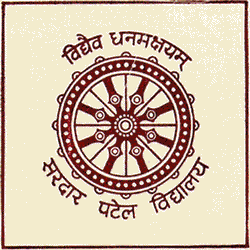
**Project Report**

**on**

**Conics with Matplotlib**



**Submitted By**

Vinayak Bector

**Under the Guidance of**

Ms. Angel Panesar

Department of Computer Science

Sardar Patel Vidyalaya

Lodhi Estate, New Delhi 110003

**CERTIFICATE**

This is to certify that Vinayak Bector Of Class XI D has prepared the report on the Project entitled “Conics with Matplotlib”. The report is the result of his efforts & endeavors. The report is found worthy of acceptance as final project report for the subject Computer Science of Class XI. He has prepared the report under my guidance.

Ms. Angel Panesar

Department of Computer Science

Sardar Patel Vidyalaya

Lodhi Estate, New Delhi 110003

**DECLARATION**

I hereby declare that the project work entitled “Conics with Matplotlib”, submitted to Department of Computer Science, Sardar Patel Vidyalaya, Lodhi Estate, New Delhi 110003 is prepared by me. The project work is result of my personal efforts.

Vinayak Bector

Class: XI D

**ACKNOWLEDGEMENT**

I would like to express my special thanks and sincere gratitude to my teacher Angel Ma’am who gave me the golden opportunity to do this wonderful project on the topic Conics with Matplotlib.This topic has helped me in doing a lot of Research and I came to know about so many new libraries, modules, functions, tips and tricks. I am really thankful to her.

Secondly I would also like to thank my parents, brother (Kartikeya Bector) and Friend (Anirudh Kumar) who helped me a lot in finalizing this project within the limited time frame.

Vinayak Bector

Class: XI D

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| S. No. | Topic | Page No. |
| 1. | [Libraries Used](#lib) | 5 |
| 2. | [Working Description](#Wd) | 6-8 |
| 3. | [Project Code](#Pc) | 9 |
| 4. | [Output Scene](#os) | 25 |
| 5. | [Conclusion](#conc) | 32 |
| 6. | [Bibliography](#bb) | 33 |

**LIBRARIES USED**

**MATPLOTLIB**

**Matplotlib** is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

One of the greatest benefits of visualization is that it allows visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

**Python Numpy**

**Numpy**is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python.  
Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data.

**WORKING DESCRIPTION**

Owing to my love for graphs I was introduced to this amazing world of Conic Sections, commonly known Conics. As I explored, pondered and discovered new and quite exciting stuff in maths, I eventually decided to take a very bold step and Extrapolate this interest of mine, in my CS project.

It started as a parametric Equation of circle, which could be printed using Trig Ratios in Matplotlib. Then slowly I watched a few lessons and courses on Youtube to expand my knowledge and understanding of the topic. With multiple drafts and code reviews, and help from all my well-wishers I was able to dexterously craft a 250 line code, with 3 nested while loops and multiple if loops within it.

The programme offers you many different choices.

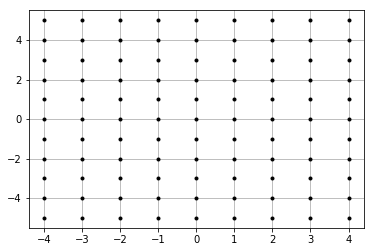
First, it introduces you with a Menu, bombarding you with multiple options, so that you can print your favourite conic section.

I have used many modules in my code, a brief description of each is provided as follows :

**Numpy Meshgrid function**

The numpy.meshgrid function is used to create a rectangular grid out of two given one-dimensional arrays representing the Cartesian indexing or Matrix indexing. Meshgrid function is somewhat inspired from MATLAB.

Consider the above figure with X-axis ranging from -4 to 4 and Y-axis ranging from -5 to 5. So there are a total of (9 \* 11) = 99 points marked in the figure each with a X-coordinate and a Y-coordinate. For any line parallel to the X-axis, the X-coordinates of the marked points respectively are -4, -3, -2, -1, 0, 1, 2, 3, 4. On the other hand, for any line parallel to the Y-axis, the Y-coordinates of the marked points from bottom to top are -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5. The numpy.meshgrid function returns two 2-Dimensional arrays representing the X and Y coordinates of all the points.



