Mapeamento de Henon com $a = 1.4$, $b = 0.3$ e $X_0 = Y_0 = 0.0001$

27 de maio de 2018

```
In [1]: from scipy.stats import moment
        from scipy.stats import kurtosis, skew, scoreatpercentile
        from scipy.stats import norm, lognorm, beta
        from scipy.optimize import minimize

        from numpy import zeros, fromiter, savetxt
        from IPython.display import Image

        import subprocess

        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns

        import auxiliar_matcomp as aux

        #%matplotlib inline

        size = 2**12
        t = fromiter((i for i in range(0,size)), int, size)
```

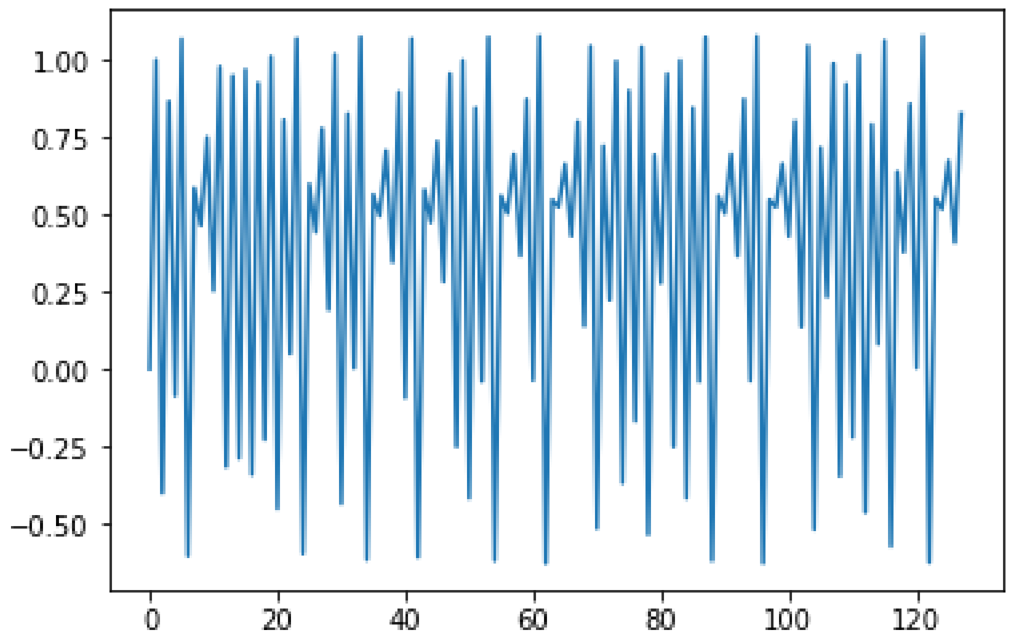
1 Série Completa

1.1 Gerando série temporal e plotando resultado

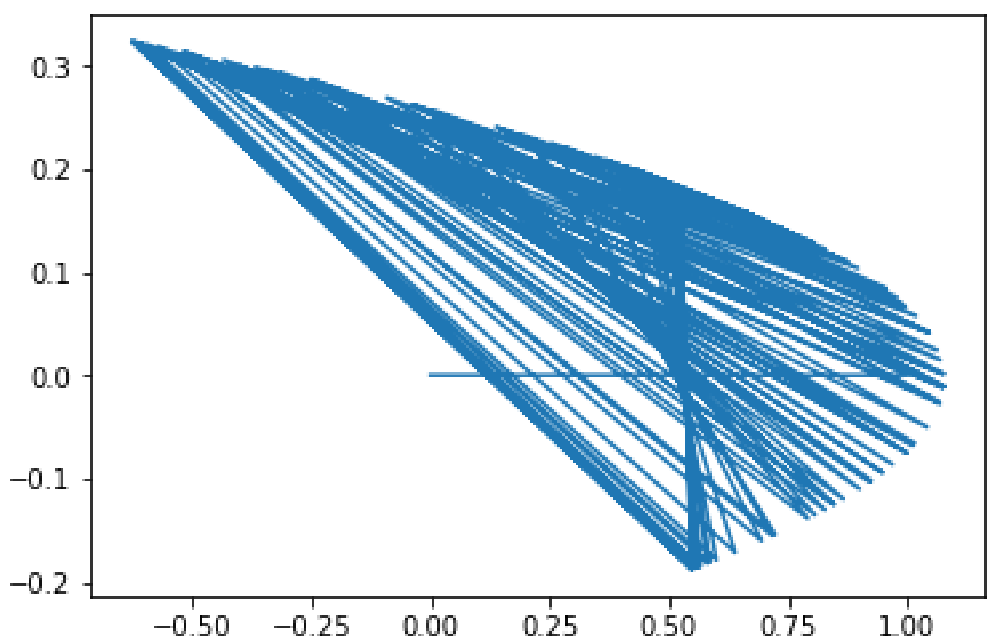
```
In [2]: name = "A.ex:1.2.a"

        a = 1.4
        b = 0.3

        # A desempenha o [apel de X]
        A = zeros(size)
        Y = zeros(size)

        A[0] = 0.0001
        Y[0] = 0.0001

        for i in range(0, size-1):
            A[i+1] = 1 - a*A[i]**2.0 + b*Y[i]
            Y[i+1] = b*A[i]

        savetxt(name + ".txt", A)
```

```
save_A = A
```

In [3]:
```
num_points = 128
plt.plot(t[0:num_points], A[0:num_points])
plt.show()
```



