Part 1a Scientific Computing
Assessed Exercise 3

Tudor Mihai Avram

Homerton College, tma33@cam.ac.uk

1 SECTION 1

1.1 *Task 1 : Isotopic case*

For this task, I chose $k_x = k_y = k_z = 0.1$, $x = y = z = 10$ and $v_x = v_y = v_z = -0.4$. The resulting graph had the shape of an elipse, as can be illustrated in **Figure 1**.

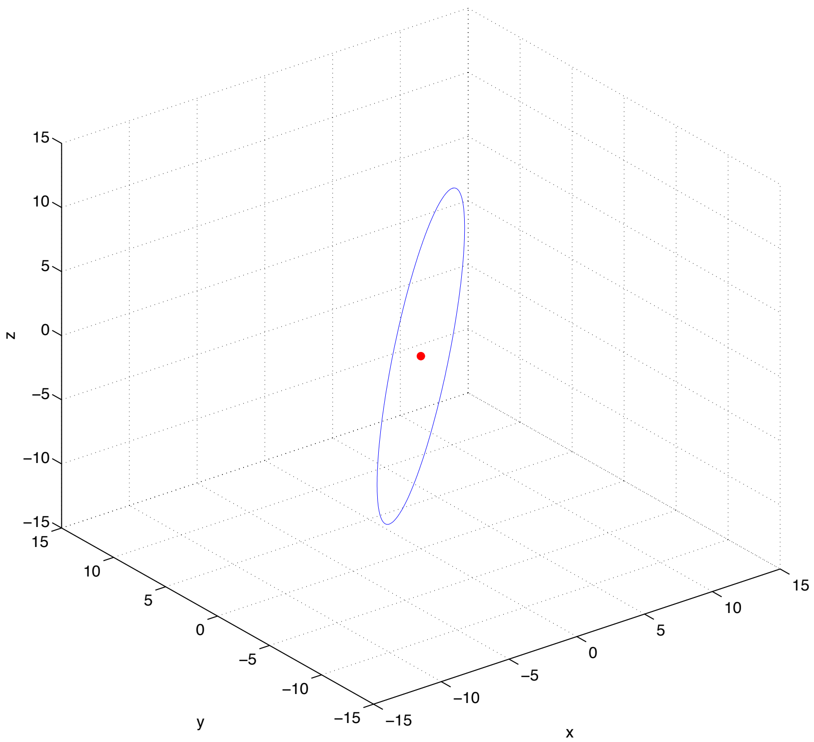

Figure 1 : As can be observed, the trajectory of the particle is an elipse, centred in the origin set by the initial conditions.

1.2 *Task 2 : 2D anisotropic*

In this case, I changed the 3 spring constants to the following values, so that they are now different : $k_x = 3$, $k_y = 4$, $k_z = 5$. I also set $v_z = z = 0$, keeping $x = y = 10$ and $v_x = v_y = -0.4$. As expected, the resulting trajectory is now just in two dimensions, not having an z component.

