Math 396L
Math for 3D Graphics Lab

Matlab Functions and Scripts

September 11, 2008

Introduction to Functions

Functions let us:

- break up a big program into small chunks:

  easier to write because we are focused

- help make a large program clearer:

  write some code once and reuse it in the program many times

Functions in MATLAB

- in math, a function $f(x)$ associates a result to each value of x
- in programming, values of x are called *inputs* or *arguments*, the results are the *output*
- MATLAB has built-in functions like $\sin(x)$, $\cos(x)$, $\sqrt{x}$, etc.
- you can create function files that can be used just like the built-in functions

Structure of a Function File

- first line: function definition

  ```
  function [output arguments] = function_name(input arguments)
  ```

- function name: same rules as variables

- input arguments:

  list of variables used in the function that provide input when the function is called

- output arguments:

  list of variables used in the function and that transfer output from the function

Structure of a Function (cont'd)

- H1 and Help text lines
  - H1 is the first comment line after the function definition
  - used by `lookfor` command for searching
  - help text lines include the remaining comment lines until the first non-comment line
  - used by `help` command

Structure of a Function (cont'd)

- local variables
  - all variables in function files are *local*
  - they can have the same name as variables used in the command window or script files but they are distinct and do not share values
  - when the function file finishes its execution, the values of local variables are lost.
- global variables
  - variables that are shared with the Command Window, script files, and other function files

```
global variable1, variable2,  variablen
```

Saving Function Files

- save before you try to use it
- give it a name that is the same as the function it defines

```
function [mpay, tpay] = loan(amount, rate, years)
```

⇒ `loan.m`

```
function trajectory(v, h, g)
```

⇒ `trajectory.m`

Using your Function

- use from command window, script file, or from another function
- assigning output to a variable

```
average_grade = CalcHWGrade(1)
```

- using in an expression

```
total_weight = weight_ring(d) + weight_base(l,w,h)
```

- type in command window

```
>> PlotMyData(x)
```

Example 1 – Evaluating an Expression

$$f(x) = \frac{x^4\sqrt{3x+5}}{(x^2+1)^2}$$

- write a function file to evaluate the above function allowing for x to be a vector
- calculate

$$f(x) \quad \text{for} \quad x = 6$$

$$f(x) \quad \text{for} \quad x = 1, 3, 5, 7, 9, 11$$

Script and Function FIles

- both have the .m suffix in their names.
- script files contain a list of commands to be executed by MATLAB as if they were being typed in the command window
- first line of function file is always the function definition
- variables in a function file are local
- variables in a script file are shared with command window and other files
- function files can accept data through input arguments and return data through output arguments
- the function file name should be the same as the function it defines

Anonymous Functions

- good for short (one-line) calculations used frequently in a longer program
- example: converting Fahrenheit to Celsius
- general form

```
function_name = @ (arguments) expression
```

```
FtoC = @ (F) 5*(F-32) ./9
```

```
cube = @ (x) x^3
```

```
circle = @ (x,y) 16*x^2+9*y^2
```

- use functions in the expression as well as vectors or matrices
- warning: predefined variable values are captured when the anonymous function is defined.

Function Functions – Example

A MATLAB function that takes a function as input

MATLAB built-in function `fzero` can find the zeros (*i.e.*, x s.t. $f(x) = 0$) of a function $f(x)$. How do we describe $f(x)$ to `fzero`?

- function functions like `fzero` accept functions as arguments in two ways:
  - function handle
  - using the name of the function in a string

Function Handles

- a function handle is a unique value associated with any function (user-defined, built-in, anonymous)
- the function handle can be obtained with the syntax `@function_name`, for example
  - `@cos`
  - `@FtoC` – as function file
  - `FtoC` – as anonymous function
- function functions must use input arguments consistently with the input function

Function Names in Strings

Older / less efficient method

- pass name of function as a string `'cos'`, `'FtoC'`
- evaluate

  `var= feval('functionname', arguments)`

Subfunctions

- a function file can contain more than one user-defined function
- first function defined, the "primary," is how the function is known to the rest of the program
- other functions, "subfunctions," are only known inside the function file and each has its own workspace (local variables)

Nested Functions

- subfunctions have separate workspaces (variables)
- by nesting function definitions, variables can be shared

```
function y = A(a1, a2)
...
    function z = B(b1, b2)
    ...
    end

    function w = C(c1, c2)
    ...
    end
end
```

SImple Example

Example of a simple MATLAB function that takes a vector `x` as input and evaluates and plots

$$f(x) = \frac{x^3 \sqrt{x^2 + 1}}{x^2 - 2}$$

over the interval defined by `x`

```
function y = f1(x)

y = (x .^ 3) .* ((x .^ 2 + 1) .^ 0.5) ./ (x .^ 2 - 2);

plot(x,y);
```