

MATH 3341 — Spring 2021

Lab 02: Variables, Arrays and Scripts

If you haven't downloaded and unzipped `Math.3341.zip`. Download and unzip it under H: (H Drive if you are working on the Remote Lab). Change the current working directory by typing `cd H:\Math.3341\Math.3341.Lab.02` in the Command Window, and type `edit lab_02_script` in the Command Window to edit `lab_02_script.m`.

1 1-D ARRAY: VECTOR

- (a) Define two evenly spaced row vectors: `vec1` of which elements start from 1 to 9 with 9 entries and `vec2` that is between 18 and 1 with step size -2 using `linspace` and `colon`, respectively. Then store lengths of `vec1` and `vec2` to `vec1Length` and `vec2Length`.
- (b) Calculate the following products:
- Store the product of all elements of `vec1` to `vec1Product`.
 - Calculate the elementwise product of `vec1` and `vec2`, then assign the result to `vecProduct`.
 - Calculate the dot product of `vec1` and `vec2` in three ways using `dot`, matrix multiplication, and `sum` with elementwise multiplication. Store the results to `dotProduct1`, `dotProduct2`, and `dotProduct3`.

2 2-D ARRAY: MATRIX

- (a) Verify that column sums, row sums, and main diagonal sum of a magic square matrix are equal. Define `mat1` to be a 3×3 magic square matrix. Calculate the column sums `mat1ColSum`, row sums `mat1RowSum`, main diagonal sum `mat1DiagSum` using `sum`.
- (b) Create a 3×3 matrix $\text{mat2} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$ by reshaping `vec1` using `reshape` and `transpose`. Then calculate `matProduct1` which is the product of `mat1` and `mat2` using matrix multiplication, and `matProduct2` using elementwise multiplication. Observe the difference between these two multiplications.
- (c) Stack `matProduct1` and `matProduct2`, then assign it to `mat3` of which the size is 6×3 . Compute the sum `sumAll` of all elements of `mat3`, column minimums `mat3ColMin`, and row maximums `mat3RowMax` of `mat3`. Find the row indices `rowIndex` and column indices `colIndex` of entries that are *no greater than* 20 in `mat3`.
- (d) Add three new columns to `mat3`:
- Extract the second column of `mat3` and assign it as the fourth column of `mat3`.
 - Append the vector `int8(rand(6, 1)) * 255` to `mat3` by specifying the column index to be `end + 1`.
 - Use bracket to append the vector `colon(0,36,200)` to `mat3` as the last column.
 - Check the size of `mat3` and store it to `mat3Size`.

3 ARRAY: CHAR ARRAY VS. STRING ARRAY

- (a) Define `helloChar`, `worldChar`, `helloString`, `worldString` to be `'hello '`, `'world'`, `"hello "`, `"world"`, respectively. Concatenate the above strings as below:
- Use bracket to concatenate `helloChar` and `worldChar` horizontally and assign it to `helloWorldChar1`. Repeat this to `helloString` and `worldString`, then store the result to `helloWorldString1`.
 - Now use `strcat` instead of bracket (use `help strcat` to check out the syntax) to repeat the above. Store the results to `helloWorldChar2` and `helloWorldString2`.
- (b) Use `class` and `length` to determine the types and lengths of `helloWorldChar1`, `helloWorldChar2`, `helloWorldString1`, `helloWorldString2`. Create variables with suffix `Class` or `Length` to store the corresponding results. For example, the length of `helloWorldChar1` should be stored as `helloWorldChar1Length`.

4 APPLICATION: IMAGE PROCESSING

Next we will process Figure 1a to produce Figure 1b in MATLAB. For this portion, suppress the output if it is more than 10 lines by putting `;` at the end of the statement.



