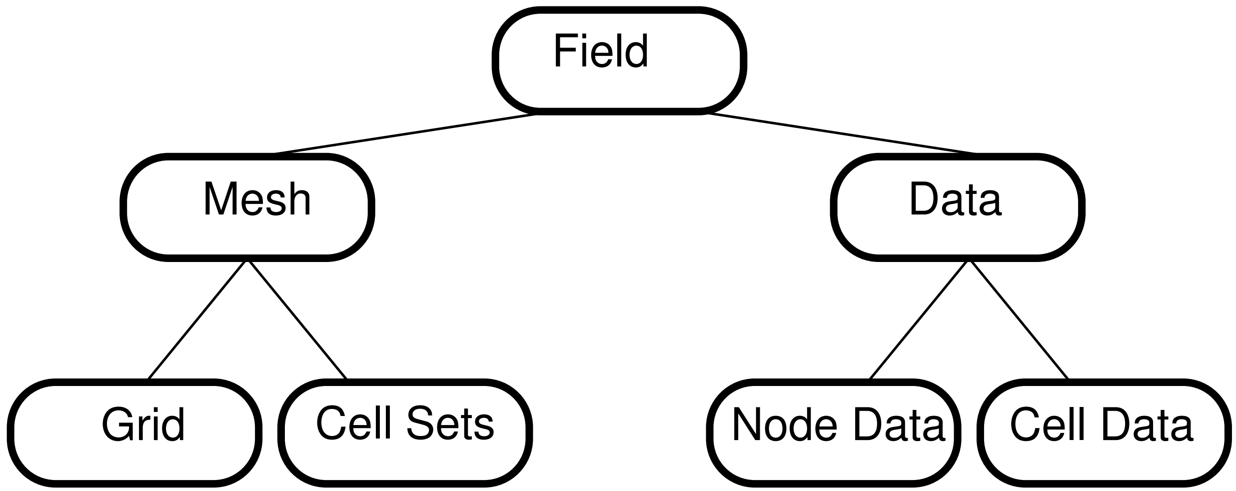
Importing data

- The Express Field
- Overview of representing different data types
- How to import data into AVS/Express

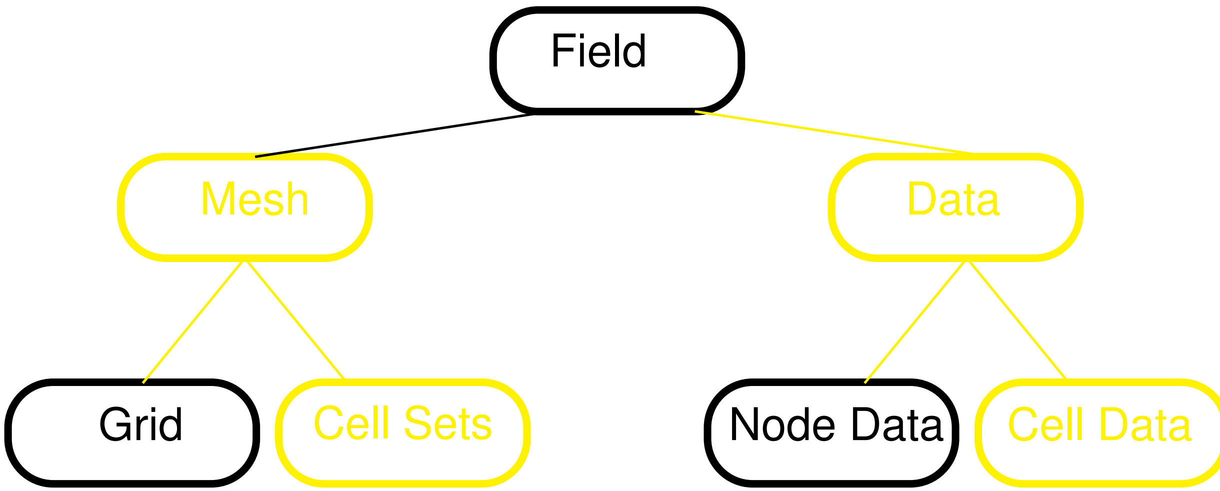
What can you import?

- ❑ Read (and Write) netCDF
- ❑ GISWorldDataBank and GISDigitalElevation
- ❑ Read (and Write) Image
  - ■ BMP, GIF, JPEG, PBM, SGI image, Sun Raster, TIFF, avs .x
- ❑ User defined data is discussed later
- ❑ The Express FLD provides support for AVS5 field, UCD and GEOM
- ❑ Lets look in more detail at the FLD...

