

Kurzeinführung Sage

Jonathan Schmidt-Dominé

RWTH Aachen

18. Oktober 2012

1 Allgemeines

2 Grundlagen

3 Analysis

4 Abstraktes

5 Ihr braucht unbedingt Maple?

6 Fazit

Eigenschaften

- Computeralgebrasystem, Computeralgebra im weitesten Sinne

Eigenschaften

- Computeralgebrasystem, Computeralgebra im weitesten Sinne
- Verwendung üblicher Programmiersprache: Python

Eigenschaften

- Computeralgebrasystem, Computeralgebra im weitesten Sinne
- Verwendung üblicher Programmiersprache: Python
- Frei (und kostenlos)

Eigenschaften

- Computeralgebrasystem, Computeralgebra im weitesten Sinne
- Verwendung üblicher Programmiersprache: Python
- Frei (und kostenlos)
- Integriert Features verschiedenster anderer freier Software: Maxima, R, Singular, GAP, ...

Eigenschaften

- Computeralgebrasystem, Computeralgebra im weitesten Sinne
- Verwendung üblicher Programmiersprache: Python
- Frei (und kostenlos)
- Integriert Features verschiedenster anderer freier Software: Maxima, R, Singular, GAP, ...
- Bei Bedarf auch Ansteuerung proprietärer Systeme

Eigenschaften

- Computeralgebrasystem, Computeralgebra im weitesten Sinne
- Verwendung üblicher Programmiersprache: Python
- Frei (und kostenlos)
- Integriert Features verschiedenster anderer freier Software: Maxima, R, Singular, GAP, ...
- Bei Bedarf auch Ansteuerung proprietärer Systeme

Eigenschaften

- Computeralgebrasystem, Computeralgebra im weitesten Sinne
- Verwendung üblicher Programmiersprache: Python
- Frei (und kostenlos)
- Integriert Features verschiedenster anderer freier Software: Maxima, R, Singular, GAP, ...
- Bei Bedarf auch Ansteuerung proprietärer Systeme

Spezielle Software (sehr kleine Auswahl)

- Numerik: Octave (zu Matlab kompatibel)
- Statistik: R, ROOT, ...
- Algebra: Singular, GAP, ...
- Diverse Bibliotheken für C++, Python, (Fortran)

Web-Frontend

- Arbeitet als lokaler HTTP-Server

Web-Frontend

- Arbeitet als lokaler HTTP-Server
- Über Browser zu bedienen, mit MathJax

Web-Frontend

- Arbeitet als lokaler HTTP-Server
- Über Browser zu bedienen, mit MathJax

Cantor

```
21>>> def rodriques():
        global rodriquesdiffs
        return 1/factorial(0)/2**0*rodriquesdiffs[0]
```

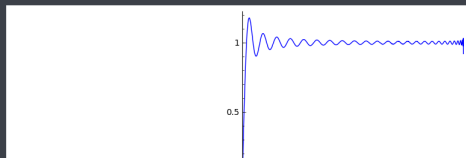
```
23>>> rodriques(0), rodriques(1), rodriques(2), rodriques(3)
```

$$\left(1, x, \frac{3}{2}x^2 - \frac{1}{2}x^3 + \frac{3}{2}(x^2 - 1)x\right)$$

```
18>>> map(theta_coeff, xrange(150))
```

$$\left[0, 1, 0, -\frac{1}{4}, 0, \frac{1}{8}, 0, -\frac{5}{64}, 0, \frac{7}{128}, 0, -\frac{21}{512}, 0, \frac{33}{1024}, 0, -\frac{429}{16384}, 0, \frac{715}{32768}, 0, -\frac{2431}{131072}, 0, \dots\right]$$

```
19>>> plot(sum(map(lambda t: (2**t+1)/2**t*theta_coeff(t) * rodriques(t), xrange(100))), (x, -1.0, 1.0))
```



- Alle Ausdrücke sind ganz normale Python-Objekte

- Alle Ausdrücke sind ganz normale Python-Objekte
- Symbolischer Ausdruck erzeugt durch `var` (wenn nicht standardmäßig vorhanden)

```
var('x,y')
```

- Alle Ausdrücke sind ganz normale Python-Objekte
- Symbolischer Ausdruck erzeugt durch `var` (wenn nicht standardmäßig vorhanden)

```
var('x,y')
```

