Paul's very short introduction to SageMath for Calc 3 students

Sage is open source mathematical software. It has the advantages (and drawbacks) of being open source.

It is easiest to use Sage though an online Sage Cell Server. This is a website that will process short bits of code for you. Since you cannot save code on the Sage Cell Server, **I suggest that you keep a text file somewhere with bits of code that you use frequently.**

- You can access a public Sage Cell Server here:
  https://sagecell.sagemath.org

If you want, you can also use Sage in the following ways:

- download Sage to your personal machine:
  https://www.sagemath.org/index.html
- use the campus Jupyter Notebook Server:
  https://jupyter.datasci.watzek.cloud

The syntax of Sage is rooted in the Python programming language. Like most programming languages, Sage is very fussy about syntax. It is important to pay attention to the details of the code you enter. For example, if you want to tell Sage to multiply 2 times x in order to form the quantity $2x$ you need to explicitly indicate the multiplication and type `2*x`. If you have trouble getting your Sage code to work,

- ask your classmates,
- look at the resources on Paul's website,
- ask Google.

And, of course, you can ask me!

Note that there is a lot more to Sage than is covered in these notes. I recommend the following book:

- *Computational Mathematics with SageMath* by Zimmerman *et al.* which is available here:
  https://aimath.org/textbooks/approved-textbooks/zimmermann/

First steps in getting to know Sage

Here is some Sage code that generates a plot of the function $f(t) = 2t^2$ on the interval $-1 \leq t \leq 2$.

```
var('t')
f(t) = 2*t^2
plot(f(t),(t,-1,2))
```

Let's unpack this code line-by-line:

1. We tell Sage that we would like to use the letter t as a variable.
2. We define the function f by $f(t) = 2t^2$
3. We tell Sage to plot f on the domain $-1 \leq t \leq 2$.

Instead of plotting the function f, we could evaluate it at $t = 10$. Do this with the following code

```
var('t')
f(t) = 2*t^2
f(10)
```

Let's now return to plotting. It is possible to add all sorts of options to the plotting function. For example:

```
var('t')
f(t) = 2*t^2
plot(f(t),(t,-1,2),
     ymin=-.5, ymax = 3,
     thickness=2, color='purple', linestyle='dashed',
     axes_labels=['time␣(s)','volume␣(mL)'])
```

For manipulating plots, it is better to give the plot a name and then to use that name to display the plot. This code should yield the same plot as the previous code.

```
var('t')
f(t) = 2*t^2
fplot=plot(f(t),(t,-1,2),
     ymin=-.5, ymax = 3,
     thickness=2, color='purple', linestyle='dashed')

fplot.show(axes_labels=['time␣(s)','volume␣(mL)'])
```

But the following code allows us to adjust the size of the graphic.

```
var('t')
f(t) = 2*t^2
fplot=plot(f(t),(t,-1,2),
```

```
    ymin=-.5, ymax = 3,
    thickness=2, color='purple', linestyle='dashed')

fplot.show(axes_labels=['time␣(s)','volume␣(mL)'],figsize=[4,3])
```

We can easily modify the code to save the graphic as a pdf file.

```
var('t')
f(t) = 2*t^2
fplot=plot(f(t),(t,-1,2),
    ymin=-.5, ymax = 3,
    thickness=2, color='purple', linestyle='dashed')

fplot.save(filename='your-file-name.pdf',
     axes_labels=['time␣(s)','volume␣(mL)'],
     figsize=[4,3])
```

Go slowly through each line of the following code. What does each line do?

```
var('t')
f(t) = 2*t^2
g(t) = 1-t^2

fplot=plot(f(t),(t,-1,2),thickness=2, color='purple')
gplot=plot(g(t),(t,-1,2),thickness=2, color='blue')

mainplot=fplot+gplot
mainplot.show(ymin=-.5, ymax = 3,
              xmin=-.3, xmax= 1.5,
              axes_labels=['time␣(s)','distance␣(m)'],
              figsize=[7,5])
```

Plotting functions $\mathbb{R}^2 \to \mathbb{R}$

Let's plot the function

$$f(x,y) = e^{-(x^2+y^2)}$$

on the domain $-1 \leq x, y \leq 1$.

```
var('x','y')
f(x,y) = exp(-(x^2+y^2))
fplot = plot3d(f(x,y), (x,-1,1),(y,-1,1))
fplot.show(figsize=[4,3])
```

How can you add the plot of the function $g(x,y) = 1 - x^2 - y^2$ to the image?

Now let's make a contour plot of the function f.

```
var('x','y')
f(x,y) = exp(-(x^2+y^2))
fplot = plot3d(f(x,y), (x,-1,1),(y,-1,1))
fcontour = contour_plot(f(x,y), (x,-1,1),(y,-1,1))
fcontour.show(figsize=[4,4])
```

Now let's make some adjustments

```
var('x','y')
f(x,y) = exp(-(x^2+y^2))
fplot = plot3d(f(x,y), (x,-1,1),(y,-1,1))
fcontour = contour_plot(f(x,y), (x,-1,1),(y,-1,1), fill=false,
    contours=20,cmap=u"BrBG")
fcontour.show(figsize=[4,4])
```

See the Sage documentation for more options.

