Courbes_collision

November 29, 2022

Probabilité qu'il n'y ait pas de collision

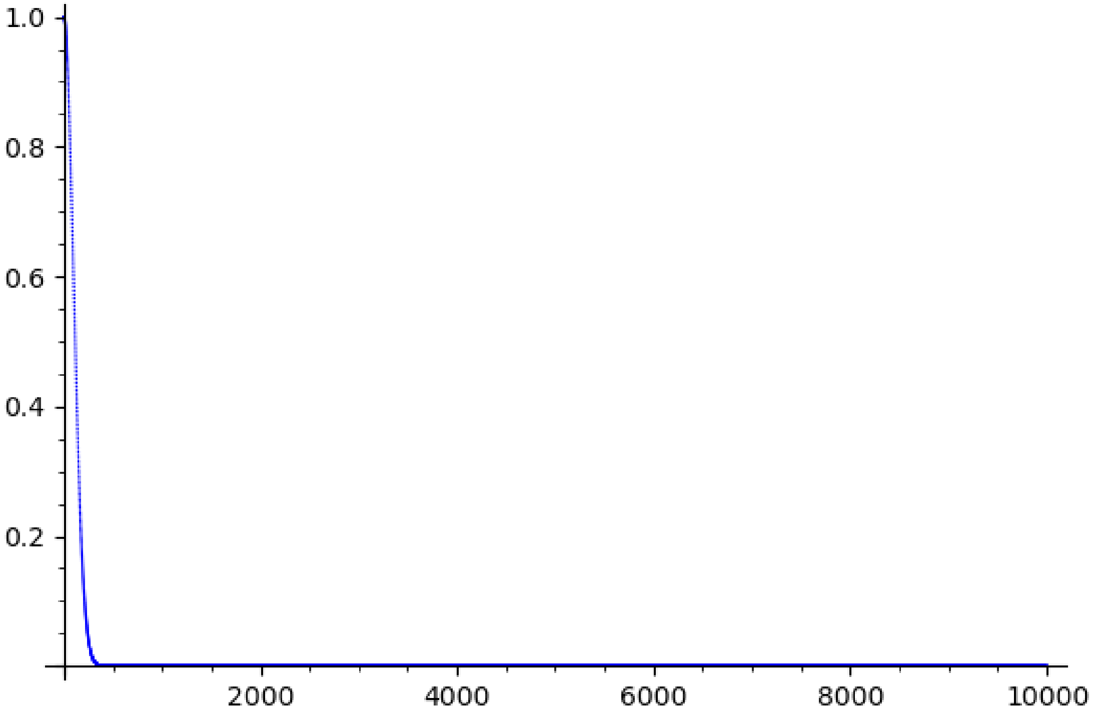
[1]:

```
m = 10**4
Pts = []
proba = 1
for n in range(m):
    proba = proba * ( (m-n) / m )
    Pts.append((n, proba))
```

[2]:

```
point(Pts, size=1)
```

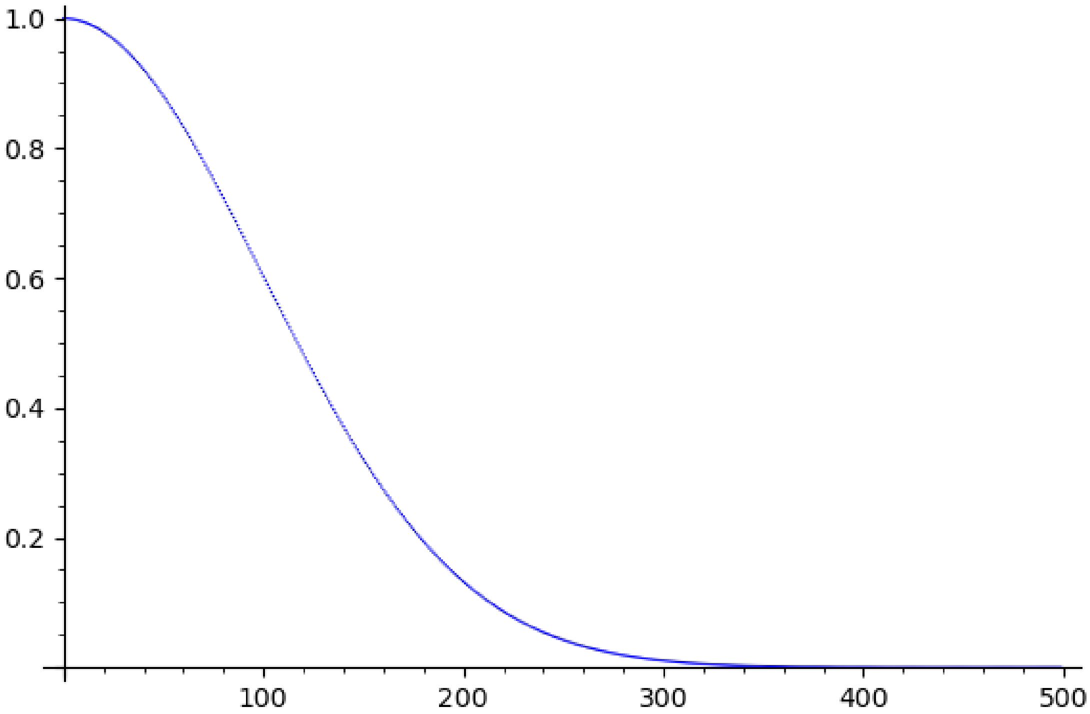
[2]:



[3]:

```
point(Pts[:500], size=1)
```

[3]:

