A New Line Database Format and Data Processing
The Potential of a Modern Database

Date of Issue: February 14, 2012
Author: Lutz Wendorff

Contents

1 Introduction

Data from airborne geophysics generally have some typical characteristic which distinguishes them from typical SQL databases: There are a bunch of channels containing a big number of data items in very regular fashion being connected by their time stamp. A data item can consist of some single value or a data structure like radiometric spectra or EM data samples. With rather little effort, the data of interest can be organized such that complete information records gained from joining all channels are simply achieved by reading from each channel the very same record. Effectively this is just a large table of long records. Then, why not storing these long records in simple binary files? Because you would need to rewrite the whole file for just changing or adding one single channel. SQL databases are not very suitable for this type of data.

2 The Solution

The solution to all these requirements is HDF5. For those who never heard of it, Wikipedia and Google may be a good starting point, and, of course, http://www.hdfgroup.org. The author made first developments back in 2009 by downloading the source code using the products from its compilation. But there are binary libraries available for Windows and Linux, for Fortran and C and more. The HDF5 files can easily be used by MatLab and Python. For example Statoil, Interaction and EMGS [emgs 2010] are proposing HDF5 as a generally accepted data format in marine EM. In the mean time, the company Interaction has become part of Fugro.

2.1 HDF5

2.1.1 General structure of HDF5

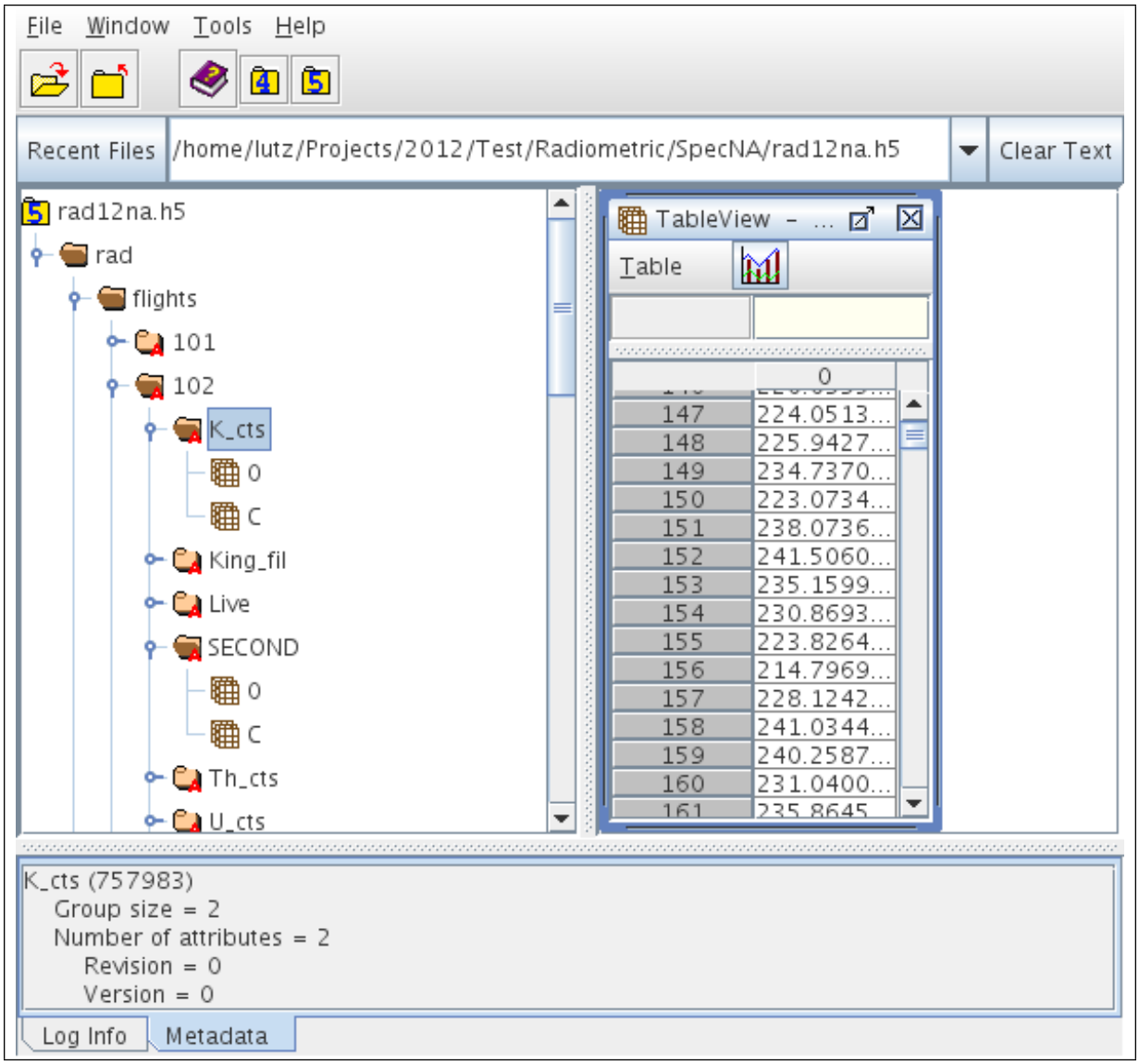
Developing a new database does not mean just to install SQL or HDF5 and you are done. The internal design is still to be developed and is a very important part of it. XYZ files of Geosoft for example is a very straight forward design and familiar to everybody working in this field. Data are kept in records, one line keeps different parameters generally belonging to one point in time and space. Some companies in airborne geophysics use a similar data structure in binary form. A record thus can contain different data types, perhaps 8-byte floating point for x and y coordinates, 4-byte floating point for height and temperature and some channels as integers or even text strings.

