インタラクティブシステム論

「数値計算ソフト SciLabに慣れる」で使ったScilab
コードの、Pythonによる書き換え

梶本裕之
Twitter ID kajimoto
ハッシュタグ #ninshiki

- Pythonを使う人は、Pythonの環境構築を自力で出来、関数を自力で調べられる人に限ります。
- 授業課題だけに限れば、Scilabの方が（環境構築も含め）楽です。Pythonに関してはこの授業ではサポート対象外です。
- 授業課題はScilabかPythonのどちらかで回答してください。

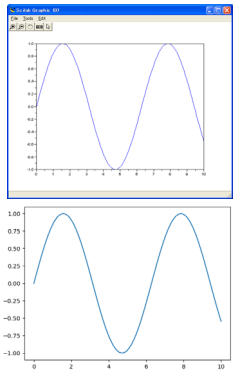
はじめの一歩（2）プロットしてみる

Scilab

```
x=[0:0.1:10];
y=sin(x);
plot (x,y);
```

Python

```
import matplotlib.pyplot as plt
import math
import numpy as np
x = np.linspace(0,10,100)
y = np.sin(x)
plt.plot(x,y)
plt.show()
```



課題：行列

次の行列の
（1）逆行列を求め
（2）逆行列と元の行列をかけると単位行列になることを確認

A=[0 1
1 2]

Scilab

```
A=[0,1;1,2]
B=inv(A)
A*B
```

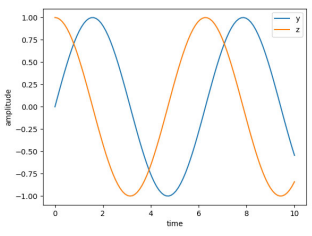
Python

```
import numpy as np
A = np.array([[0,1],[1,2]])
B = np.linalg.inv(A)
C = np.dot(A,B)
print(C)
```

2次元グラフ(2)：書式(python)

複数個のグラフを描き、x軸、y軸のラベルを書き、凡例を貼る

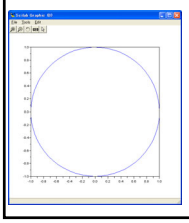
```
import matplotlib.pyplot as plt
import math
import numpy as np
x = np.linspace(0,10,100)
y = np.sin(x)
z = np.cos(x)
plt.plot(x,y,label="y")
plt.plot(x,z,label="z")
plt.xlabel("time")
plt.ylabel("amplitude")
plt.legend()
plt.show()
```



授業中課題：円を描く

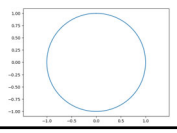
Scilab

```
rad = [0:0.1:2*%pi];
x=cos(rad);
y=sin(rad);
plot(x,y);
```



Python

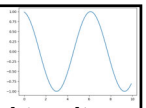
```
import matplotlib.pyplot as plt
import math
import numpy as np
rad = np.linspace(0,2.0*math.pi,100)
x = np.sin(rad)
y = np.cos(rad)
plt.plot(x,y)
plt.axes().set_aspect('equal','datalim')
plt.show()
```

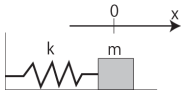


レポート課題（1）：リサージュ図形

「リサージュ図形」について調べ，
plot関数で描いてみよ
（ヒント：円を描くのとほとんど同じ）

制御文：ばねの挙動をシミュレート





Scilab

```
m=1.0;   //重さ
k=1.0;   //ばね定数
x=1.0;   //初期位置
v=0;     //初期速度
dt=0.1;  //時間刻み
record=[];//記録用
for time= 0:dt:10 //時刻
    F=-k*x;  //ばねによって生じる力
    a=F/m;  //生じる加速度
    v= v+a*dt; //速度
    x= x+v*dt;  //位置
    record = [record,x]; //記録
end
plot([0:dt:10],record);
```

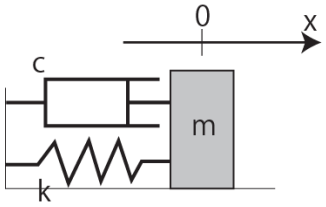
Python

```
import matplotlib.pyplot as plt
import math
import numpy as np
m = 1.0
k = 1.0
x = 1.0
v = 0.0
dt = 0.1
time = np.arange(0,10.0,dt)
record = list()
for t in time:
    print(t)
    F = -k*x
    a = F/m
    v = v+a*dt
    x = x+v*dt
    record.append(x)
plt.plot(time,record)
plt.show()
```

レポート課題(2)

速度に比例したブレーキ（ダンパ，粘性）が加わったときの様子をシミュレートせよ．
（ヒント：力の部分が変わる）

バネマス系：F=ma=-kx
バネマスダンパ系：F=ma=-kx-cv



Pythonでの音の扱い(1) 正弦波再生

```
import numpy as np
import simpleaudio as sa

frequency = 440
fs = 44100
seconds = 3
t = np.linspace(0, seconds, seconds * fs, False)
note = np.sin(frequency * t * 2 * np.pi)
audio = note * (2**15 - 1) / np.max(np.abs(note))
audio = audio.astype(np.int16)
play_obj = sa.play_buffer(audio, 1, 2, fs)
play_obj.wait_done()
```

参考
https://realpython.com/playing-and-recording-sound-python/#playsound

Pythonでの音の扱い(2) waveファイル再生

```
import simpleaudio as sa

wave_obj = sa.WaveObject.from_wave_file("aiueo.wav")
play_obj = wave_obj.play()
play_obj.wait_done()
```

参考 https://simpleaudio.readthedocs.io/en/latest/
Wavファイルを置いたディレクトリに移動して実行する。

レポート課題（3）：余裕のある場合のみ

簡単な音楽を作成せよ．提出はwaveファイルで
はなくプログラムファイルで。

（ヒント）ドレミファソラシドの各周波数を調べる。

Pythonでの音の扱い(3) waveファイルの出力

```
import numpy as np
import simpleaudio as sa
import wave
import struct
frequency = 440
fs = 44100
seconds = 3
t = np.linspace(0, seconds, seconds * fs, False) # make wave
note = np.sin(frequency * t * 2 * np.pi)
audio = note * (2**15 - 1) / np.max(np.abs(note))
audio = audio.astype(np.int16) # format to 16bit int
play_obj = sa.play_buffer(audio, 1, 2, fs) # play audio
play_obj.wait_done()
bi_wave = struct.pack("h" * len(audio), *audio)
ww = wave.open("out.wav", 'w')
ww.setnchannels(1)
ww.setsampwidth(2) #16bit
ww.setframerate(fs)
ww.writeframes(bi_wave) # output wave file
ww.close()
```