

Class Room Mathematics

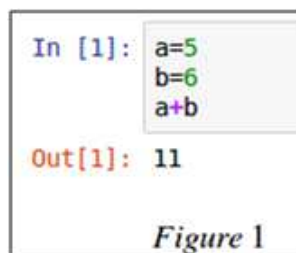
Understanding graphs using SageMath

Dr. Debashish Sharma

Introduction

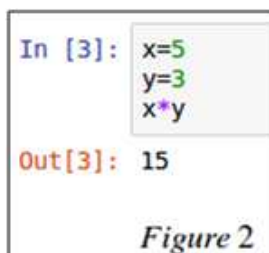
The knowledge of plotting graphs and the ability to understand concepts graphically are two of the most important skills that a mathematics student should inculcate. With computers and smartphones available and used widely, it has become a lot more easier to build up the habit of plotting graphs to analyse mathematical concepts and problems. The recent introduction of the Choice Based Credit System (CBCS) as per the guidelines of University Grants Commission (UGC) has also necessitated the use of softwares to sharpen the mathematical and computational skills of the learners. As per the UGC model curriculum, there are four practical papers in the first two years of Mathematics Honours course. The first paper is on plotting various well known graphs. The very first query that often comes up is the feasibility of buying costly softwares for the purpose. Well, this problem has a very simple solution in form of a large number of opensource softwares. These are available free of cost and are at par with any of the commercial softwares. Some well known ones are Octave, SCILAB and SageMath. In my college, I have used all three of them. However, my personal experience tells me that SageMath is quite helpful and simple to use. This software can be downloaded free of cost from www.sagemath.org. It can also be used online in the cloud platform CoCalc, the link of which is given in the same website. In this article, I shall share my experiences in conducting the practicals on plotting graphs for 1st Semester Mathematics Honours students.

Using SageMath



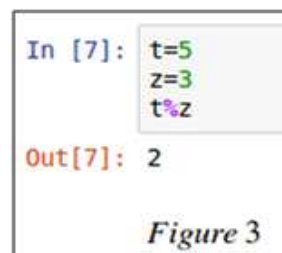
```
In [1]: a=5  
        b=6  
        a+b  
Out[1]: 11
```

Figure 1



```
In [3]: x=5  
        y=3  
        x*y  
Out[3]: 15
```

Figure 2



```
In [7]: t=5  
        z=3  
        t%z  
Out[7]: 2
```

Figure 3

To get the students acquainted with the basic syntax of SageMath, we can begin with simple arithmetic computations. For example, the code in Figure 1 assigns the values 5 and 6 to two variables a and b and gives the value of the sum $a+b$ as the output. Figure 2 shows the code for computing the product of two variables x and y . Figure 3 shows the modulo division i.e. the remainder when t is divided by z .

For plotting graphs, the students just need to understand the format of a few plotting commands. These are discussed below :

plot command :

This command is used to plot a function given by $y=f(x)$ within a certain range of values of x , say $x=a$ to $x=b$. The general syntax is `plot(f(x),(x,a,b))`. Some examples are given below. The outputs are shown in Figures 4 and 5.

Sage code to plot the straight line $y=mx+c$
 $m=2$
 $c=-3$
 $f(x)=m*x+c$
`plot(f(x),(x,-2,3))`