Data sets in HDF5 are multidimensional arrays of a homogeneous type, where each item itself can be a structure of any complexity. The author has to admit that he was first struggling with this, if your records are of an unpredictable structure as your data set develops over time. The solution is to stop thinking of records belonging to one point in time but to think of channels belonging to a flight or line (segment). These channels represent one parameter of measurement and the array is one-dimensional. This can be a coordinate, the magnetic field or a radiometric spectrum, the latter consisting of many numbers covering all spectrometer channels. A complete record is very easily re-established by just reading the same record of each channel.

Now the question may be why not just writing each channel of each line into one separate binary file. Whoever asks this question has not looked into the great features of HDF5. And even modern file systems are not optimized for this possibly huge number of files as there may be easily many thousands of line segments with 100 and more channels each, only to be distinguished by their names. Perhaps the data should be stored in a compressed format saving not only space on the hard disk but also i/o time for reading and writing. This would require the use of some compression libraries and, if channels can develop to larger sizes, some sort of chunking has to be implemented.

There is no reason to reinvent the wheel. HDF5 provides all features which may be used to store geophysical data. Some features are there which may not be considered important right away but may turn out as quite beneficial eventually. Who knows whether the data is some time to be read on a completely different computer platform. With HDF5 no problem. Data can be organized in one single HDF5 file or in a number of these files as may be logically advisable. The author used HDF5 for storing large matrices of more than 40 GB without any problem or downgrading of access time for single vectors.

Here are screen shots showing the content of two such HDF5 files. We see on the left the hierarchical data structure within the HDF5 file: mag/flights/flight-number/channel. Under "channel" is one more level called "version". The version would generally be '0', but can increment to any number if new versions of that channel are generated. There is always the version "C", which is not a version on its own but

