



Python für Wissenschaftler

Thomas McColgan
26.9.11

Inhalt

- Was ist Python?
- Beispiel-Workflow
- Getting started

Was ist Python

- Python ist **freie** Software
- Dynamische **Skriptsprache**
- einfache und eindeutige **Syntax**
- Umfangreiche **Standardbibliothek**
- Viele Pakete für **Wissenschaft**

Was ist Python

- Python ist **freie** Software
- Dynamische **Skriptsprache**
- einfache und eindeutige **Syntax**
- Umfangreiche **Standardbibliothek**
- Viele Pakete für **Wissenschaft**

Wieso frei?

- Frei wie Freibier
- Open-Source Community
 - Guido van Rossum: Benevolent Dictator For Life
 - Entwickler == Benutzer
 - Unterstützt von Google, Yahoo, CERN, NASA, ...
- Offene Standards
 - kein Lock-in

Was ist Python

- Python ist **freie** Software
- Dynamische **Skriptsprache**
- einfache und eindeutige **Syntax**
- Umfangreiche **Standardbibliothek**
- Viele Pakete für **Wissenschaft**

Dynamische Skriptsprache

2 Modi:

- Interaktiv

```
# python  
>>> print "Hello biology!"  
Hello biology!
```

- Batch

```
# python hello.py  
Hello biology!
```

Multi-paradigma

- Prozedural
 - Befehle nacheinander eingeben
- Funktional
 - Befehle in Funktionen bündeln
 - Closures: Funktionen an Funktionen übergeben.
- Objektorientiert
 - Daten und Funktionen in sinnvolle Pakete bündeln
 - Alles ist ein Objekt

Modular

- Funktionen werden hierarchisch sortiert
- Laden nach Bedarf
- Ursprung einer Funktion immer klar
- Module können interagieren

Was ist Python

- Python ist **freie** Software
- Dynamische **Skriptsprache**
- einfache und eindeutige **Syntax**
- Umfangreiche **Standardbibliothek**
- Viele Pakete für **Wissenschaft**

Syntax

- Whitespace matters

Python:

```
x = 0
for i in range(10):
    print i
    if x != 3:
        x += i**2
print x
```

Matlab:

```
x = 0;
for i = 0:9
    i
    if x ~= 3
        x += i^2;
    end
x
end
```

Was ist Python

- Python ist **freie** Software
- Dynamische **Skriptsprache**
- einfache und eindeutige **Syntax**
- Umfangreiche **Standardbibliothek**
- Viele Pakete für **Wissenschaft**