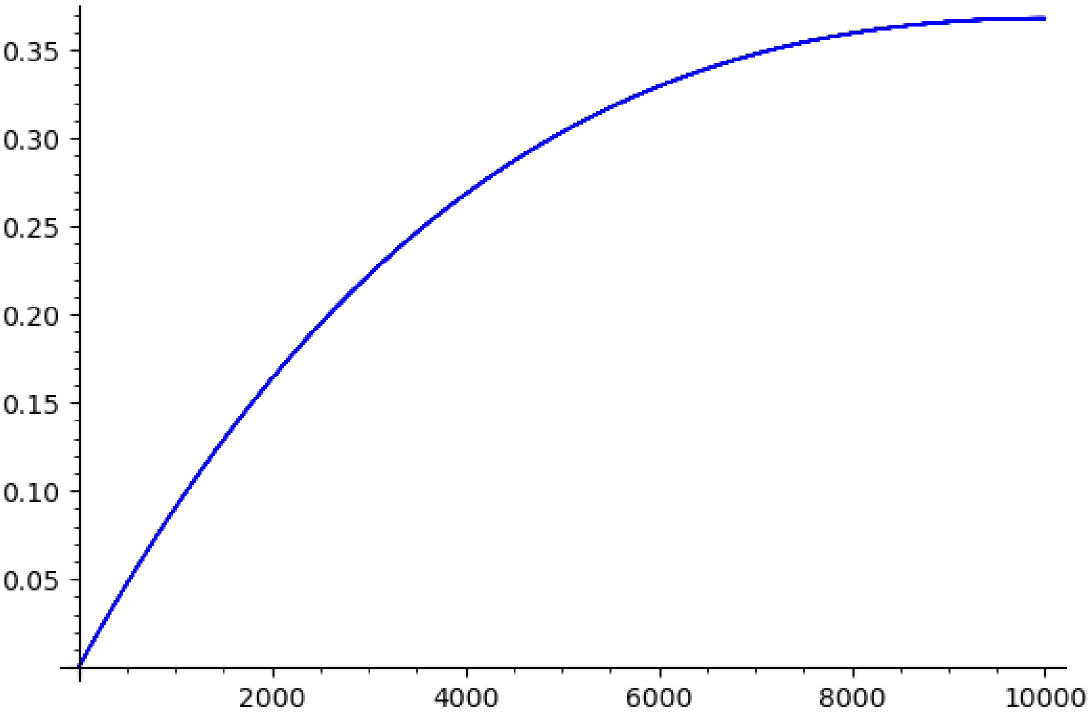


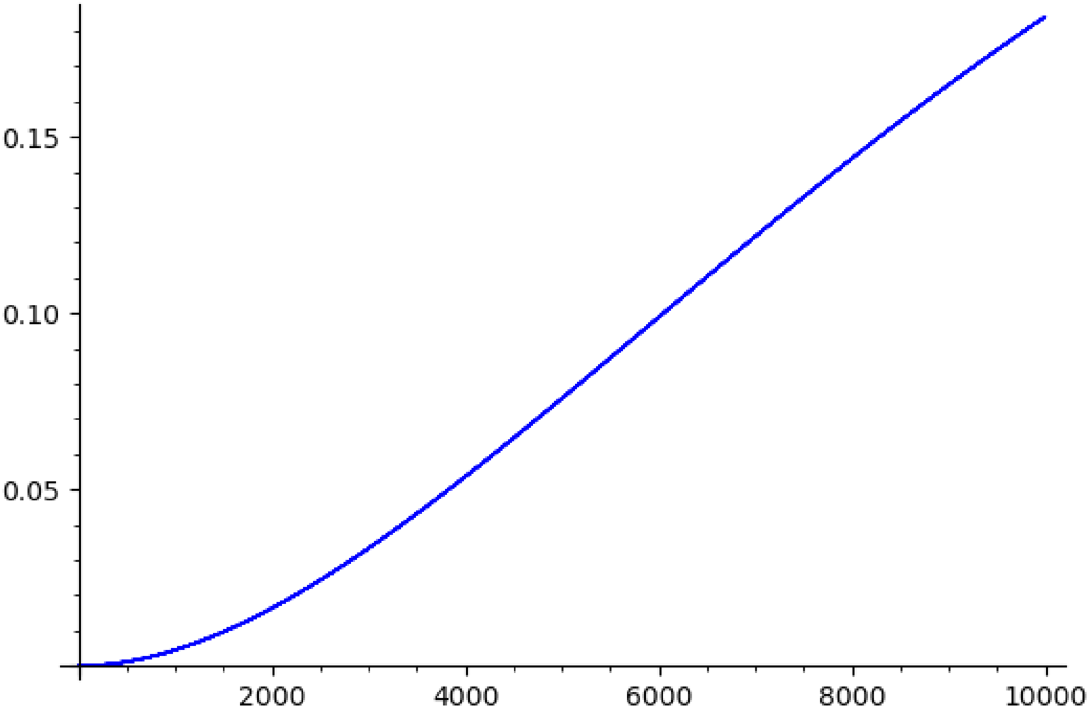
Probabilité que k clé aient la même valeur de hachage fixée

