLRC01M	1	8	1	func						
6	SL170111.Fu1	SLRC01M	1	9	1	fmap						
7	SL170111.Fu1	SLRC01M	1	11	1	func						
8	SL170111.Fu1	SLRC01M	1	12	1	func						
9	SL170111.Fu1	SLRC01M	1	13	1	func						
10	SL170111.Fu1	SLRC01M	1	14	1	func						
11	SL170111.Fu1	SLRC01M	1	15	1	func						
12	SL170111.Fu2	SLRC02M	1	4	1	anat						
13	SL170111.Fu2	SLRC02M	1	5	1	fmap						
14	SL170111.Fu2	SLRC02M	1	7	1	func						
15	SL170111.Fu2	SLRC02M	1	8	1	func						
16	SL170111.Fu2	SLRC02M	1	9	1	func						
17	SL170111.Fu2	SLRC02M	1	10	1	func						
18	SL170111.Fu3	SLRC03M	1	4	1	anat						
19	SL170111.Fu3	SLRC03M	1	5								
20	SL170111.Fu3	SLRC03M	1	7								
21	SL170111.Fu3	SLRC03M	1	8								

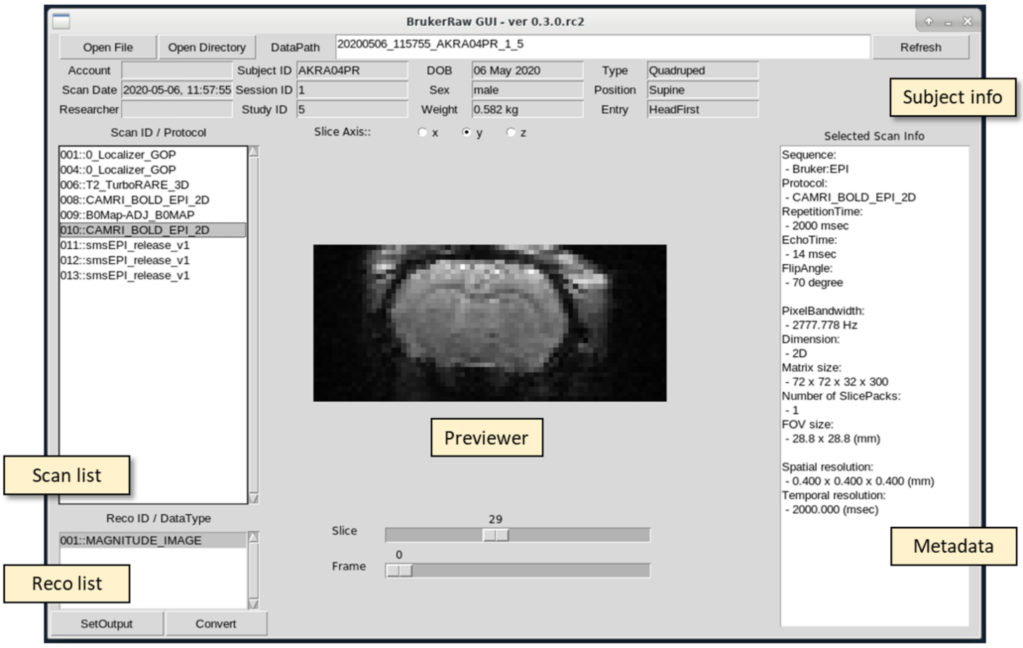
```
{
    "Manufacturer": "VisuManufacturer",
    "ManufacturersModelName": "VisuStation",
    "DeviceSerialNumber": "VisuSystemOrderNumber",
    "StationName": "VisuStation",
    "SoftwareVersion": "VisuAcqSoftwareVersion",
    "MagneticFieldStrength": {
        "Freq": "VisuAcqImagingFrequency",
        "Equation": "Freq / 42.576"
    },
    "ReceiveCoilName": "VisuCoilReceiveName",
    "ReceiveCoilActiveElements": "VisuCoilReceiveType",
    "GradientSetType": "ACQ_status",
    "MRTransmitCoilSequence": {
        "Name": "VisuCoilTransmitName",
        "Manufacture": "VisuCoilTransmitManufacturer",
        "Type": "VisuCoilTransmitType"
    }
}
```

```
$ brkraw bids_convert <input_path> <bids_datasheet.xlsx> -r <JSON syntax template>.json
```

Outputs
sub-SLRC01M
ses-1
anat
func
fmap
sub-SLRC01M_ses-1_<updated>.nii.gz
sub-SLRC01M_ses-1_<updated>.json
sub-SLRC02M
ses-1
anat
func
fmap
sub-SLRC03M
ses-1
anat
func
Converted files with BIDS standard

```
{
    "Manufacturer": null,
    "ManufacturersModelName": "Biospec 94/30",
    "DeviceSerialNumber": null,
    "StationName": "Biospec 94/30",
    "SoftwareVersion": 5.1,
    "MagneticFieldStrength": 9.402658597929491,
    "ReceiveCoilName": null,
    "ReceiveCoilActiveElements": null,
    "GradientSetType": "S090",
    "MRTransmitCoilSequence": {
        "Name": null,
        "Manufacture": null,
        "Type": null
    }
}
```

Figure 4. The example usage of the command-line tool 'brkraw' for BIDS data organization.



