Wissenschaftliches Arbeiten mit Python

Symbolisches Rechnen: SymPy

Es geht eine Menge...

- Einfache Terme umformen via einfachen arithmetischen Operationen.
- Rechnen mit Polynomen (bis hin zu Gröbnerbasen).
- Berechnen von Abeleitungen, Integralen und Taylorreihen.
- Symbolisches lösen von Gleichungen und Differentialgleichungen.
- Symbolisches Rechnen mit Matrizen.
- Rechnen mit Punkten, Schnitten und Tangenten
- Nutzen von physikalische Einheiten.

Woher bekommt man es?

- `sudo aptitude install python-sympy` (!Python2 only!), oder
- `pip3 install --user sympy`

Los geht's SymPy: Klassifizieren und in chic

```
1  import sympy as sy
2  x = sy.symbols('x') # Klassifiziert x zum ein Symbol
3  y, z = sy.symbols('y␣z')
4  print(x+y*z)
5  sy.pprint(x+y*z)  # PrettyPrint-Funktion von sympy
6
7  f = sy.Function('f')
8  sy.pprint(sy.Integral(f(x), x))
9
10 f = 1/sy.cos(x)
11 sy.pprint(f.series(x, 0, 10))
12
13 sy.pprint(sy.expand((x+y)**5))
```


SymPy: Funktionen, Grenzwerte und Integrale

```
1  from sympy import *
2  init_printing()  #Schaltet pretty printing generell an
3  x = Symbol('x')
4  f = Function('f'); f = sqrt(x**3 - 5*x + 2)
5  limit(f, x, oo); limit(f, x, -oo); limit(f, x, 0)
6
7  g = Function('g'); g = (x+1) / (x-1)
8  limit(g, x, oo); limit(g, x, -oo); limit(g, x, 0)
9
10 limit(f/g,x,0); limit(f/g,x,1)
```

SymPy: Simplify and calculus

```
from sympy import *
init_printing()
x, y, z = symbols('x␣y␣z')

simplify(sin(x)**2 + cos(x)**2)
simplify((x**3 + x**2 - x - 1)/(x**2 + 2*x + 1))
simplify(gamma(x)/gamma(x - 2))
```

```
diff(x**4, x, 2);  diff(x**4, x, 3)
diff(exp((x*y*z +y)**2),x)
integrate(cos(x), x)
integrate(exp(-x), (x, 0, oo))
integrate(exp(-x**2 - y**2), (x, -oo, oo), (y, -oo, oo))
```


SymPy: Solve my Equation!

Gleichungen werden mit Eq(*linker Term*,*rechter Term*) definiert.

```
from sympy import *
init_printing()
x, y, z = symbols('x␣y␣z')
f = Function('f')

solve(Eq(4, x*2),x)

solve(Eq(y*x-4*x*z, z**2-4*x),x)
```

```
f = Function('f')
k = symbols('k')
g = f(x).diff(x, x) + k**2*f(x)
diffeq = Eq(g,0)
dsolve(diffeq,f(x))
```

