Contents

- Clearing previous data
- Understanding the data
- Contours
- Now, finding it out for a line along the plane
- Plotting for the bottom plane
- Radially averaging the data
- Radial coordinate plotting of stress

```
% Code Summary:
% This code imports the data from a fix file (binning and averaging has
% already been done). After that, the data for a single plane is found.
% Contour plots are obtained from the x,z plane data as for each x, there
% is a corresponding z, hence the unique command has to be used. After
% that, I found it out for the lowest z>0 and found out relevant plots and
% properties on a single line. Then plotting was done for the bottomplane.
% Eventually, did the radial averaging along r and z, so that the entire
% domain can be summarised in a half-plane.
```

Clearing previous data

```
clc;
close all;
clear all;
```

```
fulldata = importdata("fix2.profile"); % This file contains the data of time averaged
data = fulldata.data;
clear fulldata
```

Understanding the data

```
x_og = data(:, 4); % Original coordinates
y_og = data(:, 3);
z_og = data(:, 2);

x = data(:, 4); % will modify later
y = data(:, 3);
z = data(:, 2);

fprintf("The minumum and maximum of x are %0.4f, %0.4f\n", min(x), max(x));
fprintf("The minumum and maximum of y are %0.4f, %0.4f\n", min(y), max(y));
fprintf("The minumum and maximum of z are %0.4f, %0.4f\n\n", min(z), max(z));
```

```
The minumum and maximum of x are -0.0515, 0.0515
The minumum and maximum of y are -0.0515, 0.0515
The minumum and maximum of z are -0.0100, 0.1200
```

For extraction along the Y = 0 plane

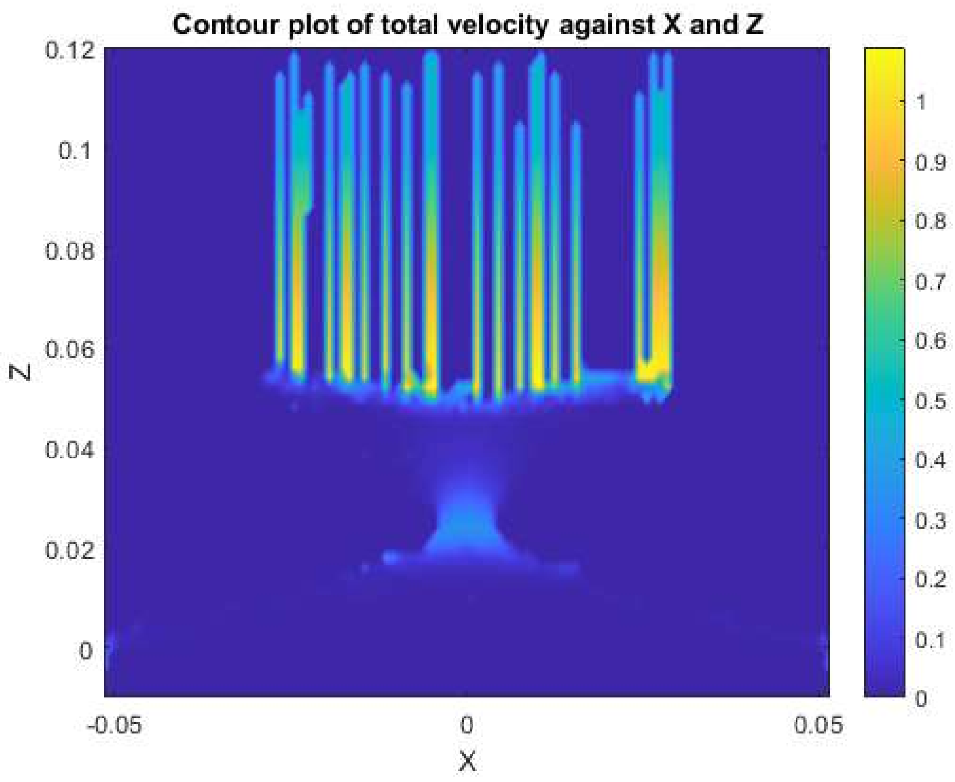
```
yind = (y_og==min(y(y>0))); % as already binned data, yindex (yind) can be taken for a certain/specific coordinate
planedata = data(yind == 1, :); % got the data for points at the specific coordinate

% save plane.mat planedata
% On uncommenting the above line, can save the data on a plane in a mat
% file

x = planedata(:, 4); % Now, x,y,z contains data for the plane
y = planedata(:, 3);
z = planedata(:, 2);
Ncount = planedata(:, 5);
vx = planedata(:, 6);
vy = planedata(:, 7);
vz = planedata(:, 8);
vs = planedata(:, 9);

fprintf("The minumum and maximum of x are %0.4f, %0.4f\n", min(x), max(x));
fprintf("The minumum and maximum of y are %0.4f, %0.4f\n", min(y), max(y)); % both are same
fprintf("The minumum and maximum of z are %0.4f, %0.4f\n\n", min(z), max(z));
```

```
The minumum and maximum of x are -0.0515, 0.0515
The minumum and maximum of y are 0.0005, 0.0005
The minumum and maximum of z are -0.0100, 0.1200
```

