

# MDF: Magnetic Particle Imaging Data Format

T. Knopp<sup>1,2</sup>, T. Viereck<sup>3</sup>, M. Hofmann<sup>1,2</sup>

<sup>1</sup>Section for Biomedical Imaging, University Medical Center Hamburg-Eppendorf

<sup>2</sup>Institute for Biomedical Imaging, Hamburg University of Technology

<sup>3</sup>Institute of Electrical Measurement and Fundamental Electrical Engineering, TU Braunschweig

December 29, 2015

Version **0.1**

## Abstract

Magnetic particle imaging (MPI) is a tomographic method to determine the spatial distribution of magnetic nanoparticles. In this document a file format for the standardized storage of MPI data is introduced. The aim of the Magnetic Particle Imaging Data Format (MDF) is to provide a coherent way of exchanging MPI data acquired with different MPI scanners worldwide. The focus of the file format is on sequence parameters, raw measurement data, calibration data, and reconstruction data. The format is based on the hierarchical document format (HDF) in version 5 (HDF5).

## 1 Introduction

The purpose of this document is to introduce a file format for exchanging Magnetic Particle Imaging (MPI) data. The Magnetic Particle Imaging Data Format (MDF) is based on the hierarchical document format (HDF) in version 5 (HDF5). HDF5 allows to store multiple datasets within a single file and is thus very flexible to use. To allow the exchange of MPI data, one has to specify a naming scheme within HDF5 files which is the purpose of this document. In order to

create and access HDF5 data, an Open Source C library is available. For most programming languages bindings to this library exist. Matlab supports HDF5 by the functions `h5read` and `h5write`. For Python the `h5py` package exists. The Julia programming language provides access to HDF5 files via the `HDF5` package. For languages based on the .NET framework the `HDF5DotNet` library is available.

In this initial version of the file format the focus is on sequence parameters, raw measurement data, calibration data, and reconstruction data. The format can store three different dataset types

1. Measurement data
2. System calibration data
3. Reconstruction data

It is possible to combine measurement data and reconstruction data into a single file. However, calibration data has to be stored in an independent HDF5 file.

## 1.1 Datatypes

For most parameters a fixed datatype is used, i.e. the drive-field amplitudes are stored as [Float64](#) values. In case of the measurement data and the calibration data the datatype is not restricted such that maximum flexibility is given. In case of no restriction we refer to the datatype as [Any](#).

## 2 Data (group: /)

**Remarks:** Within the root group metadata about the file itself is stored. Within several subgroups, metadata about the experimental setting, the MPI tracer, and the MPI scanner are stored. The actual

Since storing complex data in HDF5 is not standardized, we extend the dimensionality of an existing array and store the real and imaginary part in the last dimension with size 2 (index 0 = real part, index 1 = imaginary part). In this way the real and imaginary part of a complex datum is stored sequentially on disk. When loading the data it is possible to cast it to a complex array in most programming languages.

## 1.2 Contact

If you find mistakes in this document or the specified file format or if you want to discuss extensions to this specification, please contact us by email:

[t.knopp@uke.de](mailto:t.knopp@uke.de)

As the file format is versionized it will be possible to extend it for future needs of MPI. The current version discussed in this document is version 0.1.

data is stored in dedicated groups on measurement data, calibration data, and/or reconstruction data.

Parameter	Type	Dim	Unit/Format	Optional	Description
version	<a href="#">String</a>	1	0.1	no	Version of the file format
uuid	<a href="#">String</a>	1	f81d4fae-7dec-11d0-a765-00a0c91e6bf6	no	Universally Unique Identifier (RFC 4122)
date	<a href="#">String</a>	1	yyyy-mm-ddThh:mm:ss.ms	no	UTC creation time of MDF data set

### 2.1 Study Description (group: /study/)

**Remarks:** The study description group describes the experimental setting under which the MPI data was recorded. The study field may be used as a name tag for several experiments, which are related. The dataset at hand may then be described by a number and a short description. Additionally, the name of the imaged subject

can be provided and the starting time of the MPI measurement can be provided.

The reference field may be used to indicate, if the background signal of the scanner was recorded.