**Numpy Linspace**

The NumPy linspace function (sometimes called np.linspace) is a tool in Python for creating numeric sequences.

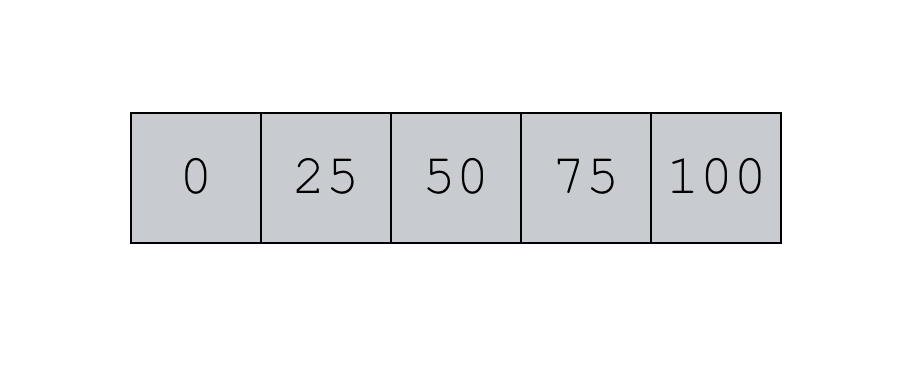
The NumPy linspace function creates sequences of evenly spaced values within a defined interval.

Essentally, you specify a starting point and an ending point of an interval, and then specify the total number of breakpoints you want within that interval (*including* the start and end points). The np.linspace function will return a sequence of evenly spaced values on that interval.

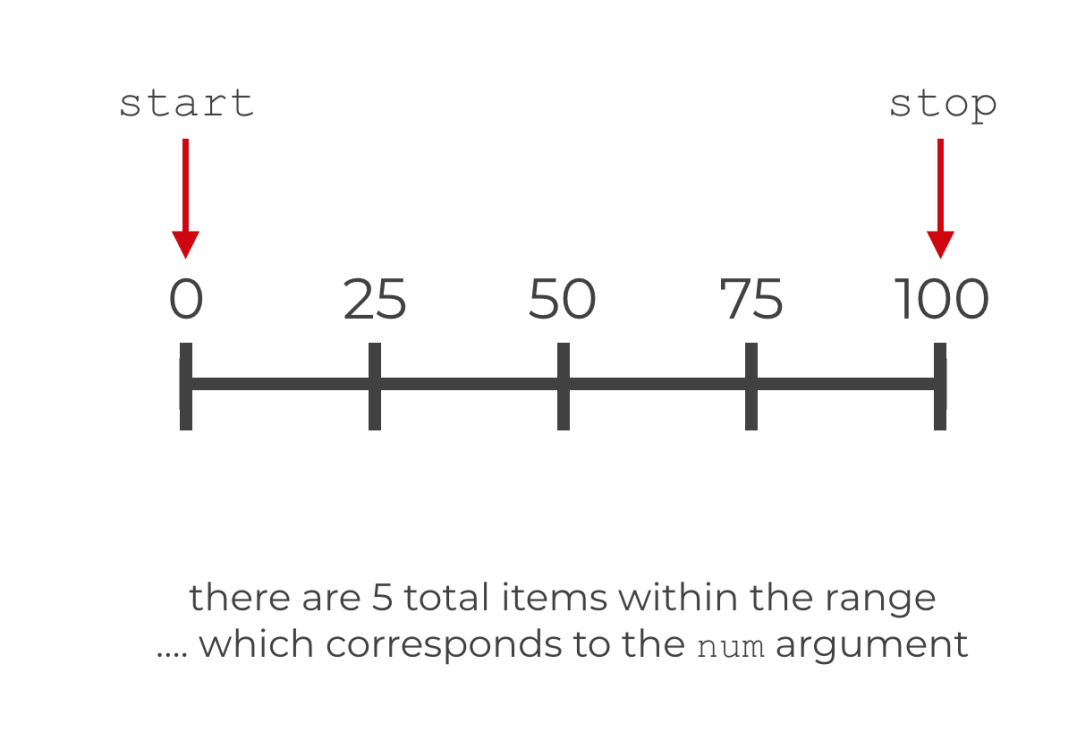
To illustrate this, here’s a quick example. (We’ll look at more examples later, but this is a quick one just to show you what np.linspace does.)

