```
%PLOT_GRAB_VER_2
%AUTHORS: AJ Fillo (filloa@oregonstate.edu) and Kyle
 Zada(zadak@oregonstate.edu)

%DESCRIPTION: The following code has been developed to pull data
 points off
%of an image of a plot. To start, save an image of a plot that is
%compatible with the Matlab software. Next, press the run button. This
 will
%open up a prompt window to search for the plot image. Once loaded
 into
%Matlab, click at the x/y-origin, x_extreme, and y_extreme in that
 order to
%calibrate the plot image. Follow the on-screen inquires and enter the
%assocated values/answers. Once complete, you can click on specific
 data
%points you wish to pull off the image (and error bars if specified).
 Once you are done, click anywhere
%outside of the plot (same number of times you click per point). For
 example,
%if you have x-error bars only, you will click three times per point
 total and therefore
%click three times to exit the program. Your data will be saved under
 the 'DATA' matrix.


%UPDATES
%Ver 2
    %Changed calculation of error bars from x,y coordinates to
 absolute error from point position
    %Changed output title of the data point plot

clear
clc
close all

[filename, pathname, filterindex] = uigetfile('*.*'); %gets File name
 and path

plot_img = imread([pathname,filename]); %reads in image file
figure
imshow(plot_img)                        %Plots image
axis image
title('In Order, Select Origin, X-Axis Extreme, Y-Axis
 Extreme','fontsize',16)
[x_pix_val,y_pix_val] = ginput(3);  %calls user mouse input from plot

x_axis_form = input('X-axis linear or log? Enter lin or log\n','s');
  %user input for axis type (linear or log)
y_axis_form = input('Y-axis linear or log? Enter lin or log\n','s');
  %user input for axis type (linear or log)
```

```
err_x = input('Are error bars present on the plot for the x-axis?
 Enter Y or N\n','s'); %user input for if error bars are present in
 the x-axis
err_y = input('Are error bars present on the plot for the y-axis?
 Enter Y or N\n','s'); %user input for if error bars are present in
 the y-axis

x_or_val = input('What is X-origin value?\n');
  %user input from x_axis value of the origin
y_or_val = input('What is Y-origin value?\n');
  %user input from y_axis value of the origin
x_axis_val = input('What is X-Extreme value?\n');
  %user input from x_axis value of the extreme
y_axis_val = input('What is Y-Extreme value?\n');
  %user input from y_axis value of the extreme

cal_x = (x_pix_val(2)-x_pix_val(1))/(x_axis_val-x_or_val);
  %calibration of the x_axis with respect to pixels per value
cal_y = (y_pix_val(1)-y_pix_val(3))/(y_axis_val-y_or_val);
  %calibration of the y_axis with repsect to pixels per value

b = 1;  %Initial condition for stopping the data picking process (=1
 means continue)
i = 1;  %Initial condition for the index for data points
title({'Select All Points of Interest.','Select error bars from
 positive to negative starting with x THEN Click outside axes to
 end'},'fontsize',12)

%The following while loop is used to pick points (pixels) from the
 plot
%until the user hits the cursor outside of the axes of the plot (data
 point
%collection will stop)
while b == 1
    if err_x == 'Y'  & err_y == 'Y'
        [x_points(i),y_points(i)] = ginput(1);
  %Graphical input of data points from the cursor, saved as x and y
 pixel coordinates
        [x_points_err_x_pos(i),x_points_err_y_pos(i)] = ginput(1);
  %Graphical input of positive error bar from the cursor, saved as x
 and y pixel coordinates
        [x_points_err_x_neg(i),x_points_err_y_neg(i)] = ginput(1);
  %Graphical input of negative error bar from the cursor, saved as x
 and y pixel coordinates
        [y_points_err_x_pos(i),y_points_err_y_pos(i)] = ginput(1);
  %Graphical input of positive error bar from the cursor, saved as x
 and y pixel coordinates
        [y_points_err_x_neg(i),y_points_err_y_neg(i)] = ginput(1);
  %Graphical input of negative error bar from the cursor, saved as x
 and y pixel coordinates
        type = 1;
    elseif err_x == 'Y'  & err_y == 'N'
```

```
        [x_points(i),y_points(i)] = ginput(1);
 %Graphical input of data points from the cursor, saved as x and y