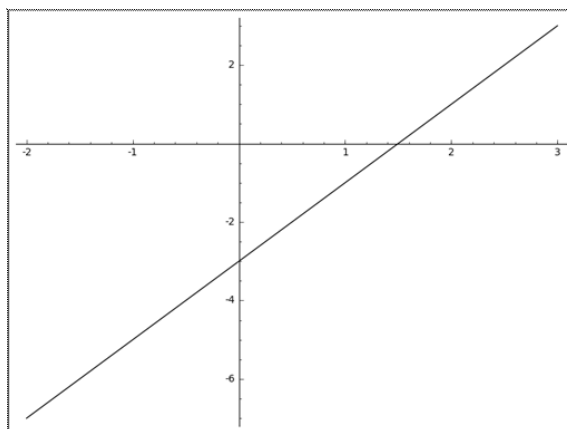


Figure 4 : Straight line

Sage code to plot the parabola $y=x^2$
 $a=-1$
 $b=1$
`plot(x^2,(x,a,b))`

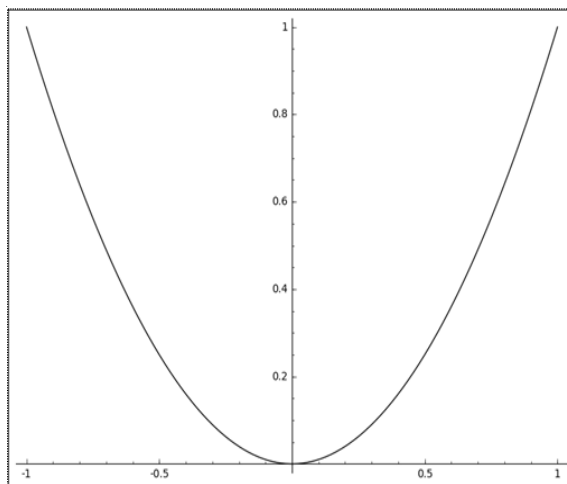


Figure 5 : Parabola

Once the students are comfortable with plotting such basic functions, we can proceed to plotting several graphs, related in some way, on the same worksheet. For example, the code for plotting three parallel straight lines is given below. We need three plot commands, each of which is stored in a different variable. The final graph is obtained by calling the sum of these variables. We can also use different colours for each plot. This is also illustrated in the Sage code below.

Sage code to plot three parallel straight lines

```
m=2
c=0
p1=plot(m*x+c,(x,-3,3))
m=2
c=2
p2=plot(m*x+c,(x,-3,3),color='green')
m=2
c=-3
p3=plot(m*x+c,(x,-3,3),color='red',width='10')
p1+p2+p3
```

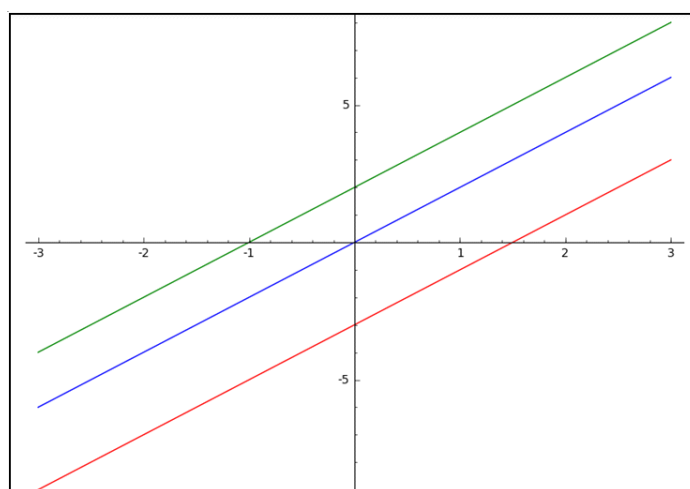


Figure 6 : Three parallel straight lines

Similarly, the students may be asked to plot three straight lines passing through the origin. It may also be necessary to use different pattern of curves for the plots. For this, we can use the `linestyle` option. The available styles are solid, dashed, dotted, dashdot which produce curves accordingly. For example, Figure 7 shows the plot of $y=\sin(ax+b)$ for three different sets of values of a and b . This example will illustrate the effect of changing the values of a and b on the graph of $y=\sin(ax+b)$.