np.linspace(start = 0, stop = 100, num = 5)

This code [produces a NumPy array](https://www.sharpsightlabs.com/blog/numpy-array-python/) that looks like the following:



That’s the ndarray that the code produces, but we can also visualize the output like this:



**Axhline:**

**Matplotlib.pyplot.axhline (Value)**

Literally means an axis line (straight line) with is Horizontal in nature; this is where ‘h’ in axhline is derived from.

Similarly

**Axvline:**

**Matplotlib.pyplot.axvline (Value)**

Produces a vertical line on the grid(i.e graph output).

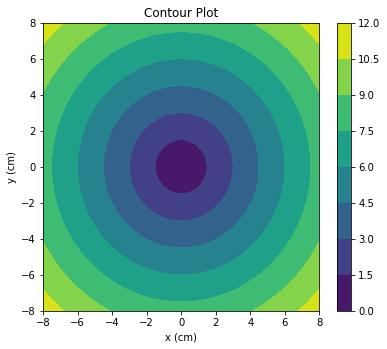
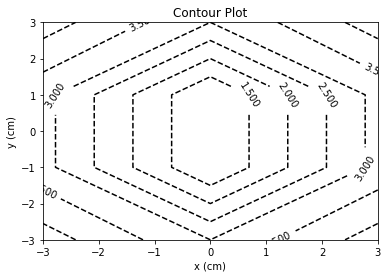
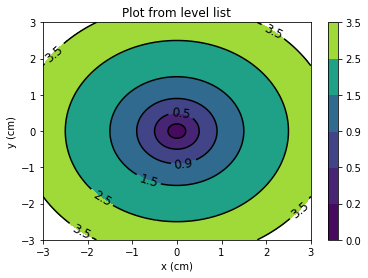
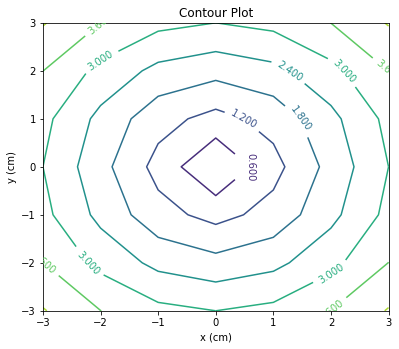
**Contour:**

**Contour function**, which is usually used to represent a 3-d array on a 2-d platform.

In the code you’ll observe that I used the meshgrid X and Y coordinates and in place of Z coordinates, the equation of conic. The Z coordinate array, in this case the equation of conic, has a parameter [Value, to satisfy the equation] to make the equation of conics mathematically sound.

Contour has many benefits, first and foremost being the ease with which you can print complex graphs, along with features like adding colors and label lines with different altitudes/ heights (called contour).

This will also help us in the future to add multiple conics on the same graphs, with different altitudes to add a parallax effect, which will help us, differentiate these graphs with ease. Some contour output is as follows:



**PROJECT CODE**

# Importing all the required Libraries

#Importing matplotlib

import matplotlib.pyplot as mpl

#Impoting numpy

import numpy as np

#Defining values for x and y

x = np.linspace(-50, 51, 200)

y = np.linspace(-50, 51, 200)

#meshgrid makes an array, this is useful in defining functions

x, y = np.meshgrid(x, y)

#Axes() as a function, this will highlight our origin

def axes():

mpl.axhline(0, alpha= .2, linewidth= 2, color='k' )

#Printing Horizontal line, X axis

mpl.axvline(0, alpha= .2, linewidth= 2, color='k' )

#Printing Vertical line, Y axis

#mpl.axis('equal')

#For taking valid inputs as values, defining a f(n)

def options(prompt):

while True:

try:

#Ensuring the input to be integer

number = int(input(prompt))

break

except ValueError :