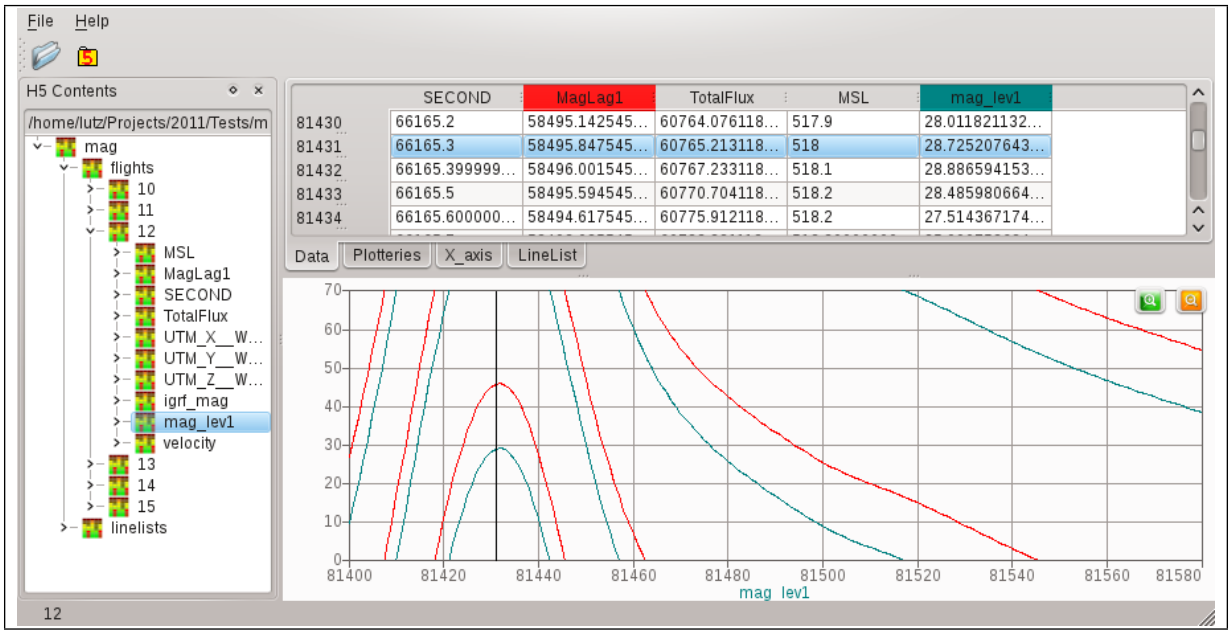


A screen shot of HDFView showing a line data base

Figure 1: Example of a new line database shown by *HDFView*

an alternative link to some selected version for further processing. This way, the finally selected version would be part of the database and does not need to be known to any program. If you select a channel in ViewLine (2nd screen shot), the version "C" will be used but you can open the channel folder to select (additionally) any other version. If you move to the next flight, only the "C" version of any channel will be shown and you would eventually need to select another version explicitly again as otherwise the selected version might not exist at the other flight while "C" always exists.

Another wonderful feature with HDF5 is that any data link or data set can have a number of named attributes attached. This way, for every flight exists besides the flight number an attribute "Doy" with the number for the day-of-the-year, and likewise the attribute "Year". In the HDF5 file you could also maintain some file access sequence number which is then also written to any changed channel when changed or written, see *Revision* in the lower part of fig. 1 for potassium counts. This way it would be



A screen shot of ViewLine

Figure 2: ViewLine (*vl*)

easy to see whether a "child-channel" is (re)written after the "parent-channels" and thus based on current data in case some data has been reprocessed.

In the hierarchy parallel to "mag" in fig. 2 or to "rad" in fig. 1 you will see at the bottom "linelists". Thereunder are lists of lines for each flight. There can be any number of linelists, perhaps for different purposes covering only some data for test processing, or with lines having different extensions beyond the actual survey limits. This is used to only read these lines in a gridding program, or it is used for a program to provide output for delivery to the client. The line lists are also with version number as described for channels above. The program "add_stats" can be used to generate statistics based on any line list and will be stored in the hierarchy as "statistcs" parallel to the linelists. For example running add_stats on a small line list just containing the lines of an attenuation test flight could provide the averages for each height and channel of interest.

2.1.2 More data to go with HDF5

A professionally used data base would also require to store some ancillary data that applies to the whole project. This could be stored in its own hierarchy, right under the root, i.e. in the example above parallel to "rad" or "mag" respectively. This could be information about the geographic region, UTM zone, ellipsoid and datum, information about used equipment or aircraft and many more things perhaps. This should again be provided in a well designed structure that certain programs which may need this information, can access this information correctly. E.g. a survey may use different aircraft and need different sets of compensation coefficients to remove manoeuvre effects or other calibration parameters.

Another data hierarchy may be under "history", showing the processing history for each data item. Any program which writes data to the file could make an entry in a hierarchy, e.g. parallel to "mag" showing all the processing under this hierarchy. The name of the link could be the program name and the file access sequence number. Any data item (channel) produced will get this as an identifier for its

history and therefore, the history is stored only once even if many channels have been produced by that program, showing a concise listing file.

2.1.3 Utilities coming with HDF5

Fig. 1 shows a screen shot from *HDFView*, which is a visual tool for browsing and editing HDF4 and HDF5 files. This tool shows the complete content of any HDF5 file and could be used to investigate such file if its content is otherwise unknown. This tool can also be used to move data, change its structure and also to export content, e.g. make an ASCII listing file of the line list.

