NIfTI-1 File Format

Summary and Rationale

RW Cox

SSCC/DIRP/NIMH/NIH/DHHS/USA/EARTH

with essential input from John Ashburner, Steve Smith, and Mark Jenkinson

NIfTI-1 Charter

- Squeeze extra metadata into 348 byte long ANALYZE™ 7.5 header (.hdr) to make it more useful
  - As a medium of data interoperability
  - As a medium of data publishing, databasing
  - NIfTI-1 files should still be usable by non-NIfTI-aware software tools
- AFNI (Cox), FSL (Smith), SPM (Ashburner) agree to support new format
  - Hope to carry other programs along with us

What Now Exists

- C header file: **nifti1.h**
  - Extensively commented (1100+ lines)
  - This is the definition of the format
- Sample I/O library C functions: **nifti1_io.c**
  - 2000+ lines, fairly well commented
  - Not mandatory, but shows my interpretation of **nifti1.h**
    - ❖ Particularly for quaternion-based spatial rotation matrix definition

Additions: Overview

❶ **Two** affine coordinate definitions

- 1 orthogonal, 1 general

❷ “Complete” set of 8..128 bit data types

❸ Single or dual file storage (.nii or .hdr/.img)

❹ Standardized way to store vector data

❺ Codes and parameters for data “intent”

❻ Affine data scaling

❼ FMRI-specific slice-timing information

❽ “Magic” string to indicate NIfTI-ness

❶ Coordinates - I

- Affine transformations give coords of voxel **centers** from voxel indexes (**`i,j,k`**).

```
[ x ]   [ m11 m12 m13 m14 ] [ i ]
[ y ] = [ m21 m22 m23 m24 ] [ j ]
[ z ]   [ m31 m32 m33 m34 ] [ k ]
[ 1 ]   [  0   0   0   1  ] [ 1 ]
```

- Units of (**`x,y,z,t`**) can be specified by xyzt_units header element:
  - meters, millimeters, microns
  - seconds, milliseconds, microseconds, Hz, ppm
- Spatial axes are always dim[1..3] = (**`x,y,z`**)

Coordinates - II

- “qform” transformation matrix is specified by rotation and by grid spacings (pixdim)
  - proper rotation specified by unit quaternion (3 float values)
  - improper rotation noted by pixdim[0] < 0
    - effect is to change sign of pixdim[3] ≡ Δz
- “sform” transformation matrix is specified by giving all 12 elements
  - pixdim not used for this case

Coordinates - III

- +x=Right, +y=Anterior, +z=Superior
- Each transformation has a code to indicate the “meaning” of its coordinate system:
  - Scanner anat = coordinates reported in image header (e.g., from DICOM)
  - Aligned anat = coordinates aligned to “truth” or to some reference image
  - Talairach
  - MNI-152

Coordinates - IV

- “qform” usually to be Scanner anat or Aligned anat
  - “q” for “quaternion”
- “sform” usually to be Talairach or MNI-152 (a standard frame)
  - “s” for “standard”
- Time axis, if present, is always dim[4]
  - Units (s, ms, μs) specified in xyzt_units
  - time_offset field specifies origin of *t*

❷ Data Types

```
#define DT_UINT8                   2  /* new names */
#define DT_INT16                   4  /* for old   */
#define DT_INT32                   8  /* ANALYZE™  */
#define DT_FLOAT32                16  /* datatype  */
#define DT_COMPLEX64              32  /* codes     */
#define DT_FLOAT64                64
#define DT_RGB24                 128
------------------------------------------------------
#define DT_INT8                  256  /* new codes */
#define DT_UINT16                512  /* for the   */
#define DT_UINT32                768  /* NIfTI-1   */
#define DT_INT64                1024  /* world     */
#define DT_UINT64               1280
#define DT_FLOAT128             1536
#define DT_COMPLEX128           1792
#define DT_COMPLEX256           2048
```

❸ Single or Dual File Storage

- Dual files: name.hdr and name.img
  - name.hdr is 348 bytes long, as always
  - ❽ magic string is "ni1"   ["1" indicates NIfTI-1]
  - data bytes start at offset 0 in name.img
- Single file: name.nii (.nif and .nft are taken)
  - ❽ magic string is "n+1"
  - data bytes start at offset (int)vox_offset in name.nii ; header occupies 1st 348 bytes of file
  - Useful for Web downloading?

format version number

❹ Vector Data - I

- Vector data ≡ more than one value stored per spatiotemporal voxel
- Vector "dimension" is always dim[5] > 1
  - Example: 1D time series of 3-vectors has
    - ❖ dim[0] = 5
    - ❖ dim[1] = dim[2] = dim[3] = 1 (no spatial extent)
    - ❖ dim[4] = # time points pixdim[4] = time spacing
    - ❖ dim[5] = 3
    - ❖ datatype = DT_FLOAT32 (e.g.)