Parameter	Type	Dim	Unit/Format	Optional	Description
name	String	1		yes	Name of the study
experiment	String	1		yes	Experiment number within study
description	String	1		yes	Short description of the experiment
subject	String	1		yes	Name of the subject that was imaged
reference	Int64	1		yes	Flag indicating if field of view was empty during the measurement

## 2.2 Tracer Parameters (group: /tracer/)

**Remarks:** The tracer parameter group contains information about the MPI tracer used during the experiment such as the tracer name, its vendor, the tracer concentration, and the total volume applied.

Note that the injection clock recording the injection time should be synchronized with the clock, which provides the starting time of the measurement.

Parameter	Type	Dim	Unit/Format	Optional	Description
name	String	1		yes	Name of tracer used in experiment
batch	String	1		yes	Batch of tracer
vendor	String	1		yes	Name of tracer supplier
volume	Float64	1	$\mu\text{L}$	yes	Total volume of applied tracer
concentration	Float64	1	mmol(Fe)/L	yes	Concentration of tracer
time	String	1	yyyy-mm-ddThh:mm:ss.ms	yes	UTC time at which tracer injection started

## 2.3 Scanner Parameters (group: /scanner/)

**Remarks:** The scanner parameter group contains information about the MPI scanner used such as the manufacturer, the model, and the facility where the scanner is installed.

Parameter	Type	Dim	Unit/Format	Optional	Description
facility	<a href="#">String</a>	1		no	Facility where the MPI scanner is installed
operator	<a href="#">String</a>	1		no	User who operates the MPI scanner
manufacturer	<a href="#">String</a>	1		no	Scanner manufacturer
model	<a href="#">String</a>	1		no	Scanner model
topology	<a href="#">String</a>	1		no	Scanner topology (e.g. FFP or FFL)

## 2.4 Acquisition Parameters (group: /acquisition/)

**Remarks:** The acquisition parameter group can describe different imaging protocols and trajectory settings. The corresponding data is organized into general information, a subgroup containing data on the particle excitation and a subgroup containing data on the receive channels.

In general each MPI dataset consists of the measurement data of  $L$  frames. A frame groups all data together that will be used to

reconstruct an image/volume. On certain MPI scanners the drive-field field-of-view (FOV) can be shifted either by magnetic fields or by mechanical movement. Therefore, a frame may consist of several sub-measurements. For instance a Cartesian 2D trajectory with 100 lines would be realized by setting `numPatches = 100`.

Parameter	Type	Dim	Unit/Format	Optional	Description
numFrames	<a href="#">Int64</a>	1	1	no	Number of available frames, denoted by $L$
framePeriod	<a href="#">Float64</a>	1	ms	no	Complete time to acquire a full frame
numPatches	<a href="#">Int64</a>	1	1	no	Number of patches within a frame denoted by $J$
gradient	<a href="#">Float64</a>	3 or $J \times 3$	T/m	no	Gradient strength of the selection field in $x$ , $y$ , and $z$ directions
time	<a href="#">String</a>	1	yyyy-mm-ddThh:mm:ss.ms	no	UTC start time of MPI measurement

#### 2.4.1 Drive Field (group: /acquisition/drivefield/)

**Remarks:** The drive field subgroup describes the details on the imaging protocol and trajectory settings. On the lowest level each MPI scanner contains  $D$  channels for particle excitation.

These excitation signals are usually sinusoidal and can be described by  $D$  amplitudes (drive field strengths), a base frequency, and  $D$  dividers. Depending on the base frequency and the divider a

periodic excitation signal is generated defining the sampling trajectory of the field-free point or field-free line. The trajectory may cover a 1D, 2D, or 3D area.

Certain parameters such as **strength**, **fieldOfViewCenter**, and **fieldOfView** can either be defined globally for the entire multi-patch sequence or individually for each patch of the sequence.

Parameter	Type	Dim	Unit/Format	Optional	Description
numChannels	Int64	1	1	no	Number of drive field channels, denoted by $D$
strength	Float64	$D$ or $J \times D$	mT	no	Applied drive field strength
baseFrequency	Float64	1	Hz	no	Base frequency to derive drive field frequencies
divider	Int64	$D$	1	no	Divider for drive fields frequencies ( <b>baseFrequency</b> / <b>divider</b> )
period	Float64	1	ms	no	Drive field trajectory period
averages	Int64	1		no	Number of internal averages (applied in hardware/software)
repetitionTime	Float64	1	ms	no	Time to complete averaged DF trajectory (averages * period)
fieldOfView	Float64	3 or $J \times 3$	mm	no	Approximate size of the area/volume captured by the trajectory
fieldOfViewCenter	Float64	3 or $J \times 3$	mm	no	Center of the drive field trajectory (relative to origin/center)

#### 2.4.2 Receiver (group: /acquisition/receiver/)

**Remarks:** The receiver subgroup describes details on the MPI receiver. For a multi-patch sequence it is assumed, that signal acquisition only takes place during particle excitation. During each

drive-field cycle,  $C$  receive channels record the superposition of the change of the particle magnetization at  $Z$  equidistant time points.

