The ISMRM Raw Data Format

Souheil J Inati[1], Peter Kellman[2], Joseph Naegele[1], Sonia Nielles-Vallespin[2], Vinai Roopchansingh[1], Thomas S Sorensen[3], Kaveh Vahedipour[4], Hui Xue[2], Nicholas R Zwart[5], and Michael S Hansen[2]

[1]National Institute of Mental Health, National Institutes of Health, Bethesda, MD, United States, [2]National Heart Lung and Blood Institute, National Institutes of Health, Bethesda, MD, United States, [3]Department of Computer Science, Aarhus University, Aarhus, Denmark, [4]Institute of Neurosciences and Medicine, Forschungszentrum Juelich, Juelich, Germany, [5]Barrow Neurological Institute, Phoenix, AZ, United States

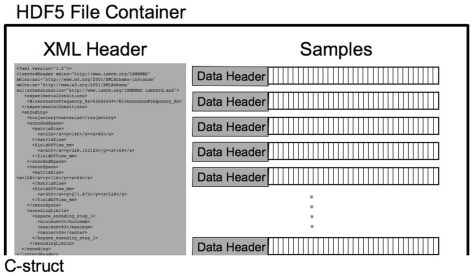
Target Audience: Researchers developing MR image reconstruction algorithms and code

Purpose: A common raw data format is a prerequisite for sharing MR image reconstruction algorithms and code, and is a necessary component of reproducible research. Ideally, this common format would be vendor neutral, and would capture the data fields needed to describe the details of the MR experiment so as to permit image reconstruction from the raw data. We propose the ISMRM Raw Data (ISMRMRD) format as an example of such a common raw data format.

Background: Reproducible research has drawn a great deal of attention recently, as highlighted for example in a recent special issue of Science[1]. The field of computational science in particular has produced several excellent examples of how such research can be carried out, e.g. the wavelab toolbox[2]. The ISMRM has begun to take steps to help to facilitate reproducible research, e.g. with the MRI unbound website[3]. Underlying many of these efforts is a discipline specific file format specification that allows scientists to easily exchange data. Much of modern astronomy research, for example, relies on telescope data in the FITS standard[4]. Other discipline specific data formats have enabled scientists to escape from the limitations imposed by vendor specific proprietary technology, leading to the development of a wide range of post acquisition data analysis tools, and enabling large scale collaborations. Medical imaging has the DICOM standard[5], which allows radiology departments to store data from different vendors on a centralized PACS, and computer scientists to implement novel image processing methods in a vendor neutral manner. The subfield of neuroimaging has NIfTI[6], which underlies large scale projects like the human connectome project[7]. For the field of MRI, these image file formats serve well, but they do not address the needs of practitioners involved in the development of image reconstruction algorithms, nor do they store the data in a format that is very closely tied to the instrument and to the particulars of the MR experiment. We believe that a first and necessary step is the development of a MR specific raw data file format.

Design: This ISMRMRD standard was developed by a subcommittee of the ISMRM Sedona 2013 workshop. It is described by an XML schema and several C-style structures and is designed to capture the details of the experiment in a way that permits image reconstruction. It is important to note, that this goal is fundamentally different from that of proprietary vendor raw data file formats that are intended to capture the state of the scanner GUI at the time of the scan to allow for reproduction of the parameters specific to a particular protocol for a particular pulse sequence. The ISMRMRD standard on the other hand is designed to permit for the exchange of the data and a physics description of the data acquisition, at least in so far as image reconstruction is concerned. The format supports storing k-space trajectories along with the data and provides a simple image format for storing the product of reconstructions.

Implementation: We have created an implementation of this standard using HDF5 files for storage, along with C++, Python, MATLAB, and JAVA libraries for reading and writing ISMRMRD files. The project follows an open source development model with a website, discussion board, and code repository at http://ismrmrd.sourceforge.net. As shown schematically below, the HDF5 container stores the variable length (extensible) XML header, followed by header/data pairs, where each fixed length header (c-struct) encodes readout specific information (k-space location, slice, etc).



C-struct

```
struct AcquisitionHeader
{
    uint16_t            version;                    /*** First unsigned int indicates the version                  */
    uint64_t            flags;                      /*** bit field with flags                                      */
    uint32_t            measurement_uid;            /*** Unique ID for the measurement                             */
    uint32_t            scan_counter;               /*** Current acquisition number in the measurement             */
    uint32_t            acquisition_time_stamp;     /*** Acquisition clock                                         */
    uint32_t            physiology_time_stamp[3];   /*** Physiology time stamps, e.g. ecg, breating, etc.          */
    uint16_t            number_of_samples;          /*** Number of samples, '0' if data                            */
    uint16_t            available_channels;         /*** Available channels                                        */
    uint16_t            active_channels;            /*** Active channels                                           */
    uint64_t            channel_mask[16];           /*** Channel mask                                              */
    uint16_t            discard_pre;                /*** Number of samples to be discarded at the beginning        */
    uint16_t            discard_post;               /*** Number of samples to be discarded at the end              */
    uint16_t            center_sample;              /*** Sample at the center of the FFT                           */
    uint16_t            encoding_space_ref;         /*** Encoding space, refer to the order in the array           */
    uint16_t            trajectory_dimensions;      /*** Trajectory dimensions, 0 means no trajectory              */
    float               sample_time_us;             /*** Time between samples in us, used for trajectory calculation */
    float               position[3];                /*** Center of the first voxel in the readout, in mm, patient coordinate system */
    float               quaternion[4];              /*** Direction cosines, 1 means PE is along x-axis, etc.       */
    float               patient_table_position[3];  /*** Position of the patient table in the coordinate system    */
    EncodingCounters    idx;                        /*** Multiple loop counters, see below                         */
    int32_t             user_int[8];                /*** Free user parameters                                      */
    float               user_float[8];              /*** Free user parameters                                      */
};
```

```
struct EncodingCounters {
    uint16_t kspace_encode_step_1;
    uint16_t kspace_encode_step_2;
    uint16_t average;
    uint16_t slice;
    uint16_t contrast;
    uint16_t phase;
    uint16_t repetition;
    uint16_t set;
    uint16_t segment;
    uint16_t user[8];
};
```

Conclusion: We have presented an initial attempt at creating a standard for storing raw MR data and have demonstrated an implementation that supports several common programming languages. We believe that as this standard is refined and evolves it will serve as a foundation upon which practitioners can base reproducible research project and collaborations. A great deal of work remains and we ask for input from the ISMRM community to help shape the direction of the format.

References: [1] Science (2011) 334. [2] http://statweb.stanford.edu/~wavelab [3] http://www.ismrm.org/mri_unbound [4] http://fits.gsfc.nasa.gov [5] http://medical.nema.org [6] http://nifti.nimh.nih.gov [7] http://www.humanconnectome.org