ISM MPSI/PCSI, Informatique
Année 2024/2025

Corrigé du TP Informatique 7

Exercice 1

1. On saisit :

```
tx=np.linspace(-6,6,10)
ty=[np.sin(x) for x in tx]
plt.plot(tx,ty)
plt.grid();plt.show()
```

2. On saisit :

```
tx=np.linspace(-6,6,100)
ty=[np.sin(x) for x in tx]
plt.plot(tx,ty)
plt.grid();plt.show()
```

3. On saisit :

```
tx=np.linspace(-6,6,100)
ty=[np.sin(x) for x in tx]
plt.plot(tx,ty,label='y=sin(x)')
plt.legend()
plt.xlabel('Temps (s)');plt.ylabel('Position (m)')
plt.grid();plt.show()
```

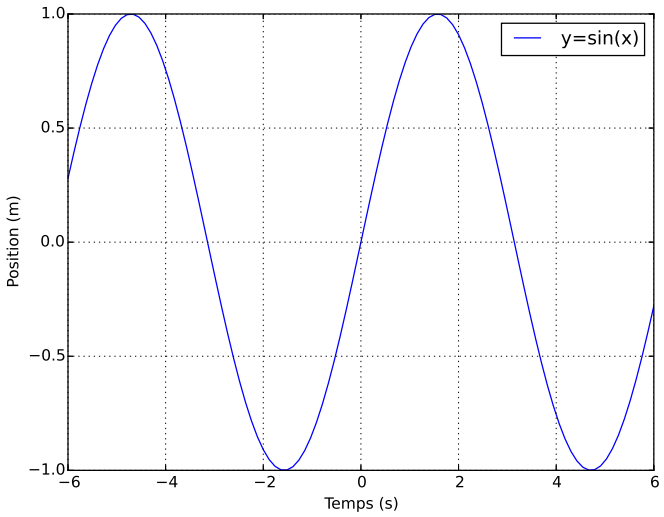

FIGURE 1 – Graphe de la fonction sin

4. On saisit :

```
tx=np.arange(0,10,.1)
ty=[np.sin(x) for x in tx]
plt.plot(tx,ty)
plt.grid();plt.show()
```

Exercice 2

1. On saisit :

```
tx=np.linspace(-6,6,100)
ty=[np.sin(x) for x in tx]
tz=[np.cos(x) for x in tx]
plt.plot(tx,ty,tx,tz)
plt.grid();plt.show()
```

2. On saisit :

```
tx=np.linspace(-6,6,100)
ty=[np.sin(x) for x in tx]
tz=[np.cos(x) for x in tx]
plt.plot(tx,ty,'b',linewidth=4)
plt.plot(tx,tz,'r--')
plt.grid();plt.show()
```

3. On saisit :

```
tx=np.linspace(-6,6,100)
ty=[np.sin(x) for x in tx]
tz=[np.cos(x) for x in tx]
plt.plot(tx,ty,'b',linewidth=4,label='y=sin(x)')
plt.plot(tx,tz,'r--',label='y=cos(x)')
plt.legend()
plt.xlabel('Temps (s)')
plt.ylabel('Position (m)')
plt.grid();plt.show()
```

4. On saisit :

```
tx=np.linspace(-6,6,100)
ty=[np.sin(x) for x in tx]
tz=[np.cos(x) for x in tx]
plt.plot(tx,ty,'b',linewidth=4,label='y=sin(x)')
plt.plot(tx,tz,'r--',label='y=cos(x)')
plt.legend()
plt.xlabel('Temps (s)')
plt.ylabel('Position (m)')
plt.axis([-7,7,-2,2])
plt.grid();plt.show()
```