A number of utility programs are part of the official HDF5 package. h5copy for example allows in a very simple way to copy one item, say a flight, from an existing HDF5 file to another or new file maintaining the whole hierarchy and copying together with all attributes.

2.2 Working with HDF5

2.2.1 Basic Programs at Rybno Geo Data

Rybno Geo Data is a very small company which has developed some processing capabilities for airborne geophysics or closely related fields. The author had to recognize that most programs in this field are relatively easy to generate and that there are extended resources freely available (e.g. Generic Mapping Tools).

Many programs in the following require some parameters which are most times not passed over the command line but by using some parameter file which all have the extension ".job".

2.2.1.1 a2cube

The program a2cube will read an ASCII file and insert the data into and HDF5 file as shown above. A program aprep and aprep2 was used as some preparation steps for data that were downloaded from a site of Natural Resources of Canada to test these applications. Any data loaded to the HDF5 file, at Rybno Geo Data also called "cube", has one (and only one) corresponding channel called "SECOND", which can be considered as the so called "fiducial", which also could be a simple sequence number if time is irrelevant.

2.2.1.2 set_date

The program set_date was used to add the attributes of "Doy" and "Year" from an added line list. This is only necessary if the data do not provide these data items as this would otherwise be done directly by the major import program like a2cube.

2.2.1.3 add_linelist

The program add_linelist will add a line list which can be provided by an ASCII file. Lines at Rybno Geo Data are identified by simple integer numbers, the last two digits could signify possibly a segment number and eventually be followed by more digits giving e.g. a reflight number, depending on the project requirements. This can be used by other programs, e.g. level.

2.2.1.4 add_stats

The program add_stats will add statistics like number of values, number of NULLs, maximum, minimum, average and standard deviation according to a given line list. The result will be stored in a hierarchy parallel to the linelists.

2.2.1.5 add_channels

As may be clear from the above description of HDF5 files, all channels will be accompanied by one time channel and are connected over the record number. If channels have a different time channel, they must be organized in different hierarchies. The program add_channels can merge channels from another hierarchy of the same or a different HDF5 file using linear interpolation to extract values for each target time. Another option is to use this program to export channels into a new HDF5 file, possibly at a new sampling rate.

2.2.1.6 igrf_line

The program igrf_line will read time from the HDF5 file from the attributes "Doy" and "Year" and the channel "SECOND" will provide the time of the day. Coordinates can be read in any projection and datum and will be converted on the fly to latitude, longitude and altitude of WGS-84 which will then be used for the syntheses of spherical harmonics of selected IGRF coefficients. The result is written into specified channels.

2.2.1.7 lagcorrection

The program lagcorrection has the purpose to correct for any time lag of some data channel. The program uses linear interpolation only and can correct for positive and negative lags. The lag is generally comprised of two parts, a constant and some variable depending on speed which may be calculated from position values.

2.2.1.8 map_lines

The program map_lines can calculate any projection and datum conversion. At the beginning, the program will analyze the required steps and will compile the whole formula from this making the process very efficient. There are at most three steps to carry out: (1) from a given projection to latitude, longitude and height at the given datum, (2) from the source datum to the target datum, (3) finally projection to the target projection using the target datum. Eventually some or all steps can be left out if not required. The projection uses the proj4 package as published by the US Geological Survey and the datum conversion by a subroutine programmed by the author following the analytical solution in [HW 1993].

2.2.1.9 level

The program level applies level corrections to traverse and control lines. The data can remain organized by flights. The line list is used to identify data sets of lines as parts of flights. The levelling function can consist of linear segments between given correction values and also interpolated by smooth functions using modified Bézier curves, effectively smoothed over the full length but limited to a predefined envelope around the function of linear segments.

2.2.1.10 fourier1d

The program fourier1d is used for any Fourier domain method applied to line data, especially filter channels in frequency domain, cosine tapered. It is using the FFTW3 package for the Fourier transformations.

2.2.1.11 function_line

Almost any formula can be defined for function_line for calculations based on values of different channels. One formula can create any number of output channels. It is using Polish Reverse Notation and very straight forward to formulate in the parameter file. The formula is first compiled when parsed by the program to create a very efficient routine.

2.2.1.12 cube2sys

The program cube2sys will export data from HDF5 files to be used with Generic Mapping Tools.