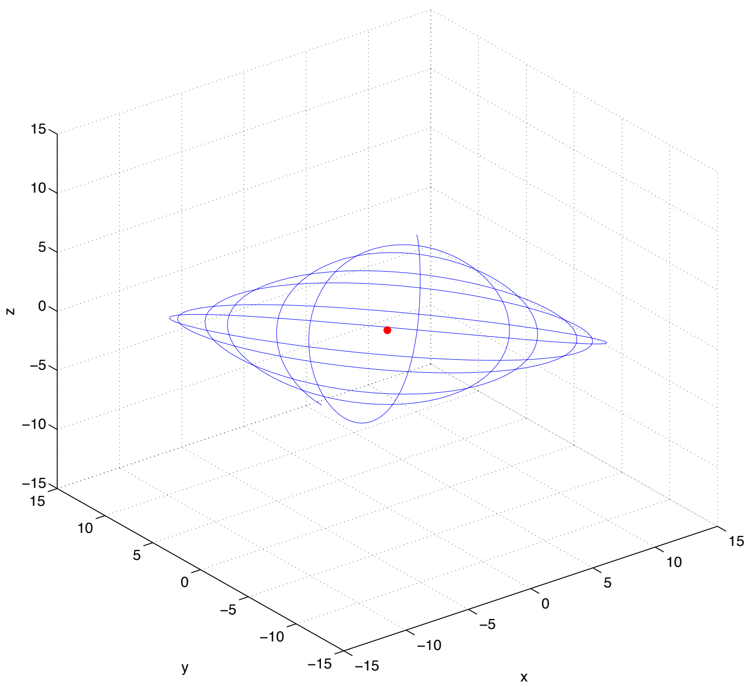

Figure 2 : This time, due to the new conditions, the particle's trajectory is restricted to a rectangle in the xy plane.

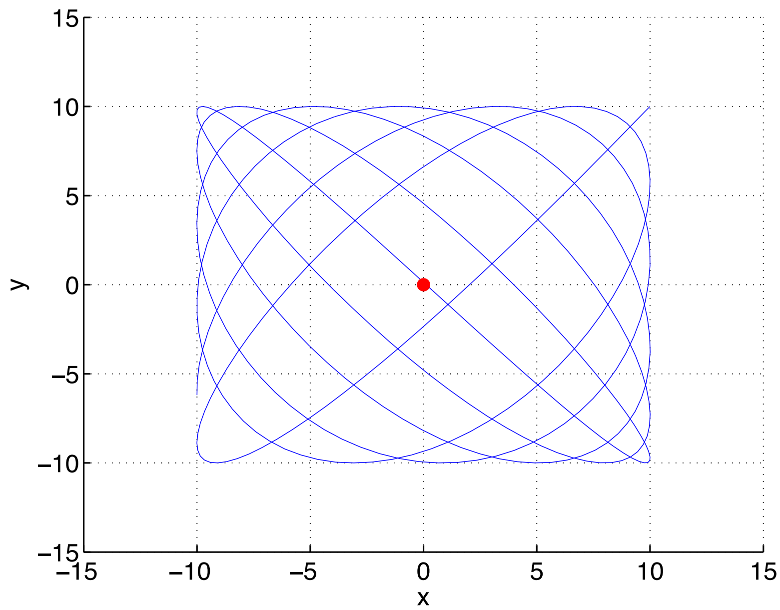

Figure 3 : The trajectory from **Figure 2**, but projected on the xy plane.

1.3 *Task 3 : Full 3D anisotropic*

The values chosen for this task were :

- $k_x = 3$, $k_y = 4$, $k_z = 6$
- $x = 6$, $y = 7$, $z = 8$
- $v_x = 3$, $v_y = 4$, $v_z = 5$