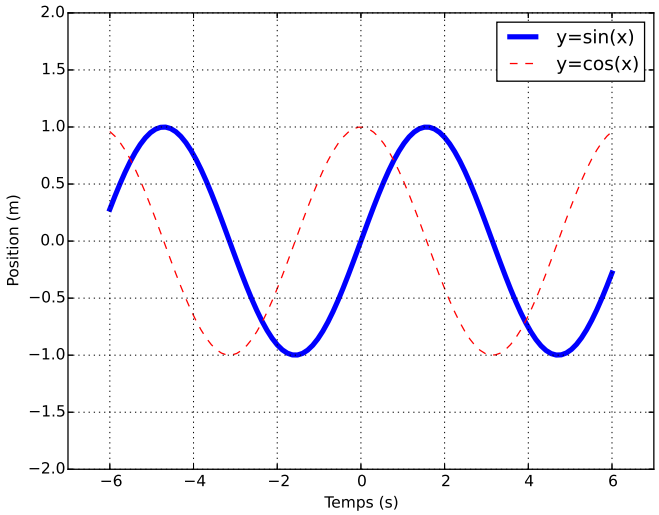

FIGURE 2 – Graphes des fonctions sin et cos

Exercice 3

1. On saisit :

```
def f(x):
    if x==0:
        return 0
    else:
        return x*np.sin(1/x)

tx=np.linspace(-1,1,1000)
ty=[f(x) for x in tx]
plt.plot(tx,ty)
plt.grid();plt.show()
```

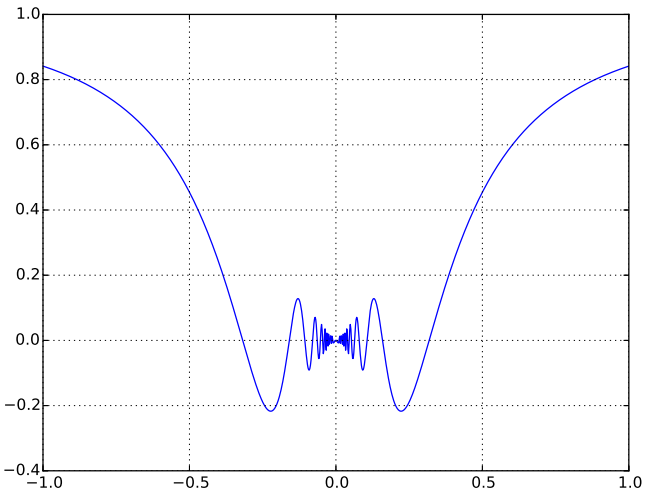

FIGURE 3 – Graphe de la fonction f

2. On saisit :

```
def g(x):
    if x<0:
        return 1-x
    elif x<1:
        return 1+x*(x-1)
    else:
        return x

tx=np.linspace(-3,3,100)
ty=[g(x) for x in tx]
plt.plot(tx,ty)
plt.grid();plt.show()
```

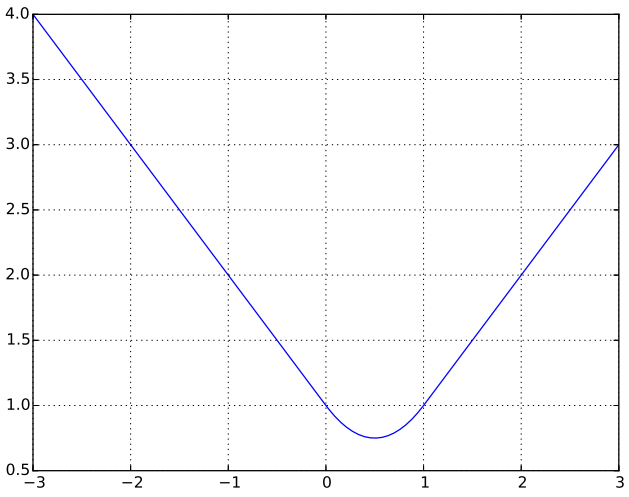

FIGURE 4 – Graphe de la fonction g

Exercice 4

1. On saisit :

```
def u(n):
    res=0
    for k in range(2,n+1):
        res+=np.log(k)
    return res

tn=range(1,100,10)
tu=[u(n) for n in tn]
plt.plot(tn,tu,'bo--')
plt.grid();plt.show()
```

