Data Handling v3

Dr Amir-Homayoun Javadi
a.h.javadi@gmail.com
www.javadilab.com

Data Handling in MATLAB

General notes

To list the files, you can use dir function:

```
List = dir('*.mat')     % lists all the files with .mat extension
List = dir('Data_*.*')  % lists all the files beginning with 'Data_'
```

List will be a structure-array with multiple fields. For example:

```
List(2).name      % refers to the name of the 2nd file/folder
List(3).isdir     % indicates whether the 3rd file/folder is a
                  % file (zero) or a folder (one)
List(4).folder    % indicates the address of the 4th file/folder
```

To refer to files in different directories, you can use fullfile function. For example the line below lists all the files beginning with 'Data_' and file extension '.mat' in the folder 'Data, Sample'.

```
List = dir(fullfile('Data, Sample', 'Data_*.mat'))
```

Of course, if you want to access the files listed in array List, you need to include the folder address as well.

```
load(fullfile('Data, Sample', List(1).name))    % loads file List(1).name from
                                                % folder 'Data, Sample'
```

Native MATLAB data

To save and load MATLAB data you can easily use save and load functions:

```
save Data   % saves all the variables in the memory to the current folder
load Data   % loads all the variables in file Data.mat
```

If you want to specify specific variables to save or load, you can use the following:

```
save Data var1 var2 …
load Data var1 var2 …
```

A better way of saving and loading files is to use standard method of calling functions using parentheses:

```
save('Data', 'var1', 'var2')
```

```
load('Data', 'var1', 'var2')
```

Excel data

To save and load Excel files, you can use `xlsread` and `xlswrite` as follows:

```
xlswrite('filename.xlsx', 'sheet')                  % sheet is optional
[num, text, raw] = xlsread('filename.xlsx', 'sheet')  % sheet is optional
```

`num`, `text` and `raw` refer to numerical values only, text values only and all content, respectively. If you want to read the text content without numerical values you can use ~ sign as follows:

```
[~, text] = xlsread('filename.xlsx')
```

Using the same commands you can save and load .xls files. .xls files are for 97-2003 excel file formats and .xlsx files are for 2007 and later file formats.

One note, when you read and write files, make sure that they are not open elsewhere.

Data Handling in Python

General notes

To list the files, you can use `listdir` function from `os` library:

```
import os
List = os.listdir('*.mat')    % lists all the files with .mat extension
List = os.listdir('Data_*.*') % lists all the files beginning with 'Data_'
```

To refer to files in different directories, you need to use `\\` in Windows machines and `/` in Mac machines to separate folders and filenames. For example the line below lists all the files beginning with 'Data_' and file extension '.mat' in the folder 'Data, Sample'.

```
List = os.listdir('Data, Sample\\Data_*.mat')
```

Of course, if you want to access the files listed in array `List`, you need to include the folder address as well.

```
Data = sio.loadmat('Data, Sample\\' + List[1])  % loads file List[1] from
                                                % folder 'Data, Sample'
```

Native Python data

Python does not have any native file format as MATLAB does. Therefore, you need to use different toolboxes to save data. There are quite a lot of options. The one that I would suggest you to use is saving and loading MATLAB data files. It has multiple advantages such as compatibility with MATLAB, benefitting from MATLAB well defined structures and data types. For this purpose, you need to use `scipy.io` library. The code below is to load a MATLAB file and access different variables.

```
import scipy.io as sio
Data = sio.loadmat('filename.mat')
var1 = Data['var1']
var2 = Data['var2']
```

To save data to a MATLAB file, you can use the following command:

```
sio.savemat('filename.mat', {'var1':var1, 'var2':var2})
```

One note, Data in the above example is a `dictionary` which contains all the variables in data file `filename.mat`. To access individual variables you need to use the keys as shown above. To access all the keys you can look at `Data.keys()`

Excel data

To read and write Excel data files you need to use two different libraries. To read an Excel file you need to use `xlrd` toolbox as below. First you need to make a link to the file, and then to the sheet to finally extract values from the file.

```
import xlrd
WB = xlrd.open_workbook('filename.xlsx')  % access to the file
WB.sheet_names()                       % lists all the sheets in the Excel file.
S = WB.sheet_by_name('sheet')          % access to the actual sheet
                                       % using the name of the sheet
S = WB.sheet_by_index(number)          % access to the actual sheet using the
                                       % index of the sheet. This index is based
                                       % on what is shown in WB.sheet_names()
                                       % number zero refers to the first sheet
```

So far we have opened the Excel file and accessed the sheet. To actually get the data out you need to use the following:

```
a = S.cell_value(i, j)  % content of row i and column j
                        % pay attention that i and j begin from zero
b = S.row_values(i)     % all the cells in row i
c = S.col_values(j)     % all the cells in column i
d = S.nrows             % number of rows
e = S.ncols             % number of columns
```

To write to Excel files you need to use `xlsxwriter` toolbox:

```
import xlsxwriter as xlsw
WB = xlsw.Workbook('filename.xlsx') % attention that Workbook is with capital W
S = WB.add_worksheet('sheet')
s.write(i, j, value)                   % stores value in row i and column j.
WB.close()                             % make sure that you close your Excel file
                                       % at the end.
```