The Express Field

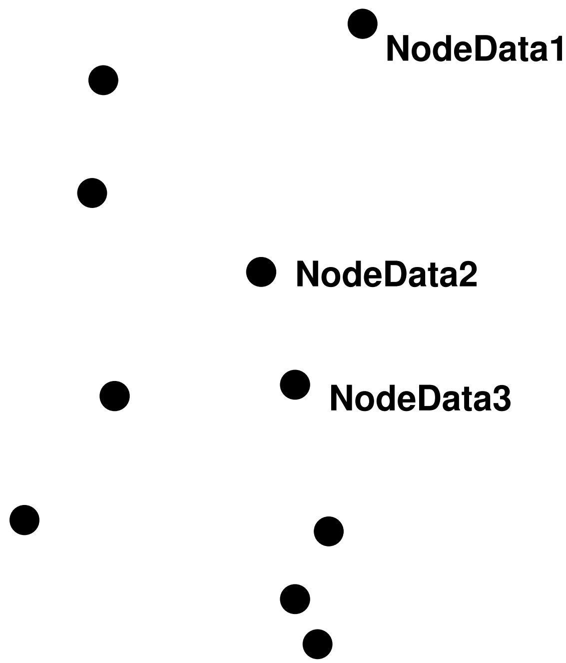


Grid and node data

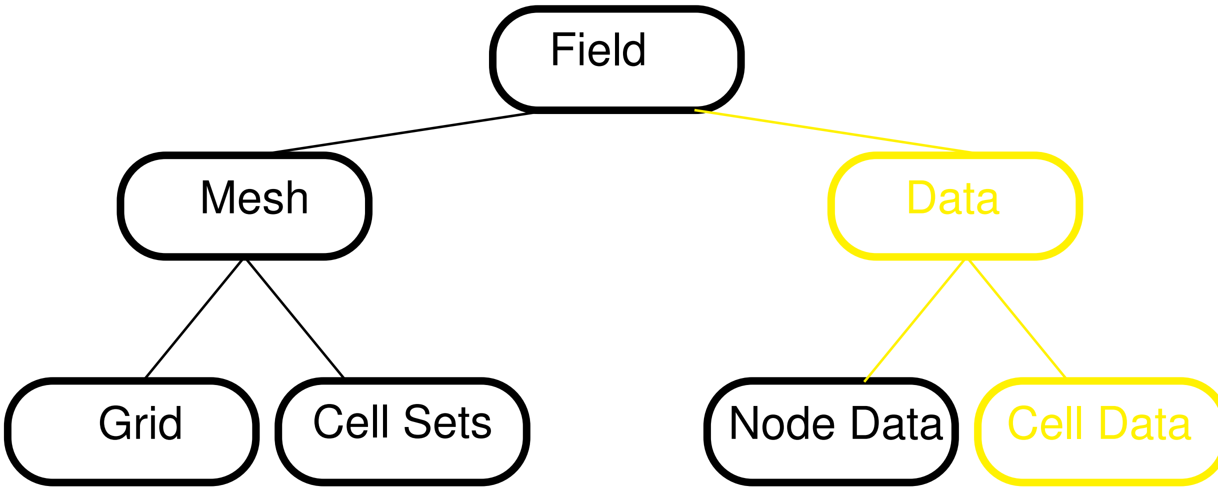


- Positions and data components
- Data components can be scalar and vector

Grid and node data

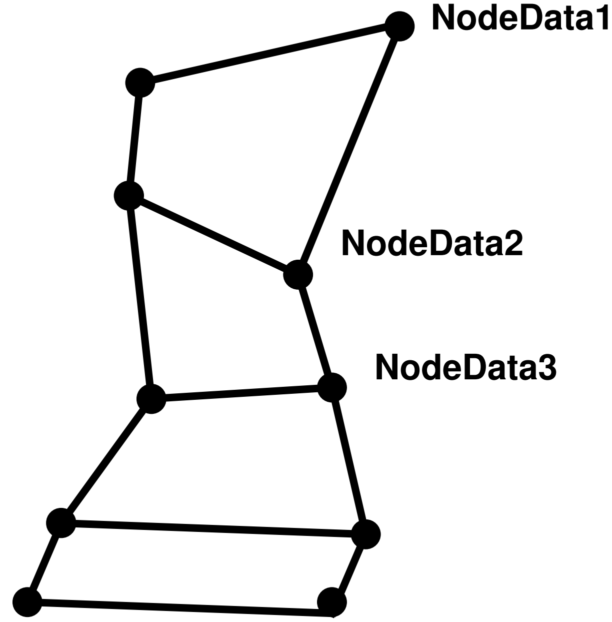


Connectivity information

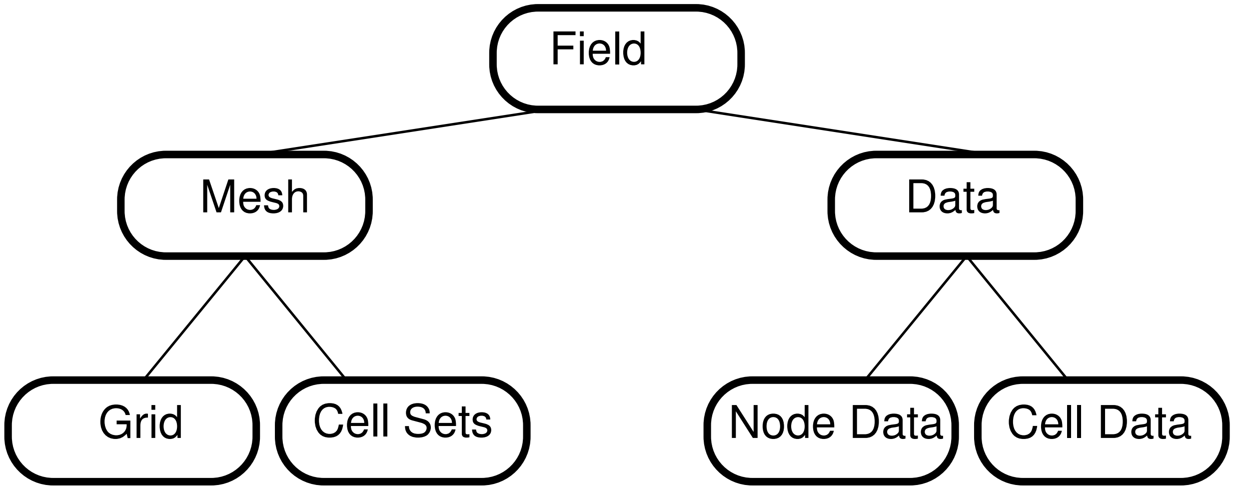


- cell is a geometric entity
- cell_sets are a collection of cells

Connectivity

