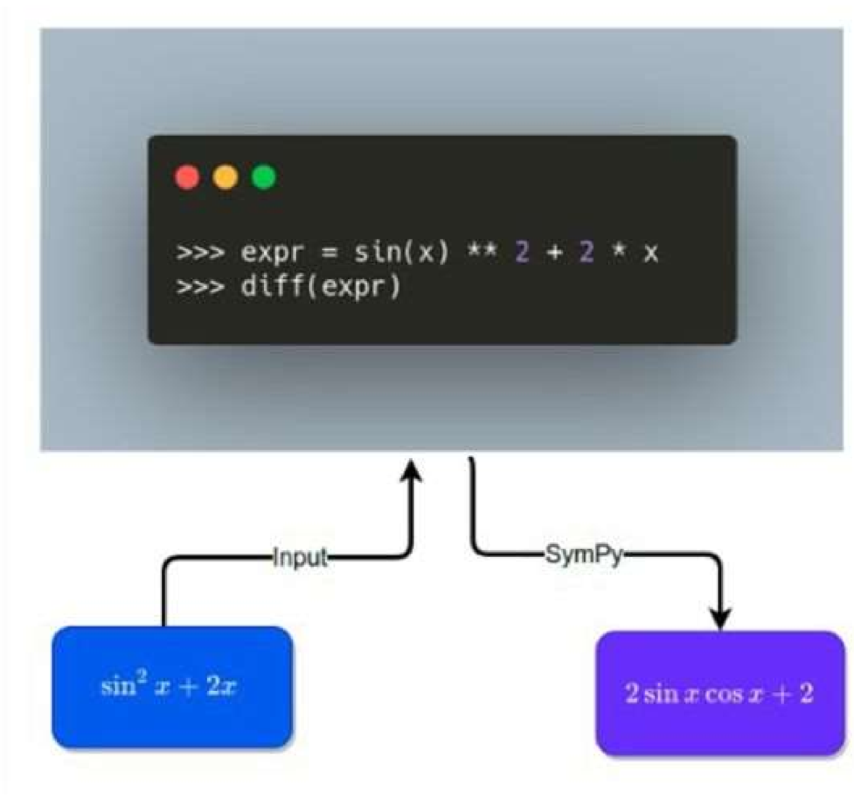
SymPy: Symbolic Computation in Python


```
>>> expr = sin(x) ** 2 + 2 * x
>>> diff(expr)
```

In [ ]:
```
# What is SymPy?
```

SymPy is a Python library that allows you to compute mathematical objects symbolically.

In [1]:
```
pip install sympy
```

```
Requirement already satisfied: sympy in c:\users\kiran\anaconda3\lib\site-packages
(1.10.1)
Requirement already satisfied: mpmath>=0.19 in c:\users\kiran\anaconda3\lib\site-p
ackages (from sympy) (1.2.1)
Note: you may need to restart the kernel to use updated packages.
```

Now let`s go over some of the amazing things that SymPy can do!

Start with importing all methods provided by SymPy

In [2]:
```
from sympy import *
```

Basic Operations

Normally, when computing a square root, we get a decimal:

In [3]:
```
18**(1/2)
```

Out[3]:
```
4.242640687119285
```

But with sympy, we can get a simplified version of the square root instead:

In [4]:
```
sqrt(18)
```

Out[4]: $3\sqrt{2}$

It is because SymPy tries to represent mathematical objects exactly instead of approximately.

This, we will get a fraction instead of a decimal when dividing 2 numbers using SymPy:

In [5]:
```
25/15
```

Out[5]:
```
1.6666666666666667
```

In [6]:
```
frac=Rational(25,15)
frac
```
Out[6]: $\frac{5}{3}$

Symbols

The real power of SymPy is its ability to work with symbols.To create symbol, use the method symbols():

In [7]:
```
x,y= symbols('x y')
expr=3*x+y
expr
```
Out[7]: $3x + y$

Cool! We can create an expression in terms of x and y. What happens if we add a number to this expression?