- Rechnen liefert symbolische Ausdrücke

```
product = x*y
```

- Alle Ausdrücke sind ganz normale Python-Objekte
- Symbolischer Ausdruck erzeugt durch `var` (wenn nicht standardmäßig vorhanden)

```
var('x,y')
```

- Rechnen liefert symbolische Ausdrücke

```
product = x*y
```

- Einsetzen möglich:

```
product(y=1)
```

ergibt

x

- Alle Ausdrücke sind ganz normale Python-Objekte
- Symbolischer Ausdruck erzeugt durch `var` (wenn nicht standardmäßig vorhanden)

```
var('x,y')
```

- Rechnen liefert symbolische Ausdrücke

```
product = x*y
```

- Einsetzen möglich:

```
product(y=1)
```

ergibt

x

- Hilfe

```
plot?
```

- Gleichung lösen mittels solve:

```
s=solve(sin(n*x)==1, x)
```

$$x = \frac{\pi}{2n}$$

- Gleichung lösen mittels solve:

```
s=solve(sin(n*x)==1, x)
```

$$x = \frac{\pi}{2n}$$

- Gleichungssystem:

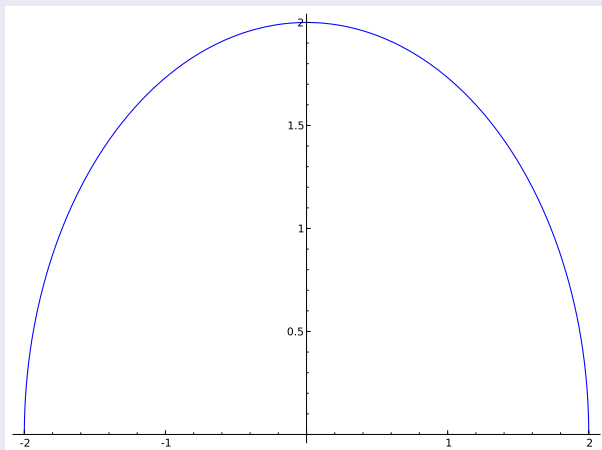
```
s=solve([x + 2*y == 1, x - y == 0], (x, y))
```

$$\left[x = \left(\frac{1}{3} \right), y = \left(\frac{1}{3} \right) \right]$$

Kreishälfte

$r=2$

```
p=plot(sqrt(r^2 - x^2), (x, -r, r))
```



- Erste Ableitung

```
d=diff(x^3, x)
```

$$3x^2$$

- Erste Ableitung

```
d=diff(x^3, x)
```

$$3x^2$$

- Zweite Ableitung

```
d=diff(x^3, x, x) # oder diff(x^3, x, 2)
```

$$6x$$

■ Erste Ableitung

```
d=diff(x^3, x)
```

$$3x^2$$

■ Zweite Ableitung

```
d=diff(x^3, x, x) # oder diff(x^3, x, 2)
```

$$6x$$

■ Mehrere Richtungen

```
f(x,y) = [x*y*z, x*y^2*z]
d = diff(f)
```

$$\left(\begin{array}{cc} (x,y) \mapsto yz & (x,y) \mapsto xz \\ (x,y) \mapsto y^2z & (x,y) \mapsto 2xyz \end{array} \right)$$

■ Geometrische Reihe

```
assume(b>1)
```

```
l = sum(1/b^n, n, 1, infinity)
```

$$l = \frac{1}{b-1}$$

■ Geometrische Reihe

```
assume(b>1)
```

```
l = sum(1/b^n, n, 1, infinity)
```

$$l = \frac{1}{b-1}$$

■ Asymptote

```
l = limit(log(exp(x) + exp(-x)) / x, x=infinity)
```

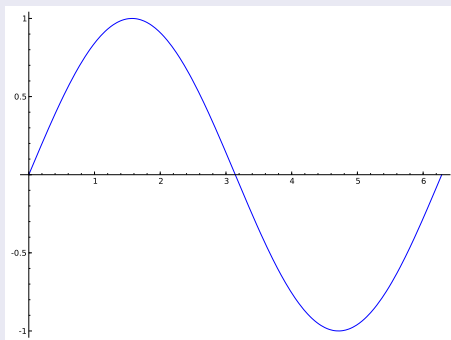
$$l = 1$$

Harmonischer Oszillator

```
assume(omega != 0)
```

```
s=desolve(diff(f, x, 2) == -omega^2*f, f, ivar=x)
```