2. On saisit :

```
tn=range(100,10000,500)
tq=[u(n)/(n*np.log(n)) for n in tn]
plt.plot(tn,tq,'bo--')
plt.grid();plt.show()
```

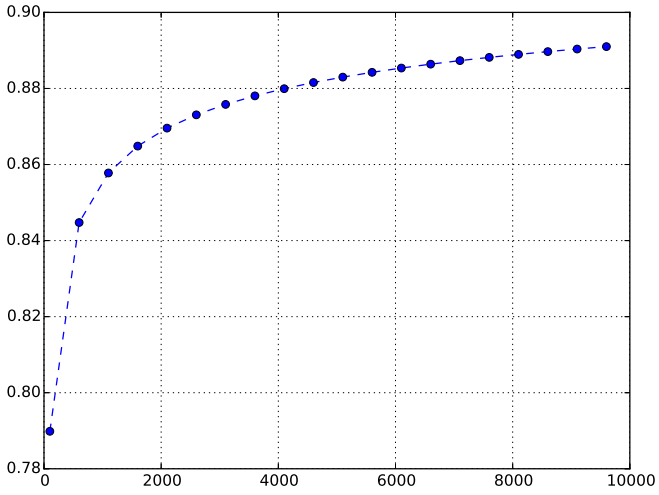

FIGURE 5 – Tracé de la suite $(v_n)_n$

On conjecture $\boxed{v_n \underset{n\to\infty}{\longrightarrow} 1}$

Ce résultat s'interprète ainsi $\ln(n!) \underset{n\to+\infty}{\sim} n\ln n$

On peut le démontrer notamment par des techniques de comparaison somme/intégrale.

Exercice 5

1. On saisit :

```
tn=range(1,100)
tu=[n**2 for n in tn]
tln=[np.log(n) for n in tn]
tlu=[np.log(u) for u in tu]

plt.plot(tln,tlu,'bo--')
plt.grid();plt.show()
```

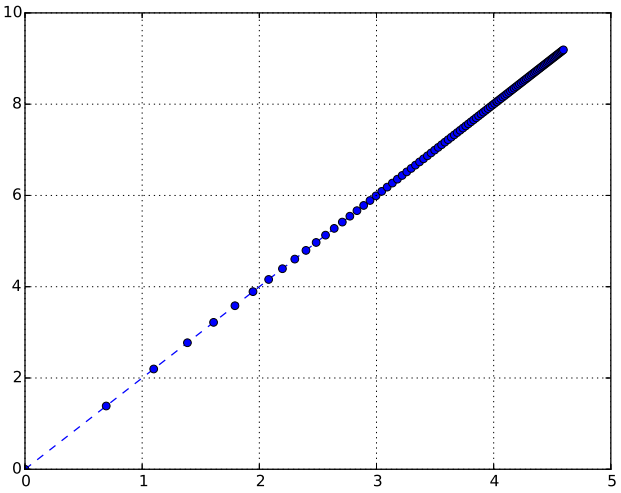

FIGURE 6 – Tracé logarithmique en abscisse et ordonnée de la suite $(u_n)_n$

2. On saisit :

```
tn=range(1,30)
tv=[2**n for n in tn]
tlv=[np.log(v) for v in tv]

plt.plot(tn,tlv,'bo--')
plt.grid();plt.show()
```

