1 Sage as a Calculator 3
2 Power Computing with Sage 5
3 Accessing Algorithms in Sage 7
This is a tour of Sage that closely follows the tour of Mathematica that is at the beginning of the Mathematica Book.
The Sage command line has a `sage:` prompt; you do not have to add it. If you use the Sage notebook, then put everything after the `sage:` prompt in an input cell, and press shift-enter to compute the corresponding output.

```
sage: 3 + 5
8
```

The caret symbol means “raise to a power”.

```
sage: 57.1 ^ 100
4.60904368661396e175
```

We compute the inverse of a $2 \times 2$ matrix in Sage.

```
sage: matrix([[1,2], [3,4]])^(-1)
[ -2  1]
[ 3/2 -1/2]
```

Here we integrate a simple function.

```
sage: x = var('x')  # create a symbolic variable
sage: integrate(sqrt(x)*sqrt(1+x), x)
1/4*((x + 1)^(3/2)/x^(3/2) + sqrt(x + 1)/sqrt(x))/((x + 1)^2/x^2 - 2*(x + 1)/x + 1) - 1/
˓→8*log(sqrt(x + 1)/sqrt(x) + 1) + 1/8*log(sqrt(x + 1)/sqrt(x) - 1)
```

This asks Sage to solve a quadratic equation. The symbol `==` represents equality in Sage.

```
sage: a = var('a')
sage: S = solve(x^2 + x == a, x); S
[x == -1/2*sqrt(4*a + 1) - 1/2, x == 1/2*sqrt(4*a + 1) - 1/2]
```

The result is a list of equalities.

```
sage: S[0].rhs()
-1/2*sqrt(4*a + 1) - 1/2
```

Naturally, Sage can plot various useful functions.

```
sage: show(plot(sin(x) + sin(1.6*x), 0, 40))
```
First we create a $500 \times 500$ matrix of random numbers.

```sage
m = random_matrix(RDF,500)
```

It takes Sage a few seconds to compute the eigenvalues of the matrix and plot them.

```sage
e = m.eigenvalues()    #about 2 seconds
w = [(i, abs(e[i])) for i in range(len(e))]
sage: show(points(w))
```
Thanks to the GNU Multiprecision Library (GMP), Sage can handle very large numbers, even numbers with millions or billions of digits.

\[
sage: \text{factorial}(100) \\
93326215443944152681699238856266700490715968264381621468592963895217599932299156089414639761565182862504163811345177680545762250762388007799280112582567950272000000000000000000000000
\]

\[
sage: \text{n = factorial}(1000000) \quad \# \text{about 2.5 seconds}
\]

This computes at least 100 digits of \(\pi\).

\[
sage: \text{N}(\pi, \text{digits}=100) \\
3.\ldots141592653589793238462643383279502884197169399375105820974944592307816406286208998628034825342117068\ldots
\]

This asks Sage to factor a polynomial in two variables.

\[
sage: \text{R.<x,y> = QQ[]}
sage: \text{F = factor(x^{99} + y^{99})}
sage: \text{F}
(x + y) \ast (x^2 - x^3y + y^2) \ast (x^6 - x^3y^3 + y^6) \ast \\
(x^{10} - x^9y + x^8y^2 - x^7y^3 + x^6y^4 - x^5y^5 + \\
x^4y^6 - x^3y^7 + x^2y^8 - x^2y^9 + y^{10}) \ast \\
(x^{20} + x^{19}y - x^{17}y^3 - x^{16}y^4 + x^{14}y^6 + x^{13}y^7 - \\
x^{11}y^9 - x^{10}y^{10} - x^9y^{11} + x^7y^{13} + x^6y^{14} - \\
x^4y^{16} - x^3y^{17} + x^2y^{19} + y^{20}) \ast (x^6y^9 + x^5y^{10} + \\
x^4y^9 - x^3y^{12} + x^2y^{14} - x^3y^{15} + x^2y^{16} - \\
x^3y^{17} + x^{18}y^{30} - x^{17}y^{33} + x^{16}y^{37} - x^{12}y^{42} - \\
x^9y^{51} + x^3y^{57} + y^{60})
sage: \text{F.expand()}
x^{99} + y^{99}
\]

Sage takes just under 5 seconds to compute the numbers of ways to partition one hundred million as a sum of positive integers.

\[
sage: \text{z = Partitions(10^{8}).cardinality()} \quad \# \text{about 4.5 seconds}
sage: \text{str(z)[:40]}
'1760517045946249141360373894679135204009'
\]
Whenever you use Sage you are accessing one of the world’s largest collections of open source computational algorithms.