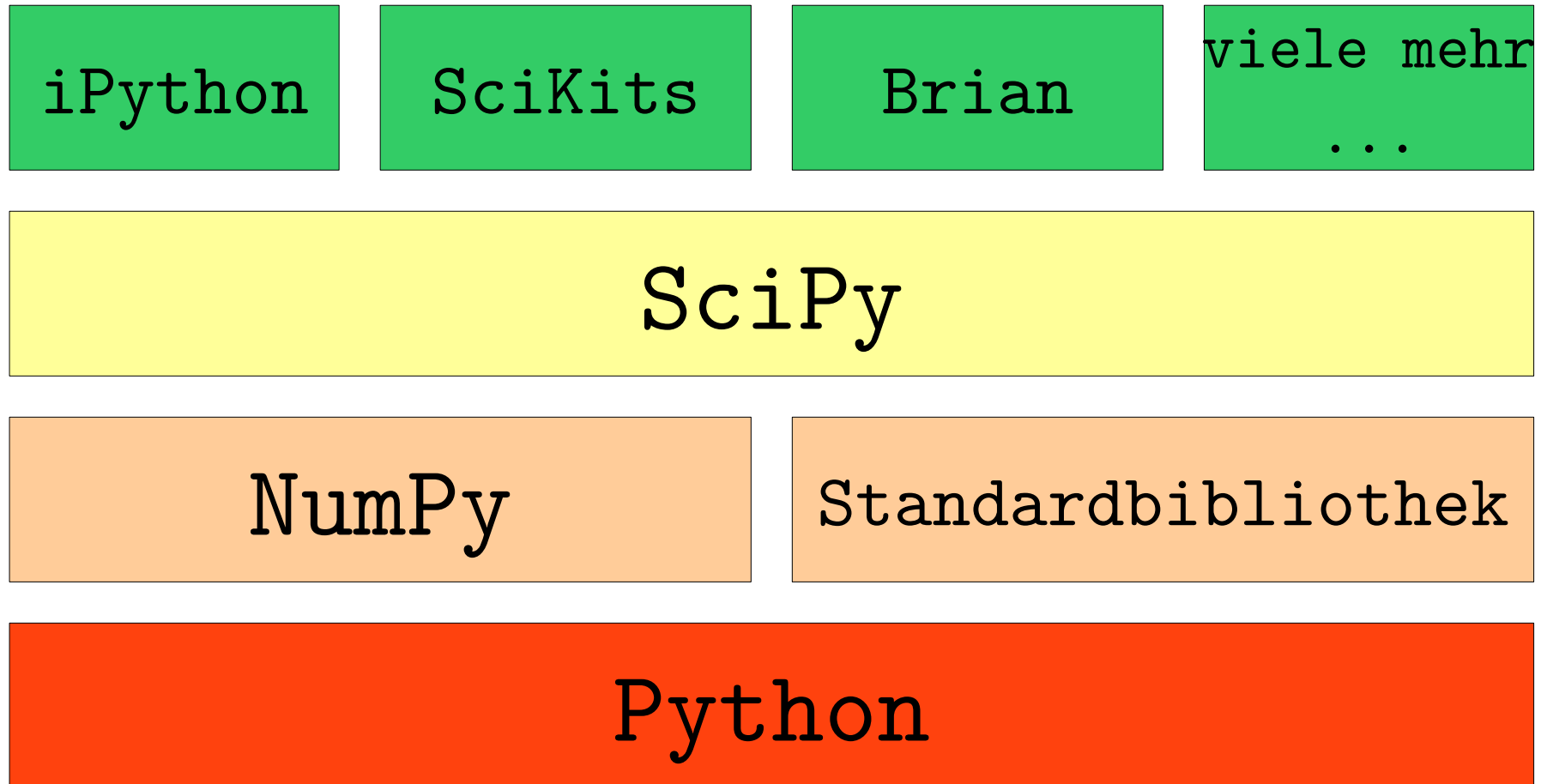
Standardbibliothek

- „Batteries included“
- Funktionen bei Bedarf importieren
- Über 100 Pakete, z.B.:
 - `datetime`
 - `csv`
 - `webbrowser`
 - `audioop`
- <http://docs.python.org/library/>

