A Tour Of Sage

Release 6.10

The Sage Development Team

21 December 2015

Indice

Questa e' una breve introduzione a Sage che ricalca il tour di Mathematica che si trova all'inizio del libro "The Mathematica Book"

CAPITOLO 1

Sage come Calcolatrice

La linea di comando di Sage ha un prompt `sage:`; non dovete aggiungerlo voi. Se usate il notebook di Sage, allora riportate quello che appare dopo il prompt `sage:` in una cella vuota, e premi MAIUSC+ENTER per ottenere l'output corrispondente

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

L'accento circonflesso indica "l'elevamento a potenza".

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

Calcoliamo l'inversa di una matrice 2×2 con Sage.

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

Qui integriamo una funzione in una variabile.

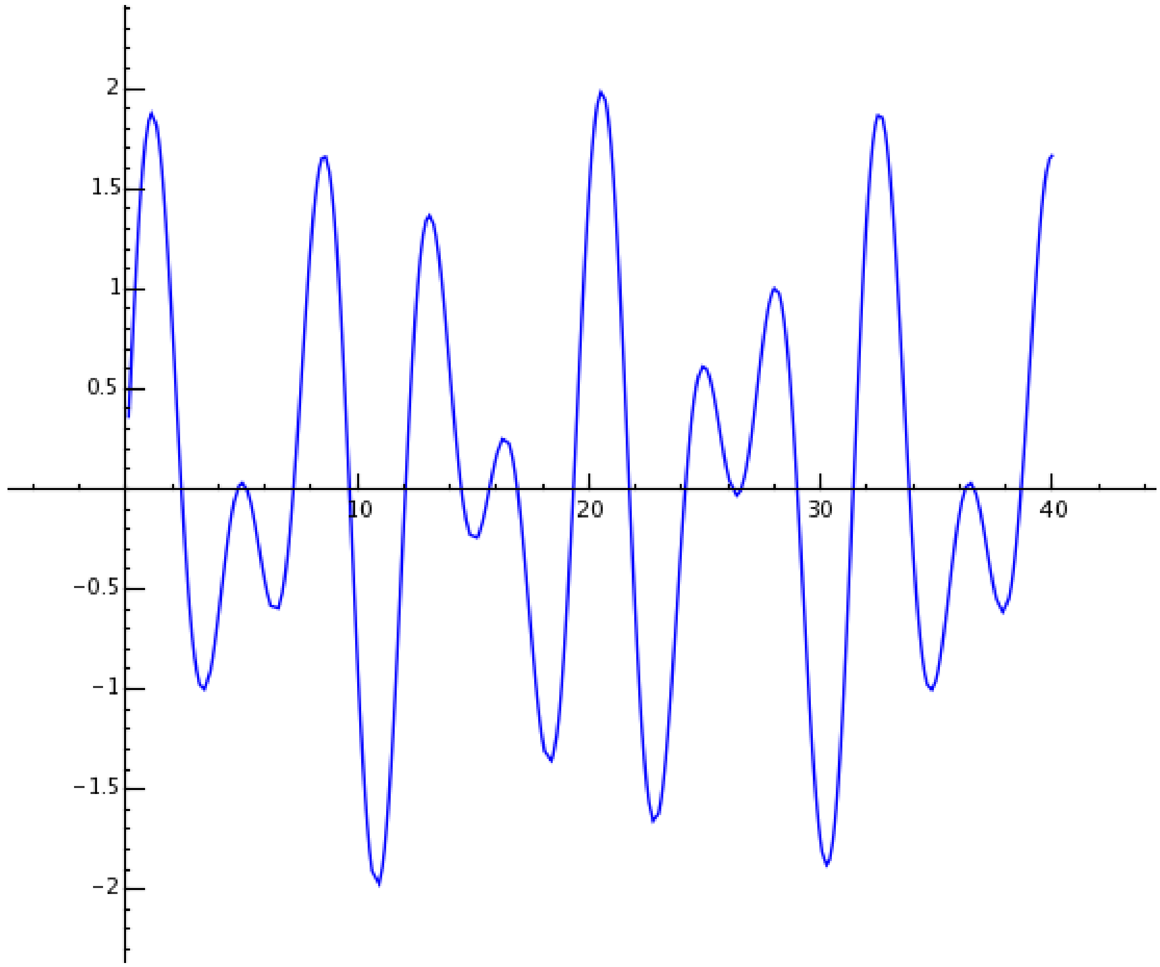
```
sage: x = var('x')   # crea una variabile simbolica
sage: integrate(sqrt(x)*sqrt(1+x), x)
1/4*((x + 1)^(3/2)/x^(3/2) + sqrt(x + 1)/sqrt(x))/((x + 1)^2/x^2 - 2*(x + 1)/x + 1) - 1/8*log(sqrt(x
```

Con questo chiediamo a Sage di risolvere una equazione quadratica. Il simbolo == rappresenta l'uguaglianza su 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]
```

Il risultato e' una lista di eguaglianze.