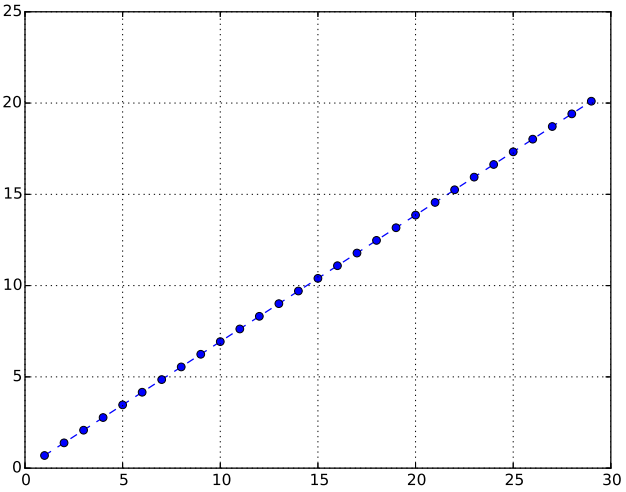

FIGURE 7 – Tracé logarithmique en ordonnée de la suite $(v_n)_n$

Exercice 6

1. On saisit :

```
tt=np.linspace(0,2*np.pi,100)
tx=[np.cos(t)**3 for t in tt]
ty=[np.sin(t)**3 for t in tt]
plt.plot(tx,ty,linewidth=4)
plt.axis('equal')
plt.grid();plt.show()
```

2. On saisit :

```
ta=np.linspace(0,1,100)

for t in tt:
    tx=[(1-a)*np.sin(t) for a in ta]
    ty=[a*np.cos(t) for a in ta]
    plt.plot(tx,ty)

plt.axis('equal')
plt.grid();plt.show()
```

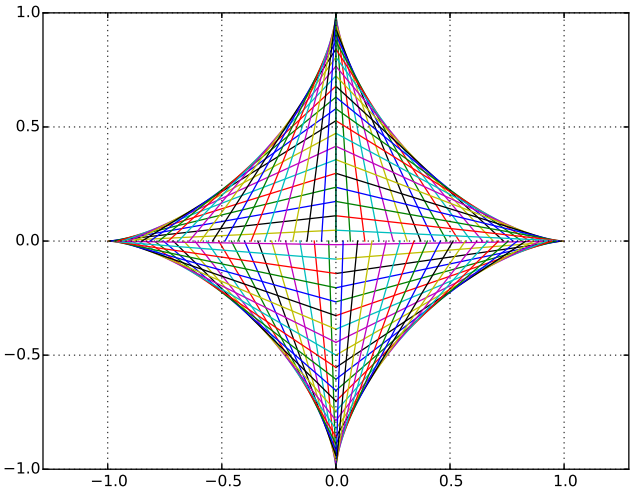

FIGURE 8 – Familles des segments d'extremités $(0, \cos t)$, $(\sin t, 0)$

On observe que la famille des segments vient envelopper la première courbe paramétrée appelée *astroïde*. On peut observer cette trajectoire dans la vie courante : lors de l'ouverture d'une porte de bus, celle-ci vient envelopper un quart d'astroïde.

Exercice 7

1. On saisit :

```
def u(n):
    res=[3]
    for k in range(n):
        res.append(np.sqrt(2+res[-1]))
    return res

n=20
tn=range(n+1)
tu=u(n)
plt.plot(tn,tu,'bo--')
plt.grid();plt.show()
```

Sans difficulté, on observe $\boxed{u_n \xrightarrow[n\to\infty]{} 2}$

On peut le démontrer par une étude classique de suite.

2. On saisit :

```
tlb=[np.log(abs(x-2)) for x in tu]
plt.plot(tn,tlb,'bo--')
plt.grid();plt.show()
```

On observe une progression linéaire avec une pente $\simeq -1.4$ ce qui laisse penser qu'on a

$$|u_n - 2| \underset{n\to+\infty}{\sim} \frac{\mathrm{C}}{1.4^n} \quad \text{avec} \quad \mathrm{C} > 0$$

On peut démontrer qu'on a en fait $|u_n - 2| \underset{n\to+\infty}{\sim} \frac{\mathrm{C}}{4^n}$ mais cette convergence est trop rapide pour être observée numériquement.