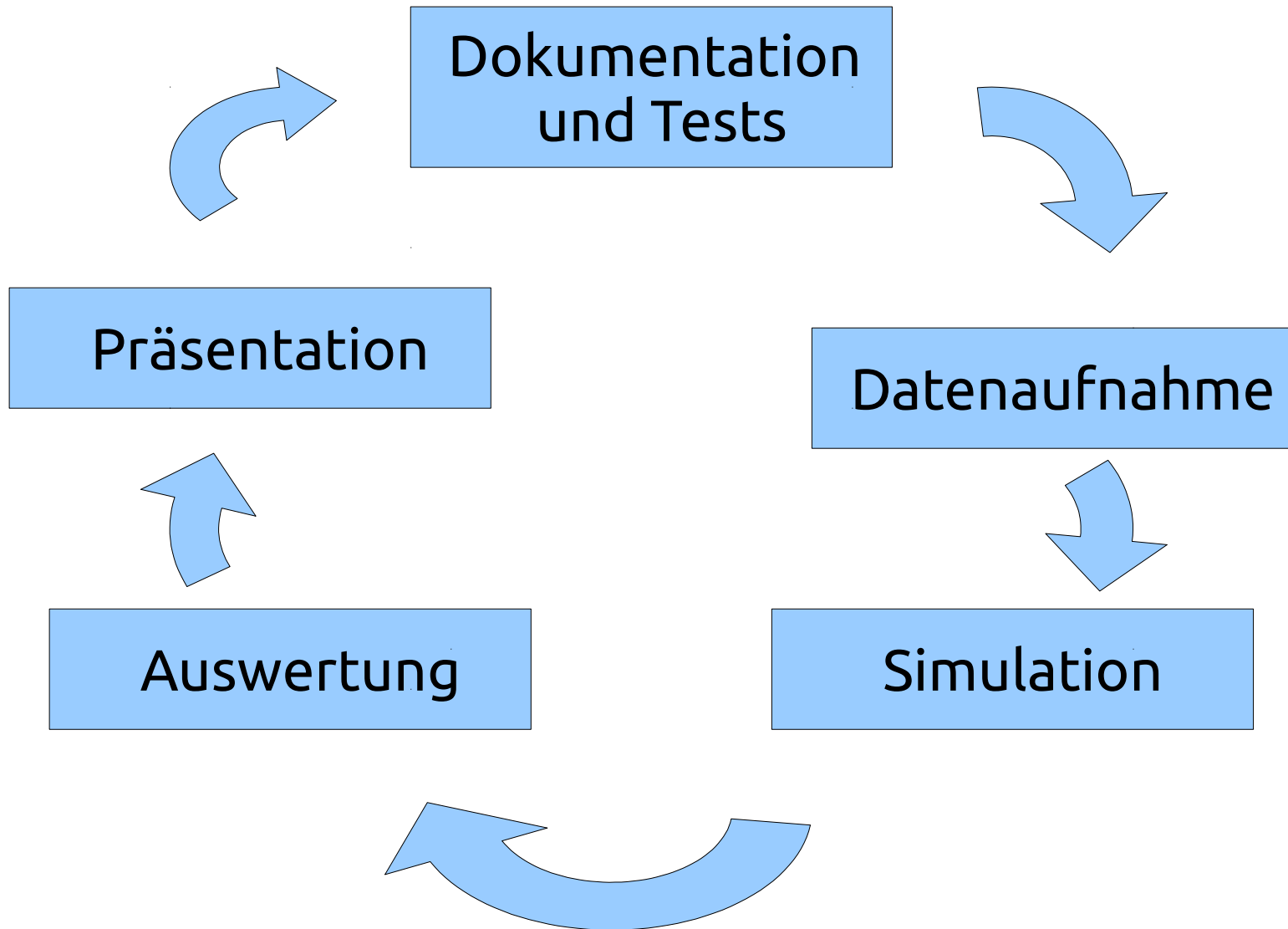
Was ist Python

- Python ist **freie** Software
- Dynamische **Skriptsprache**
- einfache und eindeutige **Syntax**
- Umfangreiche **Standardbibliothek**
- Viele Pakete für **Wissenschaft**

