MATLAB Structures

MATLAB Structures

- A structure contains one or more named fields.
  - Different from C: MATLAB structures are not data types.
  - The set of fields of a structure can be changed dynamically.
- A structure array is an array whose elements are structures with the same set of fields.
  - The same field of different array elements can contain different data types. ➔ Different from C.
  - Example:

```
a(1).f1='a';  a(1).f2=1:3;  a(1).f3=[];
a(2).f1=100;  a(2).f2={'x','y'};  a(2).f3=true;
```

- Many array operations (subarray, concatenation, etc) are applicable to structure arrays.

MATLAB Structures

- Use function `struct` to create a structure array with multiple fields and/or multiple elements in one statement.

- Syntax:

  ```
  struct(field1, value1, field2, value2, …)
  ```

  - Each non-scalar `value` needs to have the same dimension, which is the dimension of the resulting structure array.

MATLAB Structures

- Structures can be nested, that is, a field of a structure can contain another structure.

- When assigning a new field to an element of a structure array, the same field is generated for all the other elements (with their contents default to empty arrays).

- For operations that involve multiple structure arrays (e.g., concatenation), the multiple structure arrays involved need to have the same set of fields.
  - Example: `A = [B C]`, where `B` and `C` are both structure arrays.
  - The same for assignments like `A(k) = B`, where `A` and `B` are structure arrays and `k` represents the indices.

Managing Structure Fields

- Check existence of fields: `isfield(s, fields)`
- Remove fields: `rmfield(s, fields)`
- Here `fields` can be character array, a cell array of character vectors, or a string array.
- Add fields: Just do assignment
- Get the list of fields: Function `fieldnames`

More on Using Structures

- A convenient way to extract the values of a given field in all the structure array elements:
  - Syntax: `[s.field]`
  - All the values are horizontally concatenated.
  - Most useful when the values are all scalars.
- Using strings / character vectors as field names:
  - Syntax: `s.(string)`
  - This syntax allows the use of a variable inside the parentheses. This is most useful when the field names are to be assigned dynamically, such as generic programs for data processing.

Data Organization in Structures

Using a RGB image as example:

- Plane-by-plane (structure of arrays of the same size):
  - `im.R`, `im.G`, and `im.B` are all **`MxN`** arrays.
  - Example: To access the color at a particular pixel: `im.R(i,j)`.
- Element-by-element (array of structures of the same fields):
  - `im` is a **`MxN`** structure array with three fields `R`, `G`, and `B` for each element.
  - Example: To access the color at a particular pixel: `im(i,j).R`.

Data Organization in Structures

- Element-by-element organization is more like C.
- However, element-by-element organization requires the storage and processing of the array header of each field of each element.
  - Reason: The flexibility that each field of each element can have different types of content.
- As a result, when we have many elements, plane-by-plane organization is much faster and uses much less memory. (The difference is huge.)

Structures as Variable Collections

- A convenient use of structures in MATLAB is as a collection of several related but dissimilar variables.
  - This uses a scalar structure (i.e., a structure array with only one element).
  - Each variable in the collection becomes a field in the structure.
  - This allows easier handling of these variables together, such as passing into and out of functions, saving to or loading from files, etc.
    - The same is used in many C/C++ libraries / APIs.
  - Helps to avoid variable name collisions.

Application Examples

- Example1: Function `dir`:
  - Example use: `F = dir('*.mat');`
  - A structure array where each element corresponds to a file/folder/etc. in the given path.
  - Attributes of a file (name, size, etc.) are returned as fields of an array.
- Example 2: Function `load`:
  - Example use: `A = load('abc.mat');`
  - When there is an output argument, it becomes a structure whose fields are the loaded variables.
  - Helps to avoid name collisions between the loaded variables and variables in the current scope.