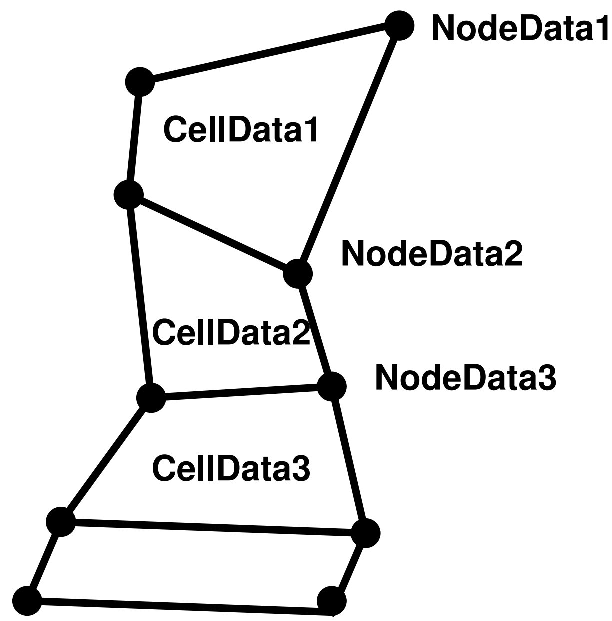


Cell Data



- Cell data completes the definition

Cell_data



NodeData1
CellData1
NodeData2
CellData2
NodeData3
CellData3

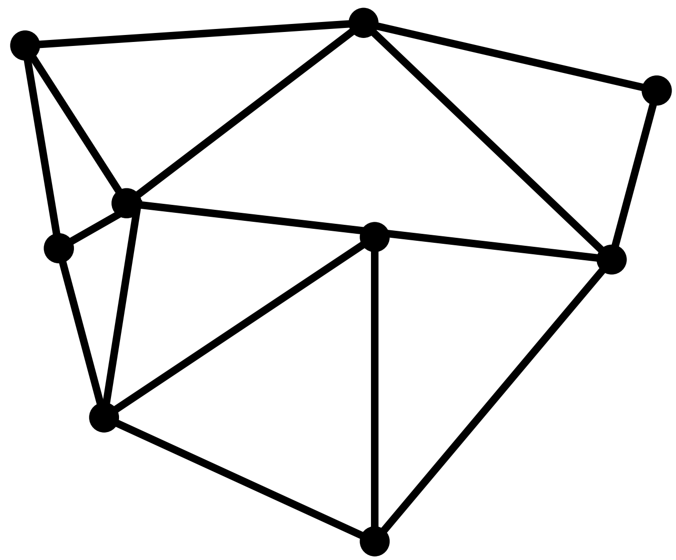
What do you specify

- Some portions of the FLD do not need to be specified explicitly but can be calculated from other information
  - connectivity
  - grid (node) coordinates
- This all depends upon the data you are trying to represent