#Printing Output for correction, Giving user another chance to enter value

print('Please enter a Valid input ')

pass

return number

def delta(a,b,c,f,g,h):

if a\*(b)\*c +2\*f\*g\*h -a\*f^2 -b\*g^2 -c\*h^2 != 0:

return True

else :

return False

while True :

print('Conic section Graphs')

print('Menu \n 1. Parabola \n 2. Ellipse and circles \n 3. Hyperbolas \n 4.EXIT')

opts = options(' Please enter any one of the options ')

if opts == 1:

while True:

print('PARABOLA')

print("MENU \n 1. Parabola in standard equation \n 2. Parabola in Expanded equation \n 3.Return to Menu")

ch = int(input('Choose an option'))

if ch == 1:

print('1. Parabola in standard equation')

a = float(input('Please enter the value of \'a\' ' ))

parabola = (y\*\*2 - 4\*a\*x)

print(' Parabolas in standard eqn are of the form \n y^2 = 4ax ')

print('Do you Want to see directrix and focus as well ? ')

duff = input (' y for YES and n for NO ')

if duff == 'y':

axes()

# Printing Focus as a DOT.

mpl.plot(a, 0, '.')

# Printing Directrix, using a vertical line axvline.

mpl.axvline(-a)

mpl.contour(x, y, parabola, [0], colors='r' )

mpl.show()

else :

axes()

mpl.contour(x, y, parabola, [0], colors='k')

mpl.show()

elif ch == 2:

while True:

print('2. Parabola in Expanded equation')

print('Parabolas in non-standard eqn are of the form :')

print('Ax^2+2Hxy+By^2+2Gx+2Fy+C=0, \n where H^2−4AB=0')

a = options(' Please enter the Value of A ')

b = options(' Please enter the Value of B ')

c = options(' Please enter the Value of C ')

f = options(' Please enter the Value of 2F ')

g = options(' Please enter the Value of 2G ')

h = options(' Please enter the Value of 2H ')

if delta(a,b,c,f,g,h) and (h)\*\*2 - 4\*a\*b == 0 :

print('Your Equation is : ')

print('\t',a,'^2x + ',h,'xy + ',b,'y^2 + ',g,'x + ',f,'y + ',c,' = 0 ' )

axes()

mpl.contour(x, y,(a\*x\*\*2 + h\*x\*y + b\*y\*\*2 + g\*x + f\*y + c), [0], colors='k')

mpl.show()

else:

print('Condition : h\*\*2 - 4\*a\*b = 0 \t NOT SATISFIED \n Try again :( ')

continue

elif ch == 3 :

break

else:

continue

elif opts == 2:

while True:

print('ELLIPSE AND CIRCLES')

print("MENU \n 1. Circle \n 2. Ellipse in Standard equation \n 3. Ellipse in Expanded equation \n 4.Return to Menu")

ch = int(input('Choose an option'))

if ch ==1 :

print('1. Circle')

r= int(input('Please enter radius of circle '))

h = int(input('Please enter x coordinate of centre '))

k = int(input('Please enter y coordinate of centre '))

print('Radius of circle ',r, 'with centre ','(',h,',',k,')')

t = np.linspace(0,2\*np.pi, 1000)

x = r\*np.cos(t) + h

y = r\*np.sin(t) + k

mpl.axis('equal')

mpl.grid()

mpl.plot(x,y)

mpl.show()

if ch==2:

print('2. Ellipse in Standard equation')

print('x^2/A^2 + y^2/B^2 = 1')

a = options(' Please enter the Value of A ')

b = options(' Please enter the Value of B ')

print('Do you Want to see directrix and focus as well ? ')

duff = input (' y for YES and n for NO ')

while True:

if duff == 'y':

axes()

h = int(input('Please enter x coordinate of centre '))

k = int(input('Please enter y coordinate of centre '))

t = np.linspace(0,2\*np.pi, 1000)

p = h + a\*np.cos(t)

q = k + b\*np.sin(t)

mpl.axis('equal')

mpl.grid()

mpl.plot(p,q)

# Defining Eccentricity.

e = np.sqrt(1 - b\*\*2/a\*\*2)

# Printing Foci as DOTs.

mpl.plot(a\*e, 0, '.', -a\*e, 0, '.')