2.2.1.13 other programs

There are a number of more programs developed, working on points files instead of HDF5 files or other utilities used in airborne geophysics. E.g. converting the cross-over-errors that can be obtained by using a program of the supplementary package of the Generic Mapping Tools to a balanced set of levelling values for traverse and control lines.

2.2.2 A Vision

As mentioned above, the HDF5 file could contain a hierarchy under "history" containing all the information of the processing that has been carried out. This could be turned around: Create a hierarchy of all the processing to be carried out with the parameters to be used. Consider the HDF5 file as a representation of a rectangular data set, from top to bottom the flights and from left to right the channels with the revision number as a processing sequence number. A control program, perhaps called *update* could then check whether this rectangle is complete and up to date and start the processing programs as required.

For example, adding some acquisition data, i.e. some channels of data, calling *update* could detect what processing steps are needed to fill or update the rectangle, e.g. generate lag time corrected channels, provide IGRF values etc. This could replace to a certain degree the need of larger shell scripts used as processing streams. Such an HDF5 file could be send into the field containing no data but "charged" with the processing stream. This design concept also shows the completeness of these files, there is no relevant information stored in any other files which can get lost.

To load even the programs into such file to make an autonomously living object would certainly be an exaggeration. Viewing the data with *ViewLine*, processing steps could eventually be triggered out of the windows program interactively. This could finally be a highly integrated processing system for airborne geophysical data.

2.3 Efficiency

2.3.1 Processing Efficiency

With processing reasonably sized data sets downloaded from the GSC website, it has been verified that the processing can be done very efficiently using HDF5 files. In general, every major processing stream would only require one single file for all its processing steps. Data can also be stored by lines in HDF5 files, eventually then without line lists. The processing times for this example were short, the HDF5 files had been internally zipped with a medium compression level.

2.3.2 Deployment Time Efficiency

The HDF5 libraries (for Fortran and for C) are extremely well documented. A program developer could install the binaries within minutes. Studying the essentials of HDF5 wouldn't take long and a programmer might get productive within a few days.

If all programs exist for geophysical processing, it is only required to provide the i/o functionality for these programs using the HDF5 file. The API is easy to use and could perhaps be further simplified by using some interface library fine-tuned for the actual requirements. The conversion from some processing system for using the here described database system would be highly efficient.

2.3.3 Development Cost Efficiency

As already mentioned in the prior subsection: Any programmer working with HDF5 will have first-class documentation and therefore be productive within days. There is probably no data storage requirement without a simple solution in HDF5. Therefore storing and reading data is extremely simple and programmers can focus on geophysical questions rather than storing technicalities. It will provide a place for any type of data and storing ancillary data with the database is very straight forward.

When it might be required to get new programmers working on this, it would be easy for anybody to get familiar with a few hundred lines of own programs dealing with HDF5 instead of some complex database management system developed in-house and probably not being as versatile. The cost of development with respect to data storage using HDF5 is all in all marginal.

3 Conclusion

Rybno Geo Data has put together a processing system for airborne geophysical data which is capable of all standard processing steps for magnetic and radiometric measurements and gravity. This document shows the high potential of the system using HDF5 as the back-end for the line database. The only fly in the ointment is the necessity to export data for use with Generic Mapping Tools, especially when these are used for generating new channels which should be integrated into the HDF5 file. As the Generic Mapping Tools work very well in a pipe or chain of programs, it is considered to program a little adaptor that data can directly be piped into applications of the Generic Mapping Tools and also back into the HDF5 file.

HDF5 would be an excellent data format to hold grid data. As the default grid format for Generic Mapping Tools is NetCDF, a format especially popular in meteorology and climate research, Rybno Geo Data decided for the format as well. NetCDF is using HDF5 as its back-end and data could be viewed with HDFView as well. There are also a number of utility and grid-viewing programs for NetCDF freely available that Rybno Geo Data can make use of these also.

References

[emgs 2010] F. Roth, A. Becht, Ø. Andersen, V. Markhus, and A. J. Kaaijk Jenssen: ftp://fileformats.emgs.com/H5EM-TS_1.0/documentation/H5EM-TS_information_sheet.pdf, 2010.

[HW 1993] B. Hofmann-Wellenhof, H. Lichtenegger, and J. Collins: GPS, Theory and Practice, Springer-Verlag, Wien - New York, 2nd edition, 1993.