Mesh types

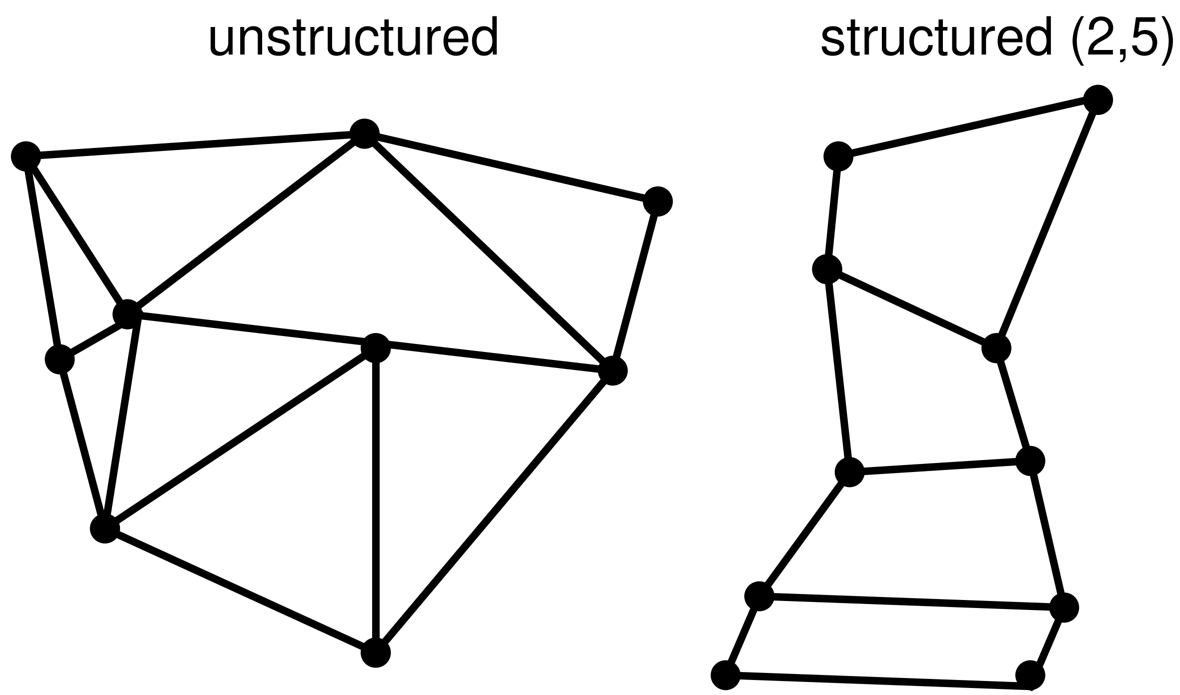
- There are four basic mesh types:
  - unstructured (c.f. UCD)
  - structured (c.f. AVS5 irregular field)
  - rectilinear (c.f. AVS5 rectilinear field)
  - uniform (c.f. AVS5 uniform field)
- Here are some examples...

Unstructured meshes



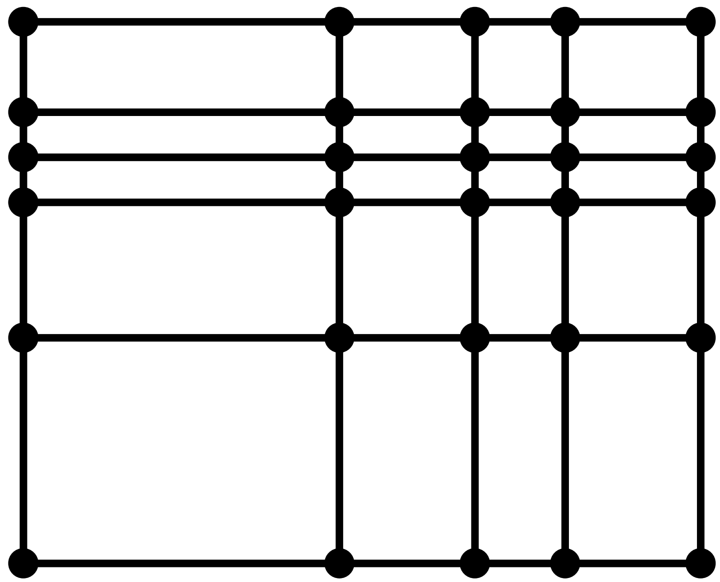
❑ The only type of mesh where cell sets are required

