

```

# simulation.py
# -*- coding: utf-8 -*-
import numpy as np
from scipy.integrate import odeint
import pylab as pl

from data_tools import create_rhs_from_1darr

def simulate(rhs):
    np.random.seed(75)
    xx0 = np.random.rand(rhs.number_of_states)

    res = odeint(rhs, xx0, tt)

    x1 = res[:,0]
    pl.plot(tt, x1)

rhs_list = []

for k in range(1,5):
    fname = "data%i.txt" % k
    try:
        x = np.loadtxt(fname)
    except ValueError, msg:
        print "Fehler:", msg
    else:
        rhs = create_rhs_from_1darr(x)
        rhs_list.append(rhs)

tt = np.linspace(0, 5, 1e3)

#rhs_list = filter(lambda q: q.number_of_states <3, rhs_list)
map(simulate, rhs_list)
pl.show()

#####
# data_tools.py
# -*- coding: utf-8 -*-
import numpy as np

def create_rhs_from_1darr(arr):
    n = arr.shape[0]
    n2 = int(np.sqrt(n))
    arr2 = arr.reshape(n2, -1)

    return rhs_factory(arr2)

def rhs_factory(A):
    n, m = A.shape
    # sicherstellen, dass A quadratisch ist
    assert n == m

    def rhs(state, time):
        x_dot = np.dot(A, state)

        return x_dot

    rhs.number_of_states = m

    return rhs

```