pixel coordinates
        [x_points_err_x_pos(i),x_points_err_y_pos(i)] = ginput(1);
 %Graphical input of positive error bar from the cursor, saved as x
and y pixel coordinates
        [x_points_err_x_neg(i),x_points_err_y_neg(i)] = ginput(1);
 %Graphical input of negative error bar from the cursor, saved as x
and y pixel coordinates
        type = 2;
    elseif err_x == 'N'  & err_y == 'Y'
        [x_points(i),y_points(i)] = ginput(1);
 %Graphical input of data points from the cursor, saved as x and y
pixel coordinates
        [y_points_err_x_pos(i),y_points_err_y_pos(i)] = ginput(1);
 %Graphical input of positive error bar from the cursor, saved as x
and y pixel coordinates
        [y_points_err_x_neg(i),y_points_err_y_neg(i)] = ginput(1);
 %Graphical input of negative error bar from the cursor, saved as x
and y pixel coordinates
        type = 3;
    else
        [x_points(i),y_points(i)] = ginput(1);
 %Graphical input of data points from the cursor, saved as x and y
pixel coordinates
        type = 4;
    end

        if x_points(i) > x_pix_val(2) || y_points(i) < y_pix_val(3)
 %If cursor clicks outside the maximum values of x or y, data
collection stops
        x_points = x_points(1:i-1); %x data points pixels are saved,
not including the stopping click
        y_points = y_points(1:i-1); %y data points pixels are saved,
not including the stopping click
            if type == 1
              x_points_err_x_pos = x_points_err_x_pos(1:i-1);
              x_points_err_y_pos = x_points_err_y_pos(1:i-1);
              x_points_err_x_neg = x_points_err_x_neg(1:i-1);
              x_points_err_y_neg = x_points_err_y_neg(1:i-1);

              y_points_err_x_pos = y_points_err_x_pos(1:i-1);
              y_points_err_y_pos = y_points_err_y_pos(1:i-1);
              y_points_err_x_neg = y_points_err_x_neg(1:i-1);
              y_points_err_y_neg = y_points_err_y_neg(1:i-1);
            elseif type == 2
              x_points_err_x_pos = x_points_err_x_pos(1:i-1);
              x_points_err_y_pos = x_points_err_y_pos(1:i-1);
              x_points_err_x_neg = x_points_err_x_neg(1:i-1);
              x_points_err_y_neg = x_points_err_y_neg(1:i-1);
            elseif type == 3
              y_points_err_x_pos = y_points_err_x_pos(1:i-1);
              y_points_err_y_pos = y_points_err_y_pos(1:i-1);
              y_points_err_x_neg = y_points_err_x_neg(1:i-1);
```

```
            y_points_err_y_neg = y_points_err_y_neg(1:i-1);
        else
            break
        end
        b = 0;
    elseif x_points(i) < x_pix_val(1) || y_points(i) >
 y_pix_val(1)  %If cursor clicks outside the minimum values of x or y,
 data collection stops
    x_points = x_points(1:i-1); %x data points pixels are saved,
 not including the stopping click
    y_points = y_points(1:i-1); %y data points pixels are saved,
 not including the stopping click

        if type == 1
        x_points_err_x_pos = x_points_err_x_pos(1:i-1);
        x_points_err_y_pos = x_points_err_y_pos(1:i-1);
        x_points_err_x_neg = x_points_err_x_neg(1:i-1);
        x_points_err_y_neg = x_points_err_y_neg(1:i-1);

        y_points_err_x_pos = y_points_err_x_pos(1:i-1);
        y_points_err_y_pos = y_points_err_y_pos(1:i-1);
        y_points_err_x_neg = y_points_err_x_neg(1:i-1);
        y_points_err_y_neg = y_points_err_y_neg(1:i-1);
        elseif type == 2
        x_points_err_x_pos = x_points_err_x_pos(1:i-1);
        x_points_err_y_pos = x_points_err_y_pos(1:i-1);
        x_points_err_x_neg = x_points_err_x_neg(1:i-1);
        x_points_err_y_neg = x_points_err_y_neg(1:i-1);
        elseif type == 3
        y_points_err_x_pos = y_points_err_x_pos(1:i-1);
        y_points_err_y_pos = y_points_err_y_pos(1:i-1);
        y_points_err_x_neg = y_points_err_x_neg(1:i-1);
        y_points_err_y_neg = y_points_err_y_neg(1:i-1);
        else
            break
        end
    b = 0;
  else
    i = i+1;    %Index point increases if cursor does not click
 outside of x or y axis
  end
end

%The following for loop organizes the pixel clicks and transforms them
 to
%actual data points (as seen on the plots).
%The data points are calculated by subtracting the pixel data point
 from the
%origin data point and adding the origin numerical value from the plot
for p = 1:length(x_points)

  if type == 1
  x_values(p) = ((x_points(p)-x_pix_val(1))/cal_x)+ x_or_val;
  y_values(p) = ((y_pix_val(1)- y_points(p))/cal_y)+ y_or_val;
```

```
    x_error_pos(p) = abs((((x_points_err_x_pos(p)-x_pix_val(1))/
cal_x)+ x_or_val)-x_values(p));
    x_error_neg(p) = abs((((x_points_err_x_neg(p)-x_pix_val(1))/
cal_x)+ x_or_val)-x_values(p));

    y_error_pos(p) = abs((((y_pix_val(1)- y_points_err_y_pos(p))/
cal_y)+ y_or_val)-y_values(p));
    y_error_neg(p) = abs((((y_pix_val(1)- y_points_err_y_neg(p))/
cal_y)+ y_or_val)-y_values(p));

    elseif type == 2
    x_values(p) = ((x_points(p)-x_pix_val(1))/cal_x)+ x_or_val;
    y_values(p) = ((y_pix_val(1)- y_points(p))/cal_y)+ y_or_val;

    x_error_pos(p) = abs((((x_points_err_x_pos(p)-x_pix_val(1))/
cal_x)+ x_or_val)-x_values(p));
    x_error_neg(p) = abs((((x_points_err_x_neg(p)-x_pix_val(1))/
cal_x)+ x_or_val)-x_values(p));

    elseif type ==3
    x_values(p) = ((x_points(p)-x_pix_val(1))/cal_x)+ x_or_val;
    y_values(p) = ((y_pix_val(1)- y_points(p))/cal_y)+ y_or_val;

    y_error_pos(p) = abs((((y_pix_val(1)- y_points_err_y_pos(p))/
cal_y)+ y_or_val)-y_values(p));
    y_error_neg(p) = abs((((y_pix_val(1)- y_points_err_y_neg(p))/
cal_y)+ y_or_val)-y_values(p));

    else
    x_values(p) = ((x_points(p)-x_pix_val(1))/cal_x)+ x_or_val;
    y_values(p) = ((y_pix_val(1)- y_points(p))/cal_y)+ y_or_val;
    end

end

%The data points are stored in this matrix depending on the type of
 error
%bars used here (if at all)
    if type == 1
    DATA = [transpose(x_values), transpose(y_values),
 transpose(x_error_pos), transpose(x_error_neg),
 transpose(y_error_pos), transpose(y_error_neg)];
    elseif type == 2
    DATA = [transpose(x_values), transpose(y_values),
 transpose(x_error_pos), transpose(x_error_neg)];
    elseif type ==3
    DATA = [transpose(x_values), transpose(y_values),
 transpose(y_error_pos), transpose(y_error_neg)];
    else
    DATA = [transpose(x_values), transpose(y_values)];
    end
```

Published with MATLAB® R2016b