Lecture: Introduction to SymPy with live demo

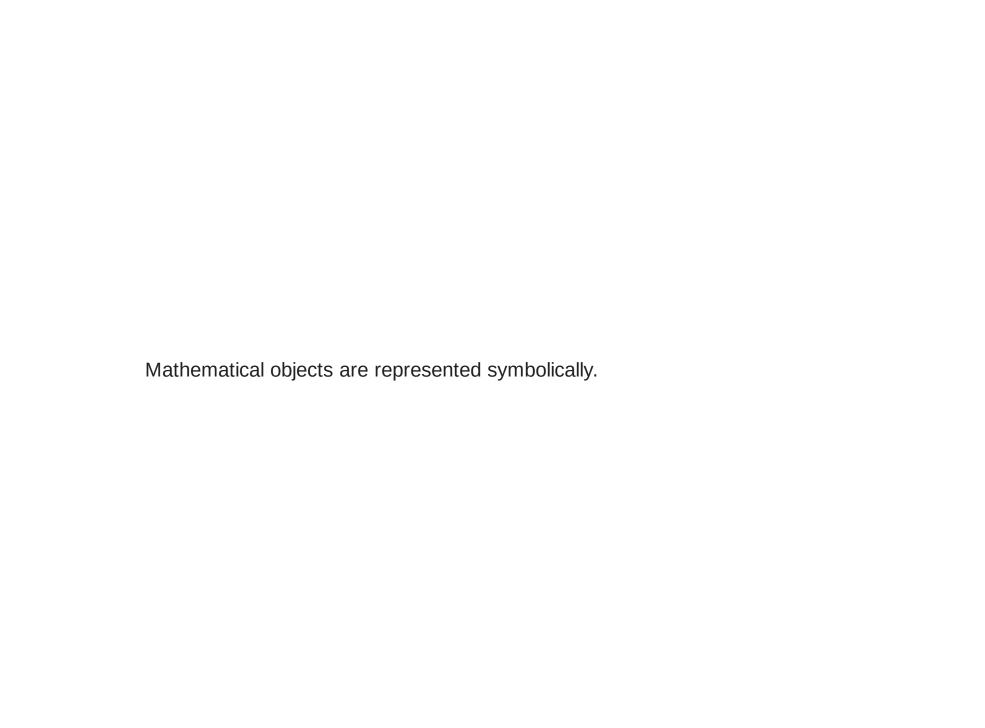
Markus Holzer

12 October 2022

SymPy is a computer algebra toolbox similar to Mathematica or Maple:

- Open Source: https://pypi.org/project/sympy/
- Low barrier to get into
- Many features like simplification, calculus, polynomials, etc.

SymPy is completely programmed in Python and offers rich tutorial sections: https://docs.sympy.org/latest/tutorial/index.html



```
In [1]: import math
  math.sqrt(3)
```

Out[1]: 1.7320508075688772

```
In [1]: import math
    math.sqrt(3)

Out[1]: 1.7320508075688772

In [2]: import sympy as sp
    sp.sqrt(3)
```

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

```
In [1]: import math math.sqrt(3)

Out[1]: 1.7320508075688772

In [2]: import sympy as sp sp.sqrt(3)

Out[2]: \sqrt{3}

In [3]: sp.sqrt(8)

Out[3]: 2\sqrt{2}
```

SymPy defines variables as symbols

SymPy defines variables as symbols

```
In [4]: x, y = sp.symbols("x, y")
type(x)
```

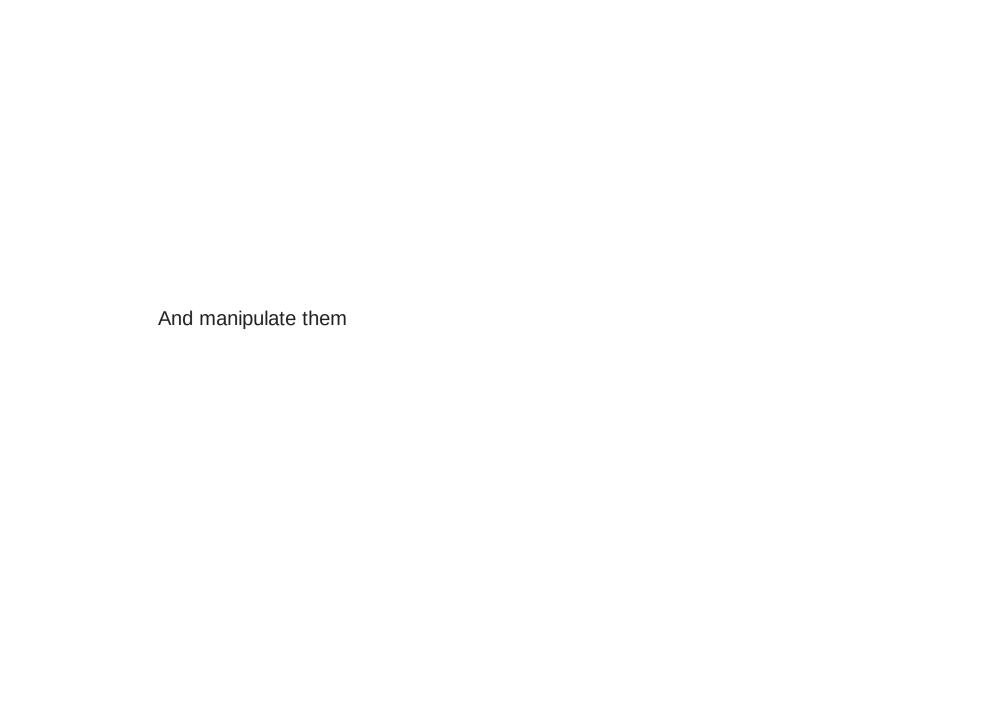
Out[4]: sympy.core.symbol.Symbol



With these symbols it is possible to define mathematical expressions

```
In [5]: expr = 2 * x + y expr
```

