

MATLAB Marina: Applying Arrays

Student Learning Objectives

After completing this module, one should:

1. Be able to solve problems involving arrays of data.

Terms

NA

MATLAB Functions, Keywords, and Operators

NA

Applying Arrays

Arrays and element by element operations are useful for many engineering applications such as: evaluating a formula for a range of values, evaluating piece-wise formulas, analyzing a set or a portion of a set of data, and computing a series summation.

Evaluating a Formula

To evaluate a formula for a range of values:

- Determine the initial and final independent variable values and determine the interval between values (or the number of values).
- Create an array of the independent variable values for the range. This is typically done using the colon operator or the `linspace` function.
- Evaluate the formula for the array of the independent variable values. Use element by element operations where appropriate.

The MATLAB program of Figure 1 evaluates and plots the formula $s(t) = s_0 + v_0t + 0.5a_0t^2$ over the range $0.0 \leq t \leq 6.0$ s.

```
clear; clc; close all;
s0 = 0.0;
v0 = 10.0;
a0 = -1.5;
% create the independent variable values
t = 0.0 : 0.05 : 6.0;
% evaluate the formula for the array of values
s = s0 + v0*t + 0.5*a0*t.^2;
figure(1)
plot (t, s)
```

Figure 1. MATLAB Program to Evaluate a Formula

Notice that when referring to the variable for the function, that `s` rather than `s(t)` is used. Parentheses cannot be used in a variable name. The syntax `s(t)` corresponds to either indexing the variable `s` or invoking a function named `s` for the argument `t`. If the statement

$$s(t) = s_0 + v_0*t + 0.5*a_0*t.^2;$$

were used, the program would have an error as MATLAB would evaluate the formula and then try to place the result in the variable *s* in the range of indices in the variable *t*. Array indices must be integers, thus this will result in an error.

Indexing the array that will hold the result of the operation is not necessary when a variable is being used for the first time or is being redefined. MATLAB will allocate enough space for the result of the operation and place the elements of the result in the variable in the same order as the evaluation results in.

Evaluating a Piece-wise Defined Formula

To evaluate a piece-wise formula, the process is similar to evaluating multiple formulae for different ranges of values.

- Determine the initial and final independent variable values and determine the interval between values for each of the pieces.
- Create arrays for the independent variable values in each range.
- Evaluate each piece of the formula for the appropriate piece of independent variable values.
- Concatenate the independent variable value pieces into one array and concatenate the formula value pieces into one array.

The MATLAB program of Figure 2 evaluates the piecewise formula

$$f(t) = \begin{cases} 5te^{-0.5t} & 0 \leq t \leq 1.5 \\ 4.116e^{-1.5(t-1.5)} & 1.5 < t \leq 2.5 \end{cases}$$

```
clear; clc; close all;
% create the two pieces for the independent variable
t1 = 0.0 : 0.05 : 1.5;
t2 = 1.55 : 0.05 : 2.5;
% evaluate each piece of the formula
f1 = 5.0*t1.*exp(-0.4*t1);
f2 = f1(end)*exp(-1.5*(t2 - 1.5));
% concatenate the pieces
t = [t1, t2];
f = [f1, f2];
```

Figure 2. MATLAB Program to Evaluate a Piece-wise Formula

Analyzing a Set of Data

Array comparisons can be used to analyze a set of data, for example determining which data meet a set of criteria. The MATLAB program of Figure 3 determines which steels have a yield strength greater than 420 MPa along with the strengths of the steels that met the criteria.

```

clear; clc;
% steel tensile strength (yield strength) in MPa
strengths = [374.7, 334.2, 416.5, 335.8, 329.8, 521.2, ...
            415.8, 368.5, 573.1, 411.3, 474.4, 559.8, 397.1, ...
            576.7, 383.2, 401.4, 415.7, 347.8, 332.7, 410.6, ...
            548.4, 581.9, 417.1, 400.0];
% determine which steels have strengths greater than 420 MPa
str_ind = find(strengths > 420);
which = str_ind;
% steel strengths greater than 420 MPa
strengthsGT = strengths(str_ind);

```

Figure 3. MATLAB Program to Determine Steels with Strength Greater than 420 MPa

The data for an operation like this would typically come from a file and not be entered directly. A comparison, `strengths > 420`, is used to determine which steels meet the criteria. Since a comparison returns an array of logicals, the `find` function is used to convert this to the corresponding indices of the array of steel strengths. The indices are then used to identify which steels met the criteria and to extract the strengths greater than 420 MPa via an indexing operation.

Evaluating a Formula and Extracting a Subrange

The MATLAB program of Figure 4 evaluates a formula and plots the formula over the full range and a subrange of interest.

```

clear; clc; close all;
s0 = 0.0;
v0 = 10.0;
a0 = -1.5;
% create the independent variable values
t = 0.0 : 0.05 : 10.0;
% evaluate the formula for the array of values
s = s0 + v0*t + 0.5*a0*t.^2;
figure(1)
plot (t, s)
% determine where s is a max
[maxS, ind] = max(s);
% extract a 2 second window of values around the max
delt = 1.0;
tL = t(ind) - delt; % lower t value for window
tU = t(ind) + delt; % upper t value for window
% extract the two second window
t2 = t(t>= tL & t <= tU);
s2 = s(t>= tL & t <= tU);
figure(2)
plot (t2, s2), axis tight;

```

Figure 4. Extracting a Subrange of a Formula

The program is based off the program of Figure 1 that evaluates a formula. Here the full range is $0.0 \leq t \leq 10.0$ s. The maximum value of the formula and location of the maximum value are then determined. The index corresponding to the maximum value is used to determine the corresponding time where the maximum value occurs and from this the times for the lower and upper bound of the window can be calculated. A comparison to determine the range of t values corresponding to the window is performed and the resulting array of logicals from the comparison is used to index the t and s values for the window. The array of logicals from the comparison is not saved as it is not needed elsewhere in the program.

Series Summation

The MATLAB program of Figure 5 approximates the geometric series $S = \sum_{k=1}^{\infty} \left(\frac{1}{2}\right)^k$ for N terms,

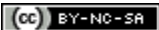
$S \approx \sum_{k=1}^N \left(\frac{1}{2}\right)^k$. This geometric series converges to $S = 1$.

```
clear; clc; close all;
% number of terms in approximation
N = 20;
% array of term numbers
k = 1:1:N;
% evaluate the geometric series formula for each term
terms = (1/2).^k;
% sum the terms to determine approximation
Sapp = sum(terms);
```

Figure 5. MATLAB Program to Approximate a Geometric Series

The number of terms is initialized and an array of all the term numbers from $k = 1$ to $k = N$ is created. The series formula, $\left(\frac{1}{2}\right)^k$, is evaluated for each of the term numbers generating an array of terms in the series. Summing all the series terms yields the series approximation.

Last modified Friday, September 18, 2020

 [MATLAB Marina](https://creativecommons.org/licenses/by-nc-sa/4.0/) is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.