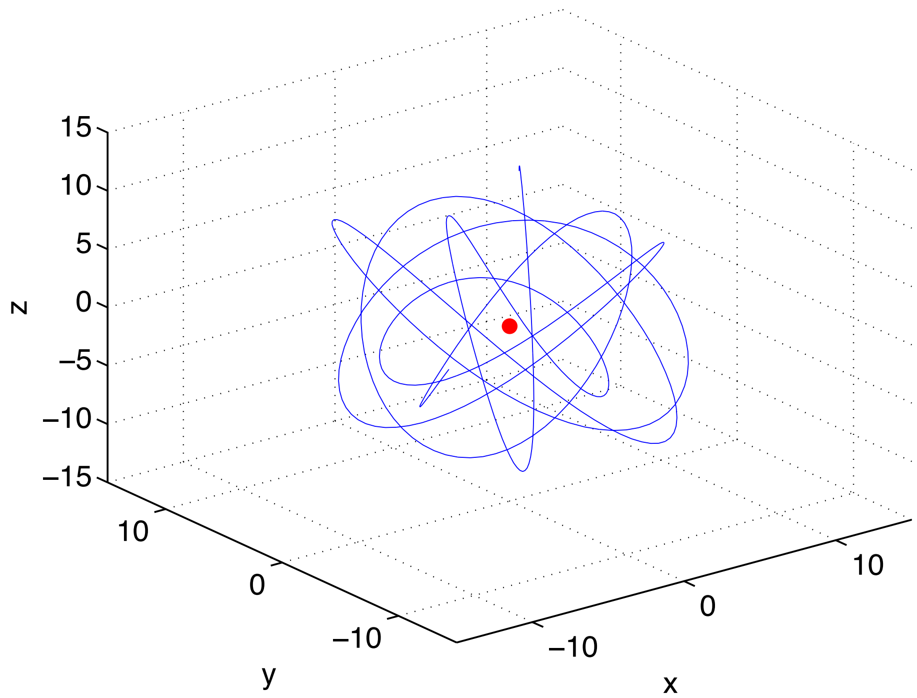

Figure 4 : A full 3D illustration of the particle's trajectory in a general anisotropic harmonic potential.

By projecting the trajectory illustrated in **Figure 4** on the xy, yz, or xz plane, we observe that it is constrained within a parallelogram. This fact can also be seen in **Figure 5**

