Matlab scripts to divide and save wave signals from a single file

Script features:

-Searches in a file divided on columns for each wave signal
-It divides each column on a different data signal.
-It stores each wave signal on a single file.
-Erases any inconsistency as blank spaces without data or a NAN value (Nor Data available).

What is need it:

-Each wave train needs to be stores in a .dat file or .txt file
-Wave trains need to be stored in a column each or in a simple row (Script 1 or Script 2)

What can be done:

-The wok can be replicated on python, C or any other programming language.
-The wave gauges can be just added repeating the code, but also an array can be added so an indefinite number of waves gauges can be on the file and be sorted if they all work on the same time frame. This was fixed on the second experimental testing.

Code sample:

1st block:

During the 1st block we only call to the data stored in the file from a certain location of the computer using a readTable() command. Then we measure the table size cause this will work as our boundaries for our loop cycles.

```
datatotal=readtable(‘Directory where the file is stored');
HD=height(datatotal);
WD=width(datatotal);
i=1;
```

datatotal=readtable() The command readtable reads the text data stores in a file, every black space is treated as a division for a different cell of data and every jump in line as a different row on the table of data.

HD=height()/WD=width(): We set the variables of the table size, so our script will only read till the final of our file (HD-final time) and the end of column (WD-final wave probe).

Then we simply set a counter to read every repetition till HD (final time is reached).

2nd block

in the second block the information from each columns is stored on and individual list, one list will be used per counting time and the other ones to store the value of the wave height.

```
while i<=HD
    for j=1:5

        if j==2
           cd=datatotal{i,1};
           CD{i,1}=cd;
           cd=datatotal{i,j};
           CD{i,j}=cd;
        end

        if j==3
           cd2=datatotal{i,1};
           CD2{i,1}=cd2;
           cd2=datatotal{i,j};
           CD2{i,j}=cd2;
        end

         if j==4
           cd3=datatotal{i,1};
           CD3{i,1}=cd3;
           cd3=datatotal{i,j};
           CD3{i,j}=cd3;
         end

         if j==5
           cd4=datatotal{i,1};
           CD4{i,1}=cd4;
           cd4=datatotal{i,j};
           CD4{i,j}=cd4;
        end

    end
   i=i+1;
end
```

In our first cycle `while i<=HD` we will repeat the process till `i` reaches HD (final time value). Then we pass trough every column that contains data being sorted as:

Column 1 = time, Column 2 = wave probe 1, Column 3 = wave probe 2…. Column n = wave probe n.

This can be seeing in detail on figure 1. If more sensors need to be added just a copy of the block if j == “wave prove number=n” needs to be added on this script.

```
        if j==“n”
           cd”n”=datatotal{i,1};
           CD”n”{i,1}=cd”n”;
           cd”n”=datatotal{i,j};
           CD”n”{i,j}=cd”n”;
         end
```

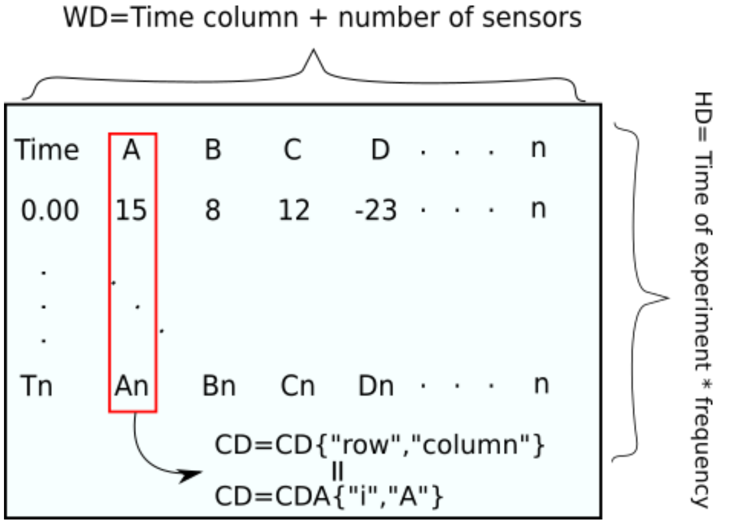

Fig. 1.- Each if structure is used to store the data of a single sensor, this will be used later to write into a table of data.

