GammaLib - Feature #989

Add support for variable length columns to FITS module.

11/18/2013 08:50 PM - Knödlseder Jürgen

Status:	In Progress	**Start date:**	11/18/2013
Priority:	Normal	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	90%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

Variable length columns are needed for RMF format reading and writing.

Variable length columns are read and written row by row. For each row, the number of elements in the column can differ. It remains to be decided how the storage of variable length information should be handled in GammaLib. CCFits uses std::vector<std::valarray<T> > to store the column data, allowing thus rows of variable length. If a similar strategy is to be implemented in GammaLib, the FITS column classes memory handler would need to be rewritten.

An alternative would be to store the variable length column in a table of maximum length. This would inflate memory but simplify the housekeeping. The only thing needed would be an additional array with one index per row that gives the number of elements per row.

Here a couple of notes from the cfitsio manual:

When reading from a variable length array field one can only read as many elements as actually exist in that row of the table; reading does not automatically continue with the next row of the table as occurs when reading an ordinary fixed length table field. Attempts to read more than this will cause an error status to be returned. One can determine the number of elements in each row of a variable column with the fits_read_descript routine.

When writing row of data to a variable length field the entire array of values for a given row of the table must be written with a single call to fits_write_col. The total length of the array is given by nelements + firstelem - 1. Additional elements cannot be appended to an existing vector at a later time since any attempt to do so will simply overwrite all the previously written data and the new data will be written to a new area of the heap. The fits_compress_heap routine is provided to compress the heap and recover any unused space. To avoid having to deal with this issue, it is recommended that rows in a variable length field should only be written once. An exception to this general rule occurs when setting elements of an array as undefined. It is allowed to first write a dummy value into the array with fits_write_col, and then call fits_write_col_nul to flag the desired elements as undefined. Note that the rows of a table, whether fixed or variable length, do not have to be written consecutively and may be written in any order.

When writing to a variable length ASCII character field (e.g., TFORM = '1PA') only a single character string can be written. The `firstelem' and `nelements' parameter values in the fits_write_col routine are ignored and the number of characters to write is simply determined by the length of the input null-terminated character string.

History

#1 - 11/19/2013 09:17 PM - Knödlseder Jürgen

- Subject changed from Add support for variable length arrays to FITS module. to Add support for variable length columns to FITS module.

- Description updated

#2 - 11/19/2013 09:52 PM - Knödlseder Jürgen

The FITS data are actually accessed using the GFitsTableCol::offset method. It would not be overly complicated to add an index array to GFitsTableCol that collects the vector lengths for variable length arrays. The GFitsTableCol::offset would then need to be modified for variable length columns.

What is more complicated is the handling of the insert method. So far this method adds rows of fixed length. Without changing the interface one could only add rows of zero length, which basically would mean that the memory remains as is but the index array becomes longer. This however also implies that a method is needed to resize the vector in a row. One could implement the following methods:

```
int  number(const int& row) const;              // Returns the number of elements in a given row
void number(const int& row, const int& number);  // Set the number of elements in a given row
```

For variable length columns,

const int& number(void) const;

would simply return the maximum length of the vector column.

#3 - 11/22/2013 05:14 PM - Knödlseder Jürgen

- Status changed from New to In Progress

- Assigned To set to Knödlseder Jürgen

- % Done changed from 0 to 90

Variable-length FITS column support has been implemented for all column types.

Variable-length string columns are not yet operational, though, leading for the moment to a segmentation fault when attempting the creation of variable-length columns. This issue needs still some investigations.

Variable-length Bit columns are also not yet handled properly due to the specific compression scheme of Bit columns. Some more thinking is needed here to properly implement the resize_data() method. Eventually, a specific elements() method needs to be implemented for this class.