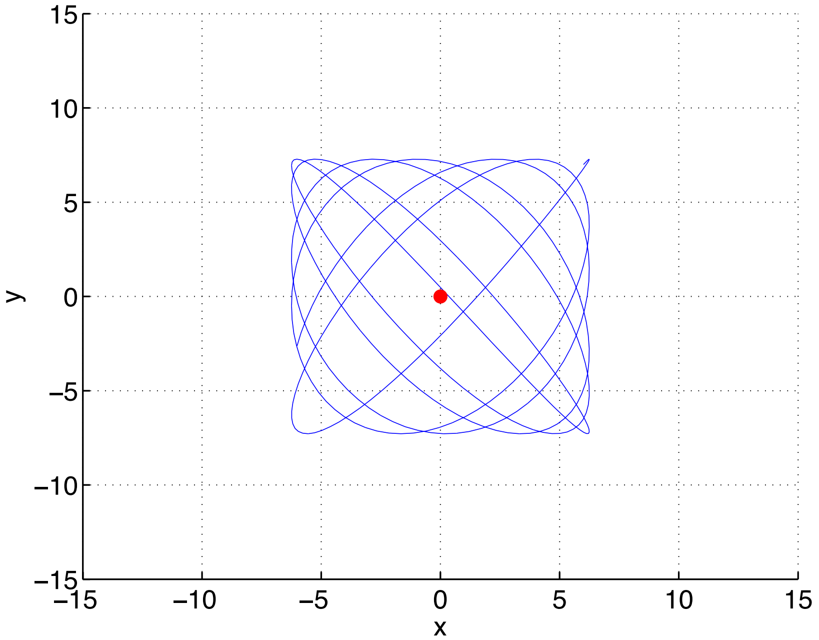

Figure 5 :The trajectory from **Figure 4**, projected on the xy plane

1.4 *Task 4 : Quality of numerics*

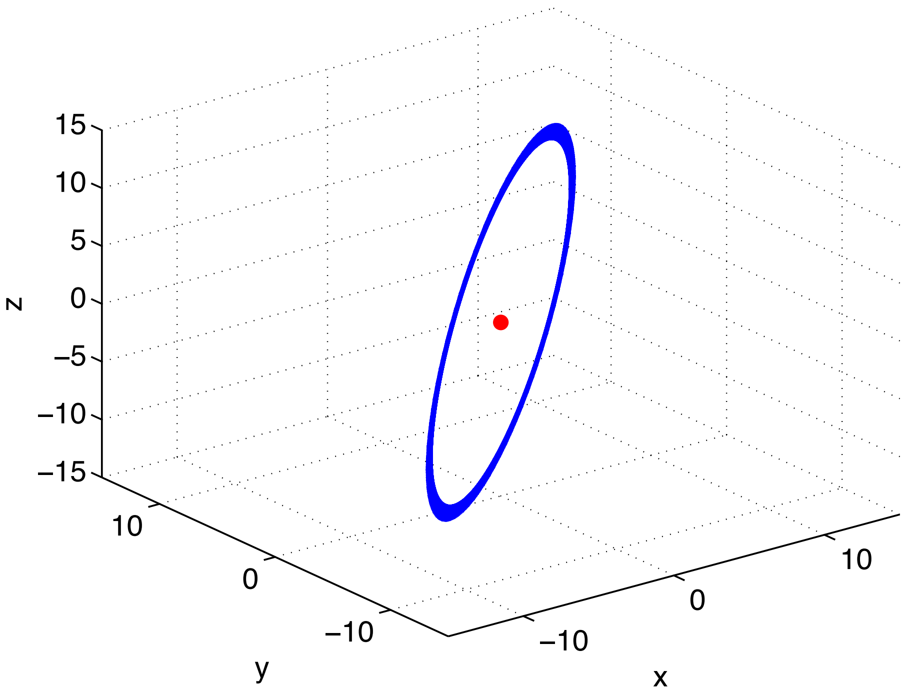

Figure 6 : This picture illustrates particle's trajectory in the isotropic case, with the following conditions : $k_x = k_y = k_z = 4$, x=12, y = 10 and y = 8, v_x = -6, v_y = -2 and v_z = 10. Also, I changed the value of dt to 0.4. As we can see in the picture, the resulting trajectory for this conditions diverges. Therefore, we can deduce that the particle does not retrieve it's path when it passes through a point for the second time.

2 SECTION 2

2.1 Orbitals 3D plots

2.1.1 *px(x,y,0) :*

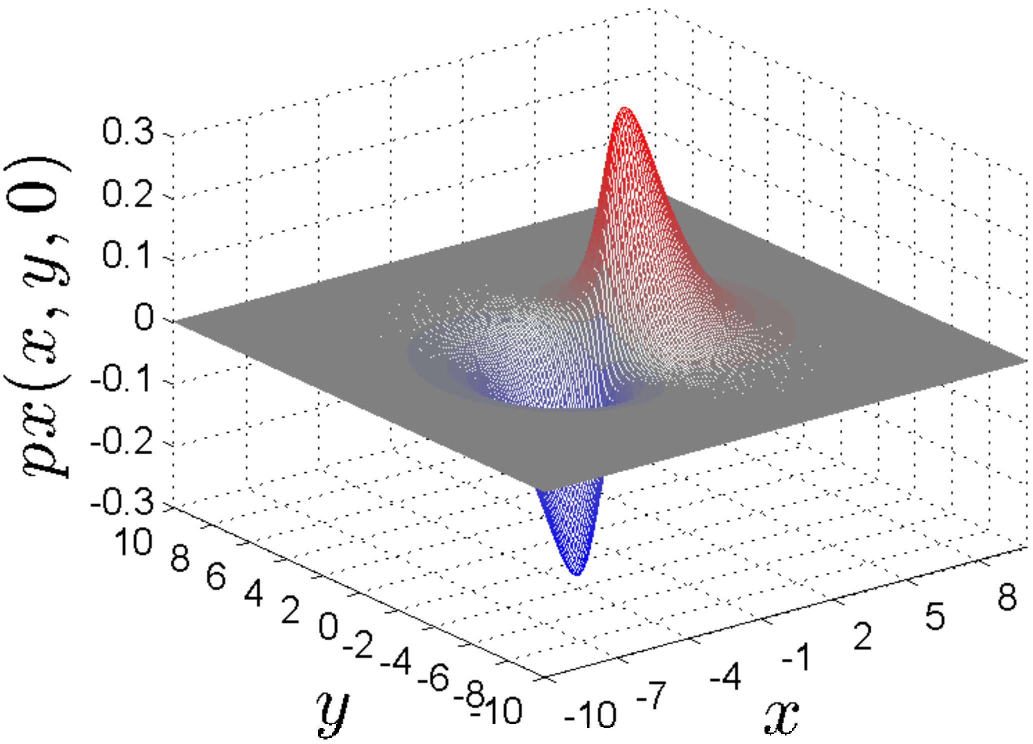

Figure 7 : 3D Graph showing the wavelength of the atomic orbital px(x,y,0)

2.1.2 *dxy(x,y,0)* :

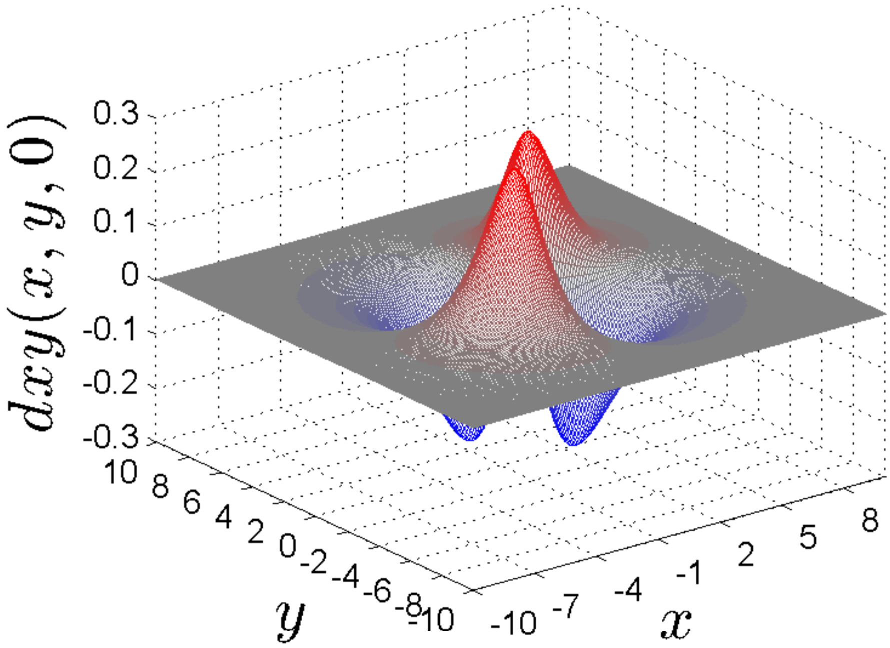

Figure 8 : 3D Graph showing the wavelength of the atomic orbital dxy(x,y,0)

2.2 Orbitals 2D contour

2.2.1 *px(x,y,0) :*

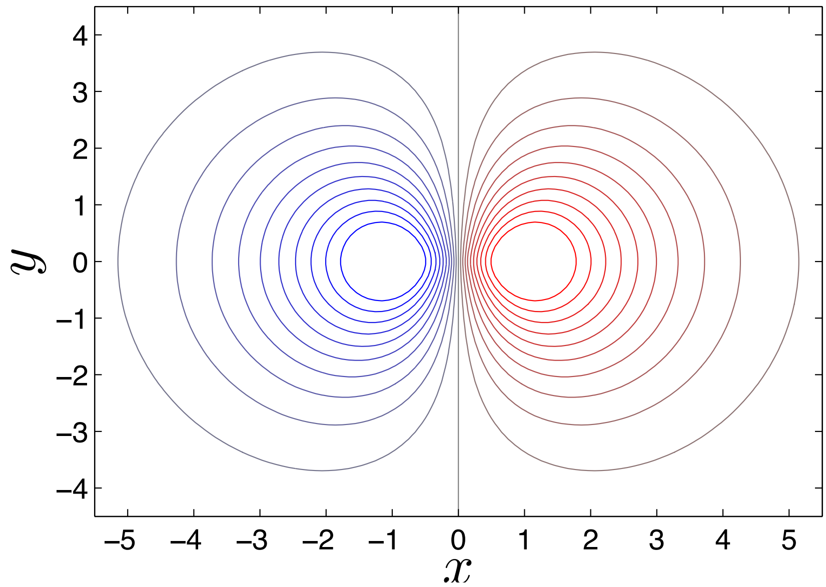

Figure 9 : 2D contour of the px(x,y,0) orbital. The blue lines represent negative values, while the red ones are positive values. The spacing between the lines is of 0.03 units.

