Data and visualization standardization

- Simulation data have many different **grid structures**:
  - equidistant cartesian grid
  - stretched cartesian grid (UCLA-MHD, Birn/Hesse MHD)
  - equidistant spherical grids (SH codes, ionosphere/thermosphere codes)
  - non-equidistant spherical (SH codes, CTIM, TIME-GCM, TDIM, LFM magnetosphere)
  - completely irregular (LFM magnetosphere, GSFC FEM magnetosphere)
  - oct-trees (Michigan code)
  - patch hierarchies (more general adaptive mesh codes)
  - .....
- Container libraries (HDF, CDF, NetCDF, ...) do **not** solve the problem of making data with different structures compatible. They merely provide a means of packaging and machine independence.
- In principle any data set could be handled as “unstructured”, i.e., full position information comes along with the data. Few viz packages can handle this. File size overhead is about a factor 1.5-5, but viz computation overhead can be 1-2 orders of magnitude: thus generally impractical.
- Interpolating data onto simpler grids with a uniform grid structure is possible, but beware:
  - There is generally a loss of information, but not necessarily so if the interpolated grid is fully contained in the original.
  - File size and memory requirements increase dramatically.

– May nonetheless be useful for a number of applications
– **Proposal:** Two universal *interpolated* formats:
  * stretched cartesian (fits some SH codes, mag codes): reasonable compromise between file size and accuracy, is understood by most viz programs.
  * spherical grid, uniform in ϕ and θ but stretched in R (fits most SH and IO/TH codes)
  * keep the actual file format simple (no libraries), simply byte streams of pre-described order. Most viz packages do it this way for their internal formats (AVS, IBM/DX)

What can be standardized?

- File namespace:
  for example:
  `model.MODEL.RUN.COORD.WHAT.TIME`
  `MODEL: batsrus/tdim/......`
  `RUN`: a run identifier: `abc12345`
  `COORD`: coordinates: `GEI/GSM/GSE/SM/...`
  `WHAT`: which parameters: `bxgse`
  `TIME`: time encoding: `1996:11:24:22:30:00.00`
  or time since start in seconds: `00014800`
- Parameter namespace and units:
  for example:
  `bxgse`: magnetic field B_x component in units of nT
  `btot`: total magnetic field in units of nT
  `press`: plasma pressure in units of pPa
  `dens`: plasma number density in units of cm^{-3}
  `Ne`: ionospheric electron density in units of cm^{-3}

- Standardization process could follow the WWW/IETF model: proposals are published on the web and iterated until they become “accepted standard practice”
- There could be some agreement on the use of certain visualization package(s). How to choose?
- Requirements:
  - should support many grid structures
  - should be widely available for many platforms
  - should support hardware accelerated rendering
  - should support scripting for non-interactive and web (background) application
  - should have rich features (cut planes, isosurfaces, flow lines, ...)

  - should be able to produce publication quality output
  - should be extendible (which implies a modular structure and/or open source)
  - should be, if possible, open source software

- Of the many packages that are in use for scientific visualization IBM Data Explorer (IBM/DX, `www.opendx.org`) probably stands out. It has all the required features and it is open source (free, that is).