Structured meshes



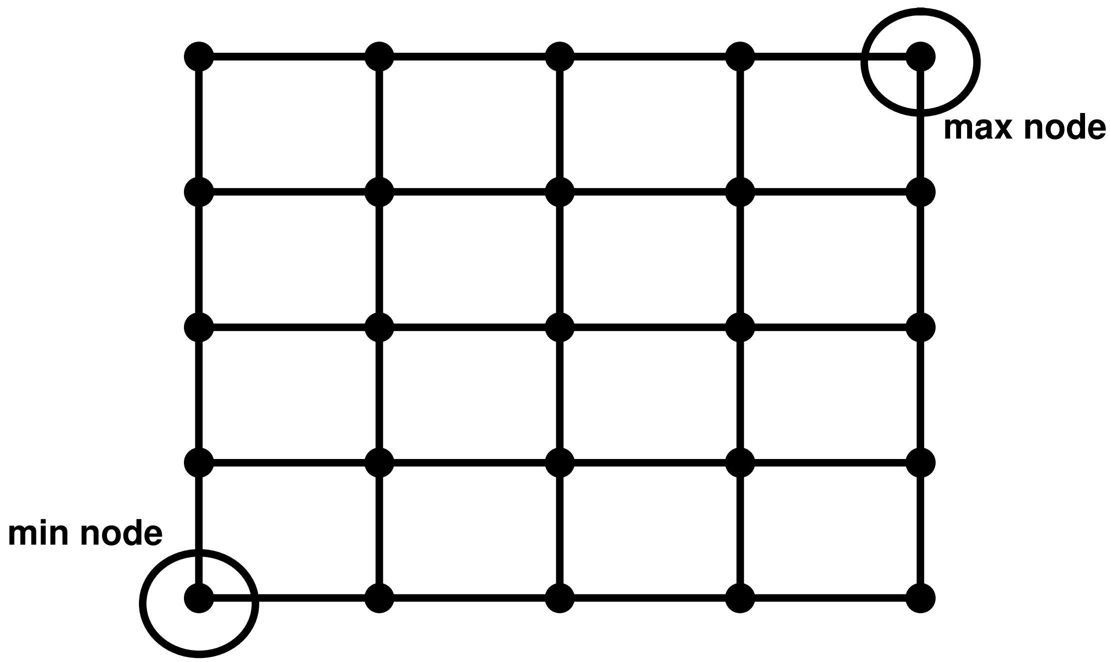
❑ Connectivity is implied by dimensions (all structured meshes have dimensions specified)