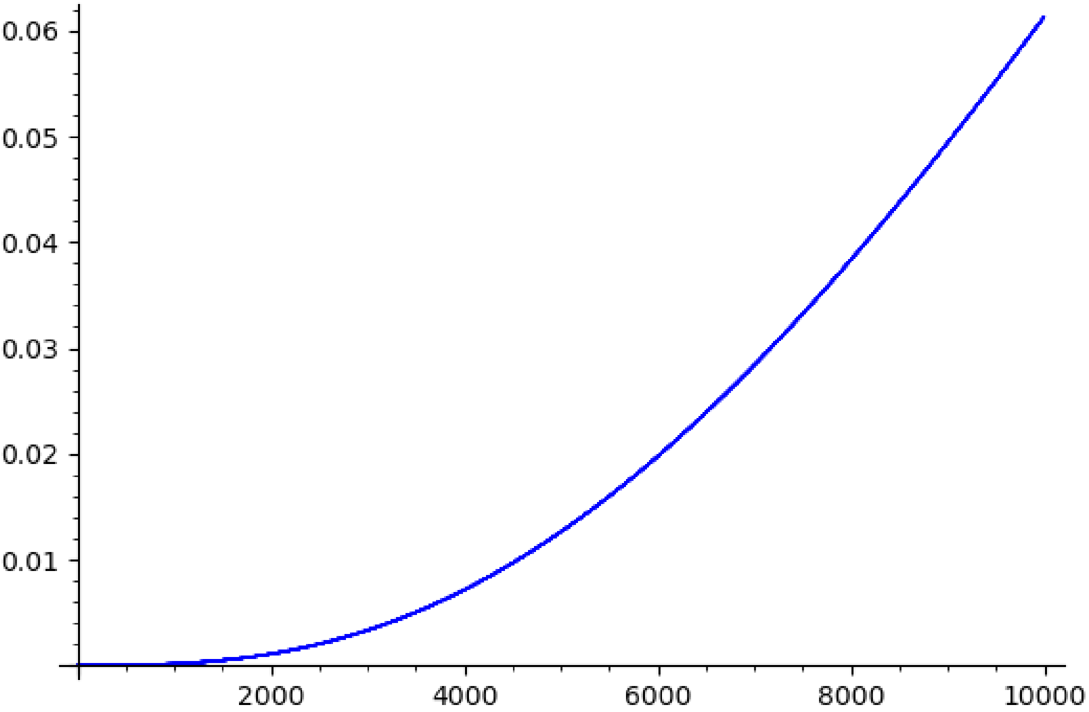
(en supposant un hachage uniforme)

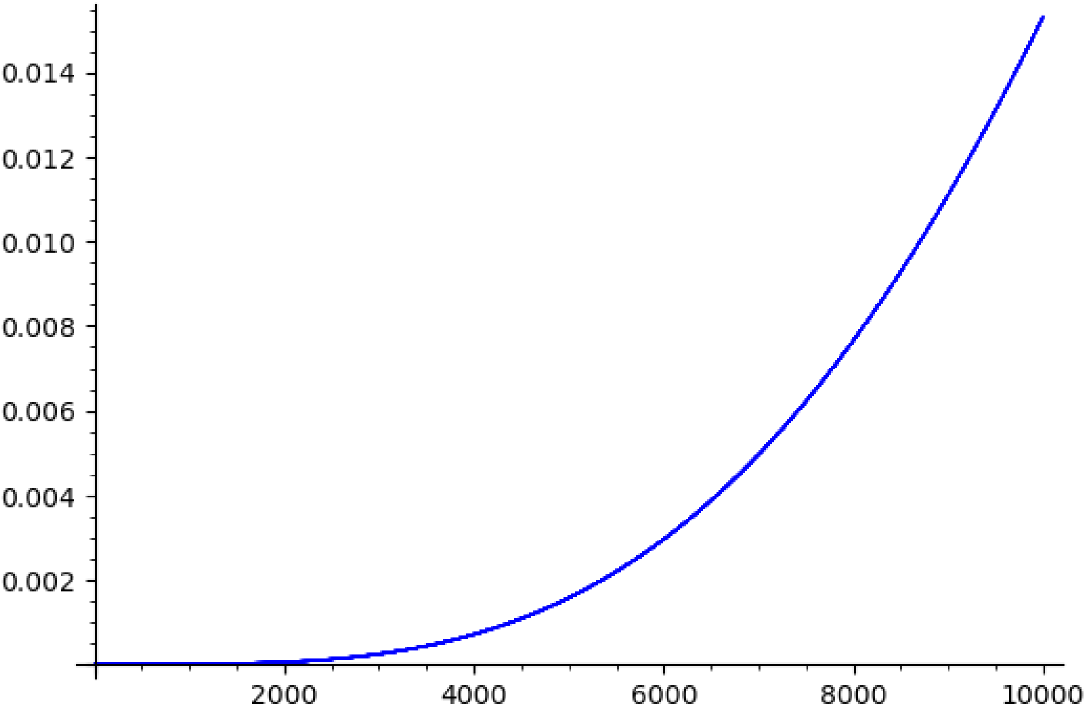
[4]:

```
m = 10**4
for k in range(1, 5):
    Pts = []
    proba = 1 / m**k
    for n in range(k, m):
        Pts.append((n, proba))
        proba = proba * ( (n+1)/(n+1-k) * (m-1) / m )
    point(Pts, size=1).show()
```