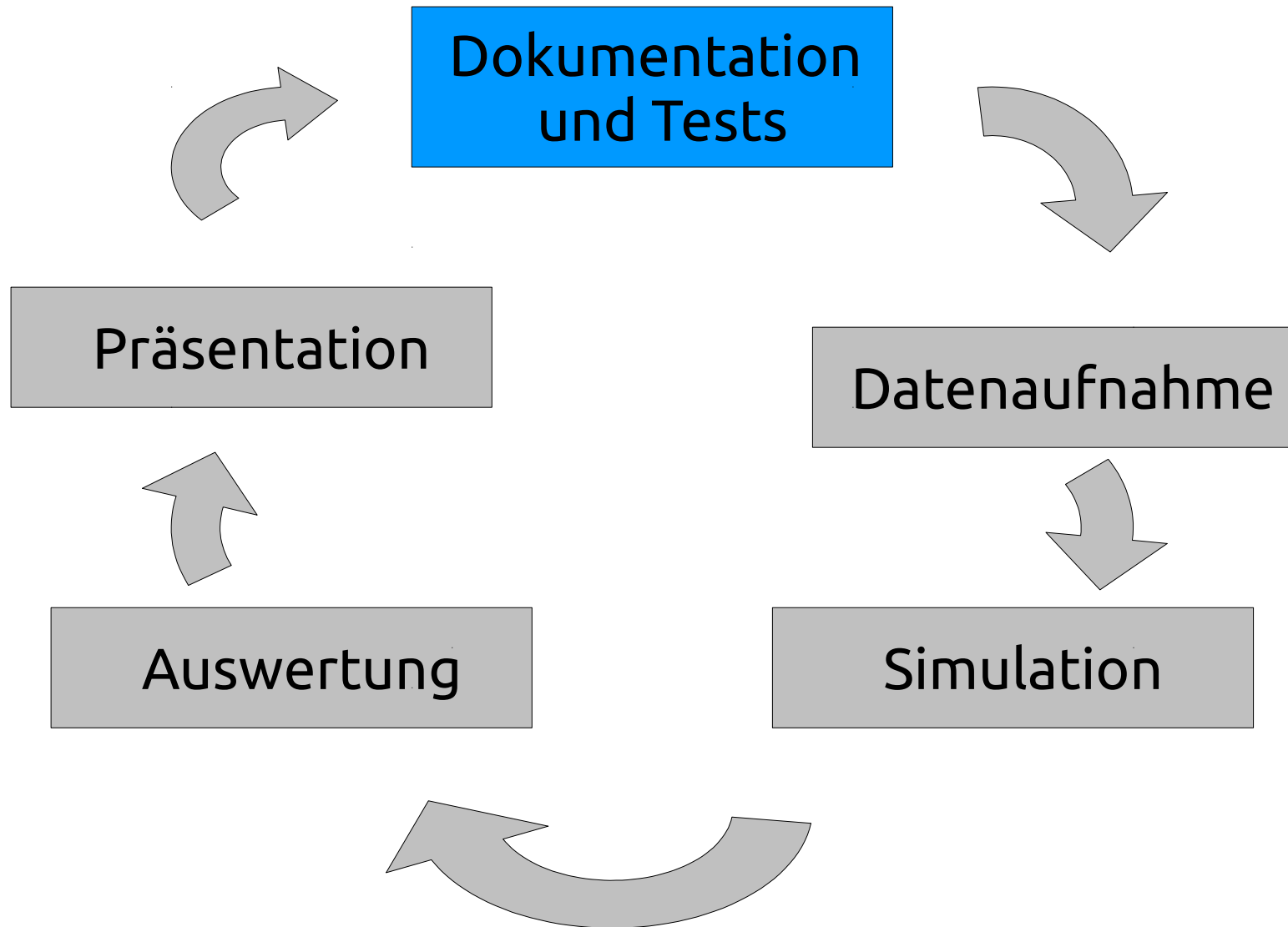
Scientific stack



Wissenschaftlicher Workflow



Wissenschaftlicher Workflow



Dokumentation

- Dokumentation:

- Docstrings:

```
def meinefunktion(x):  
    """Wert um 1 erhöhen"""  
    return x + 1
```

- pydoc (Standardbibliothek)

- Sphinx (<http://sphinx.pocoo.org/>)

- Code-Stil und Hinweise:

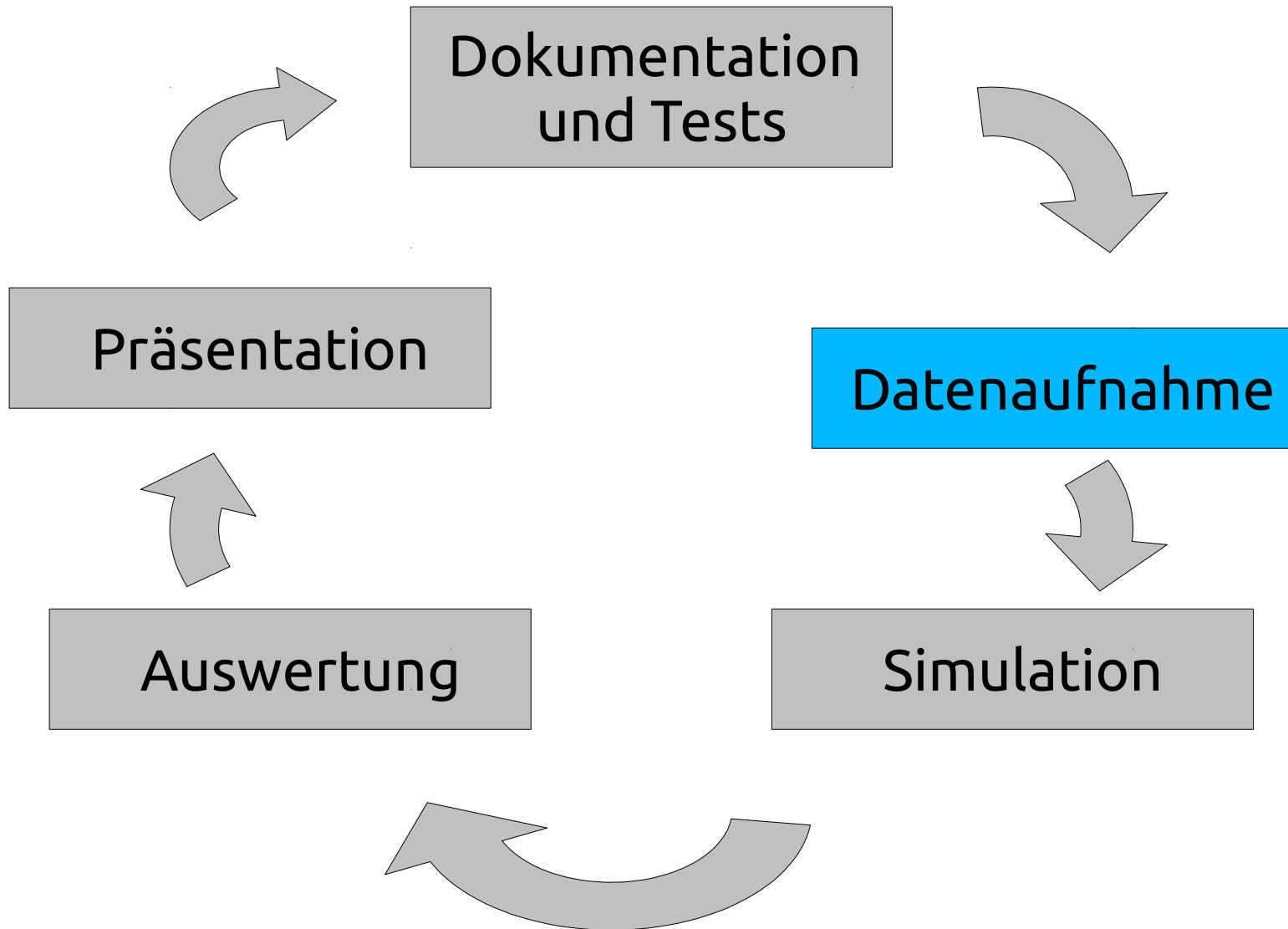
- Community-Konventionen: PEP-8 ([link](#))