Rectilinear meshes



❑ You only specify the grid (node) coordinates along each axes

Uniform mesh



- You just provide grid (node) coordinates for minimum and maximum

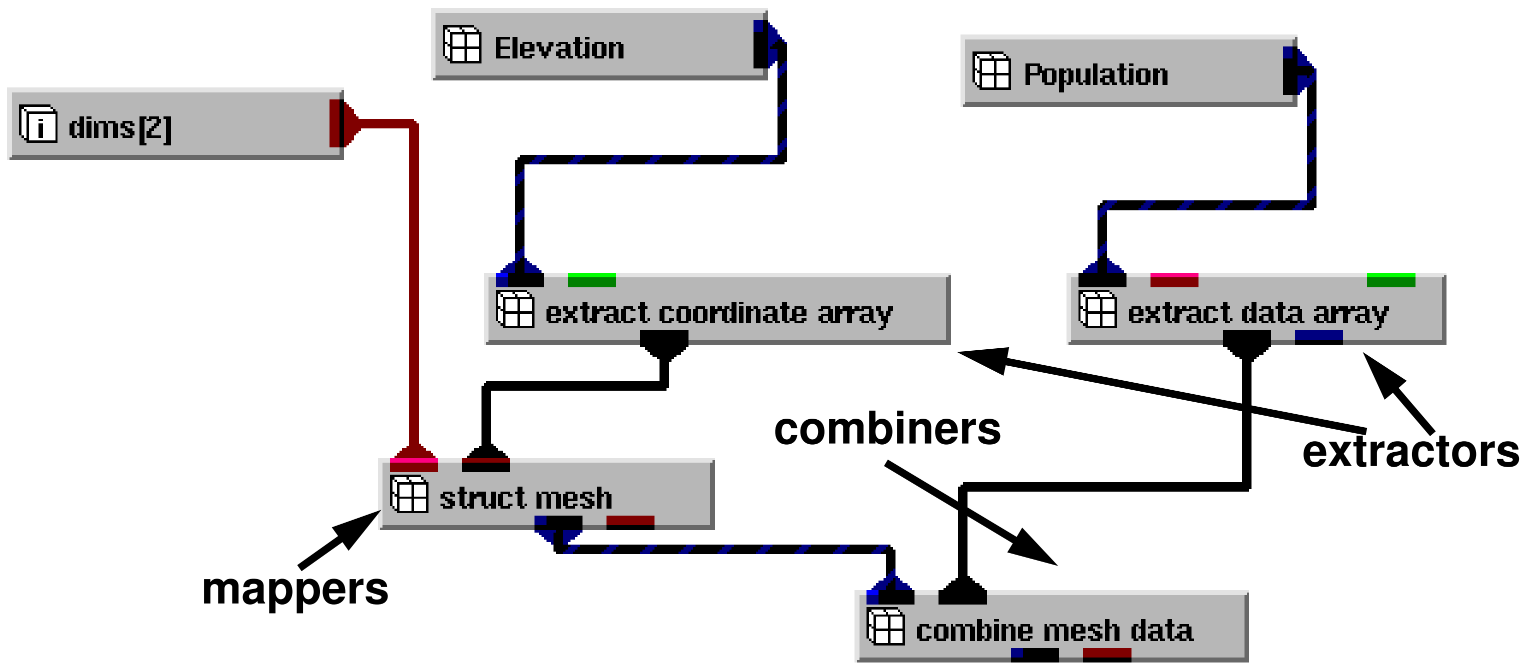
Mapping data into fields

- There are six different methods...
  - Read_Column_File
  - AVS Field file format
  - Database query
  - File access objects
  - Custom reader for arrays of data (module you write code for)
  - Custom reader to generate Express FLD
- Increase in complexity down the list
- Which one do you use?

Which one do you use?

- The good news is that there are specific reasons for each one...
- If you import the data into the system as arrays then you need to make use of:
  - mappers - produce a field from data arrays
  - combiners - merge mesh and data information
  - extractors - select part of Express FLD structure

Mappers, combiners and extractors



Read_Column_File

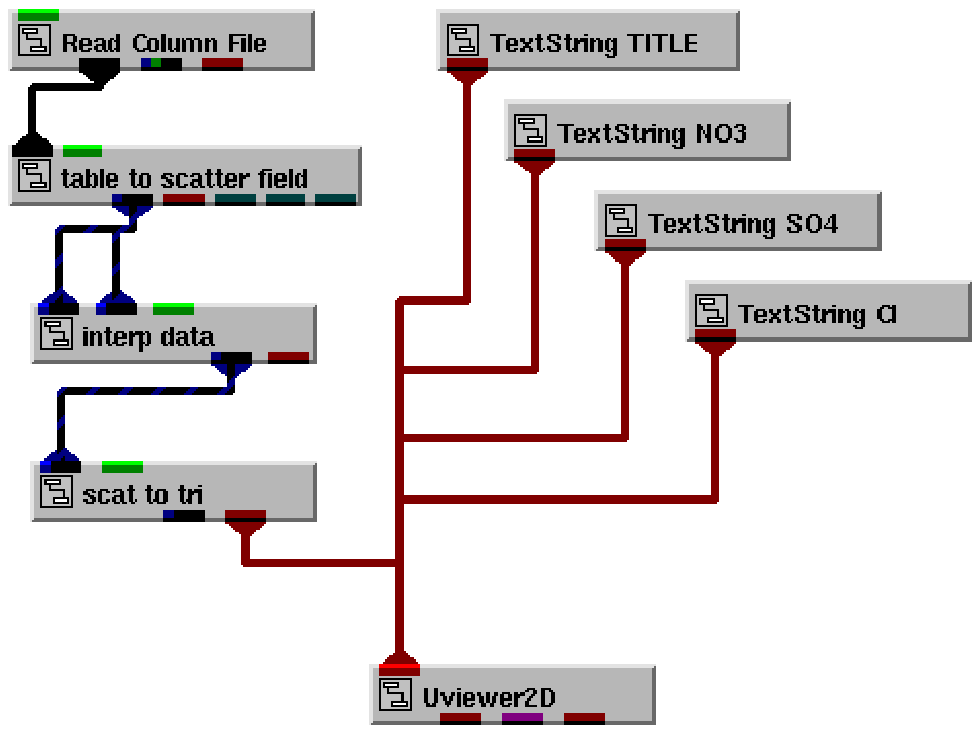
- Data file is ASCII and stored in columns
- You specify:
  - input file
  - columns to read
- To map the data you make use of the modules:
  - table_to_uniform_field
  - table_to_scattered_field

Example data for read cols

- ASCII file, each column is a different data component...

```
52 7.500000 0.866025 0.934890
52 0.488895 0.846791 0.933700
52 0.477785 0.827548 0.932470
52 0.466660 0.808279 0.931200
52 0.455555 0.789044 0.929870
.. ........ ........ ........
```

Example network for read cols



AVS Field file format

- ❑ Data is simple ASCII or binary format that can be described by the header
- ❑ It is used exactly like the AVS5 field and Read Field module
- ❑ N.B., make sure you use the AVS/Express Read Field which produces and Express FLD and not the one in the AC Kit.