0.35
0.30
0.25
0.20
0.15
0.10
0.05
2000
4000
6000
8000
10000


0.15
0.10
0.05
2000
4000
6000
8000
10000

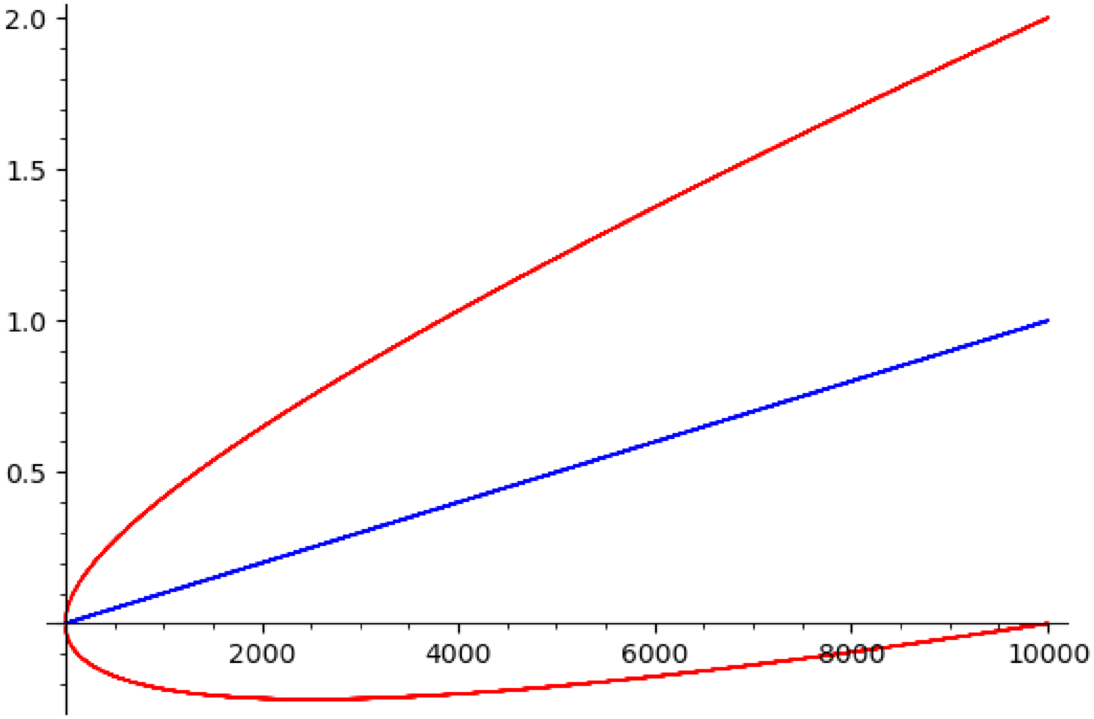

0.06
0.05
0.04
0.03
0.02
0.01
2000
4000
6000
8000
10000


0.014
0.012
0.010
0.008
0.006
0.004
0.002
2000
4000
6000
8000
10000

Moyenne et écart-type du nombre de clés par case

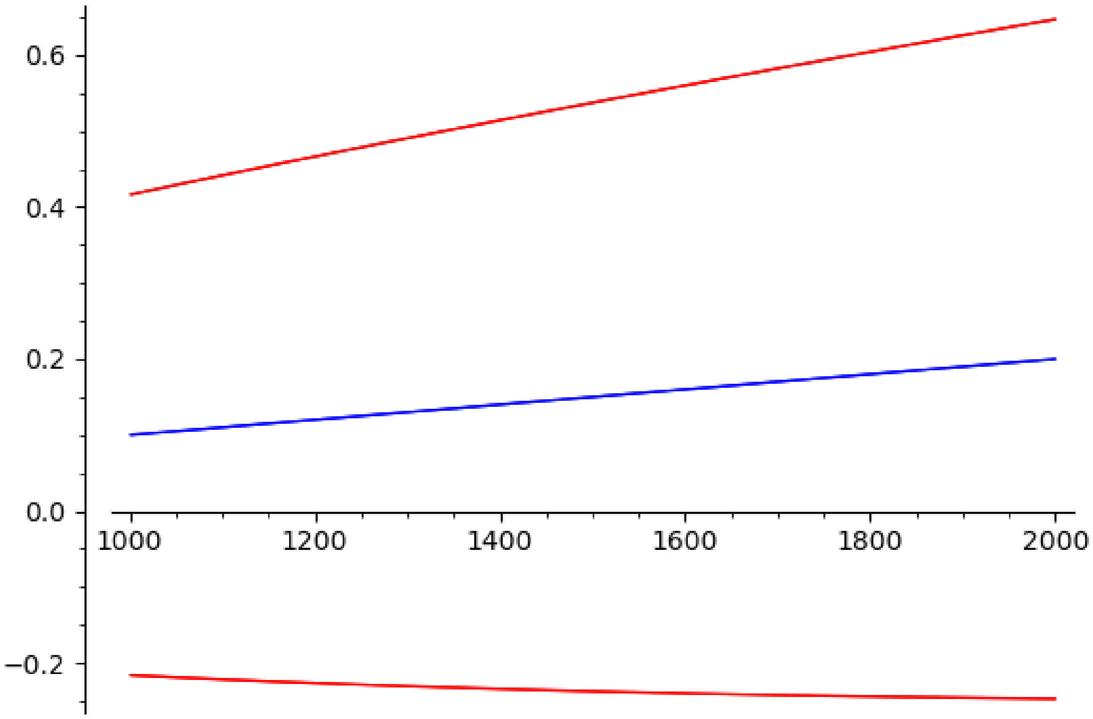
[9]:

```
m = 10**4
Pts = []
Pts2 = []
Pts3 = []
for n in range(1,m):
    proba = n/m
    ecart = sqrt(proba * ((m-1)/m))
    Pts.append((n, proba))
    Pts2.append((n, proba+ecart))
    Pts3.append((n, proba-ecart))
    p1 = point(Pts, size=1)
p2 = point(Pts2, size=1, color='red')
p3 = point(Pts3, size=1, color='red')
show(p1+p2+p3)
```



[10]:

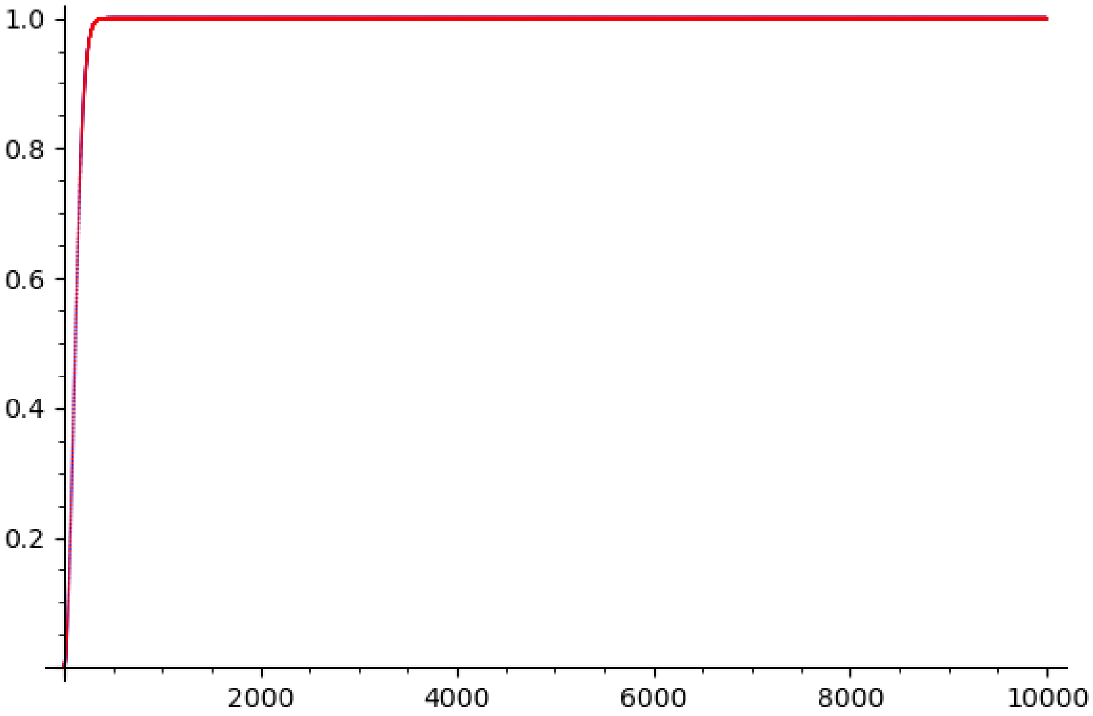
```
p1 = point(Pts[1000:2000], size=1)
p2 = point(Pts2[1000:2000], size=1, color='red')
p3 = point(Pts3[1000:2000], size=1, color='red')
show(p1+p2+p3)
```



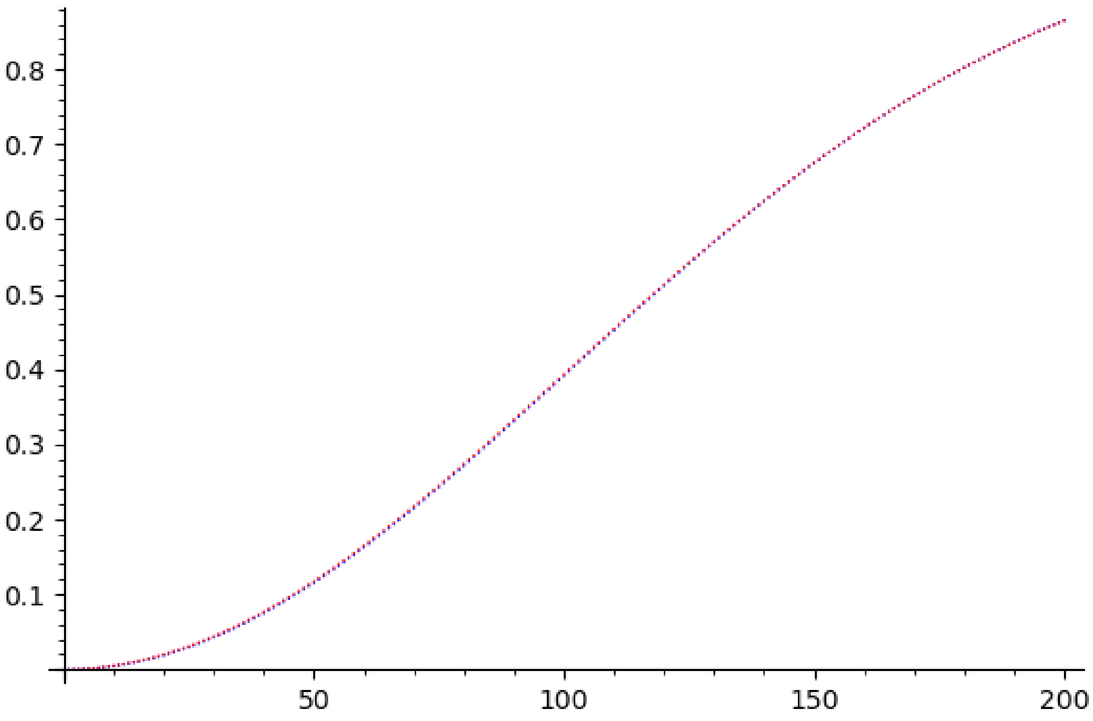
Comparaison Probabilité exacte et approximation qu'il y ait au moins une collision

(paradoxe des anniversaire)

[11]:
```
m = 10**4
Pts = []
App = []
p = 1
for n in range(1,m):
    proba = 1-p
    p = p * (1-n/m)
    Pts.append((n, proba))
    approx = 1-exp(-n**2/(2*m))
    App.append((n, approx))
p1 = point(Pts, size=1)
p2 = point(App, size=1, color='red')
show(p1+p2)
```