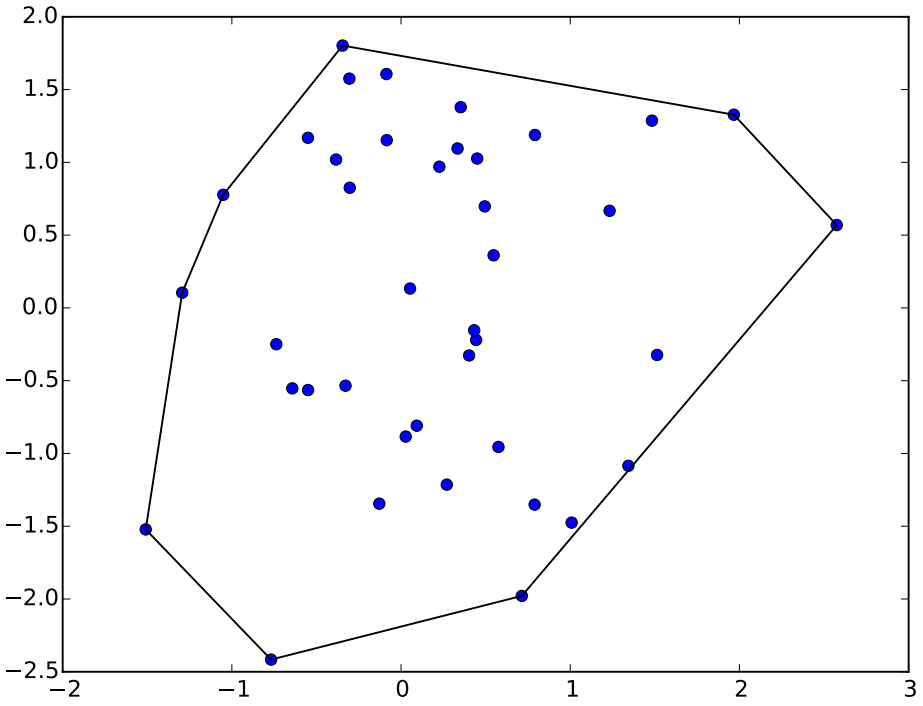
Wissenschaftliches Arbeiten mit Python

SciPy und Freunde, was geht noch?

Konvexen Hülle einer Punktmenge (scipy.spatial)

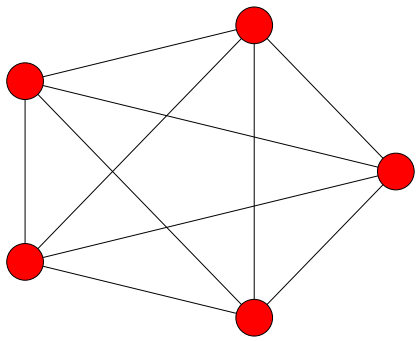
```
1  from scipy.spatial import ConvexHull
2  import numpy as np
3  import matplotlib.pyplot as plt
4
5  points = np.random.randn(40, 2)
6  hull = ConvexHull(points)
7  plt.plot(points[:,0], points[:,1], 'o')
8  for simplex in hull.simplices:
9    plt.plot(points[simplex,0], points[simplex,1], 'k-')
10
11 plt.savefig('con.pdf')
12 plt.show()
```

Konvexen Hülle einer Punktmenge (scipy.spatial) (bild)



Graphs Graphs Graphs (networkx)

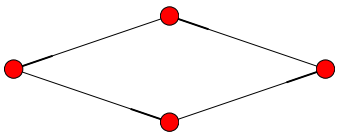
```
1 import networkx as nx
2 import matplotlib.pyplot as plt; import pylab
3
4 G = nx.complete_graph(5)
5 nx.draw_circular(G,node_size=1500)
6 pylab.show()
```



Strongly connected components (networkx)

`pip3 install --user networkx`

```
1  import networkx as nx
2  import matplotlib.pyplot as plt; import pylab
3  G = nx.DiGraph()
4  for e in [(1,2),(2,3),(3,1),(3,4),(4,5),(5,6),(6,7),(7,4)]:
5      G.add_edge(*e)
6
7  SCC  = nx.strongly_connected_components(G)
8  SCCG = nx.strongly_connected_component_subgraphs(G)
9  for c in SCC:
10     print(c)
11
12 for g in SCCG:
13     nx.draw_circular(g,node_size=1500)
14     pylab.show()
```

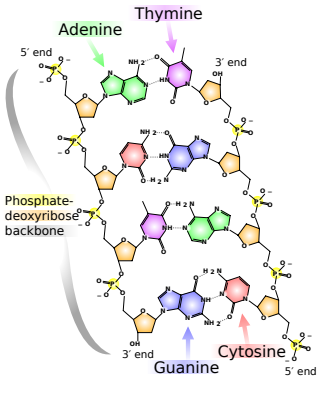


Lifeforms, (scikits.bio)

