Data input and output

Save and load binary data (1/2)

The most convenient way to save or read data when that is not used outside Matlab® is to create a *binary file*

```
>> save        % Saves the workspace in the matlab.mat
>> clear all
>> load        % Loads a workspace from matlab.mat
```

Save a given variable in a specific file by specifying arguments

```
>> A=rand(3); B=rand(4);
>> save myfile.mat A B
>> clear
>> load myfile
>> who
```

```
>> A=rand(3); B=rand(4);
>> save("myfile","A","B")
>> clear
>> load("myfile")
>> who
```

Save and load binary data (2/2)

When the name of the variables contained in the file are known *a-priori*, it is possible to load it specifically

```
>> load myfile.mat A      % Loads the variable A only
>> who
```

When loading the file content into a variable, its type is `struct` and its fields are the names of the variables stored in the file

```
>> saved = load('myfile.mat');
>> saved                          % Look in this structure
>> A_new = saved.A
>> B_new = saved.B
```

Note: save in a text file (no formatting) with `save myfile.txt -ascii`

Save formatted files

If the data has to be exported in a format or typology that is human-readable, this has to be with the command `fprintf`

```
>> fid = fopen('myfile2.txt', 'w')
>> fprintf(fid, 'Date; Mesurement\n')
>> fprintf(fid, '%s; %f\n', datestr(now), 1.245)
>> fclose(fid)
```

- The command `fprintf` has the same structure as `sprintf` in the creation of formatted sentences to be written in the file
- The choice of file extension (.txt, .dat, .csv, .any , ...) is free. Be careful however not to use extensions misleading for Maltab® (.fig, .mat) or external software (.tiff, .odt, ...)

Load formatted files

A structured way to read data from files having a predefined format is to use `textscan`. It returns a cell array, the cells' elements corresponding to the columns (see the help for more options)

```
>> fid=fopen('ListOfThings.txt');

>> % Get the 1-line header of the file
>> HDR = textscan(fid,'%s %s',1, 'delimiter',';');

>> % Get all the following rows having a same layout
>> DATA = textscan(fid,'%s %f', 'delimiter',';');
>> meastimes = datevec(DATA{1},'dd.mm.yyyy HH:MM');
>> mesval = DATA{2};

>> fclose(fid);
```

Save and load automatically CSV formatted files

When a file is simply formatted as Comma Separated Values, it is possible to save and read data automatically from it with `csvread` and `csvwrite` (see the help for more options)

```
>> A=rand(4)
>> csvwrite('file.csv',A)
```

```
>> clear all
>> B=csvread('file.csv')
```

Separation signs as comma, semicolon or blank are recognized. Specific separators can be manually defined as a 2nd argument

Further means of saving and loading external data

There are many more ways to save and read data

- Text files and human-readable (ascii) formats can be dealt with `fread`, `fwrite`, `dlmread`, `dlmwrite`, `importdata`, ...
- Specific routines for database files or spreadsheets are also available `xlsread`, `xlswrite`, ...

Each command has its own coding format. Check them in the help

```
>> help importdata
```

Note: The Matlab GUI also contains some information about how to import and export data to generate *Import-scripts* (Import Data Button)

Best practice

- When the data is wished to be used within Matlab® only, prefer using binary format
- Choose an formatting rule where you loose the less information as possible when exporting the data in a human-readable format
- Pay attention to give a meaningful file extension that is not yet commonly used when exporting data in your specific format

Exercises

Exercises: data input and output

Exercise

1. Write a function `mysum`, which takes as many inputs as desired. The output is given by partial sums:

   $$out(k) = \sum_{i=1}^{\texttt{nargin}} input(i), \quad \forall k = \{\texttt{nargin+1}, \texttt{nargout}\}$$

   if `nargin < nargout`, and

   $$out(k) = \sum_{i=1}^{k} input(i), \quad \forall k = \{1, \ldots, \texttt{nargin}\}$$

   otherwise

2. Enhance the function `curry`, in way to compute the composition of as many functions as the user wants.

$$f_1(f_2(f_3(f_4 \ldots (f_n(x)) \ldots)))$$

Exercises: data input and output

Exercise

3. a) Write a script, which imports the real weather data from the file *wetter.txt*.

   b) Plot the data in a readable way (for example a different histogram for temperature, humidity and pressure).