$$k_1 \sin(\omega x) + k_2 \cos(\omega x)$$



■ Ganze Zahlen \mathbb{Z}

- Ganze Zahlen \mathbb{Z}
- Rationale Zahlen \mathbb{Q}

- Ganze Zahlen \mathbb{Z}
- Rationale Zahlen \mathbb{Q}
- Gleitkommazahlen mit gewisser Anzahl an Stellen $\text{RealField}(n)$ oder einfach \mathbb{R}

```
RF = RealField
```

```
s = RF(2)(2.2) + RF(2)(2.5)
```

$$s = 4.0$$

- Ganze Zahlen \mathbb{Z}
- Rationale Zahlen \mathbb{Q}
- Gleitkommazahlen mit gewisser Anzahl an Stellen $\text{RealField}(n)$ oder einfach \mathbb{R}

```
RF = RealField
```

```
s = RF(2)(2.2) + RF(2)(2.5)
```

$$s = 4.0$$

- Komplexe Gleitkommazahlen $\text{ComplexField}(n)$ oder einfach \mathbb{C}

- Ganze Zahlen ZZ
- Rationale Zahlen QQ
- Gleitkommazahlen mit gewisser Anzahl an Stellen `RealField(n)` oder einfach `RR`

```
RF = RealField
```

```
s = RF(2)(2.2) + RF(2)(2.5)
```

$s = 4.0$

- Komplexe Gleitkommazahlen `ComplexField(n)` oder einfach `CC`
- Polynome `QQ['x']`

- Ganze Zahlen ZZ
- Rationale Zahlen QQ
- Gleitkommazahlen mit gewisser Anzahl an Stellen `RealField(n)` oder einfach `RR`

```
RF = RealField
```

```
s = RF(2)(2.2) + RF(2)(2.5)
```

$$s = 4.0$$

- Komplexe Gleitkommazahlen `ComplexField(n)` oder einfach `CC`
- Polynome `QQ['x']`
- Restklassenringe `ZZ.quotient(n*ZZ)`

```
ZZ5 = ZZ.quotient(5*ZZ)
```

```
s = ZZ5(3) + ZZ5(4)
```

$$s = 2$$

Wenn ihr dazu gezwungen werdet, eine Hausaufgabe mit Maple zu machen: Auf Uni-Rechnern installiert, aber auch wenn man am eigenen Rechner Zugriff braucht: Kauf nicht nötig.

Wenn ihr dazu gezwungen werdet, eine Hausaufgabe mit Maple zu machen: Auf Uni-Rechnern installiert, aber auch wenn man am eigenen Rechner Zugriff braucht: Kauf nicht nötig.

Für Physiker

```
# Mit deiner TIM-Kennung  
ssh -X ab123456@portal.physik.rwth-aachen.de  
xmaple &
```

Für Mathematiker

```
# Mit deiner Matrikelnummer  
ssh -X 123456@venus.mathepool.rwth-aachen.de  
xmaple &
```

Zugang muss einmal eingerichtet werden!

Das freie Sage kann für euer Leben sehr nützlich sein. Weiterführende Information:

- <http://sagemath.org> – Sage
- <http://www.sagemath.org/doc> Dokumentation zu Sage
- <http://ask.sagemath.org/questions> Frage-&Antwortseite
- <http://edu.kde.org/cantor> – Cantor (Oberfläche)

Weitere Software (siehe auch Repositories!):

- <http://maxima.sf.net> – Maxima (Algebra)
- <http://octave.org> – Octave (Numerik et al.)
- <http://r-project.org> – R (Statistik)
- <http://rkward.sf.net> – RKWard (Oberfläche für R)
- <http://root.cern.ch> – ROOT (Datenauswertung, speziell für Physik)
- <http://singular.uni-kl.de> – Kommutative Algebra
- <http://gap-system.org> – Gruppentheorie
- <http://gnu.org/software/gsl> – GNU Scientific Library (Numerik mit C++)
- <http://eigen.tuxfamily.org> – Eigen (lineare Algebra mit C++)
- <http://scipy.org> – NumPy, SciPy (Numerik mit Python)
- <http://matplotlib.sf.net> – matplotlib (Plotten mit Python)
- <http://gnuplot.info> – gnuplot (sehr vielseitiges Plotten)
- <http://edu.kde.org/kmplot> – KMPlot (einfaches Plotten)