Region and implicit plots

Shade the region of the plane determined by

$$x^2 + y^2 \leq 1.$$

```
var('x','y')
g(x,y) = x^2 + y^2
region_plot(g(x,y) <= 1, (x,-3,3), (y,-3,3))
```

Fill in the surface in 3D space determined by

$$x^2 - y^2 = 1$$

```
var('x','y','z')
g(x,y) = x^2 - y^2
implicit_plot3d(g(x,y) ==1, (x, -3, 3), (y, -3, 3), (z, -3, 3))
```

If we want to plot the region

$$1 < x^2 + 3 * y^2 < 3$$

then we need to split the two conditions as follows:

```
var('x','y')
region_plot([1 < x^2+3*y^2 , x^2+3*y^2 < 3],
            (x,-3,3), (y,-3,3),
            axes_labels=["$x$","$y$"])
```

Plotting parametric curves

Plot the curve

$$x(t) = t\cos(t), \quad y(t) = t\sin(t), \qquad 0 \leq t \leq 4\pi.$$

```
var('t')
x(t) = t*cos(t)
y(t) = t*sin(t)
parametric_plot([x(t),y(t)],(t,0,4*pi))
```

Plot the curve

$$x(t) = t\cos(t), \quad y(t) = t\sin(t), \quad z(t) = t, \qquad 0 \le t \le 4\pi.$$

```
var('t')
x(t) = t*cos(t)
y(t) = t*sin(t)
z(t) = t
parametric_plot3d([x(t),y(t),z(t)],(t,0,4*pi),thickness=5)
```

Plotting vector fields

Plot the vector field

$$\vec{V} = \langle x, -y\rangle.$$

```
var('x','y')
V = (x,-y)
plot_vector_field(V, (x,-3,3),(y,-3,3))
```

Let's add the plot of this vector field.

$$\vec{W} = \langle y, x\rangle.$$

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

V = (x,-y)
Vplot = plot_vector_field(V, (x,-3,3),(y,-3,3), color="blue")

W = (y,x)
Wplot = plot_vector_field(W, (x,-3,3),(y,-3,3), color="red")

mainplot = Vplot+Wplot
mainplot.show(figsize=[5,5])
```

Plotting parametrically defined regions of the plane

Plot the region of the plane defined by

$$x = u\cos(v), \quad y = u\sin(v)$$

where $0 \le u \le 3$ and $0 \le v \le \pi/4$.

```
var('u,v,z')
T = (u*cos(v), u*sin(v), z, [u,v])
plot3d(0, (u,0,3), (v,0,pi/4), transformation=T, mesh="true")
```

We can also use this transformation to plot the function

$$f(u,v) = \frac{\cos 5v}{1+u^2}$$

on the region where $0 \le u \le 3$ and $0 \le v \le 2\pi$.

```
var('u,v,z')
f(u,v)=cos(5*v)/(1+u^2)
T = (u*cos(v), u*sin(v), z, [u,v])
plot3d(f, (u,0,3), (v,0,2*pi), transformation=T, mesh="true")
```

Plotting parametrically defined surfaces

Plot the surface in 3D space defined by

$$x(u,v) = u\cos(v)$$
$$y(u,v) = u\sin(v)$$
$$z(u,v) = v/4$$

where

$$0 \leq u \leq 3 \quad \text{and} \quad 0 \leq v \leq 8\pi.$$

```
var('u','v')
x(t) = u*cos(v)
y(t) = u*sin(v)
z(t) = v/4
parametric_plot3d([x(u,v),y(u,v),z(u,v)],(u,0,3),(v,0,8*pi),
    opacity=0.7)
```

Some algebra

Suppose we want to solve the system

$$x^2 + y^2 = 1, \quad y = 2\lambda x, \quad x = 2\lambda y$$

```
var('x','y','L')
sol=solve([x^2 + y^2 == 1, y==2*L*x, x== 2*L*y],x,y,L)
show(sol)
```

Differentiation and integration

Compute

$$\frac{\partial}{\partial x}\left(e^{-(x^2+y^2)}\right)$$

```
var('x','y')
f(x,y) = exp(-(x^2+y^2))
result = diff(f(x,y),x)
show(result)
```

There are several ways to compute

$$\iint_{\text{unit disk}} (x^2 + y^2)dA$$

See if you can figure out what each of these pieces of code do.

```
var('x','y')
f(x,y) = x^2 + y^2

result = integrate( integrate(f(x,y),(x,-sqrt(1-y^2),sqrt(1-y^2)
    )), (y,-1,1))
show(result)
```

```
var('r','th')
f(x,y) = x^2 + y^2

result = integrate(integrate(f(r*cos(th),r*sin(th))*r,(r,0,1)),
    (th,-pi,pi))
show(result)
```