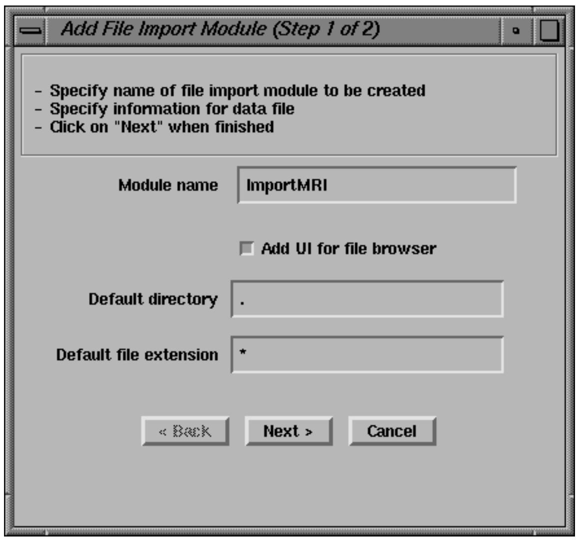
Database query

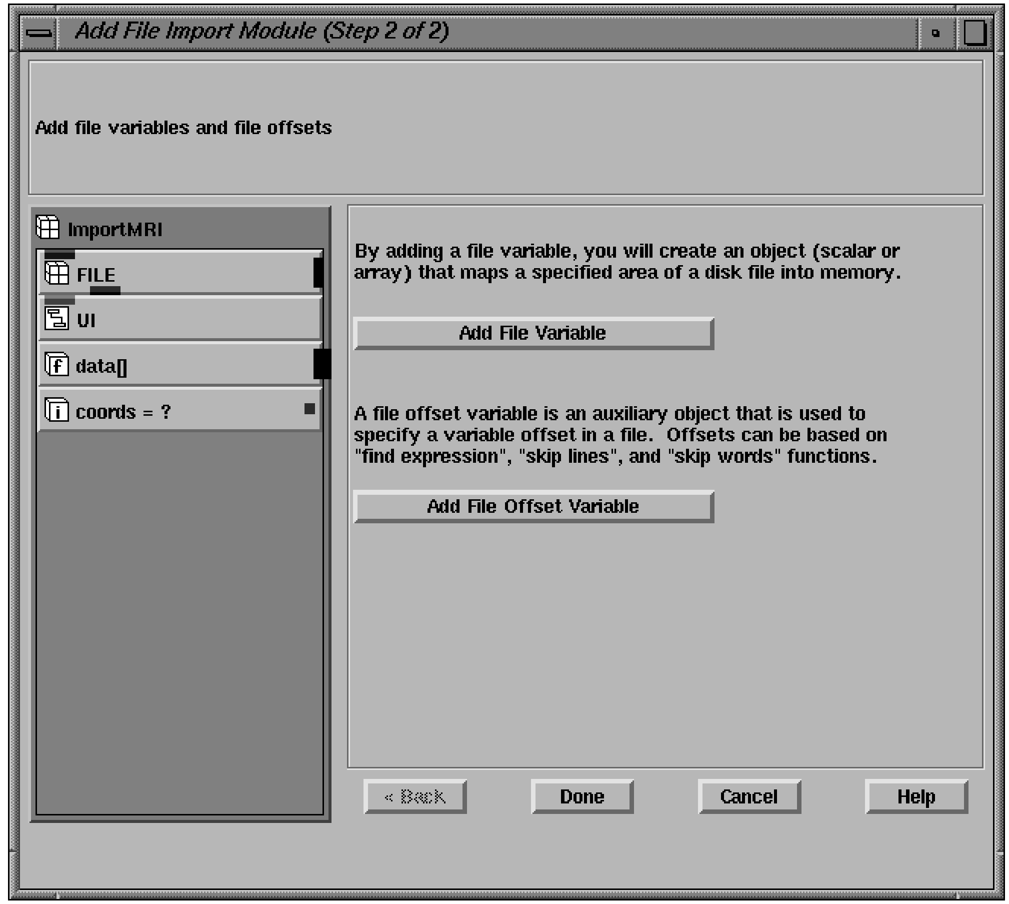
- You need to have the AVS/Express Developer Edition and the Database Kit
- Allows you to submit a query to a RDBMS and obtain data in an Express table
- You use mappers, extractors and combiners as explained before

