

Read files, determine the means, the mean of the mean and the standard deviation:

```
1 clear all
2
3 A = [importdata('test.xlsx') importdata('test2.xlsx')] ;
4 % add files if needed but excel files need to be same length!!!!
5
6 p = [2:8];
7
8 for i = 1:length(A)/10; % change i. i is same as amount of files you ...
    wanna read
9
10     neutral(:,i) = str2num(cell2mat(A(2:end,p(1))));
11     maxneutral(i) = max(neutral(:,i));
12     meanneutral(i) = mean(neutral(:,i));
13
14     happy(:,i) = str2num(cell2mat(A(2:end,p(2))));
15     maxhappy(i) = max(happy(:,i));
16     meanhappy(i) = mean(happy(:,i));
17
18     sad(:,i) = str2num(cell2mat(A(2:end,p(3))));
19     maxsad(i) = max(sad(:,i));
20     meansad(i) = mean(sad(:,i));
21
22     angry(:,i) = str2num(cell2mat(A(2:end,p(4))));
23     maxangry(i) = max(angry(:,i));
24     meanangry(i) = mean(angry(:,i));
25
26     surprised(:,i) = str2num(cell2mat(A(2:end,p(5))));
27     maxsurprised(i) = max(surprised(:,i));
28     meansurprised(i) = mean(surprised(:,i));
29
30     scared(:,i) = str2num(cell2mat(A(2:end,p(6))));
31     maxscared(i) = max(scared(:,i));
32     meanscared(i) = mean(scared(:,i));
33
34     disgusted(:,i) = str2num(cell2mat(A(2:end,p(7))));
35     maxdisgusted(i) = max(disgusted(:,i));
36     meandisgusted(i) = mean(disgusted(:,i));
37
38     p = p+10;
39 end
40
41 %%
42
43 meanmeanneutral = mean(meanneutral);
44 meanmeanhappy = mean(meanhappy);
45 meanmeansad = mean(meansad);
46 meanmeanangry = mean(meanangry);
47 meanmeansurprised = mean(meansurprised);
48 meanmeanscared = mean(meanscared);
49 meanmeandisgusted = mean(meandisgusted);
50
51 errormeanneutral = std(meanneutral);
52 errormeanhappy = std(meanhappy);
53 errormeansad = std(meansad);
54 errormeanangry = std(meanangry);
55 errormeansurprised = std(meansurprised);
56 errormeanscared = std(meanscared);
57 errormeandisgusted = std(meandisgusted);
58
```

```

59 %%
60
61 meanmaxneutral = mean(maxneutral);
62 meanmaxhappy = mean(maxhappy);
63 meanmaxsad = mean(maxsad);
64 meanmaxangry = mean(maxangry);
65 meanmaxsurprised = mean(maxsurprised);
66 meanmaxscared = mean(maxscared);
67 meanmaxdisgusted = mean(maxdisgusted);
68
69 errormaxneutral = std(maxneutral);
70 errormaxhappy = std(maxhappy);
71 errormaxsad = std(maxsad);
72 errormaxangry = std(maxangry);
73 errormaxsurprised = std(maxsurprised);
74 errormaxscared = std(maxscared);
75 errormaxdisgusted = std(maxdisgusted);
76
77 %%
78 figure
79
80 subplot(2,2,1)
81 y1 = [meanmeanneutral meanmeanhappy meanmeansad meanmeanangry ...
      meanmeansurprised meanmeanscared meanmeandisgusted];
82 erry1 = [errormeannneutral errormeanhappy errormeansad errormeanganry ...
      errormeansurprised errormeanscared errormeandisgusted];
83 str = {'Neutral'; 'Happy'; 'Sad'; 'Angry'; 'Surprised'; 'Scared'; ...
      'Disgusted'};
84 barwitherr(erry1,y1)
85 title('Expected emotion: Happy') %change name is needed
86 set(gca, 'XTickLabel',str, 'XTick',1:numel(str))
87 ylabel('Intensity')
88 xlabel('Mean emotion')
89
90 subplot(2,2,2)
91 y2 = [meanmeanhappy meanmeansad meanmeanangry meanmeansurprised ...
      meanmeanscared meanmeandisgusted];
92 erry2 = [errormeanhappy errormeansad errormeanganry errormeansurprised ...
      errormeanscared errormeandisgusted];
93 str = {'Happy'; 'Sad'; 'Angry'; 'Surprised'; 'Scared'; 'Disgusted'};
94 barwitherr(erry2,y2)
95 title('Expected emotion: Happy')
96 set(gca, 'XTickLabel',str, 'XTick',1:numel(str))
97 ylabel('Intensity')
98 xlabel('Mean emotion')
99
100 subplot(2,2,3)
101 y3 = [meanmaxneutral meanmaxhappy meanmaxsad meanmaxangry ...
      meanmaxsurprised meanmaxscared meanmaxdisgusted];
102 erry3 = [errormaxneutral errormaxhappy errormaxsad errormaxangry ...
      errormaxsurprised errormaxscared errormaxdisgusted];
103 str = {'Neutral'; 'Happy'; 'Sad'; 'Angry'; 'Surprised'; 'Scared'; ...
      'Disgusted'};
104 barwitherr(erry3,y3)
105 title('Expected emotion: Happy') %change name is needed
106 set(gca, 'XTickLabel',str, 'XTick',1:numel(str))
107 ylabel('Intensity')
108 xlabel('Maximum emotion')
109
110 subplot(2,2,4)
111 y4 = [meanmaxhappy meanmaxsad meanmaxangry meanmaxsurprised ...
      meanmaxscared meanmaxdisgusted];

