MATLAB EXPO 2019

Insights into MATLAB
Memory handling and data types

Loren Shure
Agenda

MATLAB and memory implications

- Programming
  - Passing Arrays
  - In place calculations

- Datatypes
  - How structures/cells use memory
  - tables, categorical, string arrays
MATLAB and Memory

- Passing arrays to functions
  - When does MATLAB copy memory?

```
function y = foo(x,a,b)
a(1) = a(1) + 12;
y = a*x+b;
```

- Calling `foo`
  
  `y = foo(1:3,2,4)`

  - i.e., `x = 1:3, a = 2, b = 4`
In-place Optimizations

- When does MATLAB do calculations “in-place”?

\[
x = 2 \times x + 3;
y = 2 \times x + 3;
\]
In-place Optimizations

When does MATLAB do calculations “in-place”?

function showInPlace
    xx = randn(n,1);
    xx = myfunc(xx); % vs. yy = myfunc(xx)
    xx = myfuncInPlace(xx); % vs. yy = myfuncInPlace(xx)

function x = myfuncInPlace(x)
    x = sin(2*x.^2+3*x+4);

function y = myfunc(x)
    y = sin(2*x.^2+3*x+4);

>> edit myfuncInPlace myfunc showInPlace
% separate functions, separate files
Memory Used for Different Array Types

\[
d = [1 \ 2] \quad % \text{Double array}
\]
\[
dcell = \{d\} \quad % \text{Cell array containing}
\]
\[
dstruct.d = d
\]

\[
\text{whos}
\]

>> edit overhead
MATLAB and Memory

How does MATLAB store structures?

```matlab
n = 10000;
s.A = rand(n,n);
s.B = rand(n,n);
sNew = s;

s.A(1,1) = 17;
```

>> edit structmem1
MATLAB and Memory

How does MATLAB store structures?

```matlab
im1.red = redPlane;  % each plane is m x n
im1.green = greenPlane;
im1.blue = bluePlane;
```

versus

```matlab
% each 1x3
im2(1,1).pixel = [red(1) green(1) blue(1)];
im2(2,1).pixel = [red(2) green(2) blue(2)];
...
im2(m,n).pixel = [red(m*n) green(m*n) ... blue(m*n)];
```

>> edit structmem2
**Tables**

- New fundamental data type
- For mixed-type tabular data
  - Holds both data and metadata
- Supports flexible indexing
- Built-in functionality (merge, sort, etc.)
Categorical Arrays

- New fundamental data type
- For discrete non-numeric data
  - Values drawn from a finite set of possible values ("categories")
- More memory efficient than a cell array of strings
- Can be compared using logical operators
  - Similar to numeric arrays
Strings

The better way to work with text

- Manipulate, compare, and store text data efficiently
  
  ```
  >> "image" + (1:3) + ".png"
  1x3 string array
  "image1.png"   "image2.png"   "image3.png"
  ```

- Simplified text manipulation functions
  
  - Example: Check if a string is contained within another string
    
    - Previously:  
      ```
      if ~isempty(strfind(textdata,"Dog"))
      ```
    
    - Now:  
      ```
      if contains(textdata,"Dog")
      ```

- Performance improvement
  
  - Up to 50x faster using `contains` with `string` than `strfind` with `cellstr`
  
  - Up to 2x memory savings using `string` over `cellstr`
Summary of MATLAB and Memory

- How MATLAB passes arrays to functions
  - By value, with “lazy” copy or copy on write
  - In-place optimization code pattern

- Memory use in array storage
  - Atomic types vs. cell arrays and structures
  - Array of structs vs. struct arrays
    - i.e., s(300,300).red vs. s.red(300,300)
  - Categorical arrays, tables, strings
Summary

MATLAB and memory implications

– Programming
  ▪ Passing Arrays – when are copies made
  ▪ In place calculations

– Datatypes
  ▪ How structures/cells use memory
  ▪ tables, categorical, string arrays