Contours

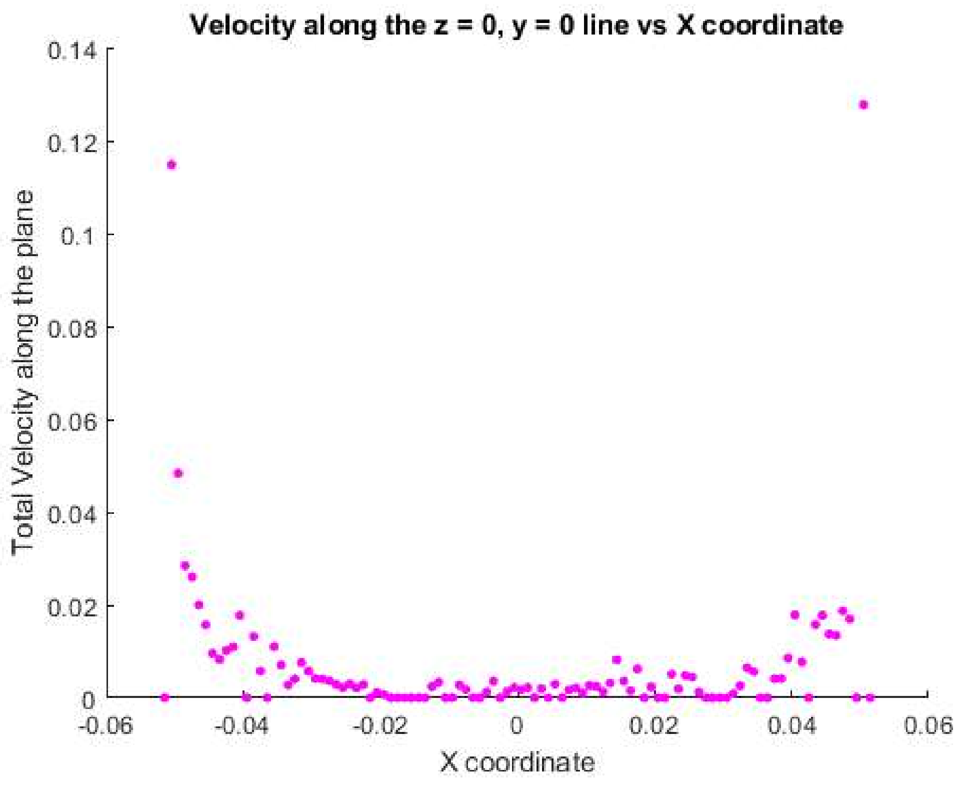
```
x_un = unique(x);
z_un = unique(z);
% for each x, we have multiple z, and vice versa, so we only find it out
% for the unique x, as contour plots are expected

fprintf("The size of x_un is %0.4f\n", length(x_un));
fprintf("The size of z_un is %0.4f\n\n", length(z_un));

[X, Z] = meshgrid(x_un, z_un);
sh = size(X);
fprintf("The size of X and Z is %0.1d, %0.1d\n", sh(1), sh(2));

vel = reshape(vs', length(z_un), length(x_un));
vel2 = reshape(vs, length(x_un), length(z_un));
vel2 = vel2'; % Matrix changes made due to reasons as told

close all;
figure;
levels = [0:.01:1.092]; % changes made for better portrayal/visualization of contours
contourf(X, Z, vel2, 5, 'LineColor', 'none', 'LevelList', levels)
% set(gca, 'CLim',[-1 0.2]); % set CLim to be the range of most of your data
% contours require value at each (x,z) point interval in the domain, hence
% meshgrid is used

% shading flat
xlabel("X")
ylabel("Z")
title("Contour plot of total velocity against X and Z")
colorbar
```

```
The size of x_un is 104.0000
The size of z_un is 66.0000

The size of X and Z is 66, 104
```