In [4]:
```
plt.plot(A[0:128], Y[0:128])
plt.show()
```



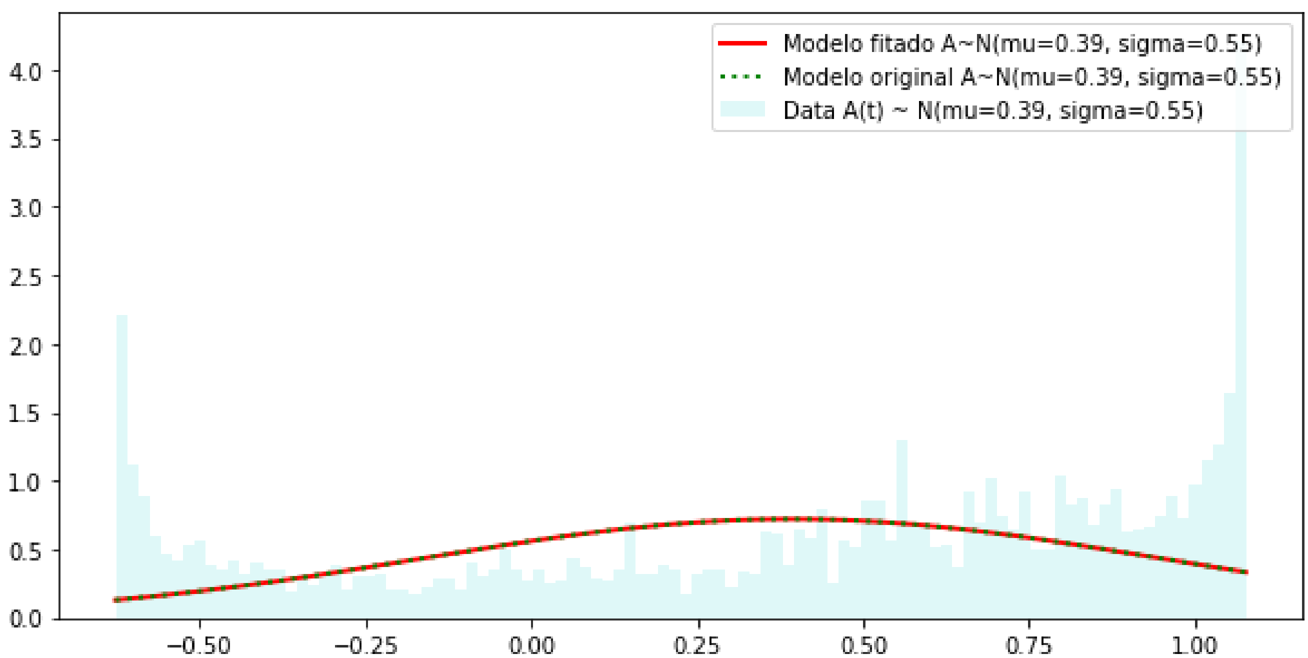
1.2 Calculando os momentos do ensemble

In [5]:
```
A_mean, A_var, A_skew, A_kurtosis = aux.calcMoments(A)

print("mean : ", A_mean)
print("var  : ", A_var)
print("skew : ", A_skew)
print("kurt : ", A_kurtosis)

A_Q1 = scoreatpercentile(A, 25)
A_Q3 = scoreatpercentile(A, 75)

print("Q1   : ", A_Q1)
print("Q3   : ", A_Q3)
```

```
mean :  0.392404431686
var  :  0.305329438832
skew :  -0.515114794419
kurt :  -1.05643374666
Q1   :  -0.0476997775696
Q3   :  0.869422966837
```