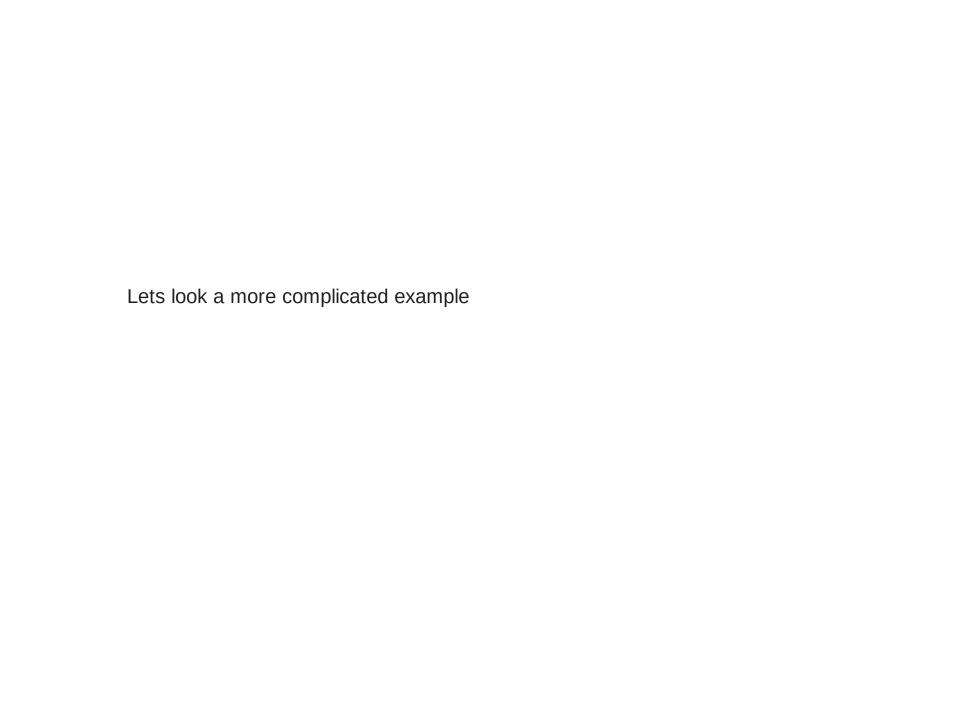
Out[5]: 2x + y



And manipulate them

In [6]: expr / 2

Out[6]: $x + \frac{y}{2}$



Lets look a more complicated example

```
In [7]: x, t, z, nu = sp.symbols('x t z nu')

expr = sp.exp(x) * sp.sin(x) + sp.exp(x) * sp.cos(x)

expr
```

 $\mathsf{Out}\left[7\right]\colon\ e^x\sin\left(x\right)+e^x\cos\left(x\right)$

- One of the most important features SymPy offers is the ability the simplify expressions
- There are many different simplifications implemented
- Most important are: factor, expand, and simplify
- Note that simplify itself applies many different simplifications at once

```
In [8]: factored = sp.factor(expr)
factored
```

Out[8]: $(\sin(x) + \cos(x))e^x$

SymPy can simplify expressions, compute derivatives, integrals, and limits, solve equations, work with matrices, and much, much more, and do it all symbolically.

Let's have a small example about integration.

$$\int \left(e^x \cdot \sin\left(x\right) + e^x \cdot \cos\left(x\right)\right) dx \tag{1}$$

SymPy can simplify expressions, compute derivatives, integrals, and limits, solve equations, work with matrices, and much, much more, and do it all symbolically.

Let's have a small example about integration.

$$\int \left(e^x \cdot \sin\left(x\right) + e^x \cdot \cos\left(x\right)\right) dx \tag{2}$$

```
In [11]: integrated = sp.integrate(sp.exp(x) * sp.sin(x) + sp.exp(x) * sp.cos(x), : integrated
```

Out[11]: $e^{x} \sin(x)$



And lets differentiate this expression again to obtain the expression we started with

```
In [12]: expr = sp.diff(integrated, x)
    expr
```

Out[12]: $e^x \sin(x) + e^x \cos(x)$

Another very important function is subs

Another very important function is subs

```
In [13]: subs_{expr} = integrated.subs({x: y}) subs_{expr}
Out[13]: e^y \sin(y)
```

Another very important function is subs

```
In [13]: subs_expr = integrated.subs(\{x: y\}) subs_expr

Out[13]: e^y \sin(y)

In [14]: subs_expr = subs_expr.subs(\{y: sp.Rational(1, 2)\})
```

This expression now only contains numbers, thus we can evaluate the expression reveal its numerical value				
		mbers, thus we can	evaluate the expre	ssion

This expression now only contains numbers, thus we can evaluate the expression to reveal its numerical value

```
In [15]: evaluated_expr = subs_expr.evalf(30)
    evaluated_expr
```

Out[15]: 0.790439083213614911843262567048

This expression now only contains numbers, thus we can evaluate the expression to reveal its numerical value

```
In [15]: evaluated_expr = subs_expr.evalf(30)
    evaluated_expr

Out[15]: 0.790439083213614911843262567048

In [16]: float(evaluated_expr)

Out[16]: 0.7904390832136149
```

Thank you very much for your attention! Questions??