```

```

112 erry4 = [errormaxhappy errormaxsad errormaxangry errormaxsurprised ...
           errormaxscared errormaxdisgusted];
113 str = {'Happy'; 'Sad'; 'Angry'; 'Surprised'; 'Scared'; 'Disgusted'};
114 barwitherr(erry4,y4)
115 title('Expected emotion: Happy')
116 set(gca, 'XTickLabel',str, 'XTick',1:numel(str))
117 ylabel('Intensity')
118 xlabel('Maximum emotion')

```

Code to make a bar graph with error:

```

1  %*****
2  %
3  %   This is a simple extension of the bar plot to include error bars.  It
4  %   is called in exactly the same way as bar but with an extra input
5  %   parameter "errors" passed first.
6  %
7  %   Parameters:
8  %   errors - the errors to be plotted (extra dimension used if assymmetric)
9  %   varargin - parameters as passed to conventional bar plot
10 %   See bar and errorbar documentation for more details.
11 %
12 %   Output:
13 %   [hBar hErrorbar] = barwitherr(..) returns a vector of handles to the
14 %   barseries (hBar) and error bar (hErrorbar) objects
15 %
16 %   Symmetric Example:
17 %   y = randn(3,4);           % random y values (3 groups of 4 parameters)
18 %   errY = 0.1.*y;           % 10% error
19 %   h = barwitherr(errY, y); % Plot with errorbars
20 %
21 %   set(gca,'XTickLabel',{'Group A','Group B','Group C'})
22 %   legend('Parameter 1','Parameter 2','Parameter 3','Parameter 4')
23 %   ylabel('Y Value')
24 %   set(h(1),'FaceColor','k');
25 %
26 %
27 %   Asymmetric Example:
28 %   y = randn(3,4);           % random y values (3 groups of 4 parameters)
29 %   errY = zeros(3,4,2);
30 %   errY(:,:,1) = 0.1.*y;    % 10% lower error
31 %   errY(:,:,2) = 0.2.*y;    % 20% upper error
32 %   barwitherr(errY, y);    % Plot with errorbars
33 %
34 %   set(gca,'XTickLabel',{'Group A','Group B','Group C'})
35 %   legend('Parameter 1','Parameter 2','Parameter 3','Parameter 4')
36 %   ylabel('Y Value')
37 %
38 %
39 %   Notes:
40 %   Ideally used for group plots with non-overlapping bars because it
41 %   will always plot in bar centre (so can look odd for over-lapping bars)
42 %   and for stacked plots the errorbars will be at the original y value is
43 %   not the stacked value so again odd appearance as is.
44 %
45 %   The data may not be in ascending order.  Only an issue if x-values are
46 %   passed to the fn in which case their order must be determined to
47 %   correctly position the errorbars.
48 %
49 %