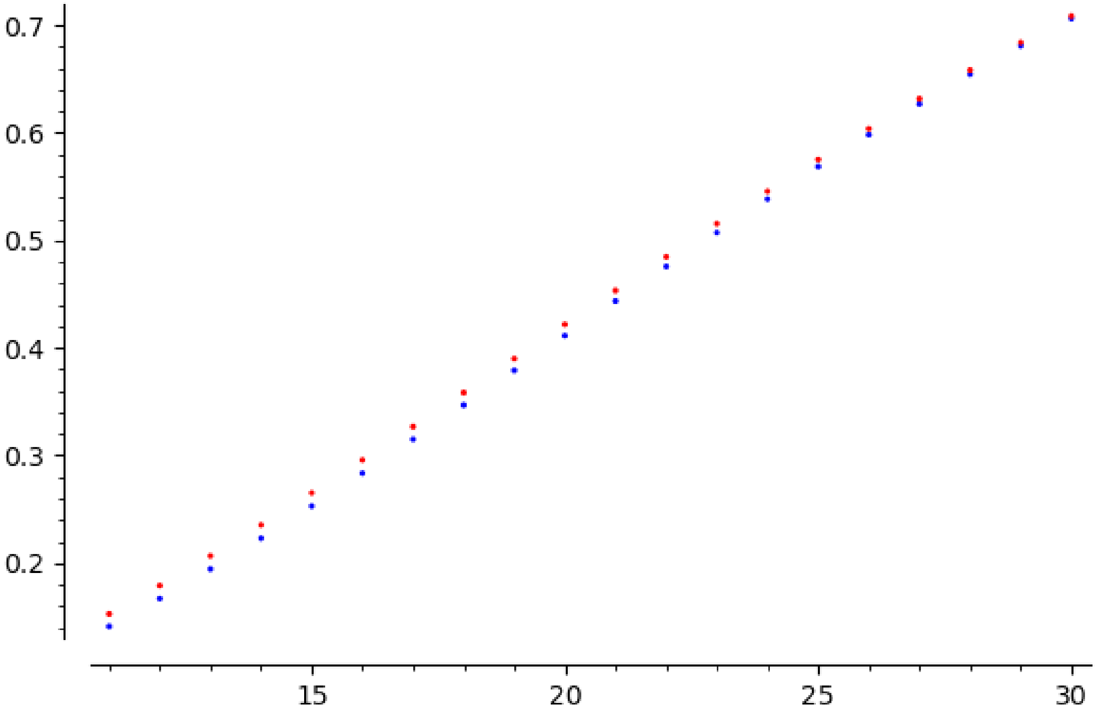
```
[12]: p1 = point(Pts[:200], size=1)
      p2 = point(App[:200], size=1, color='red')
      show(p1+p2)
```



[ ]:

[13]:

```
m = 365
Pts = []
App = []
p = 1
for n in range(1,m):
    proba = 1-p
    p = p * (1-n/m)
    Pts.append((n, proba))
    approx = 1-exp(-n**2/(2*m))
    App.append((n, approx))
p1 = point(Pts[10:30], size=5)
p2 = point(App[10:30], size=5, color='red')
show(p1+p2)
```