# Printing Directrix, using a vertical lines axvline

mpl.axvline(a/e)

mpl.axvline(-a/e)

mpl.show()

break

elif duff == 'n' :

axes()

h = int(input('Please enter x coordinate of centre '))

k = int(input('Please enter y coordinate of centre '))

t = np.linspace(0,2\*np.pi, 1000)

p = h + a\*np.cos(t)

q = k + b\*np.sin(t)

mpl.axis('equal')

mpl.grid()

mpl.plot(p,q)

mpl.show()

break

else :

continue

if ch ==3:

while True:

print('3. Ellipse in Expanded equation')

print('For ellipses, the eccentricity,e is 0<e<1 \n It is defined as e\*\*2 = 1 - B\*\*2/A\*\*2 ')

print('To define a ellipse in a non-standard position is has to follow the form :')

print('Ax^2+2Hxy+By^2+2Gx+2Fy+C=0, \n where H^2−4AB<0')

a = options(' Please enter the Value of A ')

b = options(' Please enter the Value of B ')

c = options(' Please enter the Value of C ')

f = options(' Please enter the Value of 2F ')

g = options(' Please enter the Value of 2G ')

h = options(' Please enter the Value of 2H ')

if delta(a,b,c,f,g,h) and (h)\*\*2 - 4\*a\*b < 0:

print('Your Equation is : ')

print('\t',a,'^2x + ',h,'xy + ',b,'y^2 + ',g,'x + ',f,'y + ',c,' = 0 ' )

axes()

mpl.contour(x, y,(a\*x\*\*2 + h\*x\*y + b\*y\*\*2 + g\*x + f\*y + c), [0], colors='k')

mpl.show()

mpl.axis('equal')

break

else :

print('Condition : 2h\*\*2 - 4\*a\*b < 0 \t NOT SATISFIED \n Try again :( ')

continue

elif ch==4:

break

else:

continue

elif opts == 3:

q = np.linspace(-100, 101, 200)

w = np.linspace(-100, 101, 200)

q, w = np.meshgrid(q, w)

while True:

print('HYPERBOLA')

print('\*\*\* PRO TIP : Users are requested to use values less than 50, for best results :) \*\*\*')

print("MENU \n 1. Hyperbola in standard equation \n 2. Hyperbola in Expanded equation \n 3.Return to Menu")

ch = int(input('Choose an option'))

if ch == 1:

print('1. Hyperbolas in standard equation are of the form')

print('x^2/A^2 - y^2/B^2 = 1 ')

a = options(' Please enter the Value of A ')

b = options(' Please enter the Value of B ')

hyperbola = (q\*\*2/a\*\*2 - w\*\*2/b\*\*2)

print('Do you Want to see directrix and focus as well ? ')

duff = input (' y for YES and n for NO ')

while True:

if duff == 'y':

axes()

mpl.contour(q, w,hyperbola, [1], colors='k')

# Defining Eccentricity.

e = np.sqrt(1 + b\*\*2/a\*\*2)

# Printing Foci as DOTs.

mpl.plot(a\*e, 0, '.', -a\*e, 0, '.')

# Printing Directrix, using a vertical line axvline

mpl.axvline(a/e)

mpl.axvline(-a/e)

mpl.show()

break

elif duff =='n' :

axes()

mpl.contour(q, w,hyperbola, [1], colors='k')

mpl.show()

break

else:

continue

elif ch == 2:

while True:

print('2. Hperbola in Expanded equation')

print('Hyperbolas in non-standard eqn are of the form ::')

print('Ax^2+2Hxy+By^2+2Gx+2Fy+C=0, \n where H^2−4AB=0')

a = options(' Please enter the Value of A ')

b = options(' Please enter the Value of B ')

c = options(' Please enter the Value of C ')

f = options(' Please enter the Value of 2F ')

g = options(' Please enter the Value of 2G ')

h = options(' Please enter the Value of 2H ')

hyperx = (a\*x\*\*2 + h\*x\*y + b\*y\*\*2 + g\*x + f\*y + c)

if delta(a,b,c,f,g,h) and h\*\*2 - 4\*a\*b > 0 :

print('Your Equation is : ')

print('\t',a,'^2x + ',h,'xy + ',b,'y^2 + ',g,'x + ',f,'y + ',c,' = 0 ' )

axes()

mpl.contour(x, y,hyperx, [0], colors='k')

mpl.show()

break

else:

print('Condition : h\*\*2 - 4\*a\*b > 0 \t NOT SATISFIED \n Try again :( ')

continue

elif ch == 3 :

break

else:

continue

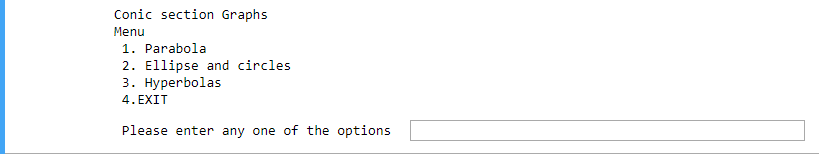
if opts == 4:

break

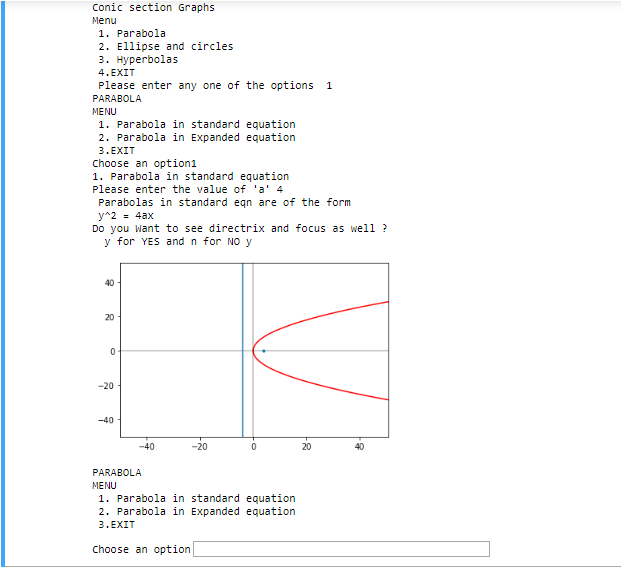
print('EOP')