Sage code to plot $y=\sin(ax+b)$ for three different sets of values of a and b :

```
a=1
b=0
p1=plot(sin(a*x+b),(x,-2*pi,2*pi))
a=1
b=pi/2
p2=plot(sin(a*x+b),(x,-2*pi,2*pi),linestyle='dashdot')
a=1
b=-pi/2
```

```
p3=plot(sin(a*x+b),(x,-2*pi,2*pi),color='red',linestyle='dashed')
p1+p2+p3
```

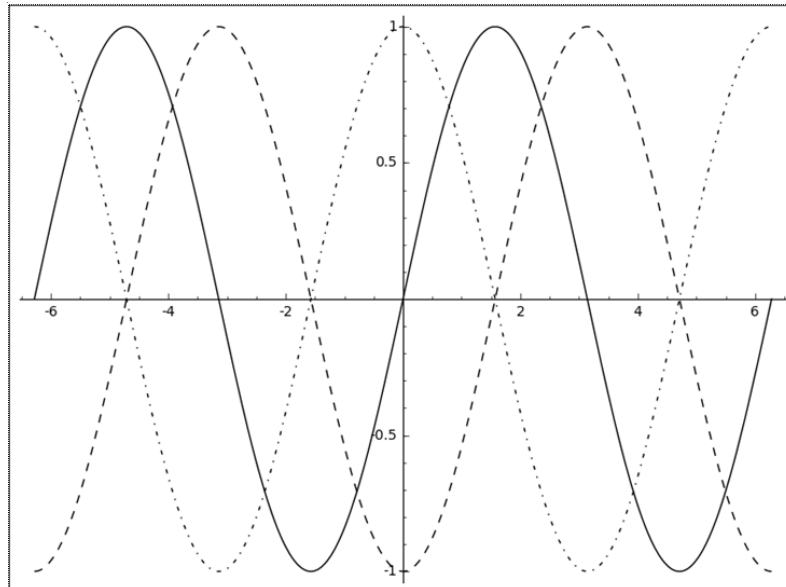


Figure 7 : Graphs of $y=\sin(ax+b)$ for three sets of values of a and b

Next, we can proceed to plotting of a family of curves given by some parameters. This will enhance the understanding of the students. For this, we require a loop structure. In SageMath, this is given by a for loop. The following code will plot the family of parallel straight lines having slope 1.

Sage code to plot the family of parallel straight lines $y= x+c$ for several values of c :

```
p=Graphics()
for c in xrange(-3,3,0.5):
    p=p+plot(x+c,(x,-5,5),color='black')
p
```

Here, `xrange(-3,3,0.5)` means the list of values starting with -3 and incremented by 0.5 upto 3 but excluding 3. `Graphics()` creates an empty graphics window and assigns it to `p`. The for loop thus plots the graphs for $c=-3$, -2.5 , -2 , \dots , 2 and 2.5 only. The plot is illustrated in Figure 8. In a similar way, we can plot several other families of curves.

Figure 8 : Family of parallel straight lines $y=x+c$

Many curves are often given by implicit equations in x and y rather than as $y=f(x)$. For such curves, we have the following command :

implicit_plot command :

Suppose we need to plot a curve given in the implicit form $f(x,y)=k$ between $x=a$ to $x=b$ and $y=c$ to $y=d$. Then we have the general syntax `implicit_plot(f(x,y)-k,(x,a,b),(y,c,d))`

Sage code to plot a circle with given centre and radius :

```
var('x,y')
```

```

h=2
k=3
r=2
implicit_plot((x-h)^2+(y-k)^2-r^2,(x,h-r,h+r),(y,k-r,k+r))

```

The output is given in figure 9.

Sage code to plot a family of circles in first quadrant which touch both the axes :

```

var('x,y')
P=Graphics()
for c in srange(0.4,3,0.1):
P=P+implicit_plot((x-c)^2+(y-c)^2-c^2,(x,0,2*c),(y,0,2*c))
P

```

The output is given in figure 10.