File access objects

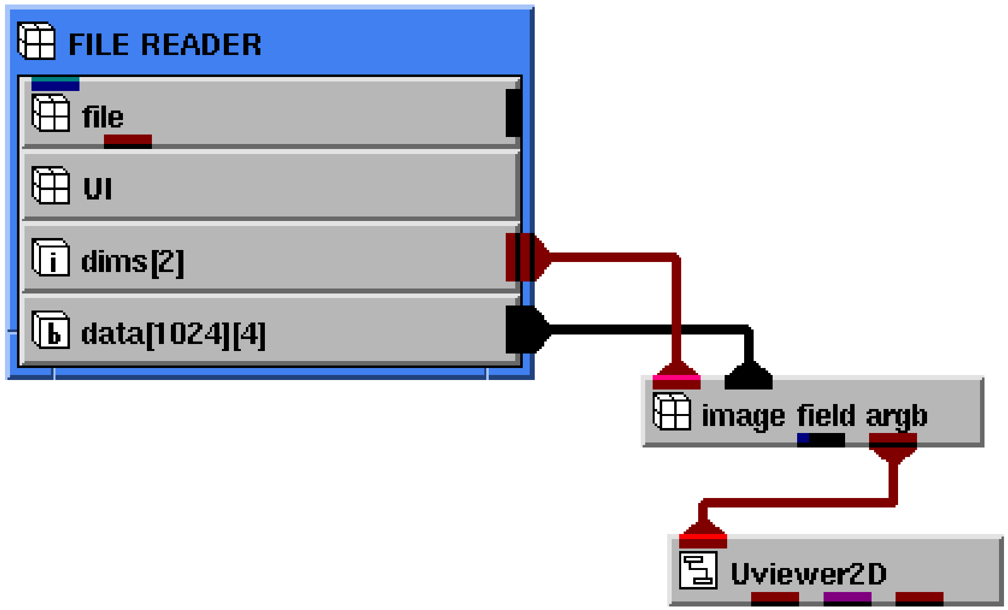
- ❑ Data is in a simple ASCII/binary format
- ❑ A mapper exists to convert the data
- ❑ The object is built using the “Add File Import Module”
- ❑ The file access object is split into two parts:
  - data variables - to be used by field mapper
  - file_obj - reads the array data from the file

Add File Import Module





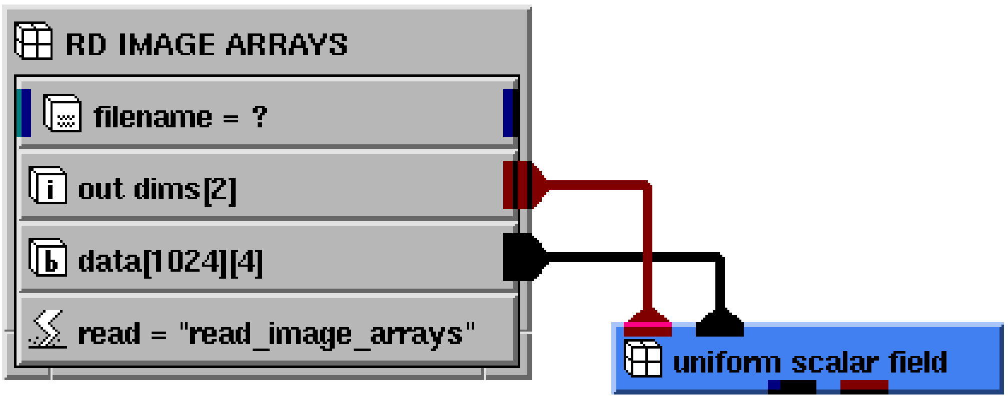
File Access Object



Custom reader for arrays of data

- Data format is too complex for other techniques
- A field mapper exists for the data you are importing
- It will contain three parts:
  - filename input
  - array outputs - needed by field mapper
  - method - to open and read data from file into output arrays

Custom reader for arrays



Custom reader to generate Express FLD

- Data is in a format too complex to read using other techniques
- There are no mappers provided for this type of data
- Your module will:
  - open and read the data from the file
  - use the FLD API routines to create and set values in an Express FLD
- This is the hardest case and is covered more in the integrating application code section

User defined data

- There is no explicit type for this data
- You simply create an object in Express to represent the data
  - e.g., Patient Scan Data

```
group {
int     pid
string  pname
Field   scandata
int     DateofScan
}
```

- Other modules could process Field
- You could create a UDT for holding the date...

Summary

- Read Column File
  - Data is arranged in columns, ASCII
  - Output is a field
  - Useful for simulating database query
- AVS Field File
  - format of file can be described using a simple header, Binary or ASCII
  - Output is a field
- Database query
  - data is stored in relational database, need AVS/Express Database Kit
  - Output is an array (need appropriate mappers)

Summary (cont)

- File Access Objects (Add File Import Module)
  - data which can be easily mapped into arrays, binary or ASCII
  - Output is an array (need appropriate mappers)
  - Can calculate key values from components in datafile eg., dims, offsets
- Custom reader module (write your own code)
  - data must be (or is easier to) parse with code to produce arrays
  - Output is an array (need appropriate mappers)
- Custom reader module (write your own code)
  - code uses FLD access libraries (C) or classes (C++), output is a field
  - use if data is complex format e.g., unstructured field and no field mapper exists

Demonstrations

- Read_Column_File
- File access objects
- Custom reader for arrays of data (module you write code for)
- Custom reader to generate Express FLD
- UDT