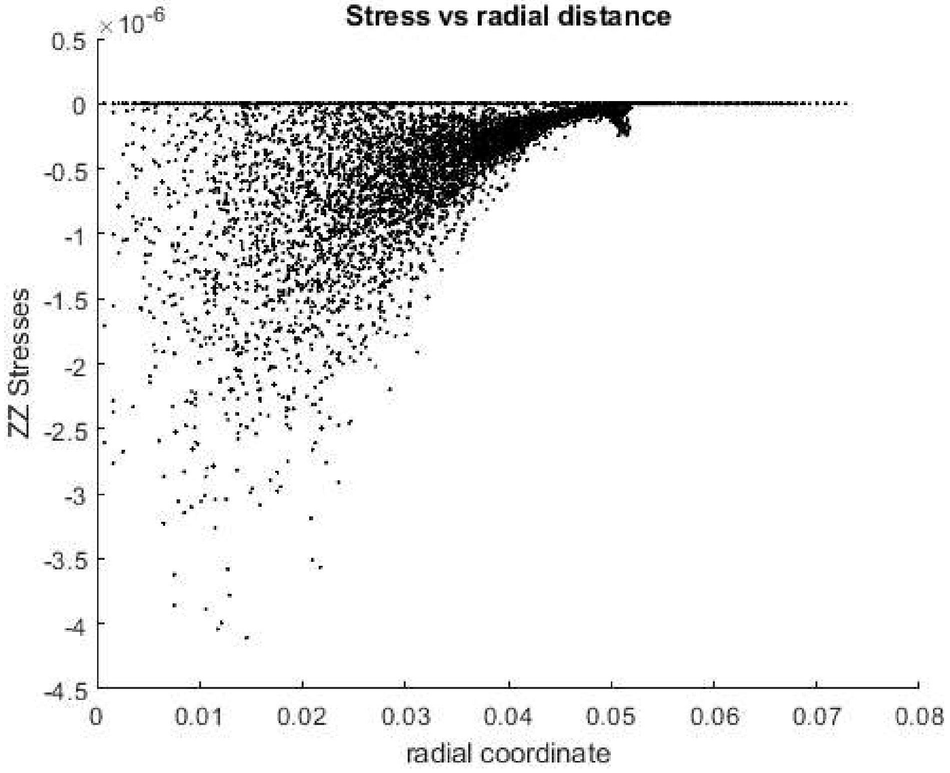
Now, finding it out for a line along the plane

```
linz = linspace(0, max(z), 100); % z already contains the data along a plane
% zind = (z < linz(2));
% zind = (z == -1.73472000000000e-18);
zind = (z == min(z(z>0))); % we only consider the z > 0 and then take the minimum of them

linedata = planedata(zind == 1, :); % from the plane data we are only considering a line
x_pl = linedata(:, 4);
vs_pl = linedata(:, 9);
z_pl = linedata(:, 2);

% So, here: y = 0.0005 and 0 < z < 1.2mm
% So, we have 624 points in this cuboid basically but it is so small that
% we call it to be a line

figure
scatter(x_pl, vs_pl, 15, "magenta", "filled", "o")
xlabel("X coordinate")
ylabel("Total Velocity along the plane")
title("Velocity along the z = 0, y = 0 line vs X coordinate")
```



Plotting for the bottom plane

```
% Plotting the data for the bottom plane, so z is the minimum positive
% number

zind1 = (z_og == min(z(z>0)));
planedata1 = data(zind1 == 1, :);

xbot = planedata1(:, 4); % xbot contains the bottom plane data
ybot = planedata1(:, 3);
zbot = planedata1(:, 2);
Ncountbot = planedata1(:, 5);
vxbot = planedata1(:, 6);
vybot = planedata1(:, 7);
vzbot = planedata1(:, 8);
vsbot = planedata1(:, 9);
tauzz = planedata1(:, 17);

[B, I] = sort(sqrt(xbot.^2 + ybot.^2), "ascend");
figure
scatter(B, tauzz(I), 1, "black");
xlabel("radial coordinate")
ylabel("ZZ Stresses")
title("Stress vs radial distance")
```