```
sage: S[0].rhs()
-1/2*sqrt(4*a + 1) - 1/2
sage: show(plot(sin(x) + sin(1.6*x), 0, 40))
```


2
1.5
1
0.5
−0.5
−1
−1.5
−2
10
20
30
40

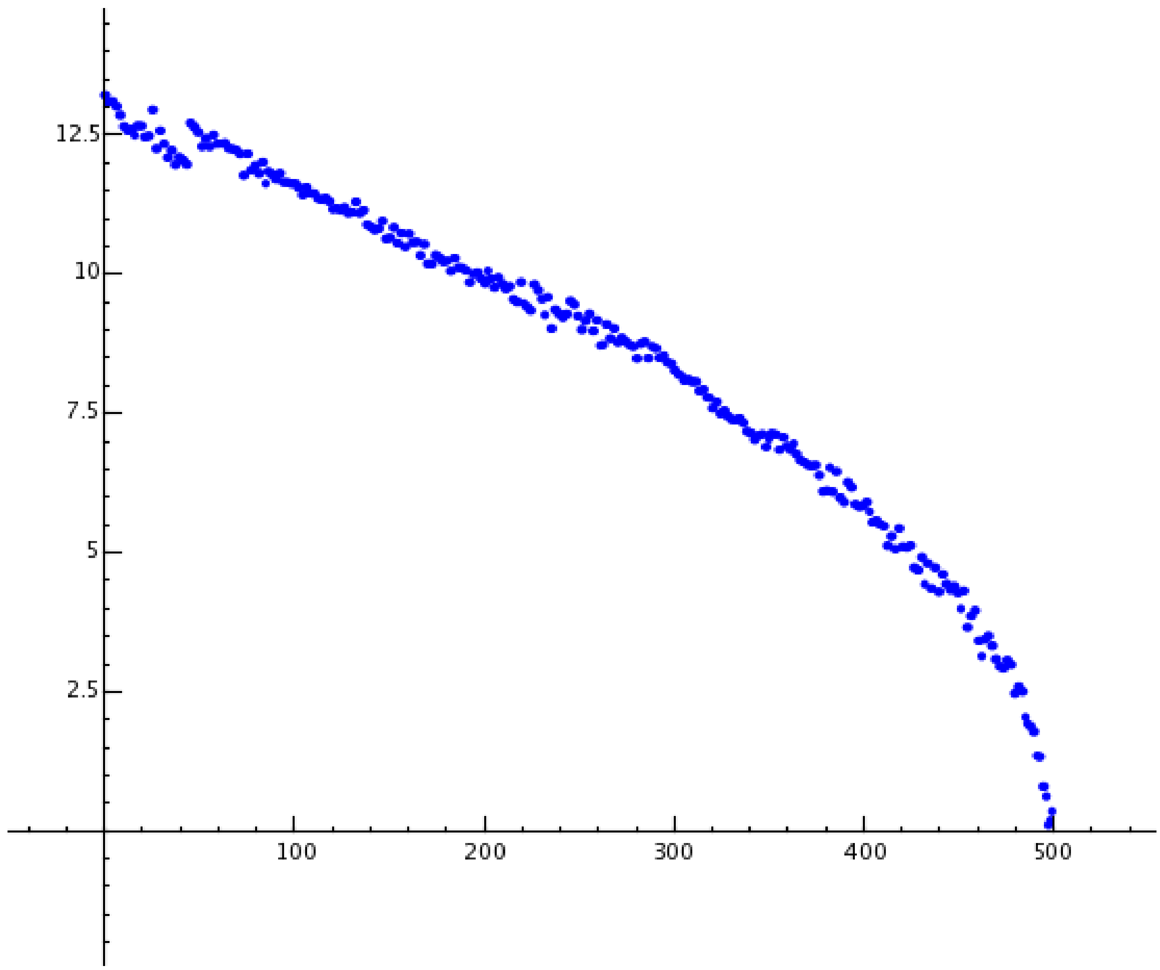
Power Computing con Sage

Iniziamo col creare una matrice random 500×500.

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

Sage impiega qualche secondo per calcolare gli autovalori della matrice e farne il plot.

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



Grazie alla GNU Multiprecision Library (GMP), Sage puo' maneggiare numeri molto grandi, persino numeri con milioni o miliardi di cifre.

```
sage: factorial(100)
93326215443944152681699238856266700490715968264381621468592963895217599993229915608941463976156518286
sage: n = factorial(1000000)   #circa 2.5 secondi
```

Il seguente comando mostra 100 cifre decimali di π.

```
sage: N(pi, digits=100)
3.141592653589793238462643383279502884197169399375105820974944592307816406286208998628034825342117068
```

Questo chiede a Sage di fattorizzare un polinomio in due variabili.

```
sage: R.<x,y> = QQ[]
sage: F = factor(x^99 + y^99)
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 impiega meno di 5 secondi per calcolare in quanti modi il numero cento milioni puo' essere scritto come somma di interi positivi.

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

CAPITOLO 3

Accesso agli Algoritmi in Sage

Con Sage avete accesso ad una delle piu' grandi raccolte al mondo di algoritmi computazionali open source.