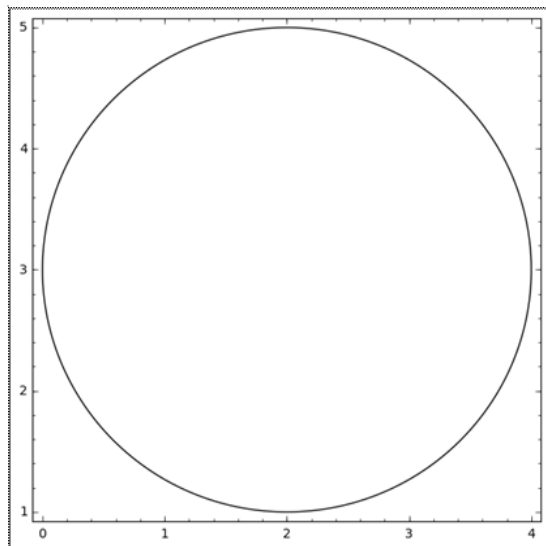


Figure 9 : Circle

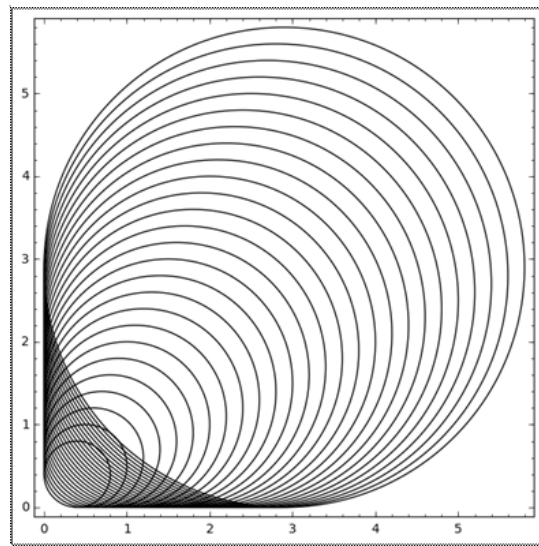


Figure 10 : Family of circles

It is also necessary to use parametric equations in order to plot curves. For this, we have the following command :

parametric_plot command :

Suppose, we want to plot a curve whose parametric equation is given by $x=f(t)$, $y=g(t)$, $z=h(t)$ for t in $[a, b]$. The general syntax is `parametric_plot((f(t), g(t), h(t)), (t,a,b))`

For example, the parametric equation of the astroid $x^{2/3}+y^{2/3}=a^{2/3}$ is given by $x=a\cos^3 t$, $y=a\sin^3 t$, $0 \leq t \leq 2\pi$. The code to plot the astroid is given below.

Sage code to plot an astroid :

```

var('t')
a=3 parametric_plot((a*(cos(t))^3,a*(sin(t))^3),(t,0,2*pi))

```

The output is shown in Figure 11.

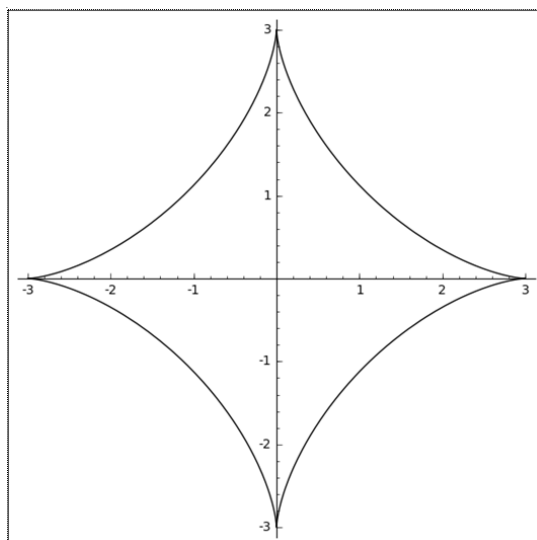


Figure 11 : Astroid

Some interesting exercises can be given to the students so that they visualize the mathematical concepts in a better way. One such exercise can be plotting tangents to curves.

Sage code to plot tangents to the curve $y=x^3-6x+3$ at any point (a,b) on the curve

```
f(x)=x^3-6*x+3
df(x)=diff(f,x)
a=-1
b=f(a)
slope=df(a)
p=plot(f(x),(x,-3,3))
t=plot(b+slope*(x-a),(x,-3,3))
p+t
```

The code given above is quite general in the sense that we just need to plug in the value of the x-coordinate and the rest is done automatically. Figure 12 gives the graphical representation for $a=1$. The command `diff(f,x)` given in the above code gives the first derivative of $f(x)$.