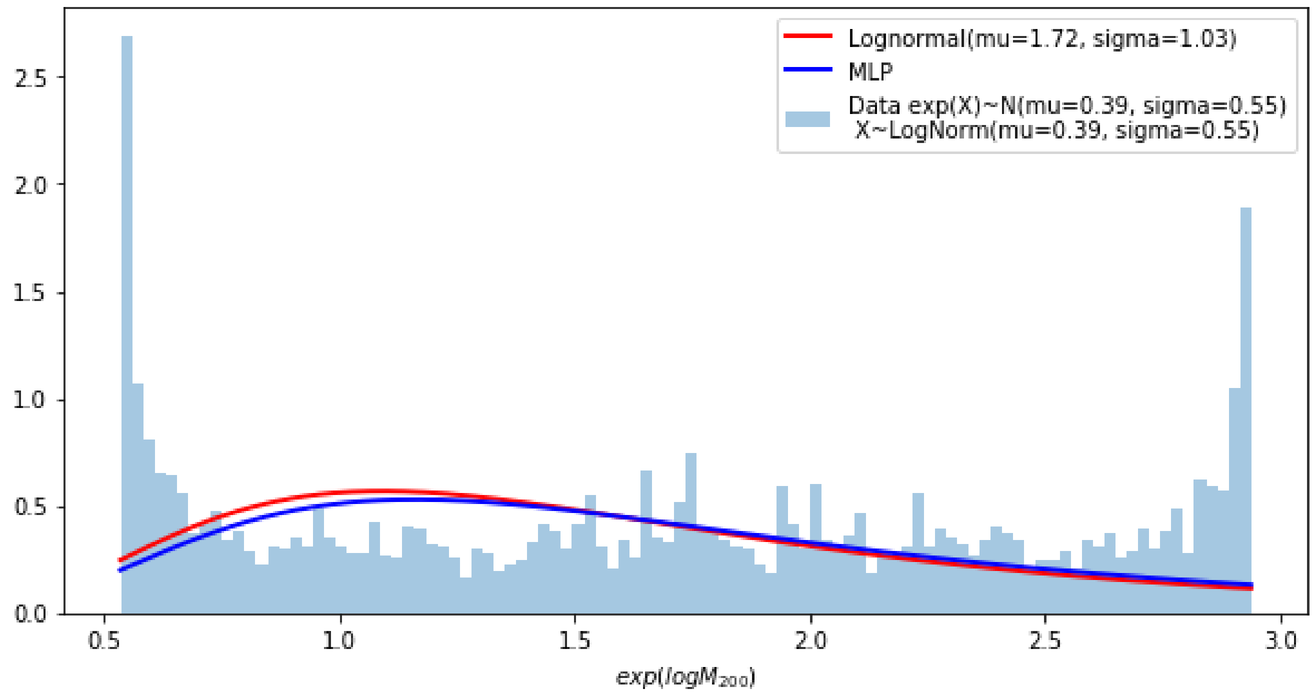
1.3 Fitando uma distribuição normal

In [6]:
```
aux.fitting_normal_distribution(A)
```



1.4 Fitando uma distribuição lognormal

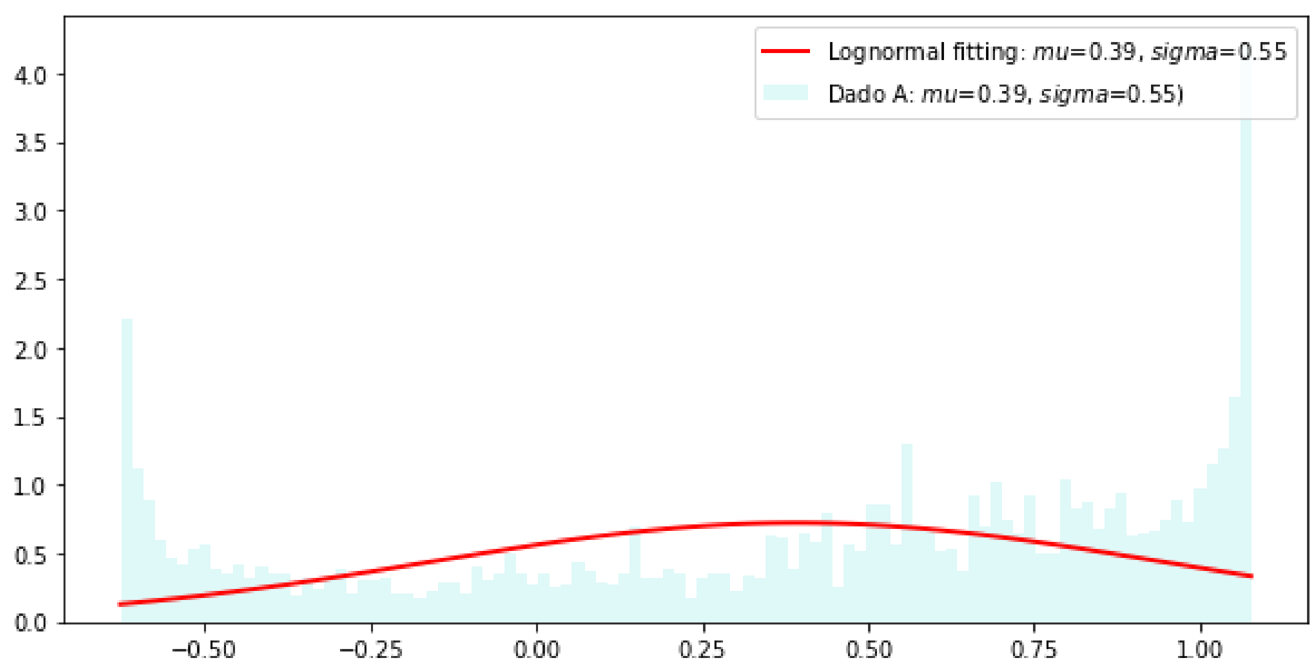
In [7]:
```
aux.fitting_lognormal_and_mlp_distribution(A)
```



1.5 Fitando uma distribuição lognormal (utlizando minha implementação)

```
In [8]: aux.fitting_lognormal_distribution(A)
```

```
parametros de fitting:  (0.002366416444840232, -232.68443150459029, 233.07761021116914)
        Fitado                         Original
mean :  0.39383131626931345          0.392404431686
var  :  0.3042195436107632          0.305329438832
skew :  0.0070992725251206885          -0.5151147944193247
kurt :  8.959954991016872e-05          -1.0564337466562803
```