- `pylint`: Detaillierte analyse des Codes, mit Gesamtpunktzahl

Tests

- Testing
 - doctest (Standardbibliothek)
 - nosetests (<http://readthedocs.org/docs/nose/en/>)
- Profiling
 - `timeit`: Schnelle Tests für Laufzeiten
 - `profile`, `hotshot`: Detaillierteres profiling
 - Alle 3 in der Standardbibliothek

Wissenschaftlicher Workflow

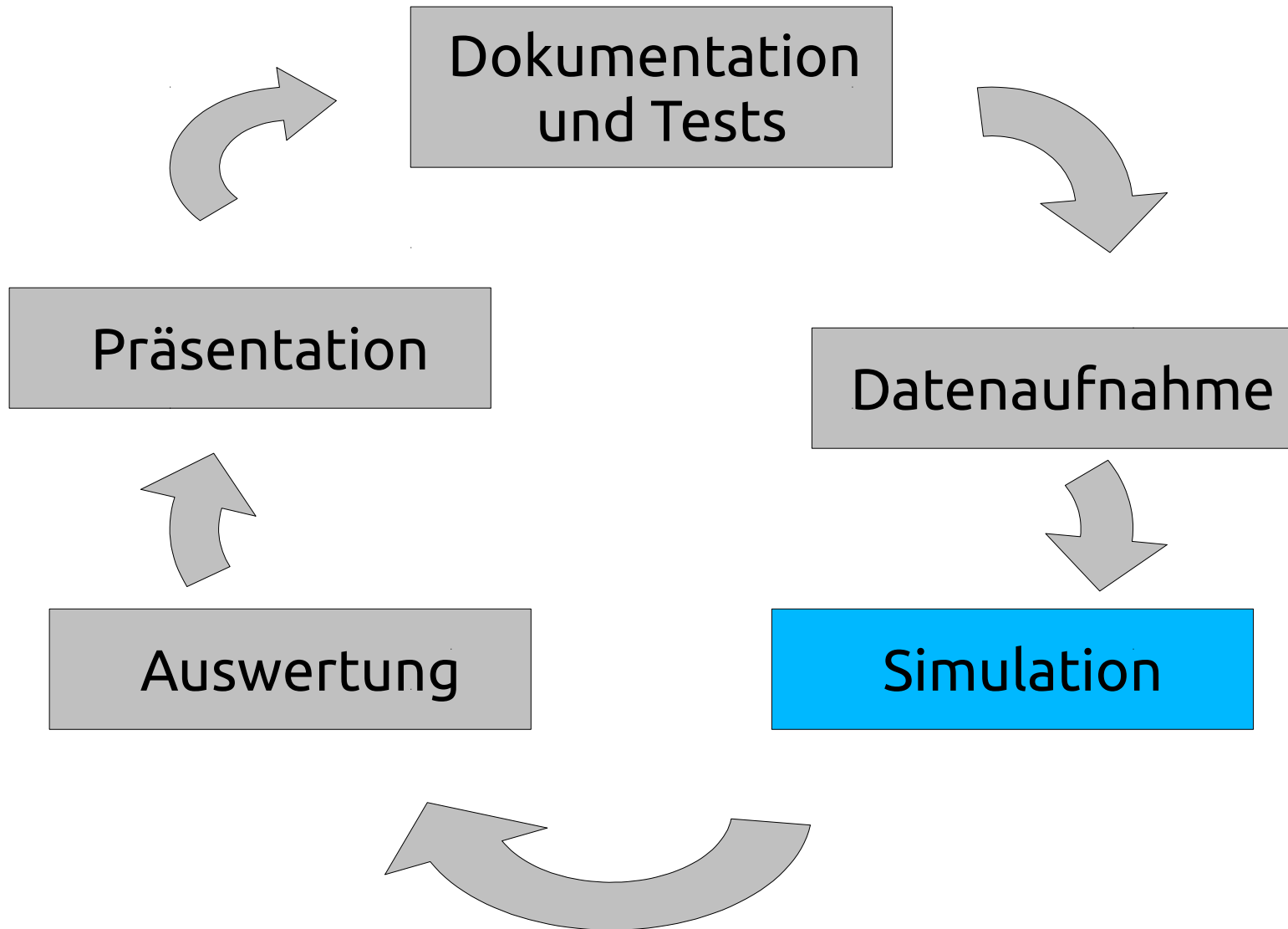


Datenaufnahme

- `csv`, `xml`, `file` (Standardbibliothek)
- Neo (<http://packages.python.org/neo/>):
 - Interface für viele E-Phys Datenformate: Plexon, Spike2, NeuroExplorer, pCLAMP and AxoScope, Tucker Davis Ttank, Micromed, Neuroshare, EEGLAB, WinWCP, WinEdr, HDF5,...
 - Erweiterbar für neue Formate