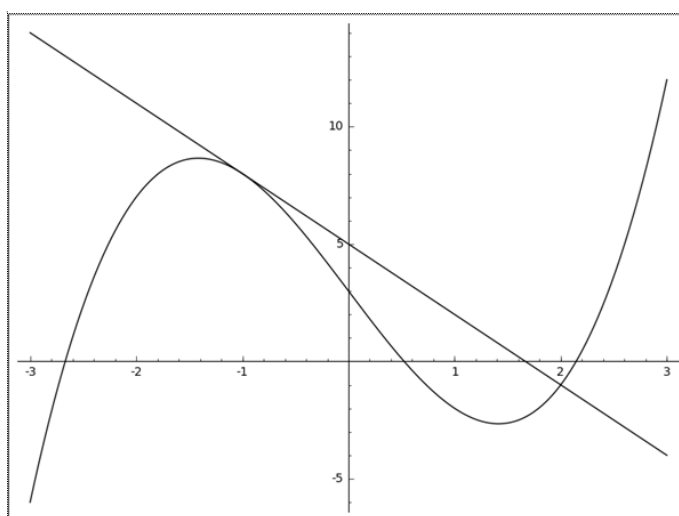


Figure 12 : Tangent at $(-1,8)$

Next, let us plot the normal to the curve as well. We have the following code then :

```
f(x)=x^3-6*x+3
a=-1
b=f(a)
```

```

slope=diff(f,x)(a)
p=plot(f(x),(x,-3,3))
t=plot(b+slope*(x-a),(x,-3,3))
n=plot(b-(1/slope)*(x-a),(x,-3,3))
p+t+n

```

But, it is to be observed that the tangent and normal in the above plot do not seem to be perpendicular to each other. This is because of the automatic scaling of the two axes by SageMath. One unit length in the X-axis is almost as large as 5 units length in Y-axis. This automatic scaling can be avoided by setting the aspect ratio of the first plot as 1. The same will be carried forward to the other subsequent plots. We use the following modified code : Figure 13

```

f(x)=x^3-6*x+3
a=-1
b=f(a)
slope=diff(f,x)(a)
p=plot(f(x),(x,-3,3),aspect_ratio=1)
t=plot(b+slope*(x-a),(x,-3,3))
n=plot(b-(1/slope)*(x-a),(x,-3,3))
p+t+n

```

The output is given in Figure 14.

Another interesting exercise for the students will be the graphical view of Lagrange's Mean Value Theorem. We use the following two sets of codes to illustrate this.

```

f(x)=x^3-6*x+3
a=-2
b=2
dfc=(f(b)-f(a))/(b-a)
g(x)=3*x^2-6-dfc
g.roots()

```

This code will give us two values of the point c in (a,b) where the tangent is parallel to the chord joining (a,f(a)) and (b,f(b)). Using those two values we write the next set of codes:

```

c1=-2/3*sqrt(3)
c2=2/3*sqrt(3)
p=plot(f(x),(x,-3,3))
ch1=plot(f(a)+dfc*(x-a),(x,a,b))
t1=plot(f(c1)+dfc*(x-c1),(x,a,b),color='red')
t2=plot(f(c2)+dfc*(x-c2),(x,a,b),color='green')
p+t1+t2+ch1

```

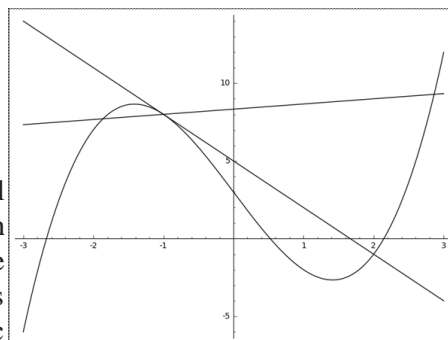


Figure 13 : Tangent and normal at (-1,8)

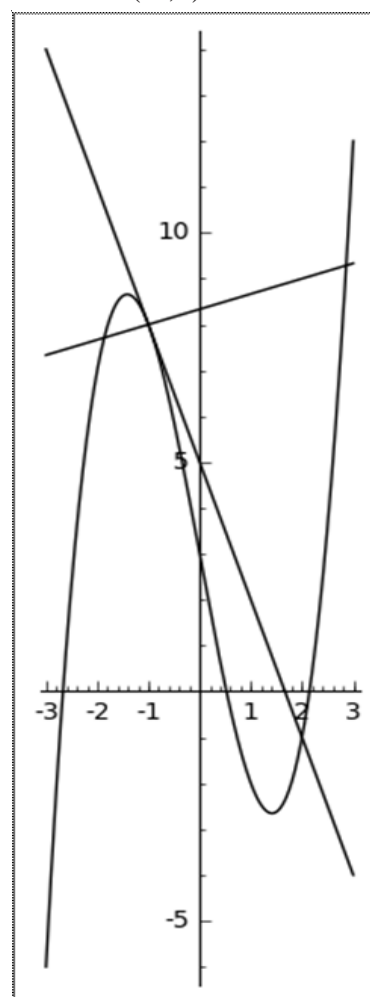


Figure : 14

The plots showing the curve, chord joining the points $(a, f(a))$ and $(b, f(b))$ and the tangents corresponding to c_1 and c_2 are shown in Figure 15. Similar commands exist for plotting three dimensional objects as well.

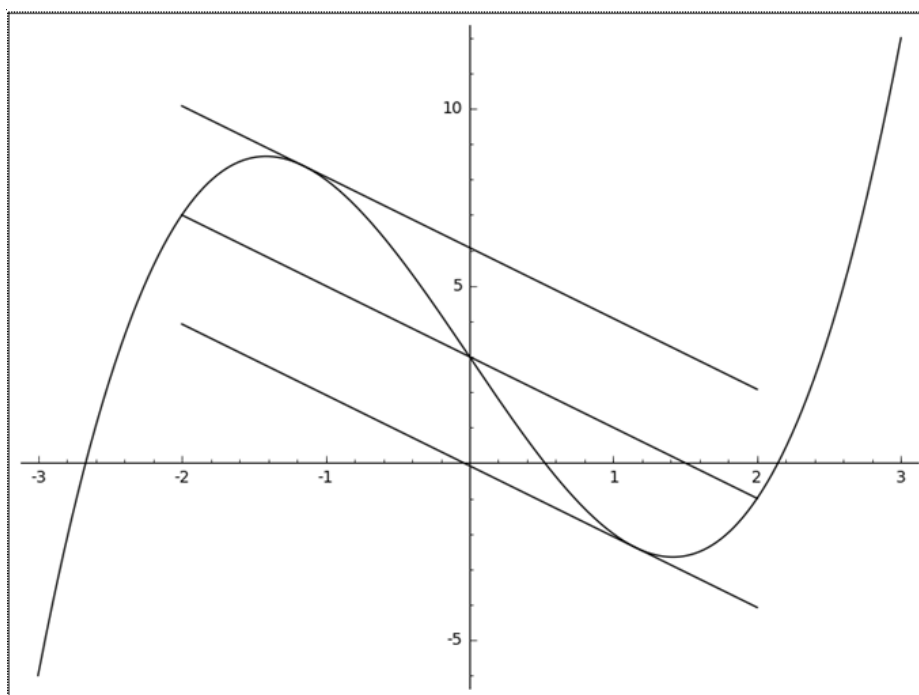


Figure 15 : Depiction of Lagrange's Mean Value Theorem

Concluding notes

SageMath can be used as an effective tool in the practical classes for mathematics honours students. It has a wide variety of features to deal with graphs, equations, calculus and linear algebra. Most importantly, SageMath is an opensource software and so students and teachers can use it absolutely free of cost. SageMath also has a cloud platform called SageMathCloud where users can create an account and perform the computations online. It will be good if teachers take the initiative to introduce SageMath to the students.

Useful resources :

1. Website for downloading the software : <http://www.sagemath.org>
2. Website for the cloud based version : <https://cocalc.com/app>

Dr. Debashish Sharma is at present assistant professor in the
Department of Mathematics of G.C. College, Silchar