Class EPS 109 "Computer Simulations in Earth and Planetary Science"

Computer lab assignment 1

Instructor: Burkhard Militzer

```
# For all labs and homeworks, please load python's numerical (numpy) and gr
import numpy as np
import matplotlib.pyplot as plt
%config InlineBackend.figure_format = 'retina'
```

(1) Simple calculations in Python

Type the follow commands and see what happens. Press Shift+Return after every line.

Please note this section contains two errors.

```
1+1
```

```
#what does Python to with this funny line?
3 -- 3
```

```
#there are two different way to divide two integers
1001/10
```

```
1001//10
```

```
cos(0)
```

```
np.cos(0)
```

```
np.cos(90)
```

```
np.cos(np.radians(90))
```

```
4^4
```

```
4**4
```

```
1.0e6
```

```
log(10.0)
```

```
np.log(10.0)
```

```
np.log10(10.0)
```

```
np.pi
```

(2) Now define your own variables (This section contains four errors ;=)

```
a=6
```

```
a
```

```
# How many lines does this print?
a
print(a)
print(10*a)
```

```
c = a + b
```

```
d = 90
a + d
```

```
np.sin(2*D)
```

```
np.sin(2*d)
```

```
2d = 2*d
```

(3) Compute the density, ρ, of the Earth by defining two variables for the radius R=6371 km and the mass M=5.9736×10^24 kg. Use the formulas ρ=M/V and volume V=4π/3*R^3. Compare your answer to the ambient densities of water (1 g/cc), rock (2.7 g/cc), and iron (7.8 g/cc).

(4) Vectors in Python. There are three options to choose from: tuples, lists and Numpy arrays

```
# (4a) Let's start with Numpy arrays, which we will use most often in this c
a = np.array( [ 1, 2, 3, 3, 5, 5 ] )
print(a)
print(len(a))
print(a.shape)
print(a.shape[0])
print(type(a))
```

```
# See if any of these commands work
print(2.0*a)
a.append(6)
```

```
# (4b) Now let's use a 'list' that is denoted by [ ]
list = [ 1, 2, 3, 3, 5, 5 ]
print(list)
print(list[1])
print(len(list))
print(type(list))
```

```
# See if any of these commands work
list.append(5)
print(list)
list.append('AA')
print(list)
print(2.0*list)
```

```
# (4c) Now let's try a 'tuple' that is denoted by ( )
# For differences between lists and typles read this page:
# https://www.geeksforgeeks.org/python/python-difference-between-list-and-t
tuple = ( 1, 2, 3, 3, 5, 5 )
print(tuple)
print(tuple[1])
print(len(tuple))
print(type(tuple))
```

```
# See if any of these commands work
tuple.append('AA')
print(tuple)
print(2.0*tuple)
```

```
# Let's practise some more with NumPy arrays because we will use them most
b = np.zeros(15)
print(b)
```

```
# Let us perform some operations on entire vectors.
# see what happens in every case
c = np.ones(6)
print(c)
print(a+c)
print(a*(c+1))
print(np.sqrt(a))
```

(5) Let us generate some XY plots

```
# Compute the x and y coordinates for points on a sine curve
x = np.linspace(0.0, 11.0, 10)
y = np.sin(x)
print(x)
print(y)

# Plot the points using matplotlib
plt.plot(x,y)
plt.show()  # In some cases, you need to call plt.show() to make the graph
```

```
# For a test, please make this curve look very smooth.
# You only need to change one number in the cell above
# but then change it back so that the following commands work.
```

```
y2 = np.cos(x)

# Plot the points using matplotlib
plt.plot(x, y, label='sin(x)')
plt.plot(x, y2,label='cos(x)')
plt.legend()
plt.show()
```

```
#let us make this plot pretty (please use only 10 x values)
plt.plot(x,y*0,'k--',linewidth=1,dashes=(10,3));
plt.plot(x,y,'rD-',linewidth=2,markersize=8,mec='r',
         mew=2, mfc='pink',label="sin(x)");
plt.plot(x,y2,'bo--',linewidth=4,markersize=10,
         mec='b',mew=2, mfc='white',dashes=(5,1,1,1),label="cos(x)");
plt.fill_between(x,y,y*0,color=(1.0,0.94,0.94),lw=0)
plt.xlabel('My instructor made me add this axis.')
plt.ylabel('My dog ate my online homework.')
plt.legend()
plt.show()
```

(6) Let us generate a 3D plot

```
n = 51
L = 10.0
x = np.linspace(-L, L, n)
y = np.linspace(-L, L, n)
X, Y = np.meshgrid(x, y)
```

```
R = np.sqrt(X**2+Y**2) + 1e-14 # Why do we need to add 10^-14 ?
Z = np.sin(R) / R
```

```
#the following 3D plot requires some extra libraries
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
from matplotlib.ticker import LinearLocator, FormatStrFormatter

fig = plt.figure(figsize=(10,10))
ax = fig.gca(projection='3d')
plt.rcParams['figure.figsize'] = [10, 10]
#Try both plot command one after the other
#surf = ax.plot_surface(X, Y, Z, cmap=cm.coolwarm,
#                       linewidth=0, antialiased=False)
surf = ax.plot_surface(X, Y, Z, cmap='viridis',
                       linewidth=10, antialiased=False,rcount=n)
#resolution is specified by 'rcount' or 'ccount'
plt.show()
```

(7) Load data file that contains the Earth temperature record in columns 3 (time in years) and 5 (temperature in Kelvin)

```
data = np.loadtxt('ice_core_temperature_data.txt', usecols=(2,4))
#why 2 and 4 instead of 3 and 5?
print(data)
```

```
# There is a good chance that the np.loadtxt command above
# has failed because the data file was not in the right folder.
# Use 'pwd' (for print working directory) and 'ls' (for list)
# to determine the the current folder and the files it contains.
pwd
ls
```

```
plt.plot(data[:,0],data[:,1],'r-',linewidth=1,markersize=0,
         mec='r',mew=2, mfc='pink',label="sin(x)");
plt.show()
```

```
#Hey, this plot is too small. I cannot see anything!
plt.rcParams['figure.figsize'] = [15, 5]
plt.plot(data[:,0],data[:,1],'r-',linewidth=1,markersize=0,
         mec='r',mew=2, mfc='pink',label="sin(x)");
plt.show()
```

```
temp = data[:,1]
print(sum(temp))
print(temp.shape)
average = sum(temp) / temp.shape[0]
print(average)
```

```
plt.rcParams['figure.figsize'] = [15, 5]
plt.plot(data[:,0],data[:,1],'r-',linewidth=1,
         markersize=0,mec='r',mew=2, mfc='pink',label="sin(x)");
plt.fill_between(data[:,0],data[:,1],
                 average,color=(1.0,0.90,0.90),lw=0)
plt.xlabel('Years before 1950')
plt.ylabel(r'Temperature - reference temperature [K]')
plt.show()
```

(8) Optional part for geeks only: Compare the role of the different import statements. In this course, we will always use "import numpy as np" but there are other options for you to try and know about. The following three import statement are all valid but

they imply that the cos() function and all other numpy functions must be called in different ways. So figure out which one is the appropriate print statement in every case. Please restart your kernel after each attempt to erase any previosuly loaded package.

```
import numpy as np
print(cos(0.0))
print(np.cos(0.0))
print(nympy.cos(0.0))
```

```
import numpy
print(cos(0.0))
print(np.cos(0.0))
print(numpy.cos(0.0))
```

```
from numpy import *
print(cos(0.0))
print(np.cos(0.0))
print(numpy.cos(0.0))
```