`pip3 install --user scikit-bio`

Ein Problem aus der Biologie: Finde die längste Folge von Purinen (Guanin oder Adenin) in einem gegebenen DNS-Strang.

```
1  from skbio import NucleotideSequence
2  nuc = NucleotideSequence.read(
3          'single_sequence1.fasta', seq_num=1)
4
5  max(nuc.find_features('purine_run'), key=lambda e: len(e[2]))
```



Making things faster: Numba

„Numba works by generating optimized machine code using the LLVM compiler [..] and is designed to integrate with the Python scientific software stack.“

Beispiel aus Numba Docu

```
1  from numba import jit; from numpy import arange
2  # jit decorator tells Numba to compile this function.
3  @jit
4  def sum2d(arr):
5      M, N = arr.shape; result = 0.0
6      for i in range(M):
7          for j in range(N):
8              result += arr[i,j]
9      return result
10
11 a = arange(9).reshape(3,3)
12 print(sum2d(a))
```

SciPy und Freunde, was geht noch?

NLTK - Natural Language Toolkit

„NLTK is a leading platform for building Python programs to work with human language data.“

Beispiel

```
1  import nltk
2  s = """The lecture by Tom about Python was fantastic."""
3  tokens = nltk.word_tokenize(s)
4  ['The', 'lecture', 'by', 'Tom', 'about',
5    'Python', 'was', 'fantastic', '.']
6  tagged = nltk.pos_tag(tokens)
7  [('The', 'DT'), ('lecture', 'NN'), ('by', 'IN'),
8   ('Tom', 'NNP'), ('about', 'IN'), ('Python', 'NNP'),
9   ('was', 'VBD'), ('fantastic', 'JJ'), ('.', '.')]
```

Wobei: `DT - Determiner`, `NN - Noun, singular or mass`,
`IN - Preposition or subordinating conjunction`
`NNP - Proper noun, singular`, `VBD - Verb, past tense`

Pandas - data manipulation and analysis

Beispiel Pandas Docu

```
1  import pandas as pd #this is how I usually import pandas
2  names = ['Bob','Jessica','Mary','John','Mel']
3  births = [968, 155, 77, 578, 973]
4  BabyDataSet = list(zip(names,births))
5  [('Bob', 968), ('Jessica', 155), ('Mary', 77),
6   ('John', 578), ('Mel', 973)]
7  pd.DataFrame(data = BabyDataSet, columns=['Names', 'Births'])
8
9       Names  Births
10 0       Bob     968
11 1   Jessica     155
12 2      Mary      77
13 3      John     578
14 4       Mel     973
```


Tolle Pakete für die keine Zeit war.

- PsychoPy, cognitive science und neuroscience Experimente durchführen
- ScientificPython, theoretische Biophysik Simulation und mehr
- Thuban, georaphische Datenanalyse
- TensorFlow
- Plotly
- SciKit-Learn
- Theano
- Scrapy
- ...

Wissenschaftliches Arbeiten mit Python

Sage

Let's have some sage tea

Sage ist ein Konglomerat von Programmen, die jeweils für sich für spezielle Problemstellungen entwickelt wurden, vereint in einem Interface.

Algebra	Singular, PolyBoRi
Analysis	Maxima, SymPy
Zahlentheorie	PARI/GP, NTL
Numerik	NumPy, SciPy
Statistik	R
Lineare Algebra	LinBox, LAPACK
Graphentheorie	NetworkX
Gruppentheorie	GAP

Wie kann man es benutzen?

- Mit einem browser, z. B. via `https://cloud.sagemath.com/`
- Lokal installiert und mittels `IPython` basierter Konsole.
- Via Python durch Nutzung der Sage-Bibliotheken.

Wissenschaftliches Arbeiten mit Python

Final Chapter

Lift off to „Jupyter“

„ALL THESE WORLDS ARE YOURS - EXCEPT EUROPA.
ATTEMPT NO LANDINGS THERE.“

-- HAL

Now: Life demonstration of `Jupyter`.