Algebraic Manipulation

This lesson discusses different techniques of algebraic manipulation.

WE'LL COVER THE FOLLOWING

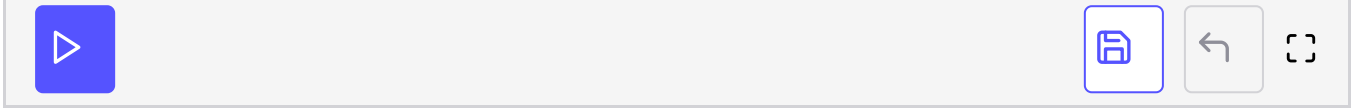
Algebraic simplification is useful in its own right. Within SymPy, though, it is mostly used indirectly as a tool in other areas of the library. In fact, many mathematical problems in symbolic computing are first expressed using entities from the symbolic core before they are preprocessed and transformed into a problem in polynomial algebra, where generic and efficient algorithms are used to solve the problem. The solution to the original problem is then subsequently recovered from the results.

Simplification

The generic way to simplify an expression is by calling the `simplify()` function. It must be emphasized that simplification is not a rigorously defined mathematical operation. The `simplify()` function applies several simplification routines along with heuristics to make the output expression *simple*.

```
from sympy import *
```

```
x = Symbol('x')
y = Symbol('y')
a = (x + x**2)/(x*sin(y)**2 + x*cos(y)**2)
print(simplify(a))
```



> 💡 In some cases, applying `simplify()` may actually result in a more complicated expression.

Due to this issue, we use specialized simplification methods. One example that we will discuss below is `trigsimp()`.

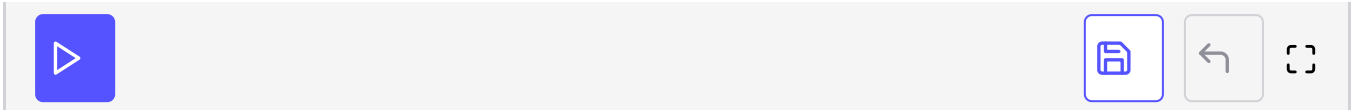
Simplifying trigonometric expressions

The `trigsimp()` method is a specialized method used to simplify trigonometric expressions. Let's see an example of this below:

```
from sympy import *

x = Symbol('x')
a = (2*tan(x) / (1 - tan(x)**2)) + ((1 -  cos(2*x)) / 2)

print(trigsimp(a))
```

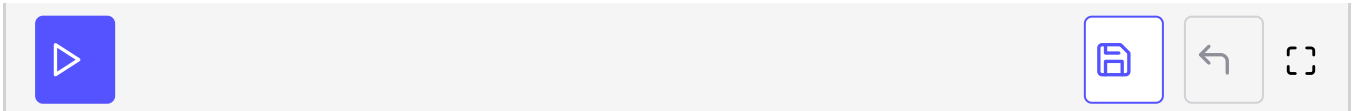


Expand

The `expand()` method expands the mathematical expression. Let's see an example of this below:

```
from sympy import *

x = Symbol('x')
a = ((x + 2) * (x + 3) * (x + 4) * (x + 5))
print(expand(a))
```



> 💡 See `help(expand)` to explore the various types of expansions that `expand()` can perform.

Factor

The opposite of a product expansion is, of course, factoring. The `factor()` method factors a polynomial into irreducibles and returns them. Let's look at an example of this below:

```
from sympy import *

x = Symbol('x')
a = (x**4 + 14*x**3 + 71*x**2 + 154*x + 120)
print(factor(a))
```

Collect

The `collect()` method collects the polynomial coefficients of the symbol specified as the second argument of the method. The first argument is the expression itself. Let's look at an example of this below:

```
from sympy import *

x = Symbol('x')
y = Symbol('y')
a = (x**2 + 3*y**2*x - 3*x**2*y + x - 1)
print(collect(a, x))
```

Fractions

Decomposition

`apart()` computes the partial fraction decomposition of a rational function:

```
from sympy import *

x = Symbol('x')
a = (68*x**2 + 34*x - 27)/(3*(4*x**3 + 4*x**2 - 5*x - 3))
print(apart(a))
```

Combination

The `together()` method combines rational functions into a single fraction and factors it.

```
from sympy import *

x = Symbol('x')
a = 1/x + 1/x + 1/x**2
print(together(a))
```

Let’s test your knowledge with a quick quiz.