Sage ガイドツアー

リリース *9.0*

The Sage Group

2020 年 01 月 01 日

目次

第 1 章　電卓としての Sage　3

第 2 章　Sage で力まかせに計算　5

第 3 章　Sage におけるアルゴリズム群の利用　9

以下では,『Mathematica ブック』冒頭の Mathematica 紹介をなぞって，Sage の紹介を試みる．

第 1 章

電卓としての Sage

Sage のコマンドラインに表示されている sage: プロンプトを入力する必要はない．Sage ノートブックを使っている場合， sage: に続く全てを入力セルに入れて shift-enter と押すと計算出力が得られる．

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

キャレット記号 ^ は「べき乗」を表わす．

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

Sage で 2 × 2 行列の逆行列を計算してみよう．

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

初等的な関数を積分する．

```
sage: x = var('x')    # 記号変数を定義
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)
```

以下では Sage に 2 次方程式を解かせる．Sage では等号として記号 == を使う．

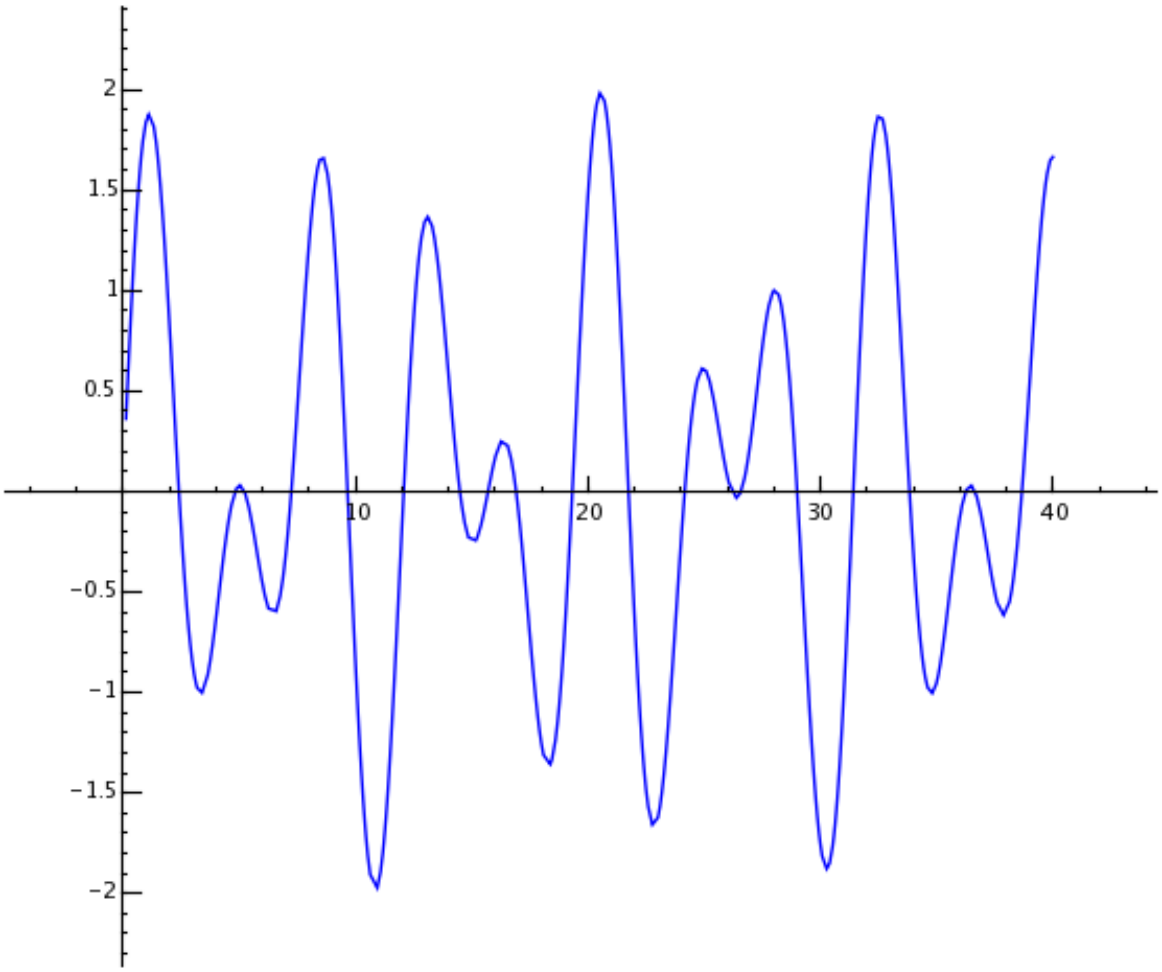
```
sage: a = var('a')
sage: S = solve(x^2 + x == a, x); S
[x == -1/2*sqrt(4*a + 1) - 1/2, x == 1/2*sqrt(4*a + 1) - 1/2]
```

結果は等式のリストになっている．

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

もちろん，よく使われる種々の関数をプロットすることもできる.

```
sage: show(plot(sin(x) + sin(1.6*x), 0, 40))
```