**OUTPUT SCREENS**

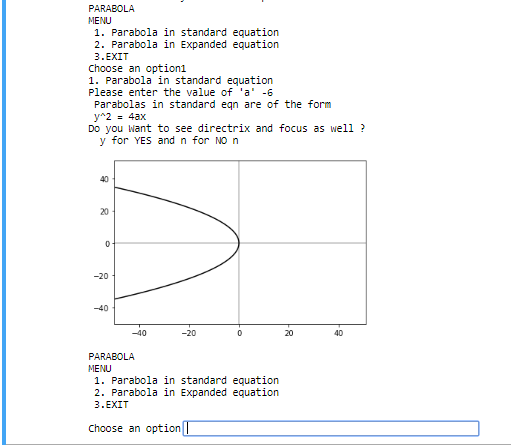
* **Main Menu :**



1. **Parabola, with Directrix and Focus**

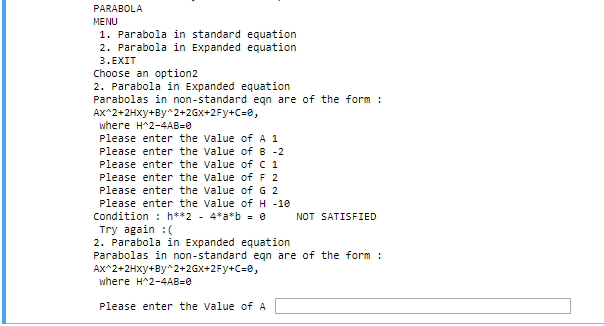


1. **Parabola, without Directrix and Focus**

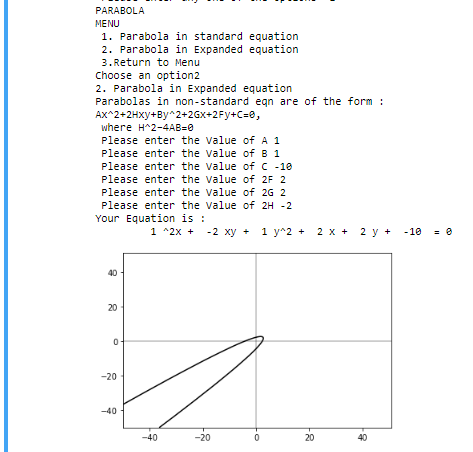


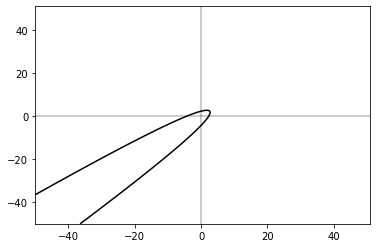
1. **Parabola in Expanded form**

**(If input is incorrect)**

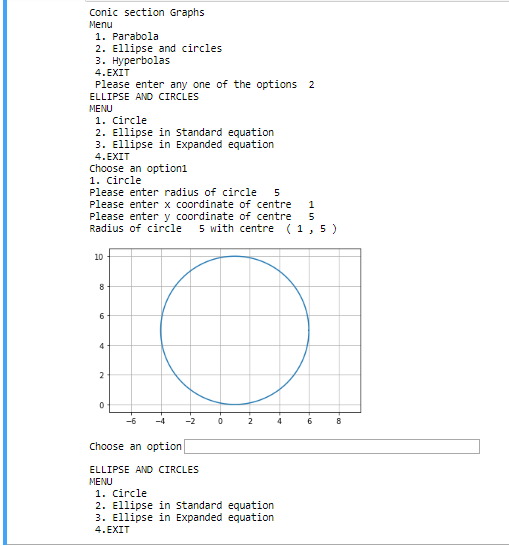


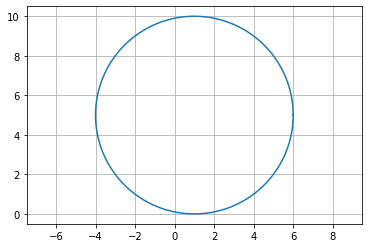
1. **Parabola in Expanded form:**



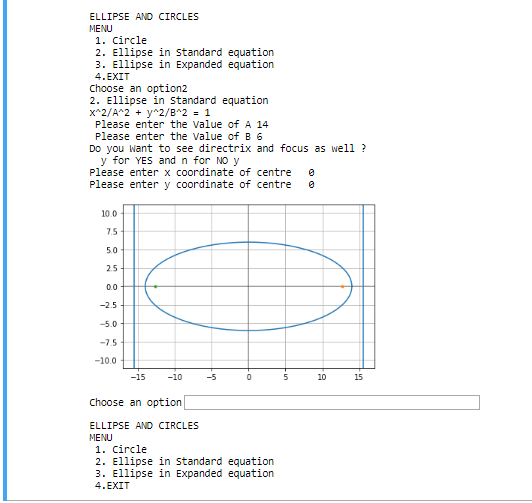


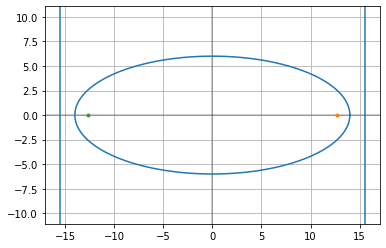
1. **Circle, with given radius and centre**