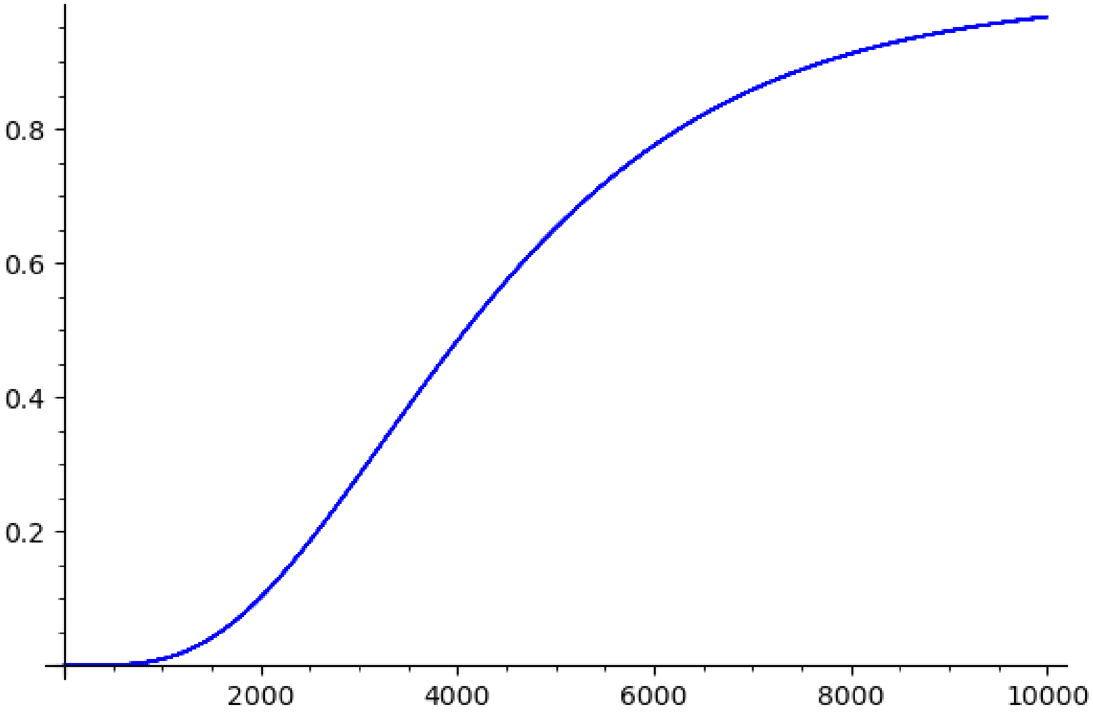


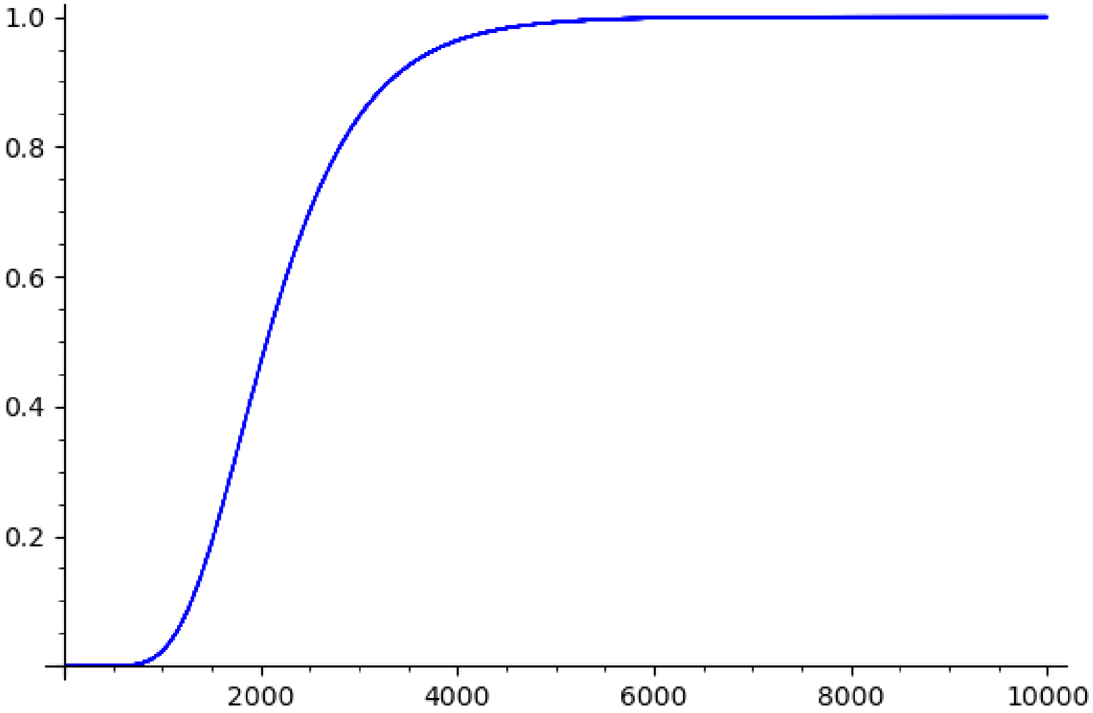
```
[14]: (sqrt(2*ln(2))).n()
```

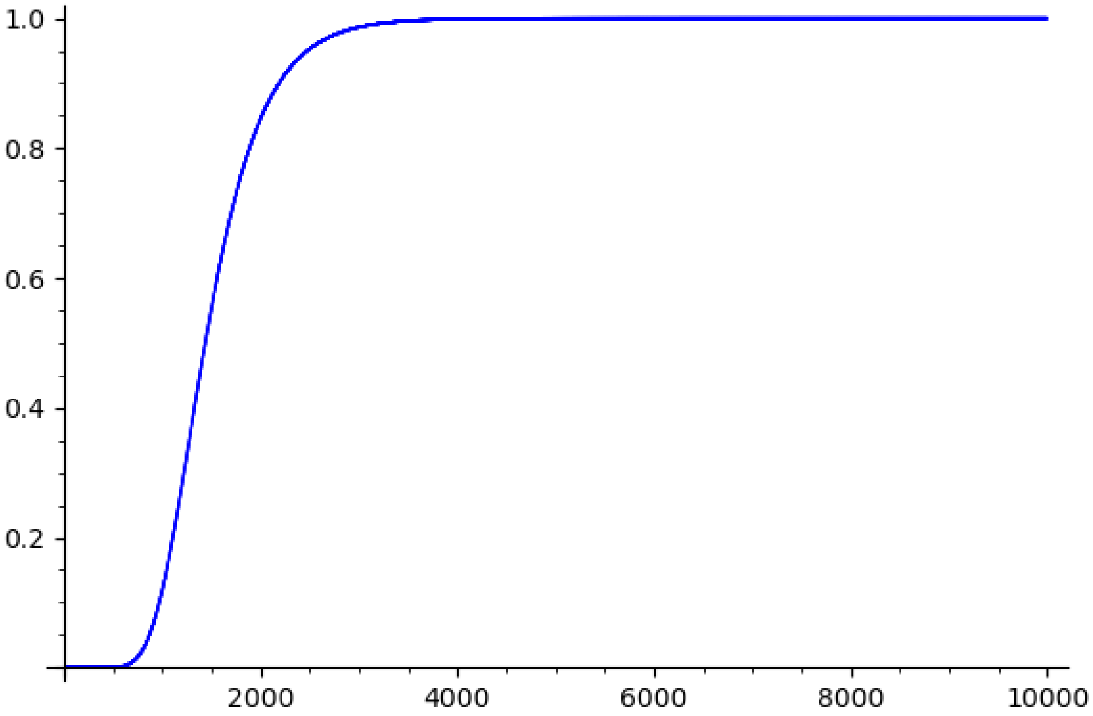
```
[14]: 1.17741002251547
```

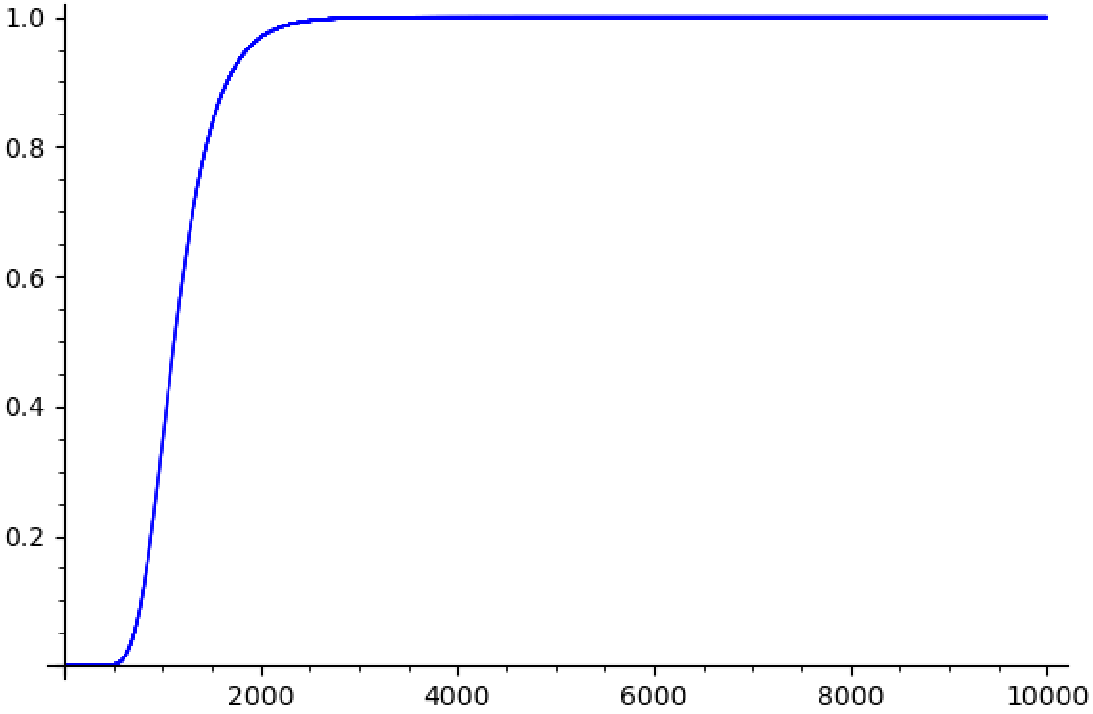
Probabilité d'erreur filtre de Bloom

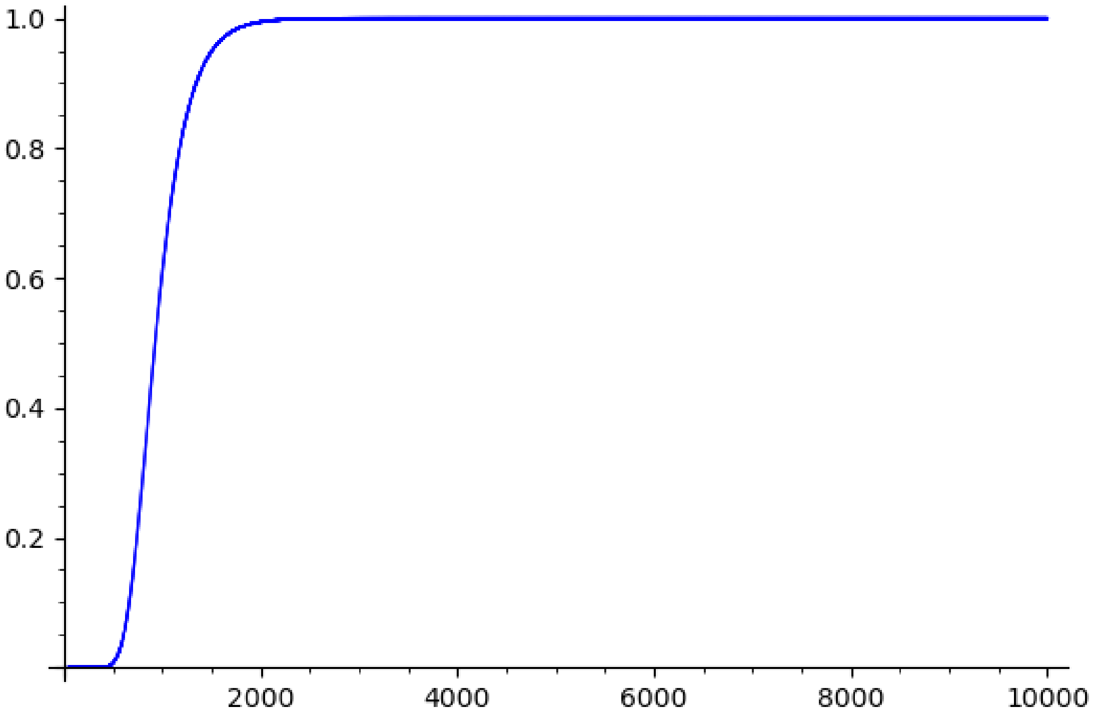
```
[22]: m = 10**4
      for k in range(5, 55, 10):
          Pts = []
          for n in range(k, m):
              proba = (1 - exp((-k*n)/m))**k
              Pts.append((n, proba))
          point(Pts, size=1).show()
```


0.8
0.6
0.4
0.2
2000
4000
6000
8000
10000


1.0
0.8
0.6
0.4
0.2
2000
4000
6000
8000
10000


1.0
0.8
0.6
0.4
0.2
2000
4000
6000
8000
10000


1.0
0.8
0.6
0.4
0.2
2000
4000
6000
8000
10000


1.0
0.8
0.6
0.4
0.2
2000
4000
6000
8000
10000

[ ]: