Matlab Exercises 2

Dr. Bernhard Kainz

14 January, 2016

Tasks

You can find all the necessary files on my homepage:
`http://www.doc.ic.ac.uk/~bkainz/Matlab/labfiles.zip`

1. Put the sequence of commands you used for Q.10 on labsheet 1 into a script file, called `plotsine.m`. Call that script file to draw the graph.

2. (a) Copy the file `data.mat` to the working directory, and load the data. This gives you variables `X` and `Y`. Look at the first row of variable `X`, i.e. `X(1,:)`, can you plot a histogram of this data (see `help hist`)?

   (b) Write a function `plothists` that takes a matrix input and plots histograms of every row of the data separately as subplots. The subplots should appear in the smallest square grid possible. For instance if the input matrix has 32 rows, then you should use a 6×6 grid of subplots and fill all but the last 4 spaces. Try saving the results of `plothists(X)` and `plothists(Y)` to files. What differences can you see between the two datasets?

3. Rewrite the solution to Q.9 from labsheet 1 as an m-file function, which takes as input argument an upper bound on the sequence. So if you want to evaluate all values in the sequence less than 1000, then you would pass in 1000. Call the function `mysequence`. Use a subfunction called test that returns 1 if a value belongs to the sequence and 0 otherwise.

4. (a) Import the `steps.csv` and `activity_mix.csv` files into your workspace with `importdata`. These contain step count and activity level data for a number of users. Each row is the data from a given day for a given user. Across both files, the columns are as follows (you can ignore the grey items):

   - `user_id` – the id of the user.
   - `steps` – a step count for that day for that user.
   - `sedentaryMinutes` – number of minutes the user was sedentary.
   - `lightlyActiveMinutes` – number of minutes the user was lightly active.

- `fairlyActiveMinutes` – number of minutes the user was fairly active.
- `veryActiveMinutes` – number of minutes the user was very active.
- `steps_at` and `activity` at – are time stamps for the data.
- `day` – number of days from the start of the study.
- `dow` – the day of the week.

(b) Now write a package `lifelogging`, containing functions:

- `step_scores`: which takes vector of step counts, and calculates the `step_score` for every element, where

  `step score = steps/500`

- `activity_scores`: which takes the $N \times 4$ array whose columns are `sedentaryMinutes`, `lightlyActiveMinutes`, `fairlyActiveMinutes`, and `veryActiveMinutes`, and calculates the `activity_score` for every row, where
  `activity_score` $= \frac{1}{48}$ `lightlyActiveMinutes`+ $\frac{4}{48}$ `fairlyActiveMinutes`+ $\frac{20}{48}$ `veryActiveMinutes`
  Do not use a `for` loop!
- `mean_cores`: which takes a vector of scores and a vector indicating which user generated that score, and returns the mean score for each user.
- `summarise_data` which takes the two imported data objects, and creates a struct array with fields: `user_id` (the user id); `step_score`, the user's mean step score; and `activity_score`, the user's average activity score. `summarise_data` will use the other functions.

(c) Now call `summarise_data` from the parent folder, and use the output to scatter-plot `step_score` versus `activity_score` for all users.

(d) (optional) Can you think of other interesting ways to represent the data?

5. Load `oarfish_small.jpg` (or any other small jpg image) with `imread` (make sure it is at most 256×256 pixels), you can display it in matlab with the image function. Find the segmented image using the provided function `segment_image`. You can display the resulting image data again with `image`. Now write a new function `get_boundary_image` that reuses the clustered image data to find the outline of regions in the image in a black and white image. Hint: locate pixels in the clustered image which neighbour at least one different coloured pixel.