(a) `UW_gray.png`



(b) `UW_gray_new.png`

Figure 1: UWyo Logo

- (a) Read `UW_gray.png` and store it to `uwGray` using `imread`. Check the size of `uwGray` and store the size to `uwGraySize`. Find the maximal eigenvalue of the matrix `double(uwGray)` and store it to `maxEigenvalue`.
- (b) Crop the Steamboat out of the logo by extracting a submatrix from `uwGray` which starts from row 1 to row 650 and from column 171 to column 650, and save it to `steamboatLeft`. Flip `steamboatLeft` and store it to `steamboatRight`. Then create `steamboat` by concatenating matrix `steamboatLeft` and `steamboatRight`.
- (c) Extracting the name `uwName` from `uwGray` which starts from row 651 to row 960 (with all columns). Stack `steamboat` and `uwName` to create `uwGrayNew`. Check the size of `uwGrayNew` which should be 960×960 . If it is correct, write `uwGrayNew` to a file named `UW_gray_new.png` using `imwrite`. List all the variables in the workspace using `whos`.

In the Command Window enter the command `diary('lab_02_output.txt')`, run the script file `lab_02_script.m`, then type `diary off` to store the output to `lab_02_output.txt`. Then upload the script file `lab_02_script.m`, output file `lab_02_output.txt`, and `UW_gray_new.png` to the folder `src` on Overleaf. Next open `body.tex` under the folder `LaTeX`. In the last section of the report, you will reproduce the following using \LaTeX . Recompile, and submit the generated `.pdf` file to WyoCourses.

5 BASICS OF L^AT_EX

5.1 SINE FUNCTIONS

For given $x \in [0, 2\pi]$ with step size $\pi/12$, we can obtain the evaluations of (5.1) at x (see Table 1), and the corresponding plot (see Figure 2).

$$\begin{cases} y_1 = \sin(x/2) \\ y_2 = \sin(x) \\ y_3 = \sin(2x) \end{cases} \quad (5.1)$$

Table 1: Sine functions

x	$\sin(x/2)$	$\sin(x)$	$\sin(2x)$
0	0	0	0
$\pi/2$	$\sqrt{2}/2$	1	0
π	1	0	0
$3\pi/2$	$\sqrt{2}/2$	-1	0
2π	0	0	0

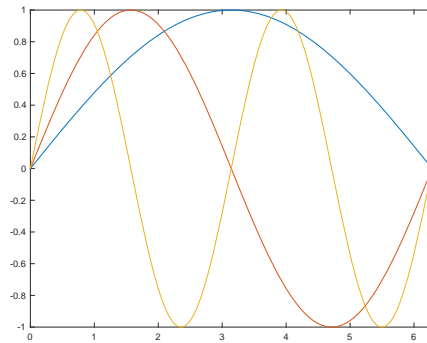


Figure 2: Sine functions

5.2 GOLDBACH'S CONJECTURE

Pursuing this type of analysis more carefully, Hardy and Littlewood in 1923 conjectured (as part of their famous *Hardy–Littlewood prime tuple conjecture*) that for any fixed $c \geq 2$, the number of representations of a large integer n as the sum of c primes $n = p_1 + \dots + p_c$ with $p_1 \leq \dots \leq p_c$ should be asymptotically equal to

$$\left(\prod_p \frac{p \gamma_{c,p}(n)}{(p-1)^c} \right) \int_{2 \leq x_1 \leq \dots \leq x_c: x_1 + \dots + x_c = n} \frac{dx_1 \cdots dx_{c-1}}{\ln x_1 \cdots \ln x_c}, \quad (5.2)$$

where the product is over all primes p , and $\gamma_{c,p}(n)$ is the number of solutions to the equation $n = q_1 + \dots + q_c \pmod p$ in modular arithmetic, subject to the constraints $q_1, \dots, q_c \not\equiv 0 \pmod p$.

This formula (5.2) has been rigorously proven to be asymptotically valid for $c \geq 3$ from the work of Vinogradov, but is still only a conjecture when $c = 2$. In the latter case, the above formula simplifies to 0 when n is odd, and to

$$2\Pi_2 \left(\prod_{p|n; p \geq 3} \frac{p-1}{p-2} \right) \int_2^n \frac{dx}{(\ln x)^2} \approx 2\Pi_2 \left(\prod_{p|n; p \geq 3} \frac{p-1}{p-2} \right) \frac{n}{(\ln n)^2},$$

when n is even, where Π_2 is Hardy-Littlewood's twin prime constant

$$\Pi_2 := \prod_{p \geq 3} \left(1 - \frac{1}{(p-1)^2} \right) = 0.6601618158 \dots$$

This is sometimes known as the extended Goldbach conjecture.

Reference: [Goldbach's conjecture](#).