 - Einfaches Einlesen zur Verwendung in Python:

```
>>> from neo.io import MyFormatIO
>>> file = MyFormatIO("myfile.dat")
>>> file.supported_objects
[Segment , AnalogSignal , SpikeTrain, Event, Spike ]
>>> seg = file.read_segment()
>>> seg.get_analogsignals()
```

Wissenschaftlicher Workflow



Simulation

- Brian Hears

<http://www.briansimulator.org/docs/hears.html>

```
from brian import *
from brian.hears import *

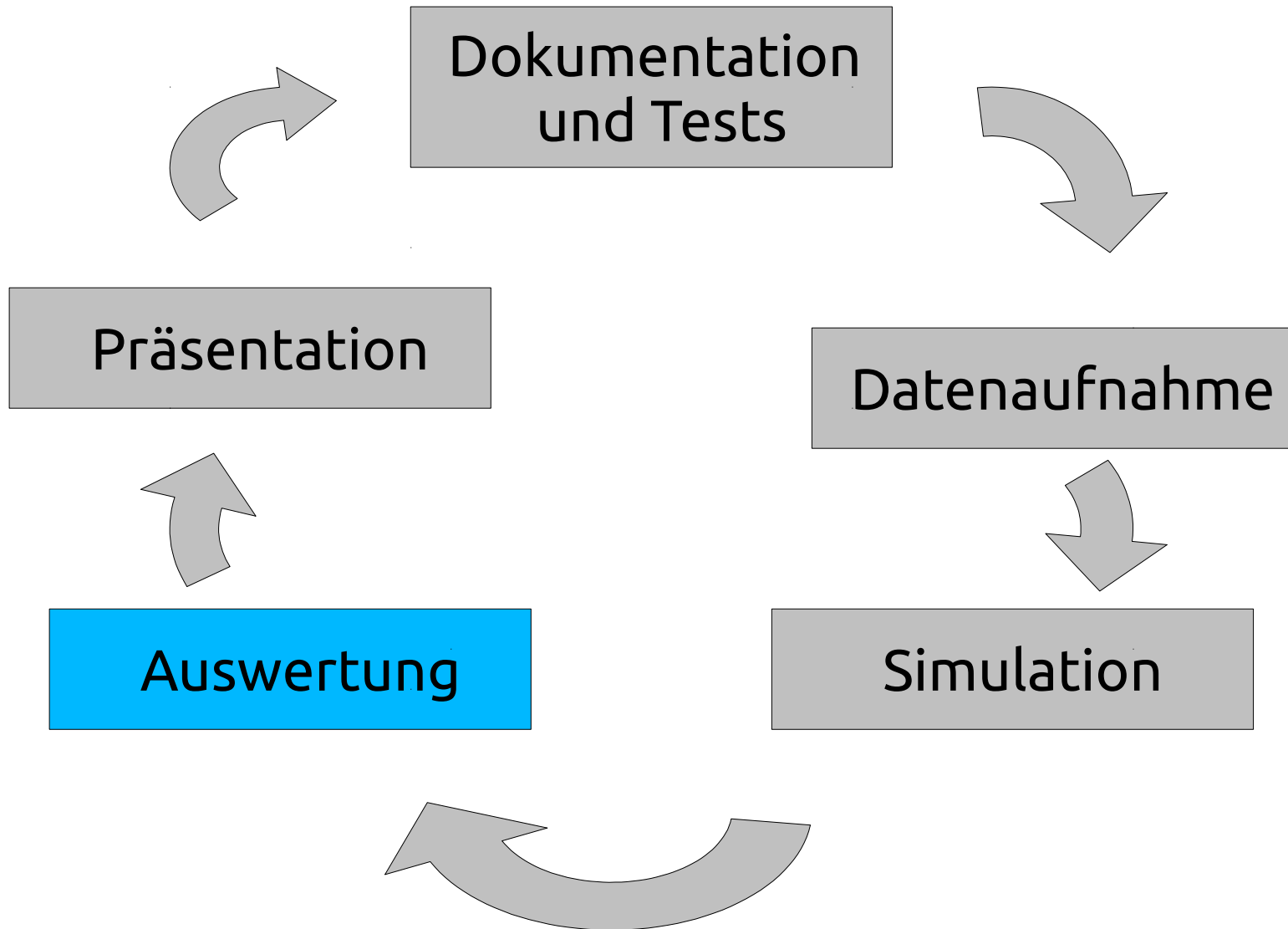
sound1 = tone(1*kHz, .1*second)
sound2 = whitenoise(.1*second)

sound = sound1+sound2
sound = sound.ramp()

sound.play()

cf = erbspace(20*Hz, 20*kHz, 3000)
fb = Gammatone(sound, cf)
output = fb.process()
```