Radially averaging the data

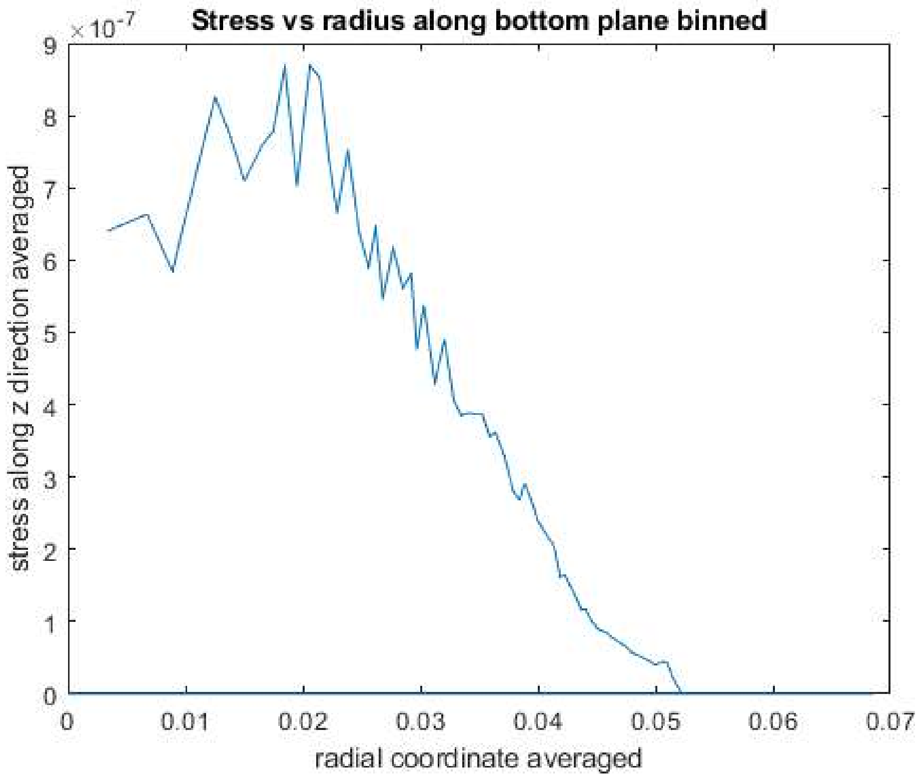
So, the system is axisymmetric. I will now average the data radially and also along z axis, so that we can have even better and averaged data, which is of almost 1/10th size x_og and y_og consist of my original x and z coordinates

```
radial_coordinate = sqrt(x_og.^2 + y_og.^2);
radius_unique = unique(radial_coordinate);
z_uni = unique(z_og);
fprintf("The size of radco is %0.1d, %0.1d\n", size(radial_coordinate, 1), size(radial_coordinate, 2));
fprintf("The size of uniq(radco) is %0.1d, %0.1d\n", size(radius_unique, 1), size(radius_unique, 2));
fprintf("The size of uniq(z) is %0.1d, %0.1d\n", size(z_uni, 1), size(z_uni, 2));

num1 = size(radius_unique, 1);
num2 = size(z_uni, 1);
radially_averaged_data = zeros(num1*num2, 20 + 1); % 20 columns for the existing columns and 1 extra in which I
% store the radial coordinate

% Used the following 2 lines to validate the find command
% radco = sort(radco);
% find(radco == radco(1))

k=1;
for i=1:num1

    samedist = find(radial_coordinate == radius_unique(i)); % get the indices of unique radii

    for j=1:num2

        samez = find(data(:, 2) == z_uni(j)); % get the indices of unique z
        common = intersect(samedist, samez); % get the coordinates of common radial distance and z
        % The intersect command will give me the common data points which
        % correspons to rows
        radially_averaged_data(k, 1:20) = mean(data(common, :)); % take mean of the common data points
        radially_averaged_data(k, 21) = radius_unique(i); % also saving the radius for ease
        k = k + 1; % update counter of row index of radially_averaged_data

    end

end

bottomdata = radially_averaged_data(radially_averaged_data(:, 2) == min(z(z>0)), :);
% Get the bottom plane's data in the array named bottomdata. Can save it using
% save bottom_radially_averaged_data.mat bottomdata
```

```
The size of radco is 713856, 1
The size of uniq(radco) is 1108, 1
The size of uniq(z) is 66, 1
```

Radial coordinate plotting of stress

```
twobottom = [bottomdata(:, 21), bottomdata(:, 17)];
[~, I] = sort(twobottom(:, 1), "ascend");
twobottom = twobottom(I, :);
lessbottom = zeros(ceil(size(bottomdata, 1)/2), 2);

j = 1;
for i=1:ceil(size(bottomdata, 1)/10) + 10
    if j <=size(bottomdata, 1)-10
    lessbottom(i, :) = mean(twobottom(j:j+10, :));
    j = j + 10;
    end
end

figure;
% scatter(lessbottom(:, 1), lessbottom(:, 2), 3, "black")
plot(lessbottom(:, 1), -1*lessbottom(:, 2))
xlabel("radial coordinate averaged")
ylabel("stress along z direction averaged")
title("Stress vs radius along bottom plane binned")
```



Published with MATLAB® R2023a