第 2 章

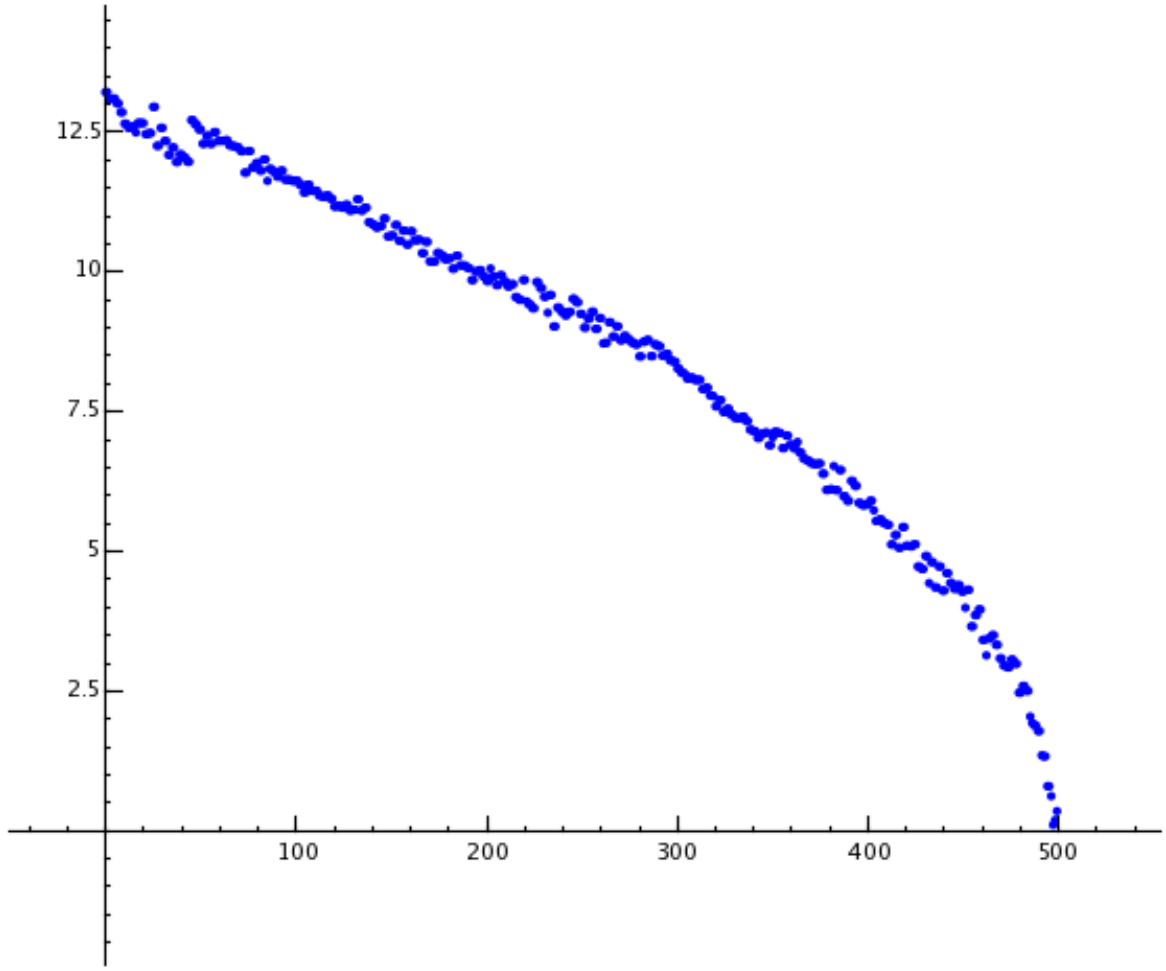
Sage で力まかせに計算

まず要素値が乱数で与えられる 500 × 500 行列を作っておく．

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

Sage でこの行列の固有値を計算してプロットするのも二，三秒程度の仕事だ．

```
sage: e = m.eigenvalues()  # 約 2 秒
sage: w = [(i, abs(e[i])) for i in range(len(e))]
sage: show(points(w))
```



GNU 多倍長ライブラリ (GMP) のおかげで，Sage は数百万から数十億桁までの非常に大きな数を扱うことができる．

```
sage: factorial(100)
933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862
sage: n = factorial(1000000)  # 2.5秒ほどかかる
```

以下では π を，少なくとも 100 桁まで計算する．

```
sage: N(pi, digits=100)
3.
↪14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211706
```

Sage に 2 変数多項式を因数分解させる．

```
sage: R.<x,y> = QQ[]
sage: F = factor(x^99 + y^99)
```

(次のページに続く)

(前のページからの続き)

```
sage: F
(x + y) * (x^2 - x*y + y^2) * (x^6 - x^3*y^3 + y^6) *
(x^10 - x^9*y + x^8*y^2 - x^7*y^3 + x^6*y^4 - x^5*y^5 +
 x^4*y^6 - x^3*y^7 + x^2*y^8 - x*y^9 + y^10) *
(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^9*y^11 + x^7*y^13 + x^6*y^14 -
 x^4*y^16 - x^3*y^17 + x*y^19 + y^20) * (x^60 + x^57*y^3 -
 x^51*y^9 - x^48*y^12 + x^42*y^18 + x^39*y^21 - x^33*y^27 -
 x^30*y^30 - x^27*y^33 + x^21*y^39 + x^18*y^42 - x^12*y^48 -
 x^9*y^51 + x^3*y^57 + y^60)
sage: F.expand()
x^99 + y^99
```

Sage では，1 億を正整数の和として表す仕方を計算するにも 5 秒以下しかかからない.

```
sage: z = Partitions(10^8).cardinality() # 約 4.5 秒
sage: str(z)[:40]
'1760517045946249141360373894679135204009'
```

第 3 章

Sage におけるアルゴリズム群の利用

Sage を使うということは，オープンソース計算アルゴリズムの世界最大級の集大成を利用できることを意味している．