Lecture 14: Function handles

Announcements

• Last lecture!
• A3 due tonight (Mon, Oct 18)
• Textbook challenge activities due tomorrow
• Office hours through this week; consulting remains available

Agenda

• Referring to functions in variables
• Passing functions to other functions
• Anonymous functions
• Parameterized functions
Variables

• Conceptually, a box that stores a value
  • Array variables: a big box broken up into smaller ones

• “Variable” – something that can change
  • Can assign different values
  • Can pass different arguments to functions (input parameters)

• What if we want to change a computation?
Example: accumulation pattern

Sum

\[
\text{function } \text{acc} = \text{sumof}(v) \\
\text{acc} = v(1); \\
\text{for } k = 2: \text{length}(v) \\
\quad \text{acc} = \text{sum}([\text{acc} \ v(k)]); \\
\text{end}
\]

Maximum

\[
\text{function } \text{acc} = \text{maxof}(v) \\
\text{acc} = v(1); \\
\text{for } k = 2: \text{length}(v) \\
\quad \text{acc} = \text{max}([\text{acc} \ v(k)]); \\
\text{end}
\]
Examples: mathematics

• Where does a function cross zero? (rootfinding)
• What is the area under a function’s curve? (integration)
• Where is a function the smallest? (optimization)
Examples: event handling

• Graphical user interface
  • When the user clicks this button, execute *this function*
Function handles

• Allows a variable to refer to a function
• Syntax: `@function_name`
• Examples:
  • `h = @sum;`
  • `s = h([1 1]);  % s = 2`
  • `h = @max;`
  • `m = h([3 1]);  % m = 3`

• `function_name` can be a:
  • built-in function
  • user-defined function
  • local function (in the same file)
Function functions

- Can write functions that take other functions as arguments
  - Input parameter will be a function handle

```matlab
function acc = accof(v,f)
    acc = v(1);
    for k = 2:length(v)
        acc = f([acc v(k)]);
    end
```
Demo: mathematics

- `fzero(func,xguess)`
  - Find root near xguess
- `integral(func,xmin,xmax)`
  - Definite integral from xmin to xmax
- `fminbnd(func,xmin,xmax)`
  - Minimize between xmin and xmax
Anonymous functions

• Creating a new .m file just to use as an argument to function- functions feels excessive

• Using local functions is more convenient, but still need to pick a name

• For simple functions, can define anonymously in the expression in which they’re used

• Syntax: @(params) expr
  • Function body must be a 1-line expression evaluating to the output value

• Example: @(x) sin(x) – x
  • Declares a handle to a function of one argument, x, that returns the value of sin(x)-x
Parameterized functions

• Function handles must take exactly the number of arguments that a function-function expects to provide
  • But user-defined functions often take additional arguments for flexibility
  • For a given operation, want to hold some arguments constant

• Use anonymous functions to “bind” values for other input parameters

• Example:
  @(x) quadratic(x,2,0,-18)
  • Binds parameter values 2, 0, -18 to the 2nd-4th arguments of a named function quadratic()
Example: solving differential equations

• $[\text{ts}, \text{ys}] = \text{ode45}(\text{rhs}, \text{tspan}, \text{y0})$
  • $\text{dydt} = \text{rhs}(t, y)$
  • $\text{tspan} = [t0 \text{ tf}]$
  • $y, y0, \text{dydt}$: column vectors
  • $[\text{length(ts)}, \text{length(y)}] = \text{size(ys)}$
Where to go from here?

- **mathworks.com** – Many free tutorials on specific topics
- *Getting Started with MATLAB* by Rudra Pratap – Good for independent learning with science/engineering applications
- Read function documentation – lots of informative examples
- Just play! (take advantage while it’s free) Check out MATLAB Community forums, “File Exchange”