Wissenschaftlicher Workflow



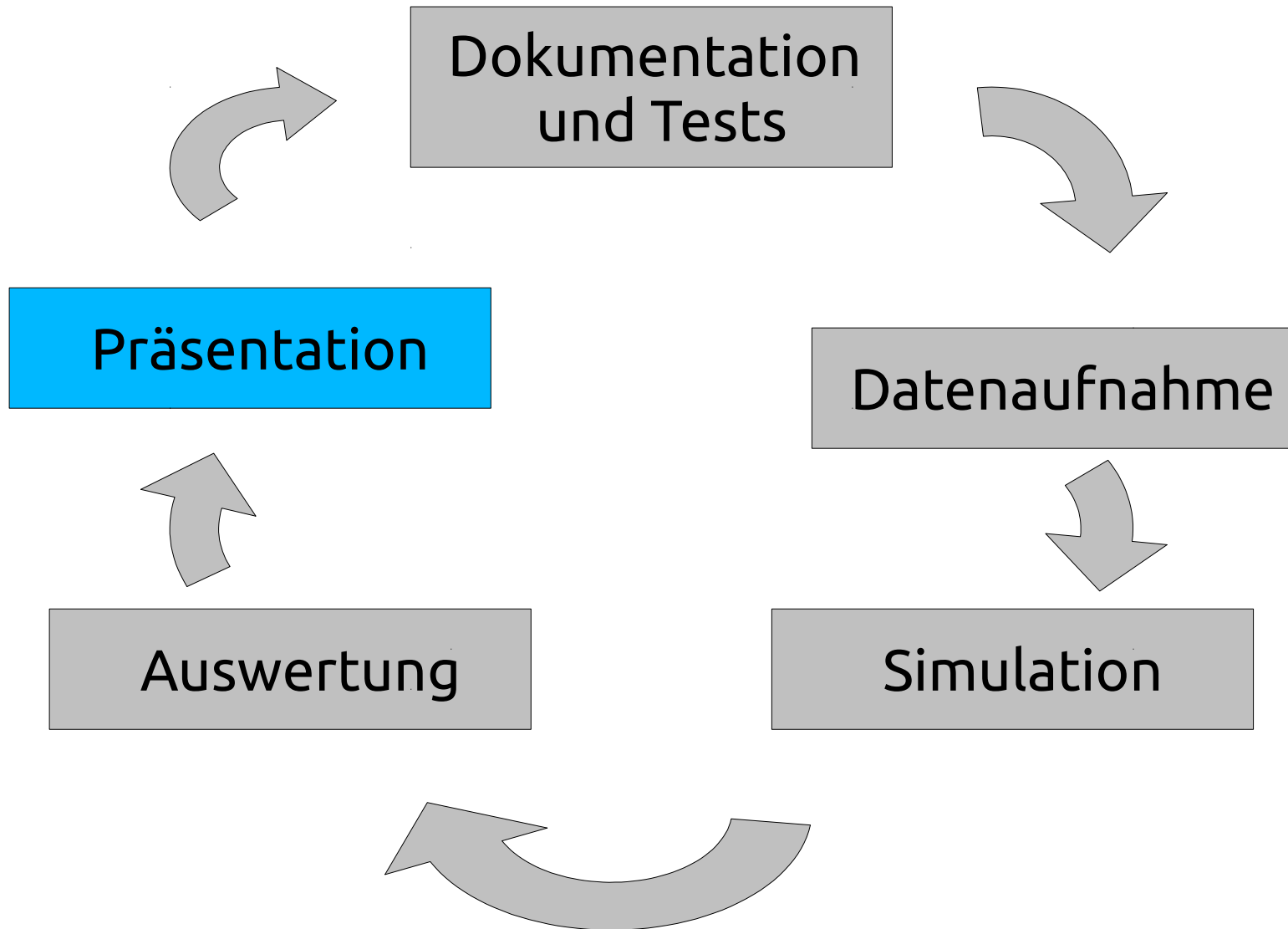
Analyse

- Scipy (<http://www.scipy.org/>)
 - viele Funktionen aus lin. Algebra, Statistik, etc...

```
>>> D = np.diag((-1,1))  
>>> scipy.linalg.eigvals(D)  
array([-1.,  1.]
```