```

```

50 % 24/02/2011 Martina F. Callaghan Created
51 % 12/08/2011 Martina F. Callaghan Updated for random x-values
52 % 24/10/2011 Martina F. Callaghan Updated for asymmetric errors
53 % 15/11/2011 Martina F. Callaghan Fixed bug for asymmetric errors &
54 % vector plots
55 % 14/06/2013 Martina F. Callaghan Returning handle as recommended by
56 % Eric (see submission comments)
57 % 08/07/2013 Martina F. Callaghan Only return handle if requested.
58 % 18/07/2013 Martina F. Callaghan Bug fix for single group data that
59 % allows asymmetric errors.
60 % Also removed dot from display as
61 % per Charles Colin comment. The
62 % handle can be returned to control
63 % appearance.
64 % 27/08/2013 Martina F. Callaghan Ensuring errors are always stored
65 % as lowerErrors and upperErrors even
66 % if symmetric.
67 % 29/10/2014 Martina F. Callaghan Updated for 2014b graphics
68 %
69 %*****
70
71 function varargout = barwitherr(errors,varargin)
72
73 % Check how the function has been called based on requirements for "bar"
74 if nargin < 3
75     % This is the same as calling bar(y)
76     values = varargin{1};
77     xOrder = 1:size(values,1);
78 else
79     % This means extra parameters have been specified
80     if isscalar(varargin{2}) || ischar(varargin{2})
81         % It is a width / property so the y values are still varargin{1}
82         values = varargin{1};
83         xOrder = 1:size(values,1);
84     else
85         % x-values have been specified so the y values are varargin{2}
86         % If x-values have been specified, they could be in a random order,
87         % get their indices in ascending order for use with the bar
88         % locations which will be in ascending order:
89         values = varargin{2};
90         [tmp xOrder] = sort(varargin{1});
91     end
92 end
93
94 % If an extra dimension is supplied for the errors then they are
95 % asymmetric split out into upper and lower:
96 if ndims(errors) == ndims(values)+1
97     lowerErrors = errors(:,:,1);
98     upperErrors = errors(:,:,2);
99 elseif isvector(values)~=isvector(errors)
100     lowerErrors = errors(:,1);
101     upperErrors = errors(:,2);
102 else
103     lowerErrors = errors;
104     upperErrors = errors;
105 end
106
107
108 % Check that the size of "errors" corresponds to the size of the y-values.
109 % Arbitrarily using lower errors as indicative.
110 if any(size(values) ~= size(lowerErrors))
111     error('The values and errors have to be the same length')

```

```

112 end
113
114 [nRows nCols] = size(values);
115 handles.bar = bar(varargin{:}); % standard implementation of bar fn
116 hold on
117 hBar = handles.bar;
118
119 if nRows > 1
120     hErrorbar = zeros(1,nCols);
121     for col = 1:nCols
122         % Extract the x location data needed for the errorbar plots:
123         if verLessThan('matlab', '8.4')
124             % Original graphics:
125             x = get(get(handles.bar(col), 'children'), 'xdata');
126         else
127             % New graphics:
128             x = handles.bar(col).XData + [handles.bar(col).XOffset];
129         end
130         % Use the mean x values to call the standard errorbar fn; the
131         % errorbars will now be centred on each bar; these are in ascending
132         % order so use xOrder to ensure y values and errors are too:
133         hErrorbar(col) = errorbar(mean(x,1), values(xOrder,col), ...
134             lowerErrors(xOrder,col), upperErrors(xOrder, col), '.k');
135         set(hErrorbar(col), 'marker', 'none')
136     end
137 else
138     if verLessThan('matlab', '8.4')
139         % Original graphics:
140         x = get(get(handles.bar, 'children'), 'xdata');
141     else
142         % New graphics:
143         x = handles.bar.XData + [handles.bar.XOffset];
144     end
145     hErrorbar = errorbar(mean(x,1), values, lowerErrors, upperErrors, ...
146         '.k');
147     set(hErrorbar, 'marker', 'none')
148 end
149 hold off
150
151 switch nargin
152     case 1
153         varargin{1} = hBar;
154     case 2
155         varargin{1} = hBar;
156         varargin{2} = hErrorbar;
157 end

```