DataPath 20200506_115755_AKRA04PR_1_5

Account
Scan Date 2020-05-06, 11:57:55
Researcher
Subject ID AKRA04PR
Session ID 1
Study ID 5
DOB 06 May 2020
Sex male
Weight 0.582 kg
Type Quadruped
Position Supine
Entry HeadFirst

001::0_Localizer_GOP
004::0_Localizer_GOP
006::T2_TurboRARE_3D
008::CAMRI_BOLD_EPI_2D
009::B0Map-ADJ_B0MAP
010::CAMRI_BOLD_EPI_2D
011::smsEPI_release_v1
012::smsEPI_release_v1
013::smsEPI_release_v1

001::MAGNITUDE_IMAGE

Sequence:
 - Bruker:EPI
Protocol:
 - CAMRI_BOLD_EPI_2D
RepetitionTime:
 - 2000 msec
EchoTime:
 - 14 msec
FlipAngle:
 - 70 degree

PixelBandwidth:
 - 2777.778 Hz
Dimension:
 - 2D
Matrix size:
 - 72 x 72 x 32 x 300
Number of SlicePacks:
 - 1
FOV size:
 - 28.8 x 28.8 (mm)

Spatial resolution:
 - 0.400 x 0.400 x 0.400 (mm)
Temporal resolution:
 - 2000.000 (msec)

Figure 5. The graphical user interface (GUI) for previewing image and parameters

Acknowledgements

We thank to the researchers of the Rorden lab at the University of South Carolina, especially Drs. Chris Rorden and Sebastiano Ferraris, regarding their pioneer works that inspire us to start this project, as well as their support on sharing accumulated know-how. We also thank Dr. Mikael Naveau at Cyceron and Gabriel A. Devenyi at Douglas Mental Health University Institute who shared the dataset for benchmarking converter. Lastly, We thank to the staff and colleagues in the Center for Animal MRI (CAMRI) at the University of North Carolina at Chapel Hill for the testing and providing helpful feedback. Especially thanks to Ms. Tzu-Wen Wang for the test the data management tools and Ms. Alicia M. Stevans for the help of critical reading of the manuscript. This work was supported by NIH (Grant No: RF1MH117053, R01MH111429, and R01NS091236).

References

Brett, M., Markiewicz, C. J., Hanke, M., Côté, M.-A., Cipollini, B., McCarthy, P., Cheng, C. P., et al. (2020). *Nipy/nibabel: 2.5.2*. Zenodo. doi:10.5281/zenodo.3745545

Chavarrias. (2017). Bruker2nifti - bruker raw to nifti conversion in matlab. *GitHub Repository*. GitHub. Retrieved from https://github.com/CristinaChavarrias/Bruker2nifti

Ferraris, S., Shakir, D., Merwe, J. V. D., Gsell, W., Deprest, J., & Vercauteren, T. (2017). Bruker2nifti: Magnetic resonance images converter from bruker paravision to nifti format. *Journal of Open Source Software*, *2*(16), 354. doi:10.21105/joss.00354

Gorgolewski, K. J., Auer, T., Calhoun, V. D., Craddock, R. C., Das, S., Duff, E. P., Flandin, G., et al. (2016). The brain imaging data structure, a format for organizing and describing outputs of neuroimaging experiments. *Scientific Data*, *3*(1). doi:10.1038/sdata.2016.44

Lowekamp, B. C., Chen, D. T., Ibáñez, L., & Blezek, D. (2013). The design of SimpleITK. *Frontiers in Neuroinformatics*, *7*. doi:10.3389/fninf.2013.00045

Mikael Naveau, Axel Mathieu, & Devenyi, G. (2019). MRI dataset in the paravision bruker format to test bruker 2 nifti conversion softwares. *GitLab Project*. GitLab. Retrieved from https://gitlab.com/naveau/bruker2nifti_qa

Rorden. (2018). Bru2Nii - a simple tool for converting bruker paravision mri data to the nifti file format. *GitHub Repository*. GitHub. Retrieved from https://github.com/neurolabusc/Bru2Nii

Brett, M. (2002). Pvconv.pl - a bruker data converter. *SourceForge*. SourceForge. Retrieved from http://pvconv.sourceforge.net