Vector Data - II

- Limitations of ANALYZE™ format mean all elements of vector must be same type
- Why always dim[5] ?
  - ☺ **Pro**: Reserving dim[1..3] for space and dim[4] for time means that non-NIfTI-aware programs may still make some sort of sense out of a NIfTI-1 dataset
  - ☹ **Con**: must reformat data array to bring all components of vector together in memory

❺ Data Intent - I

- intent_code field describes “meaning” of data values
  - intent_p1, intent_p2, intent_p3 parameters
  - intent_name string
- intent_codes from 2..22 are for various common statistical distributions
  - e.g., 2 = t-statistic (intent_p1=DOF)
- intent_codes > 1000 label other cases
  - e.g., 1005 = symmetric square matrix (intent_p1=matrix dimension)

Data Intent - II

Statistical Codes

NIFTI_INTENT_CORREL	2	NIFTI_INTENT_FTEST_NONC	12
NIFTI_INTENT_TTEST	3	NIFTI_INTENT_CHISQ_NONC	13
NIFTI_INTENT_FTEST	4	NIFTI_INTENT_LOGISTIC	14
NIFTI_INTENT_ZSCORE	5	NIFTI_INTENT_LAPLACE	15
NIFTI_INTENT_CHISQ	6	NIFTI_INTENT_UNIFORM	16
NIFTI_INTENT_BETA	7	NIFTI_INTENT_TTEST_NONC	17
NIFTI_INTENT_BINOM	8	NIFTI_INTENT_WEIBULL	18
NIFTI_INTENT_GAMMA	9	NIFTI_INTENT_CHI	19
NIFTI_INTENT_POISSON	10	NIFTI_INTENT_INVGAUSS	20
NIFTI_INTENT_NORMAL	11	NIFTI_INTENT_EXTVAL	21
		NIFTI_INTENT_PVAL	22

- If distributional parameters don’t depend on voxel, intent_p? is used
- If distributional parameters depend on voxel, dim[5] is used
- Plan: provide C code for densities and cdfs of these distributions

Data Intent - III

Non-Statistical Codes

```
#define NIFTI_INTENT_ESTIMATE   1001
#define NIFTI_INTENT_LABEL      1002
#define NIFTI_INTENT_NEURONAME  1003
#define NIFTI_INTENT_GENMATRIX  1004 // below
#define NIFTI_INTENT_SYMMATRIX  1005 // here,
#define NIFTI_INTENT_DISPVECT   1006 // would
#define NIFTI_INTENT_VECTOR     1007 // need
#define NIFTI_INTENT_POINTSET   1008 // dim[5]
#define NIFTI_INTENT_TRIANGLE   1009
#define NIFTI_INTENT_QUATERNION 1010
```

❻ Data Scaling

- scl_slope and scl_inter fields define how data should be scaled
  - if scl_slope ≠ 0, then

    $$v_{\text{true}} = \text{scl\_slope} \times v_{\text{file}} + \text{scl\_inter}$$

    for each voxel **v**alue from the file
- cal_min and cal_max fields describe data range to be mapped for display
  - display paradigm (e.g., colormap) not defined in NIfTI-1

❼ FMRI Slice Information

- dim_info field contains freq_dim, phase_dim, and slice_dim subfields
  - Each value is 0 (indicating no info) or in 1..3 (indicating which dim[] has which MRI role)
  - Example: freq_dim=1 phase_dim=2 slice_dim=3 means
    - ❖voxel index i = frequency encoding, index j = phase encoding, index k = slice direction
  - If concepts don't apply (e.g., spiral), set subfields to zero (e.g., freq_dim=phase_dim=0)

FMRI Slice Information - II

- slice_duration field, if positive and if slice_dim > 0, indicates inter-slice timing
  - slice_duration * dim[4] can be less than pixdim[4] for clustered acquisition schemes
- slice_code, if positive, indicates slice-timing pattern
  - Also must have slice_duration > 0 and slice_dim > 0

FMRI Slice Information - III

- 4 possible values for slice_code:
  - NIFTI_SLICE_SEQ_INC = sequential increasing
  - NIFTI_SLICE_SEQ_DEC = sequential decreasing
  - NIFTI_SLICE_ALT_INC = alternating increasing
  - NIFTI_SLICE_ALT_DEC = alternating decreasing
- Timing runs over slice indexes from slice_start to slice_end
  - These fields allows for padding slices off the edges of the true MRI volume
  - If present, padding slices wouldn't fit into the slice timing pattern given by slice_code

FMRI Slice Information - IV- Example

dim[slice_dim]=7 (7 slices total, indexed 0..6)

slice_duration=0.1

slice_start=1, slice_end=5 (1 padding slice on each edge)

Table shows time offset of each slice

index	SEQ_INC	SEQ_DEC	ALT_INC	ALT_DEC
6 —	n/a	n/a	n/a	n/a
5 —	0.4	0.0	0.2	0.0
4 —	0.3	0.1	0.4	0.3
3 —	0.2	0.2	0.1	0.1
2 —	0.1	0.3	0.3	0.4
1 —	0.0	0.4	0.0	0.2
0 —	n/a	n/a	n/a	n/a

Some Things *NOT* Added

- Multiple headers and image arrays in 1 file
  - Goal: store more complex objects, such as surface definitions, which aren't expressible as a set of values over a tensor product grid
- Some way to store user-defined types
  - Or at least define more datatype codes for things very likely to arise commonly
    - e.g., 3-vectors, 3×3 matrices
- Separate code for byte ordering of data (vs. byte ordering of header)

Further Efforts [Sep 2003]

- Documentation of API in **nifti1_io.c**
  - Mostly just formatting top-of-function comments and explaining the concepts
- Sample C functions for use with all statistical distributions
  - Functions for intent_code=2..10 exist in AFNI already
- Full incorporation into AFNI!

reads but doesn't write