Subject: Re: HDF SDS array access in IDL
Posted by davidf on Thu, 29 Oct 1998 08:00:00 GMT
View Forum Message <> Reply to Message

William accidentally sent this article to me instead of
posting it here. I've posted it to the newsgroup with
his permission.

D.

David Fanning wrote:
>
> William Clodius (wclodius@lanl.gov) writes:
>
>> [Long, well-written article snipped.]
>
> I've read William's excellent article three times now and
> I'm *still* confused. :-(
>

I have thought about this overnight and may be able to be clearer. There are four different aspects of arrays, how they are stored in memory, how they are accessed by the language, how matrices are mapped to arrays, and how they are displayed on the screen.

1. Memory storage: Many of the aspects of how they are stored in memory are determined by the OS/processor which provides blocks of memory with optimal alignment properties. Any sensible single processor implementation of arrays will try to lay out all elements of the array in one contiguous block for efficient access. (Many languages restrict this block to the heap, but some allow stack allocation). An array oriented language, e.g., APL, IDL, Algol 68, Fortran 90/95, will also create a small descriptor defining the layout.

2. Array access: Because contiguous elements are usually more efficiently accessed than non-contiguous elements, and most languages want to make it easy for users to implement efficient code, most languages define a mapping of array elements to memory locations in the block. Historically, it is this mapping that is usually described by the terms column major versus row major. I am not certain how this convention started. Under this definition both IDL and Fortran are column major.

3. Matrices are mathematical constructs that can be efficiently implemented as arrays with special operations. The mapping of matrices onto arrays in IDL and Fortran may be different, but I have not looked in detail at this. The interpretation of this mapping depends on whether a vector by default is described as a row or column vector. While using

the wrong convention with a given language would be a source of error, using the correct convention should be of comparable efficiency. Note, however, that IDL does not discuss matrices in its discussion of its array naming convention.

4. Array display: Many (most) languages (including C and Fortran) are not concerned with the display of arrays. Such a display for these languages is only defined by external library packages, and can follow the library's arbitrary conventions. A sensible package would map display lines to the most efficient access method, I.e., the last index for C, the first index for Fortran. The IDL language defines the same mapping as an efficient library for Fortran, each line mapping to a different value of the second index.

>> (Note because of the current limitations of C arrays I suspect that
>> IDL's arrays are actually implemented as C pointers with indexing
>> similar to F2C's.)
> <snip>

Forget that part, its not important. For what its worth, I suspect that the main reason that Fortran is not commonly used for image processing is that the language does not define a small integer comparable to the byte.

--

William B. Clodius Phone: (505)-665-9370
Los Alamos Nat. Lab., NIS-2 FAX: (505)-667-3815
PO Box 1663, MS-C323 Group office: (505)-667-5776
Los Alamos, NM 87545 Email: wclodius@lanl.gov