Parameter	Type	Dim	Unit/Format	Optional	Description
numChannels	Float64	1		no	Number of receive channels $C$
bandwidth	Float64	1	Hz	no	Bandwidth of the receiver unit
numSamplingPoints	Float64	1		no	Number of sampling point within one drive-field period denoted by $Z$
frequencies	Float64	$K$	Hz	no	Vector containing recorded frequencies

## 2.5 Measurements (group: /measurement/)

**Remarks:** Measured data can be stored in Fourier domain (FD) representation as well as time domain (TD) representation. Usually only one of the representation is stored and one has to calculate the missing representation if it is needed but not available.

One measurement consists of the voltages recorded in all receive channels for all patches. The number of measurements in time domain is thus given by  $\widetilde{M} = JCZ$ . In frequency domain one measure-

ment has  $M = KCZ$  data points. On disc the temporal/frequency index is the fastest, while the index over the patches is the slowest. If several measurements are acquired (indicated by *numFrames*), the measurements are concatenated. The number of measurements is denoted by  $L$ .

Parameter	Type	Dim	Unit/Format	Optional	Description
dataFD	Any	$L \times C \times K \times 2$ or $L \times J \times C \times K \times 2$		yes	Measurement data stored in Fourier domain representation. The last dimension is used for storing the complex data.
dataTD	Any	$L \times C \times Z$ or $L \times J \times C \times Z$		yes	Measurement data stored in time domain representation

## 2.6 Calibration (group: /calibration/)

**Remarks:** Calibration data is usually acquired by shifting a delta sample through the FOV of the scanner. The resulting system matrix can be stored in time or frequency domain. The number of spatial points is indicated by  $N$ . For practical reasons the system matrix is stored in a transposed way compared to the measurement data. Hence the index over the positions is the fastest while the index over

all data points of a particular measurement is the slowest. Depending on the programming language used (column-major or row-major order) the system matrix may appear in a transposed representation when loaded into main memory.

If a regular grid is used for sampling, the fastest index is in  $x$  direction, the second fastest index is in  $y$  direction, while the slowest

index is in  $z$  direction. For non-regular sampling points there is the possibility to explicitly store all positions.

Parameter	Type	Dim	Unit/Format	Optional	Description
dataFD	Any	$C \times K \times N \times 2$ or $J \times C \times K \times N \times 2$		yes	System matrix stored in its Fourier space representation. The last dimension is used for storing the complex data.
snrFD	Float64	$C \times K$		yes	Signal-to-noise estimate for recorded frequency components
dataTD	Float64	$C \times Z \times N$ or $J \times C \times Z \times N$		yes	System matrix stored in its time domain representation
fieldOfView	Float64	3	mm	yes	Field of view of system matrix
fieldOfViewCenter	Float64	3	mm	yes	Center of the system matrix (relative to origin/-center)
size	Int64	3		yes	Number of voxels in each dimension
positions	Float64	$N \times 3$	mm	yes	Position of each of the grid points, stored as $(x, y, z)$ tripels
deltaSampleSize	Float64	3	mm	yes	Size of delta Sample used for calibration scan

## 2.7 Reconstruction Results (group: /reconstruction/)

Reconstruction results are stored in the parameter **data**. Since the grid of the reconstruction data can be different than the system matrix grid, the grid parameter are mirrored in the reconstruction parameter group.

Usually the data is stored in a real data format but it is also possible to store complex data if the reconstruction output is complex.

Parameter	Type	Dim	Unit/Format	Optional	Description
data	Any	$L \times N$		yes	Reconstructed data
fieldOfView	Float64	3	mm	yes	Field of view of reconstructed data
fieldOfViewCenter	Float64	3	mm	yes	Center of the reconstructed data (relative to origin/center)
size	Int64	3		yes	Number of voxels in each dimension
positions	Float64	$N \times 3$	mm	yes	Position of each of the grid points, stored as $(x, y, z)$ tripels