`CD` is an array that will contain only two columns `CD{n,2}`, taken from `datatotal{mxn}`, the first one that we will call on each column `datatotal{i,1}` it store the time and the second one `datatotal{i,2}` will store the wave height collected for each sensor.

The process will be repeated for every wave probe variable:

```
     if j==3
           cd2=datatotal{i,1};
           CD2{i,1}=cd2;
           cd2=datatotal{i,j};
           CD2{i,j}=cd2;
        end

         if j==4
           cd3=datatotal{i,1};
           CD3{i,1}=cd3;
           cd3=datatotal{i,j};
           CD3{i,j}=cd3;
         end

         if j==5
           cd4=datatotal{i,1};
           CD4{i,1}=cd4;
           cd4=datatotal{i,j};
           CD4{i,j}=cd4;
        end
```

3rd block

On the thirds block the tables are put into an individual file using the character / as a delimiter, this will add black spaces that can be then used to save it as a .csv file

```
Table=cell2table(CD);
writetable(Table,'Directory where the file will be stored, Sensor
1','Delimiter','\t');

Table2=cell2table(CD2);
```

```
writetable(Table2,'Directory where the file will be stored, Sensor
1','Delimiter','\t');

Table3=cell2table(CD3);
writetable(Table3,'Directory where the file will be stored, Sensor
1','Delimiter','\t');

Table4=cell2table(CD4);
writetable(Table4,’Directory where the file will be stored, Sensor
1','Delimiter','\t');
```

Table=cell2table() is used to write a cell type array like excel type to convert it into a text table. Then writetable(“tablename”, “path and name of file to be saved”) is used to store it. Here a lot of delimiters can be used to be able to separate the files.

4th block

In this block files are checked for blank spaces or other non-numeric data, this one will be erased from the files. Most of the time this data is just inserted wrong by the sensor and is not lost, data file then will contain a black space instead of the data; however this correct data will be stored on the next line.

```
% Read the file as cell string line by line:
fid = fopen('Directory where the file is stored, Sensor 1', 'r');
if fid < 0, error('Cannot open file: %s'); end
Data = textscan(fid, '%s', 'delimiter', '\n', 'whitespace', '');
fclose(fid);

% Remove empty lines:
C    = deblank(Data{1});   % [EDITED]: deblank added
C(cellfun('isempty', C)) = [];
% Write the cell string:
fid = fopen('Directory where the file is stored, Sensor 1', 'w');
if fid < 0, error('Cannot open file: %s', FileName); end
fprintf(fid, '%s\n', C{:});
fclose(fid);


fid = fopen('Directory where the file is stored, Sensor 2', 'r');
if fid < 0, error('Cannot open file: %s'); end
Data = textscan(fid, '%s', 'delimiter', '\n', 'whitespace', '');
fclose(fid);

C    = deblank(Data{1});   % [EDITED]: deblank added
C(cellfun('isempty', C)) = [];
% Write the cell string:
fid = fopen('Directory where the file is stored, Sensor 2', 'w');
if fid < 0, error('Cannot open file: %s', FileName); end
fprintf(fid, '%s\n', C{:});
fclose(fid);



fid = fopen('Directory where the file is stored, Sensor 3', 'r');
if fid < 0, error('Cannot open file: %s'); end
Data = textscan(fid, '%s', 'delimiter', '\n', 'whitespace', '');
fclose(fid);
```

```
C    = deblank(Data{1});   % [EDITED]: deblank added
C(cellfun('isempty', C)) = [];
% Write the cell string:
fid = fopen('Directory where the file is stored, Sensor 3', 'w');
if fid < 0, error('Cannot open file: %s', FileName); end
fprintf(fid, '%s\n', C{:});
fclose(fid);


fid = fopen('Directory where the file is stored, Sensor 4', 'r');
if fid < 0, error('Cannot open file: %s'); end
Data = textscan(fid, '%s', 'delimiter', '\n', 'whitespace', '');
fclose(fid);

C    = deblank(Data{1});   % [EDITED]: deblank added
C(cellfun('isempty', C)) = [];
% Write the cell string:
fid = fopen('Directory where the file is stored, Sensor 4', 'w');
if fid < 0, error('Cannot open file: %s', FileName); end
fprintf(fid, '%s\n', C{:});
fclose(fid);
```