2.2.2 *dxy(x,y,0) :*

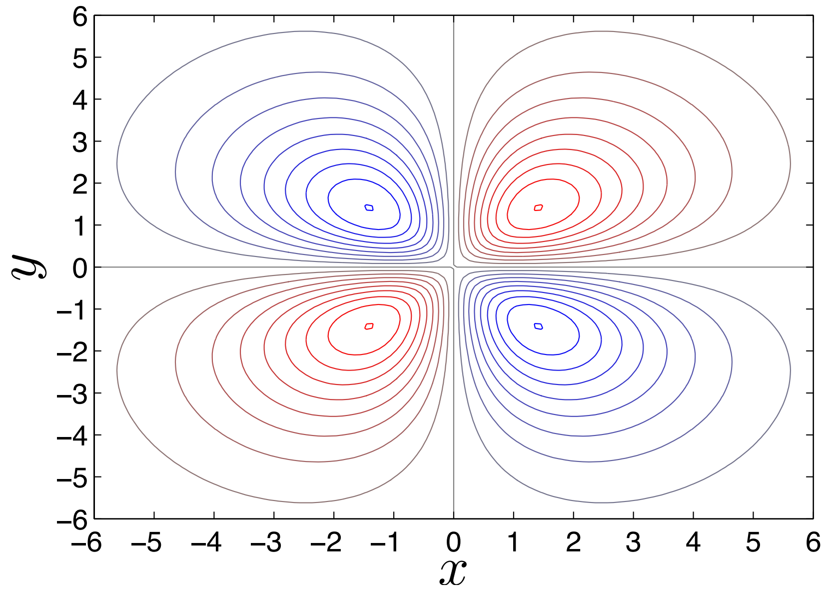

Figure 10 : 2D contour of the dxy(x,y,0) orbital. The blue lines represent negative values, while the red ones are positive values. The spacing between the lines is of 0.03 units.

2.3 dxy orbital along x = y

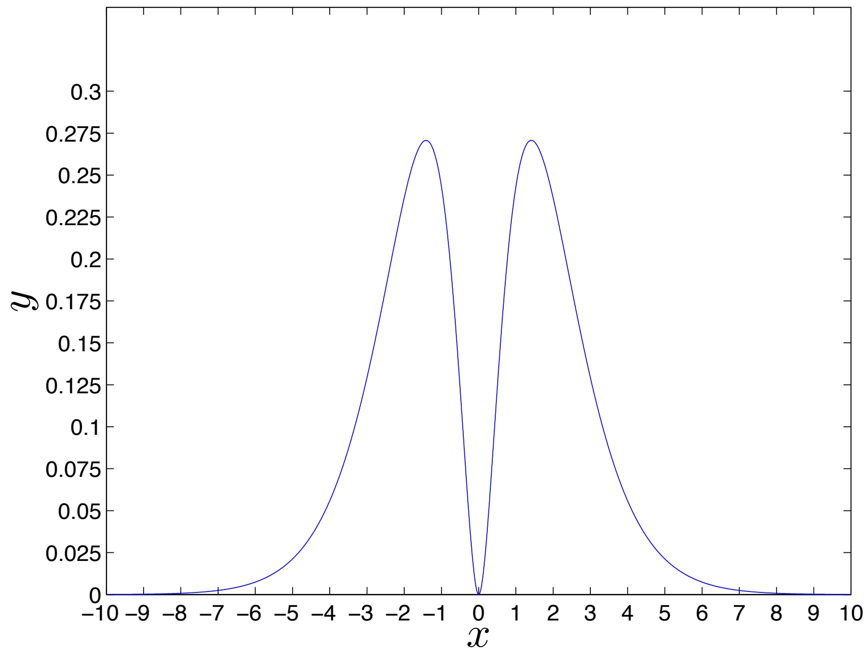

Figure 11 : The dxy(x,y,0) orbital ploted along the x = y diagonal. We can observe that it has two maximums at x= -2 and x=2 and one minimum at x=0.

