Programme Python pour démonstration du théorème de Morley par Alain Connes

(Denise Vella-Chemla, 28.8.2020)

On voudrait fournir ici des résultats surprenants qui nous ont été inspirés par un travail tout l'été autour de la preuve par Alain Connes du théorème de Morley ([1]).

En effet, on souhaitait programmer la preuve pour essayer de vérifier que le théorème n'est pas applicable pour les triangles sphériques (voir notamment la conférence [2]).

Programme de vérification et visualisation du Théorème de Morley dans le plan

```
from math import *
from matplotlib import *
from matplotlib.pyplot import *

def vecteur(point1, point2):
  return [y - x for x, y in zip(point1, point2)]

def add(vecteur1, vecteur2):
  return [x + y for x, y in zip(vecteur1, vecteur2)]

def norme(vecteur):
  return sqrt(prodscal(vecteur, vecteur))

def prodscal(vecteur1, vecteur2):
  return sum([x*y for x, y in zip(vecteur1, vecteur2)])

def determ(vecteur1, vecteur2):
 return vecteur1[0]*vecteur2[1]-vecteur1[1]*vecteur2[0]

def angle(vecteur1, vecteur2):
  cosinus=prodscal(vecteur1, vecteur2)/(norme(vecteur1)*norme(vecteur2))
  sinus=determ(vecteur1, vecteur2)/(norme(vecteur1)*norme(vecteur2))
  return atan2(sinus,cosinus)

def rotation(u, theta):
  return [u[0]*cos(theta)-u[1]*sin(theta),u[0]*sin(theta)+u[1]*cos(theta)]

def intersecte(x, y, z, t):
  a1 = (y[1]-x[1])/(y[0]-x[0])
  b1 = x[1]-a1*x[0]
  a2 = (t[1]-z[1])/(t[0]-z[0])
  b2 = z[1]-a2*z[0]
  return [(b2-b1)/(a1-a2),((b2-b1)/(a1-a2))*a1+b1]

a=[8.83, 47.89]
b=[72.74, 8.55]
c=[90.72, 86.63]
ab = vecteur(a,b) ; ac = vecteur(a,c)
ba = vecteur(b,a) ; bc = vecteur(b,c)
ca = vecteur(c,a) ; cb = vecteur(c,b)
anglea=angle(ab,ac)  ; angleb=angle(bc,ba)  ; anglec=angle(ca,cb)
aprime = add(a, rotation(ab,anglea/3.0))  ; aseconde = add(a, rotation(ab,anglea*2.0/3.0))
bprime = add(b, rotation(bc,angleb/3.0)) ; bseconde = add(b, rotation(bc,angleb*2.0/3.0))
cprime = add(c, rotation(ca,anglec/3.0)) ; cseconde = add(c, rotation(ca,anglec*2.0/3.0))
p=intersecte(a,aseconde,b,c) ; q=intersecte(a,aprime,b,c)
r=intersecte(c,cseconde,a,b) ; s=intersecte(c,cprime,a,b)
t=intersecte(b,bseconde,c,a) ; u=intersecte(b,bprime,c,a)
n=intersecte(a,aseconde,c,cprime)
o=intersecte(c,cseconde,b,bprime)
x=intersecte(b,bseconde,a,aprime)
print('Normes des cotes %3.15f '% norme(vecteur(n,o)))
print('Normes des cotes %3.15f '% norme(vecteur(o,x)))
print('Normes des cotes %3.15f '% norme(vecteur(x,n)))

fig = matplotlib.pyplot.figure()
ax = fig.add_subplot(111)
matplotlib.pyplot.plot([a[0],b[0],c[0],a[0]],[a[1],b[1],c[1],a[1]], 'g', alpha=0.7)
matplotlib.pyplot.fill([a[0],p[0],q[0]],[a[1],p[1],q[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([c[0],r[0],s[0]],[c[1],r[1],s[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([b[0],t[0],u[0]],[b[1],t[1],u[1]], 'g', 2, alpha=0.4)
matplotlib.pyplot.fill([n[0],o[0],x[0]],[n[1],o[1],x[1]], 'g', 2, alpha=0.8)
matplotlib.pyplot.xlim(0,100)
matplotlib.pyplot.ylim(0,100)
ax.set\_aspect('equal')
matplotlib.pyplot.show()
```

Le programme ci-dessus produit la visualisation ci-après, et imprime comme longueur des normes des côtés du triangle de Morley (le triangle équilatéral de sommets les intersections des trissectrices adjacentes du triangle quelconque "externe") la valeur 14.863746806168091 pour deux côtés et la valeur 14.863746806168082 pour