ENGG1003 - Friday Week 9

Scripts
Matrix Indexing

Brenton Schulz

University of Newcastle

May 10, 2019

Scripts

- ▶ It is tempting to use MATLAB from the command line only
    - ▶ It sounds easier, right?
    - ▶ Very low barrier to entry
    - ▶ Very fast to get results

Scripts

- It is tempting to use MATLAB from the command line only
  - It sounds easier, right?
  - Very low barrier to entry
  - Very fast to get results
  - Useless for non-trivial problems

Scripts

- ▶ It is tempting to use MATLAB from the command line only
  - ▶ It sounds easier, right?
  - ▶ Very low barrier to entry
  - ▶ Very fast to get results
  - ▶ Useless for non-trivial problems
- ▶ Scripts are used for multiple reasons:
  - ▶ They are necessary for realistic problems
  - ▶ They can be modified and re-executed
  - ▶ They can be reused by other people

Comments

▶ MATLAB comments start with a % symbol and end at a new line

▶ Comment guidelines:
  ▶ Describe the script's purpose, inputs, and outputs at the top
  ▶ Comment any lines which aren't "obvious"
    ▶ Yes, this depends on the audience

Scripts And Scalar Arithmetic Example

▶ Example: (From last year's slides) Write a
  MATLAB script which calculates the rate at
  which the Sun loses mass due to nuclear fusion

▶ Data required:

  ▶ $E = mc^2$
  ▶ Sun's energy output: $E = 385 \times 10^{24}$ J/s
  ▶ Speed of light: $c = 3.0 \times 10^8$ m/s

1D Array Indexing

- ▶ Element indexing follows this general rule:
  - ▶ `name(list of elements)`
- ▶ The list is, itself, a 1D array
  - ▶ It can be a single number
    - ▶ eg: `a(2)`
  - ▶ You can create it using `[ ]` concatenation syntax
    - ▶ This can be hard-coded numbers or other variables
    - ▶ eg: `a([1 4 8])`
  - ▶ It can be a list of **integers** created with `A:B:C`
    - ▶ eg 1: `a(1:10)`
    - ▶ eg 2: `a(1:2:10) % Every 2nd element`
- ▶ Things can get complicated *fast*

Multi-Dimensional Indexing

- ▶ MATLAB dimensions are named:
  - ▶ Row
  - ▶ Column
  - ▶ Page
- ▶ The indexing syntax is:
  - ▶ `name(row, column, page)`
- ▶ A good visualisation is in the MATLAB documentation: `https://au.mathworks.com/help/matlab/math/multidimensional-arrays.html`

Dimensional Indexing Notes

- ▶ 1D arrays can be row or column vectors
  - ▶ The indexing is still always in the form `a(n)`
  - ▶ Indexing does not make a distinction between row and column vectors
  - ▶ Arithmetic *does*

Dimensional Indexing Notes

- ▶ 1D arrays can be row or column vectors
    - ▶ The indexing is still always in the form `a(n)`
    - ▶ Indexing does not make a distinction between row and column vectors
    - ▶ Arithmetic *does*
- ▶ There are special syntaxes we can use when indexing:
    - ▶ Index all elements with `a(:)`
        - ▶ Useful with multi-dimensional arrays
        - ▶ eg: `a(:, [2 3])`
    - ▶ When lengths are unknown you can use the `end` keyword
        - ▶ eg: `a(2:end)`

Example - Image Analysis and Editing

▶ Perform the greyscale assessed lab task in MATLAB with a real image
▶ Shrink the image by a factor of 1/10th along each axis while developing code
▶ Knowledge:
  ▶ Images are read with `imread()`
  ▶ Colour images stored as a 3D array
    ▶ Indexing: `var(row,column,[r g b])`
    ▶ `var(0,0,:)` is the top left pixel
  ▶ Image data can be displayed with `image()`
  ▶ 2D data will be displayed with a false colour map
    ▶ Greyscale display needs custom map
▶ Do it live...

More Examples

- ▶ Simple bightness adjustment
    - ▶ Couple of methods:
        - ▶ Add or subtract a constant value to each RGB value in each pixel
        - ▶ Apply a *transfer function*. This needs a sketch...
- ▶ Contrast adjustment
    - ▶ This applies a particular transfer function, will sketch
- ▶ All of the above can be applied to all channels equally or differently to the RGB channels