Geophysical Research Abstracts,
Vol. 10, EGU2008-A-06932, 2008
SRef-ID: 1607-7962/gra/EGU2008-A-06932
EGU General Assembly 2008

Visualization and data compression for multi-variable geophysical flow simulations

J. Schmalzl (1) and A. Loddoch(2)

(1) Inst. of Geophysics, Muenster University, Germany (joergs@uni-muenster.de), (2) Chevron ETC, Technical Computing, Houston, U.S.A.

Numerical simulations play a key role in understanding the Earth's dynamic. Many of the problems investigated are intrinsically three dimensional and exhibit a time dependent behavior. The temporal fluctuations occur over a wide range of different time scales. Another complication comes from the number of flow variables involved. For a simulation designated to understand the Earth's dynamo one has at least three velocity components, three magnetic field component, a temperature, a composition and the pressure. This combined with a typical numerical resolution of 256x256x256 a single model run can easily produce output data in the Terra-byte range. In order to understand the physical mechanism it is often essential to visualize the spatial and temporal evolution of the data and to postprocess the result. One way of doing this is to visualize the data while calculating the data. This has the advantage of not being forced to store the data. One of the problems with this approach is that only a small subset of the information is preserved and one may need to rerun the complete simulation in order to extract additional information. Our approach to the problem is to use lossy data compression similar to still image data storage(JPEG). Since the governing equations from which the data are calculated contain a diffusive term the spatial variation of the quantities is smooth. This makes them well suited for compression using a discret cosines transform (DCT) like in the JPEG algorithm. We will present an investigation of compression ratio, compression/decompression time, and resulting errors for different schemes. For visualizing and post-processing the data we have developed our own OpenGL based program. We consider this approach superior to using a general purpose package. The program can easily be adopted for utilizing the latest

hardware developments like multi-textures and is portable across different machine architectures.