1. Assignment

Getting to know Octave

Deadline: 21.10.2016

Description of assignment

This assignment aims at making you familiar with Octave.

Guidelines

(1) It is of utmost importance that your **code is executable**. Make sure that all results are printed and plotted by executing `main.m` and no errors occur.
(2) Create a **new window for each result** (hint: command `figure`. Do not overwrite results! (This assignment should produce five separate figure-windows).
(3) In the **second line of every file** fill in the following information as a comment: first name, last name and student id.
(4) All provided script and function files have to be used and completed if necessary. If you create additional files make sure that the code is still executable by `main.m`.
(5) All files have to be collected in a **zip-archive** and be uploaded in TUWEL (until the deadline).

Task 1: Hello Octave

Points: 1
Files: `hello.m`, `main.m`

Create a function in a separate file called `hello.m`. Input parameters are two strings: `firstname` and `lastname`. The output parameter `sentence` is one string concatenating the input strings into the following sentence: "Hello `firstname lastname`. How are you?"

For example for the input strings "Susi" and "Maier" the output parameter stores the string: "Hello Susi Maier. How are you?".

Add one line of code in the provided file `main.m` to call your function `hello` and store the output so that it can be printed out by `disp` in line 12 onto the console.

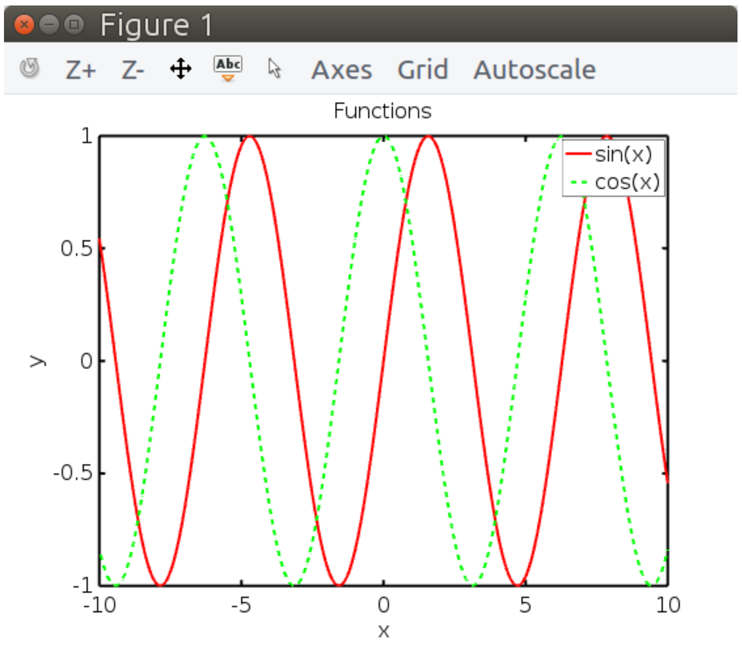

FIGURE 1. Output of plotting function.

Task 2: Plotting in Octave

Points:2
Files: `plotting.m`, `main.m`

The file `plotting.m` contains a function, which should be used to plot sin(x) and cos(x). In `main.m` the function is already called by `plotting(-10, 10, 0.1, 16, 2);` in line 15. The four parameters of the plotting function have to following purpose:

- `start` and `stop` define an interval for the plotting function
- `delta` sets the step size of the plotting function
- `fontsize` and `linewidth` allow the user to influence the look of the output

The plotting function should produce a graphical output like in Figure 1. Each function should be plotted in a different color and line style. The title of the figure should be "Functions" and the labels of the axes "x" and "y". In the legend you should display "sin(x)" and "cos(x)". All text should be set to the provided font size (input parameter `fontsize`). All lines (functions, boarder of drawing) should be set to the provided line width (input parameter `linewidth`).

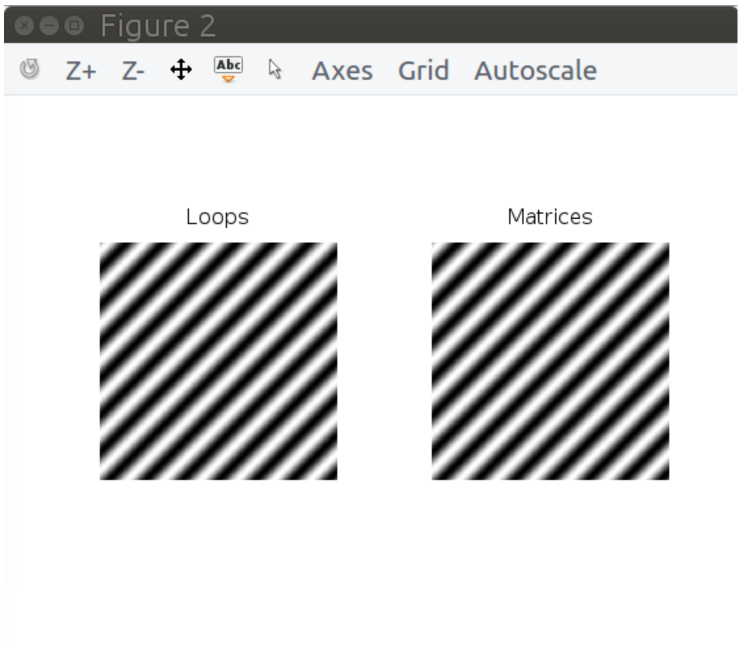

FIGURE 2. for-loops vs. matrices

Task 3: Matrices vs. Loops

Points: 3
Files: `vs.m`, `main.m`

This task is about comparing the performance of loops and matrices in Octave. The results will convince you to prefer matrices over loops.

The following function:

$$f(x,y) = A \cdot sin(u_0 \cdot x + v_0 \cdot y),$$

where $x = 1, 2, 3, \ldots, M$ and $y = 1, 2, 3, \ldots, N$. The input parameters of the function are $A = 1$, $u_0 = \frac{1}{4\cdot\pi}$, $v_0 = \frac{1}{4\cdot\pi}$, $M = 512$, and $N = 512$ (see `main.m` line 18).

Your task is to calculate $f(x,y)$ with the help of `for`-loops and afterwards with matrices. Then the running times are compared.

For this task you will already find some code in the file `vs.m`. In three places you have to complete the code. This places are marked with `TASK 3.1`, `TASK 3.2` and `TASK 3.3`.

For both solutions (with loops and matrices) the output should look the same (see Figure 2).

Task 4: Feature Extraction

Points: 4
Files: `extractFeatures.m`, `main.m`
Binary input images: `bunny.pgm`, `dog.pgm`, `hand.pgm`

This task aims at working with images, extracting simple features from binary images and creating a data set stored in a matrix.

The first part of this task (see TASK 4.1 in file `extractFeatures.m`) is to implement a function, which is able to do the following:

- read in the image with the provided filename (see input parameter `imgName`)
- invert the image so that the objects (bunny, dog, hand) are in the foreground
- plot the image in a new window
- extract at least five features of the binary image
- store the features in the output vector `features`
- plot the features and a description in the console (see example below)

Example for plotting of features in console:

```
Features of bunny.pgm
Area: 3401
Length of major axis: 87.192
Length of minor axis: 53.924
Euler number: 1
Perimeter: 266
-----------------
```

The last task (see TASK 4.2 in `main.m` line 22) is to store the extracted features of the three binary input images in a matrix. Call the matrix "database" and store the three output vectors so that each column of the database belongs to an image (i.e. each row shows the values of the same feature in the three images).