2.4 The differential of the dxy orbital

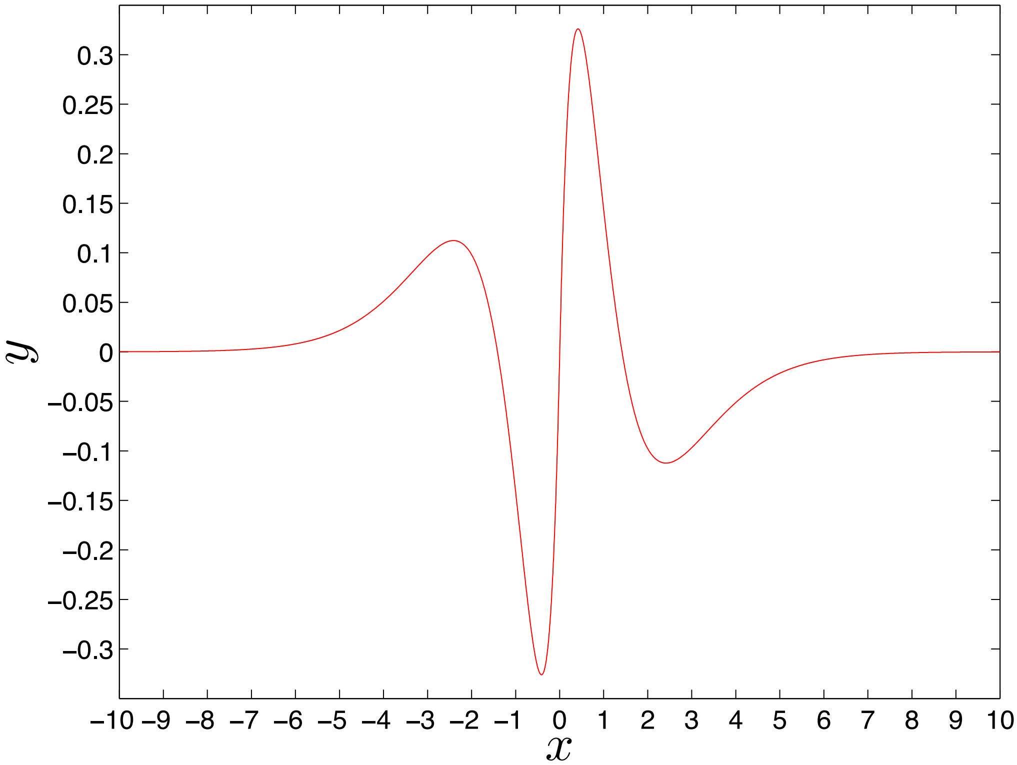

Figure 12 : The differential of the dxy(x,y,0) orbital. We can observe that it changes signs at x=-2, x=0 and x=2, fact that supports the maximums and the minimum seen in **Figure 11.**

---

3 ***APPENDIX : Matlab code***

3.1 *pxOrbital.m* : the function that computes the values of px(x,y,z) :

```
% function for computing the value of px
function w = pxOrbital(x, y, z)
   w = x.*exp(-sqrt(x.^2 + y.^2 + z.^2));
end
```

3.2 *dxyOrbital.m :* the function that computes the values for dxy(x,y,z) :

```
% function that computes the value for the dxy orbital
function w = dxyOrbital(x, y, z)
    w = x.* y.*exp(-sqrt(x.^2 + y.^2 + z.^2));
end
```