In [9]:
```
expr+2
```
Out[9]: $3x + y + 2$

Aha! +2 is added to the expression and the expression remains unevaluated.

Why would working with symbols be useful? Because we can now use all kinds of math tricks we learn in school such a expanding, factoring and simplifying an equation to make our life easier.

Equations

Expand,Factor and Simplify

We know that the expansion of the expression on the left is equal to the expression on the right.

x(3x+y)=3x2+xy

Can this be done with Sympy? Yes! SymPy allows us to expand an equation using expand:

In [10]:
```
expansion=expand(x*expr)
expansion
```
Out[10]: $3x^2 + xy$

Another cool thing we can do with SymPy is to simplify an equation using simplify:

In [11]:
```
expr=(6*x**2+3*x)/(3*x)
expr
```

Out[11]: $\frac{6x^2 + 3x}{3x}$

```
In [13]: simplify(expr)
```

Out[13]: $2x + 1$

Solve an Equation

One of the most common questions we see when dealing with mathematical symbols is to solve an equation. Luckily, this can also be done with SymPy:

To solve an equation, use solve:

```
In [14]: eq=(2*x+1)*3*x
         eq
```

Out[14]: $x(6x + 3)$

```
In [15]: solve(eq,x)
```

```
Out[15]: [-1/2, 0]
```

Substitution

If we substitute the equation below by 2, what do we get? x(6x+3)

We can figure that out using eq.subs(x,2):

```
In [16]: eq.subs(x,2)
```

Out[16]: 30

We can also substitute x with another variable to get an expressin like below:

```
In [17]: eq.subs(x**2,2)
```

Out[17]: $x(6x + 3)$

Trigonometric

Remember all the fun trigonometric identities we learn in high school:

To simplify expressions using trigonometric identities, use trigsimp()

```
In [19]: trigsimp(1/sec(x))
```

Out[19]: $\cos(x)$

```
In [20]: trigsimp(sin(x)/cos(x))
```

Out[20]: $\tan(x)$

```
In [21]: trigsimp(sin(x)**2+cos(x)**2)
```

Out[21]: 1

```
In [22]: trigsimp(1+cot(x)**2)
```

Out[22]: $\frac{1}{\sin^2(x)}$

Derivatives, Integrals and Limit

With SymPy, you can also do calculus! What is the derivative of the expression below?

```
In [23]: expr=sin(x)**2+2*x
         expr
```

Out[23]: $2x + \sin^2(x)$

If you cannot figure it out, don`t worry. We can use SymPy to figure it out.

```
In [24]: res=diff(expr)
         res
```

Out[24]: $2\sin(x)\cos(x) + 2$

We can also take the limit as x approaces infinity: lim x-> inf * 1/x2

```
In [25]: limit(
         1/(x**2),
         x,oo,
         )
```

Out[25]: 0

```
In [26]: limit(
         1/(x**2),
         x,
         2,
         )
```

Out[26]: $\frac{1}{4}$

Special Functions

SymPy also provides special functions to:

Find the factorial of a number

```
In [28]: factorial(x)
```

Out[28]: $x!$

```
In [29]: factorial(3)
```

Out[29]: 6

Rewrite the expression in terms of another

```
In [30]: tan(x).rewrite(cos)
```

Out[30]:

$$\frac{\cos\left(x - \frac{\pi}{2}\right)}{\cos\left(x\right)}$$

Print Latex

If you like the output and want to get the LaTex form of an expression,use latex:

```
In [31]: print(latex(integrate(res)))
```

```
2 x + \sin^{2}{\left(x \right)}
```

You could paste this LaTex form to yur notebook`s markdown and get a nice mathematical expression like below!

$2x + \sin^{2}\left(x\right)$

Conclusion

You have just learned how to compute mathematical objects symbolically in Python using SymPy. The next time you solve math, try using SymPy to make your life easier.

There are so many other useful methods SymPy provides that I could not covere here. I encourage you to check out this documentation for more inspiration.

```
In [ ]:
```