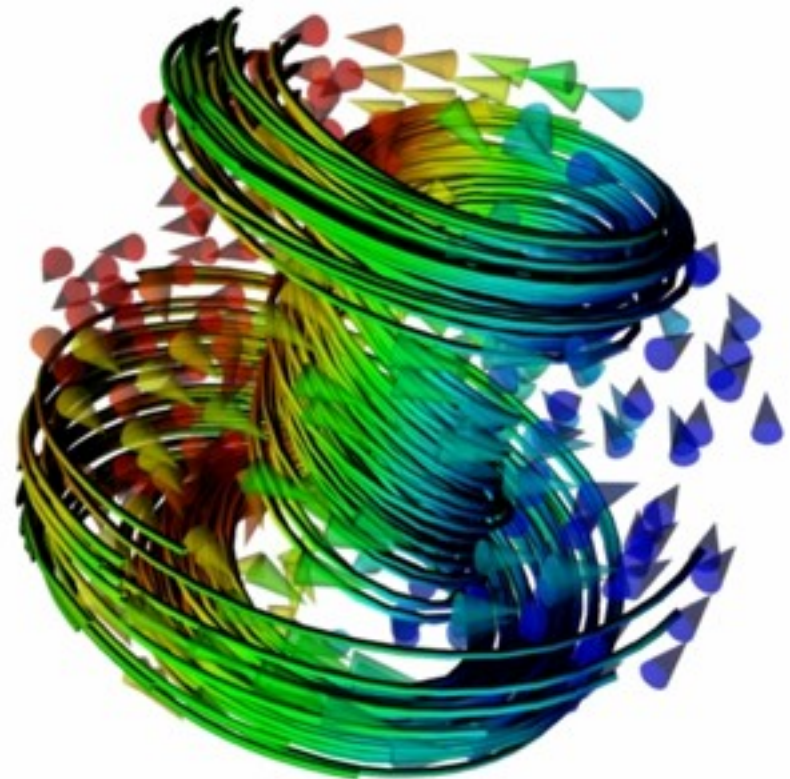
- Ipython (<http://ipython.org/>)
 - verbesserte interaktive Shell
 - ideal für explorative Analyse
 - Skalieren von Einzelrechner bis zum Cluster

Wissenschaftlicher Workflow



Präsentation (2)

- Mayavi2
(<http://code.enthought.com/projects/mayavi/>)
- Hochwertige 3D-Visualisierungen
- Interaktives Interface



Getting started

Python installieren

- Linux:
 - trivial
- Mac OS:
 - Python vorinstalliert (alte Version)
 - Installer von Webseite
 - fink, macports, homebrew, ...
- Windows (seufz):
 - Installer von Webseite
 - Besser: EPD (<http://www.enthought.com/>)

Pakete installieren

- Verzeichnis (fast) aller Pakete: PyPI
<http://pypi.python.org/>
- Installer: easy_install (alt) und pip
<http://www.pip-installer.org/>
- Alternativ: Paket herunterladen und selber installieren

```
# cd meinpaket  
# python setup.py install
```

Further reading

- Dive into Python (<http://diveintopython.org/>)
- Getting Started (offizielles Tutorial)
<http://www.python.org/about/gettingstarted/>
- Neural Ensemble (<http://neuralensemble.org/>)

Python vs. Matlab

- Matlab:
 - Weit verbreitet
 - Gute IDE
 - speziell für Wissenschaftler/Ingenieure entwickelt
- Python:
 - Open Source
 - Moderne Sprache
 - kann viele Funktionen von Matlab imitieren
 - schnell wachsende Community
- Detaillierter Vergleich:
http://www.scipy.org/NumPy_for_Matlab_Users

One last thing...

- Viele dieser Tools werden von Wissenschaftlern entwickelt
- Wer sie verwendet, sollte sie zitieren.
- Zum Beispiel:

```
@Misc{  
  author =    {Eric Jones and Travis Oliphant and Pearu Peterson and others},  
  title =     {{SciPy}: Open source scientific tools for {Python}},  
  year =     {2001--},  
  url = "http://www.scipy.org/"  
}
```

Vielen Dank

<http://bit.ly/py4bio2>