3.3 *AssessedTask3.m* : the main code for making the graphs and computing the auxiliary values :

```
% Creating the colormap for the mesh and contour
mult = 0.025;
for i=0:40
    colorMap(i+1, 1) = i * mult;
    colorMap(i+1, 2) = (20 - abs(20 - i)) * mult;
    colorMap(i+1, 3) = (40 - i) * mult;
end

% Surface discratization

plotSpace = linspace(-10, 10, 200);
[x, y] = meshgrid(plotSpace,plotSpace);

pxOrb = pxOrbital(x, y, 0); % values of px(x,y,0)

figure1 = figure;

mesh(x,y,pxOrb);
gca1 = gca;
colormap(colorMap);
% Setting the axis limits
xlim([-10 10]);
ylim([-10 10]);
zlim([-0.3 0.3]);
% Setting the graphic & labels
set(figure1,'Units','centimeters');
set(figure1,'PaperUnits','centimeters');
set(figure1,'PaperPosition',[0 0 15 10]);
set(gca1,'fontsize', 13, 'fontname', 'arial');
set(gca1,'XTick',[-10:3:10]);
set(gca1,'YTick',[-10:2:10]);
set(gca1,'ZTick',[-0.3:0.1:0.3]);
xlabel('$x$','fontsize',25,'interpreter','latex');
ylabel('$y$','fontsize',25,'interpreter','latex');
zlabel('$px(x,y,0)$','fontsize',25,'interpreter','latex');

% doing the same thing for dxy(x,y,0)

dxyOrb = dxyOrbital(x,y,0);
figure2 = figure;
```

```
mesh(x,y,dxyOrb);
gca2 = gca;
colormap(colorMap);
% Setting the axis limits
xlim([-10 10]);
ylim([-10 10]);
zlim([-0.3 0.3]);
% Setting the graphic & labels
set(figure2,'Units','centimeters');
set(figure2,'PaperUnits','centimeters');
set(figure2,'PaperPosition',[0 0 15 10]);
set(gca2,'fontsize', 13, 'fontname', 'arial');
set(gca2,'XTick',[-10:3:10]);
set(gca2,'YTick',[-10:2:10]);
set(gca2,'ZTick',[-0.3:0.1:0.3]);
xlabel('$x$','fontsize',25,'interpreter','latex');
ylabel('$y$','fontsize',25,'interpreter','latex');
zlabel('$dxy(x,y,0)$','fontsize',25,'interpreter','latex');


% making the contour for px(x,y,0)

figure3 = figure;

contour(x,y,pxOrb,[-0.3 : 0.03 : 0.3]);
gca3 = gca;
colormap(colorMap);
% Setting the axis limits
xlim([-5.5 5.5]);
ylim([-4.5 4.5]);
% Setting the graphic & labels
set(figure3,'Units','centimeters');
set(figure3,'PaperUnits','centimeters');
set(figure3,'PaperPosition',[0 0 15 10]);
set(gca3,'fontsize', 13, 'fontname', 'arial');
set(gca3,'XTick',[-5:1:5]);
set(gca3,'YTick',[-5:1:5]);
xlabel('$x$','fontsize',25,'interpreter','latex');
ylabel('$y$','fontsize',25,'interpreter','latex');


% making the contour for dxy(x,y,0)

figure4 = figure;

contour(x,y,dxyOrb,[-0.3 : 0.03 : 0.3]);
gca4 = gca;
colormap(colorMap);
% Setting the axis limits
xlim([-6 6]);
ylim([-6 6]);
% Setting the graphic & labels
set(figure4,'Units','centimeters');
set(figure4,'PaperUnits','centimeters');
set(figure4,'PaperPosition',[0 0 15 10]);
set(gca4,'fontsize', 13, 'fontname', 'arial');
```

```
set(gca4,'XTick',[-6:1:6]);
set(gca4,'YTick',[-6:1:6]);
xlabel('$x$','fontsize',25,'interpreter','latex');
ylabel('$y$','fontsize',25,'interpreter','latex');

% calculating and ploting d(x,x,0) against x
x=[-10:0.005:10];
dxxOrb = dxyOrbital(x,x,0);

figure5 = figure;
plot(x,dxxOrb,'-b');
gca5 = gca;
% axis limits
xlim([-10,10]);
ylim([0,0.35]);
% Graphics & labels
set(figure5,'Units','centimeters');
set(gca5,'fontsize', 13, 'fontname', 'arial');
set(gca5,'XTick',[-10:1:10]);
set(gca5,'YTick',[0:0.025:0.3]);
xlabel('$x$','fontsize',25,'interpreter','latex');
ylabel('$y$','fontsize',25,'interpreter','latex');

%calculating the differential
ddxx = diff(dxxOrb);
ddxx = ddxx./(0.005);

x = [-9.995:0.005:10];

% ploting the differential
figure6 = figure;
plot(x,ddxx,'-r');

gca6 = gca;
% axis limits
xlim([-10,10]);
ylim([-0.35,0.35]);
% Graphics & labels
set(figure5,'Units','centimeters');
set(gca6,'fontsize', 13, 'fontname', 'arial');
set(gca6,'XTick',[-10:1:10]);
set(gca6,'YTick',[-0.3:0.05:0.3]);
xlabel('$x$','fontsize',25,'interpreter','latex');
ylabel('$y$','fontsize',25,'interpreter','latex');
```