L11: Array Operations and Loops

Increasing run-time efficiency

Lucas A. J. Bastien

E7 Spring 2017, University of California at Berkeley

February 10, 2017

Version: release

Announcements

Lab 04 is due on February 17 at 12 pm (noon)

Today:

- Array operations versus loops
- Pre-allocating arrays versus resizing arrays at every iteration

Next week:

- Debugging
- Error handling
- More data structures: `struct` arrays and `cell` arrays
- Binary representation of data

Grades for labs 01 and 02 should be released within the next few days

Array operations versus using loops

Certain tasks can be performed using either:

- Array operations (also known as vectorized operations); or
- Loops

By "using array operations", I mean "relying on Matlab built-in features (other than loops) designed to manipulate entire arrays at once (e.g., element-wise operations, matrix multiplication, `sum`)

In most cases, **using array operations is much more efficient than using loops** to perform the same task (see `L11.m` for a demonstration)

There are situations, however, where using loops is necessary

Some components of a computer

The **processor (CPU)** does the calculations

The **memory (RAM)** is used to store data temporarily. Data stored in memory can be accessed relatively quickly. The data stored in memory is lost at computer shut down

The **hard-drive** is used to store data in the long term. Access to data stored on a hard-drive is typically slower than access to data stored in memory. Data is not lost at computer shut down

Peripherals (*e.g.*, mouse, keyboard, monitor) are used to interact with users

Pre-allocating and resizing arrays

Memory allocation
When you create a new variable (or when you resize an array to add more values) in Matlab, your operating system gives ("allocates") Matlab a chunk of memory that is not currently being used by other programs, so that Matlab can store these new values. The bigger the array, the more memory you need

By "**pre-allocating an array**" I mean "creating an array of the desired size (usually full of zeros or ones) before calculating all its values"

In most cases, **it is much more efficient to pre-allocate an array and fill in its values later on, rather than resizing the array every time a new value is added to the array** (See `L11.m` for a demonstration)

Practice question

Assuming that we start with an empty workspace, what will the value of the variable "a" be after executing the following code?

```
>> a = 0;
>> for i = [1, 2, 10]
>>     a = a + i;
>> end
>> a = i;
```

(A) Matlab will throw an error (Undefined variable "i")

(B) 3

(C) 13

(D) 0

(E) 10

Practice question

What will the values of the variables "i" and "j" be after executing the following code?

```
>> for i = 1:26
>>     for j = 1:26
>>         if i*j == 130
>>             break
>>             break
>>         end
>>     end
>> end
```

(A) i is 13 and j is 10

(B) i is 10 and j is 13

(C) i is 26 and j is 5

(D) i is 5 and j is 26

(E) i is 26 and j is 26

Practice question

What will the value of the variable "a" be after executing the following code?

```
>> a = 3;
>> f = @(x) a*x + 3;
>> a = 0;
>> for i = 1:3
>>     a = a +f(i);
>> end
```

(A) Matlab throws an error

(B) 42

(C) 23

(D) 27

Practice question

What will the value of the variable "`fruit`" be after executing the following code?

```
>> fruit = '?';
>> color = 'green';
>> if strcmp(color, 'green')
>>     fruit = 'apple';
>> end
>> if strcmp(color, 'orange')
>>     fruit = 'orange';
>> end
>> if numel(fruit) == 5
>>     fruit = 'lemon';
>> end
```

(A) 'apple'

(B) 'orange'

(C) 'lemon'

(D) '?'