Data Types in Matlab

Tyler Ransom
Duke University

July 16, 2012

Data Types in Matlab

- Numeric Classes
- The Logical Class
- Characters and Strings
- Structures
- Cell Arrays
- Function Handles
- Map Containers

Why worry about data types?

- Matlab makes it difficult to combine different data types
- For numeric classes, data types may affect accuracy
  - e.g. rounding error
- It's important to know about data types when working with data in any program, because different data types take up different amounts of disk space
- Don't waste disk space by saving a dummy variable in a format that has precision to 15 decimal points

Numeric Classes

Integers
- can take 8, 16, 32, or 64 bits
- can be signed or unsigned
- if unsigned, user can get twice as long of integers per byte

Double Floating-Point Numbers
- always take up 64 bits
- most precise numerical format (16 digits of accuracy)

Single Floating-Point Numbers
- always take up 32 bits
- only 8 digits of accuracy

Complex Numbers
- can be formatted in any fashion above, but Matlab has an additional flag to denote complexity

`Inf` and `NaN`
- stored as 64-bit double

Working with Numeric Classes

- Viewing which variables are which type
  - At the command line, type `whos`
  - User can see a list of variables and which class they are formatted
- Generating a variable of a certain type:

```
y = int64(-589324077574); x = double(y)
```

How can I optimize my data storage?

- Suppose a user has data she wants to save in a mat-file
- Suppose also that she has a number of variables that are stored in "too big" of numeric classes
  - e.g. she has a dummy variable stored as a 32-bit integer and a 16-bit integer stored as a 64-bit double
- How can she quickly and easily convert her numerical variables to the most efficient numerical class?
- The easiest way is the `save` command
- Matlab's `save` command is streamlined to automatically convert variables to the optimal format
- Beyond this, there are user-written scripts that store data even more efficiently than Matlab's `save` command

Logical Class

- Any variable that is a logical (e.g. "`Z=(x<5);`") is stored as an 8-bit *logical*.

Characters and Strings

- Strings in Matlab are stored as 16-bits per character
- Character arrays must have all elements be the same length, so spaces are automatically inserted to make the dimensions match
- Thus, if you tried you would get an error because the dimensions don't match up
- Instead, use the `char` function:
- Can convert strings to cell arrays (see below)

Structures

Structures are invoked with a period, e.g. (`A.data`). `A` is a structure with substructure `data`.

Cell Arrays

Cell arrays are the only way to store data of two different types in the same array, side-by-side.

- Cell arrays contain multiple data containers called cells
- Each cell can contain any type of data
- When referencing elements, use `()` to reference cells (data containers) and use `{}` to reference individual elements within a cell

Operations with Cell Arrays

- Generate a blank cell array `A` with the command `A=cell(3,4,2);`
- concatenate cell arrays along dimension k with the command `cat(k,array1,array2,...arrayN);`
- delete the (i, j) observation of cell array `myarray` with the command `myarray{i,j}=[];`
- create a nested cell array called super_array with the command `super_array = {array1; array2;};`
  - super_array is now a 2x1 cell array with array1 in the first cell and array2 in the second cell

Operations with Cell Arrays

- In order to turn a string into a cell array, use the command `cellstr`
- In order to turn a numerical array into a cell array, use `num2cell`
- Once this formatting is taken care of, the string and numerical cells can be concatenated together with the `cat` command
- Example: want to list variable names next to their point estimates in a regression output table:

```
1 varnames = cellstr(char('Var1','Var2',...,'VarK'));
2 estimate = num2cell(X\y);
3 result   = cat(2,varnames,estimate);
```

Function Handles

From the Mathworks website:
"A function handle is a callable association to a MATLAB function. It contains an association to that function that enables you to invoke the function regardless of where you call it from. This means that, even if you are outside the normal scope of a function, you can still call it if you use its handle.
With function handles, you can:

- Pass a function to another function
- Capture data values for later use by a function
- Call functions outside of their normal scope
- Save the handle in a MAT-file to be used in a later MATLAB session"

Map Containers

- Map containers are data structures that are organized by a key and a data value
- In regular arrays, the key is a number (corresponding to the element of the matrix)
- The advantage to map containers is that the key can be a string