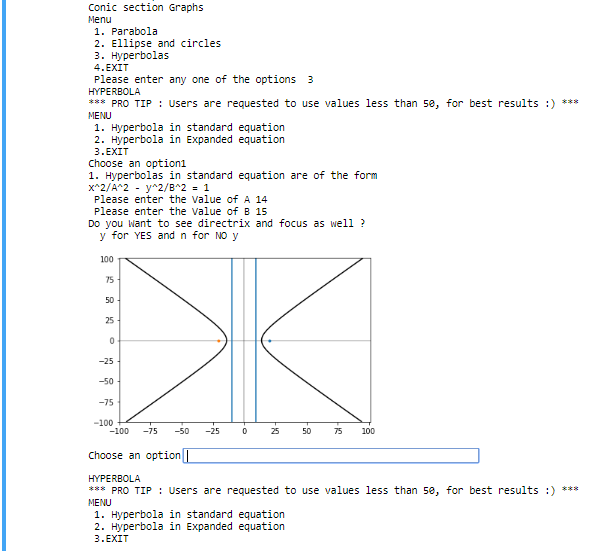


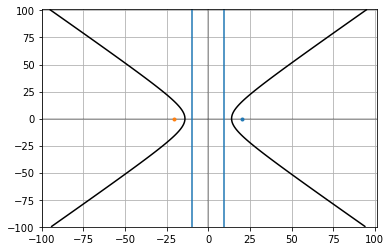
1. **Ellipse**



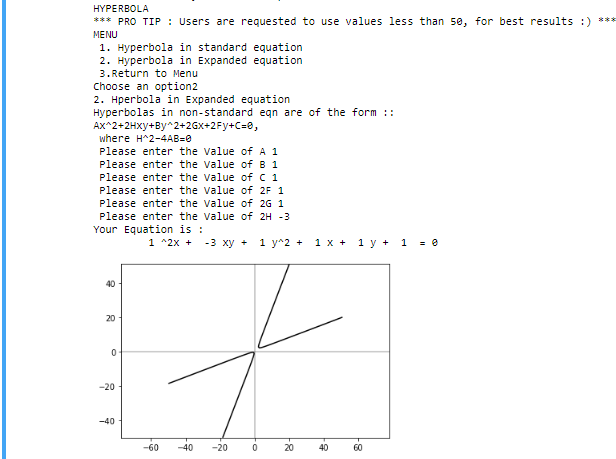


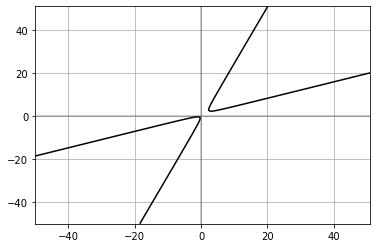
1. **Hyperbola with Directrix and Focus**





1. **Hyperbola In Expanded Form**





**CONCLUSION**

Matpoltlib coupled with Numpy is a dynamic and robust duo which can be used to generate many complex outputs easily. However, just using the plot function may produce our desired output. After exploring,probing and researching the vast domains and documentation on the internet, I came to know how Contour function is a pragmatic choice which will help my code to stand the test of time. This future proof solution can be very helpful in adding additional features like colors, labels and multiple outputs on a single screen without much clutter.

Using defined variables and custom functions, like 'Hyperx' for equations and 'Options' for integral inputs can be very convenient, to ensure correct input, output and providing necessary changes with minimal efforts.

Making a detailed mind chart also aided me in visualizing my project in a better manner.

It is important to realize that there can be multiple ways of bringing the same output, referring to multiple sites often deviates you from your original goal. Going through a single place to learn the whole library first, and then exploring new endeavours was a plus point for me.

Under the able guidance of Angel Ma’am and Multiple code discussions with my dear friend Anirudh helped me in finalizing a substantial framework and subsequently work on it.

Countless sleepless nights and thousands of code drafts helped me FINALLY build a 250 line code, which satisfied my heart and soul.

I just wonder how coders can pull off such huge applications?

I would like to conclude by saying that I wish to perfect my code so that one day it can be used to teach students in the maths lab of our Vidyalaya.

**BIBLIOGRAPHY**

1. Sumita Arora, *Computer Science with Python*, 2019
2. <https://www.geeksforgeeks.org/python-introduction-matplotlib/>
3. <https://www.geeksforgeeks.org/numpy-meshgrid-function/>
4. <https://www.geeksforgeeks.org/python-matplotlib-pyplot-ticks/>
5. <https://mmas.github.io/conics-matplotlib>
6. <https://www.youtube.com/user/sentdex>
7. <https://www.geeksforgeeks.org/multiplication-two-matrices-single-line-using-numpy-python/>
8. <https://www.geeksforgeeks.org/python-numpy/>
9. <https://www.geeksforgeeks.org/numpy-in-python-set-1-introduction/>
10. <https://www.geeksforgeeks.org/numpy-python-set-2-advanced/>
11. <https://www.geeksforgeeks.org/find-and-draw-contours-using-opencv-python/>
12. <https://www.python-course.eu/matplotlib_contour_plot.php>
13. <https://www.youtube.com/watch?v=yZTBMMdPOww>
14. <https://www.youtube.com/watch?v=8JfDAm9y_7s>