Variables, Basic Programming

Leigh J. Little SUNY Brockport September 15, 2021

# Contents



# 1 MATLAB Variables

To this point we have used generic names for our variables. This is acceptable in some cases, but generally it is a much better idea to choose variable names that are reflective of the quantity the variable represents. For example, if you are computing an acceleration, then variable names like `a`, `acc`, `accel`, *etc.* make more sense than a name like `orange`. MATLAB variables must conform to the rules below:

- Variables must start with a letter.
- Variables can contain letters, numbers and the underscore character.
- Variables can be up to 63 characters long.
- Variables should not have the same name as an existing MATLAB function. For example, you can't use `sin` as a variable name because this is also the name of the built-in sine function. If you want to check if a potential variable name is already being used in some context, you can use the `exist` command. Type in the name of the variable you want to use, for example,

```
>> exist sin
ans =
    5
>> exist evilsteve
ans =
    0
```

  If the output from this command is anything but 0, you can't use that variable name.
- Variables can't have the same name as an existing script file.

# 2 The `input` Command

Sometimes it is handy to be able to get input from the keyboard while a program or script is running. This can be done using the `input` command. For example,

```
>> b = input('Input a value for b: ');
input a value for b: 27
>> b
b =
    27
```

In this example, the system will stop and wait for the user to input a value. This will be stored in the variable `b`. You should always include a meaningful prompt indicating what the user is supposed to input.

As another example, consider the example

```
>> c = input('Input a value for c: ');
input a value for c: pi/6
>> c
c =
    0.5236
```

This example indicates that inputs are not limited to scalars. You can input vectors, matrices or an algebraic calculation. This is a good feature to have. In many programming languages, keyboard inputs are limited to scalar values. In Fortran/C/C++, if you wanted to input the value of $\frac{\pi}{6}$ you would have to manually compute the value somewhere else, then type in 0.523598775598299.

# 3 Basic Programming

The first few programs we will write are going to resemble simple calculator type exercises. For example, type the following into a script file named `carea.m` (you can leave out the comments)

```
% Program to compute the area of a circle given the radius

% Start clean
clear

% Get the radius from the user and compute the area
%
% Be sure to use the built-in value   pi   instead
% of 3.14
r = input('input the radius: ');
area = pi*r^2;

% Print out the result
disp('The area is ')
area
```

Then run the script

```
>> carea
input the radius: 1
The area is
area =
       3.1416
```

If you ever want to see all the digits in the result, you can use the format command

```
>> format long
>> area
area =
       3.141592653589793
>> format longe
>> area
area =
       3.141592653589793e+00
```

The first format command is open format with full double precision and the second is double precision in scientific notation.

## 3.1 Order of Operations

It as assumed that you are familiar with the usual order of operations when translating a mathematical formula to a one-line version.

- Operations in parentheses are performed first. Start with the inner most set in the case of nested parentheses.
- Exponents (left to right) are computed second.
- Multiplications and divisions (left to right) are computed third.
- Additions and subtractions (left to right) are computed last.

For example, if you needed to compute

$$y = \frac{x}{x^2 - 1},$$

you would translate this as

```
y = x/(x^2 - 1)
```

## 3.2 Breaking up Complex Calculations

One way to improve the reliability of transcribing a mathematical formula into a one-line version is to break up the calculation into smaller pieces. For example, if you needed to compute

$$z = \frac{a + b}{\frac{1}{c+d} + \frac{\sin a}{b}},$$

You could break this down as:

```
>> (assume a, b, c and d have been given values)
>> t = 1/(c+d);
>> s = sin(a)/b;
>> z = (a+b)/(t+s);
```