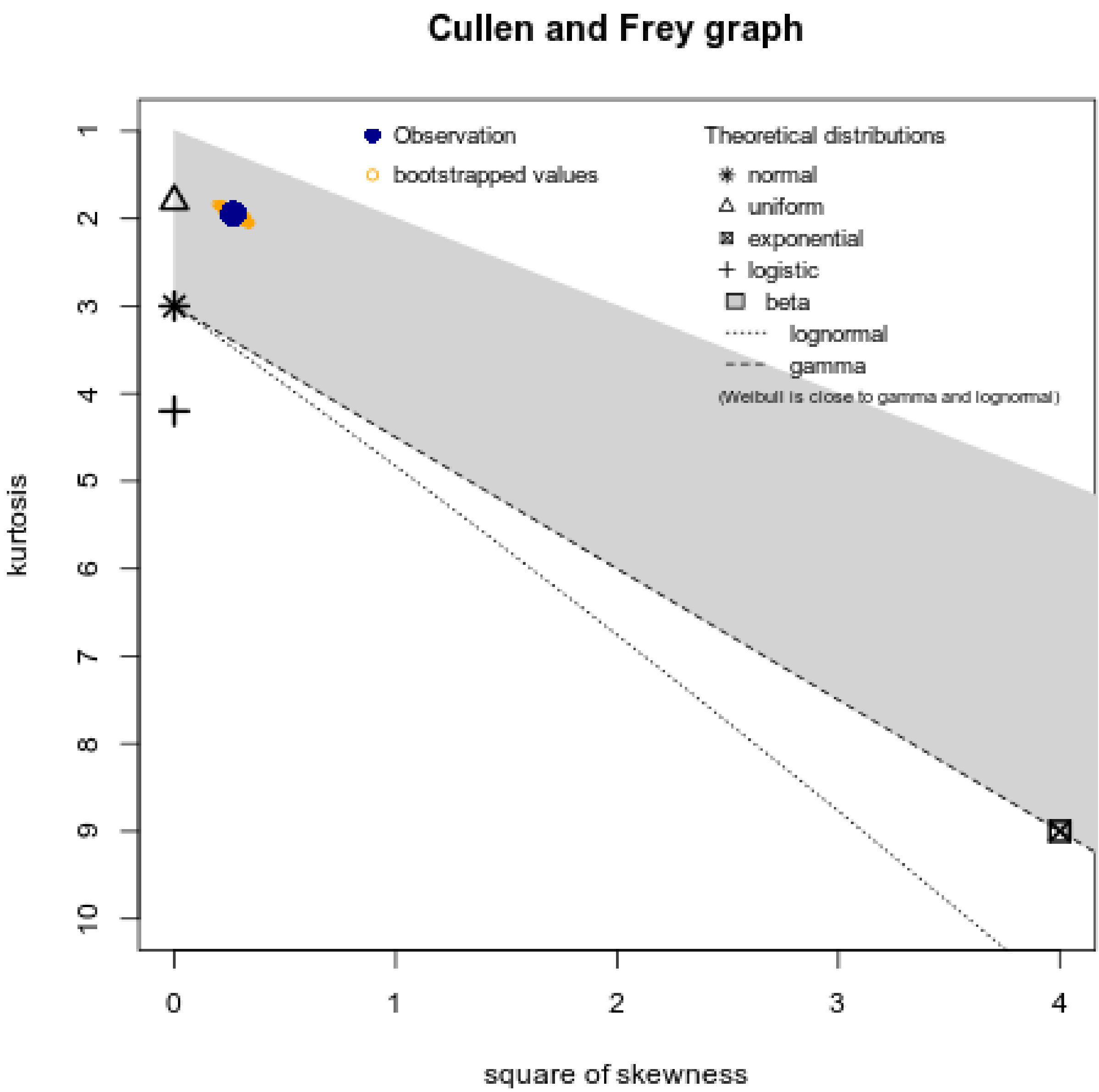


1.6 Plotando dados no espaço de Cullen-Frey

In [9]:
```
command = 'Rscript'
path_script = 'cullen_frey_script.R'

# define arguments
args = [name,]

# build subprocess command
cmd = [command, path_script] + args

x = subprocess.check_output(cmd, universal_newlines=True)
print(x)

Image(name+".png")
```

```
summary statistics
------
min:  -0.6273038   max:  1.077587
median:  0.5360758
mean:  0.3924044
estimated sd:  0.5526337
estimated skewness:  -0.5153035
estimated kurtosis:  1.943742
```

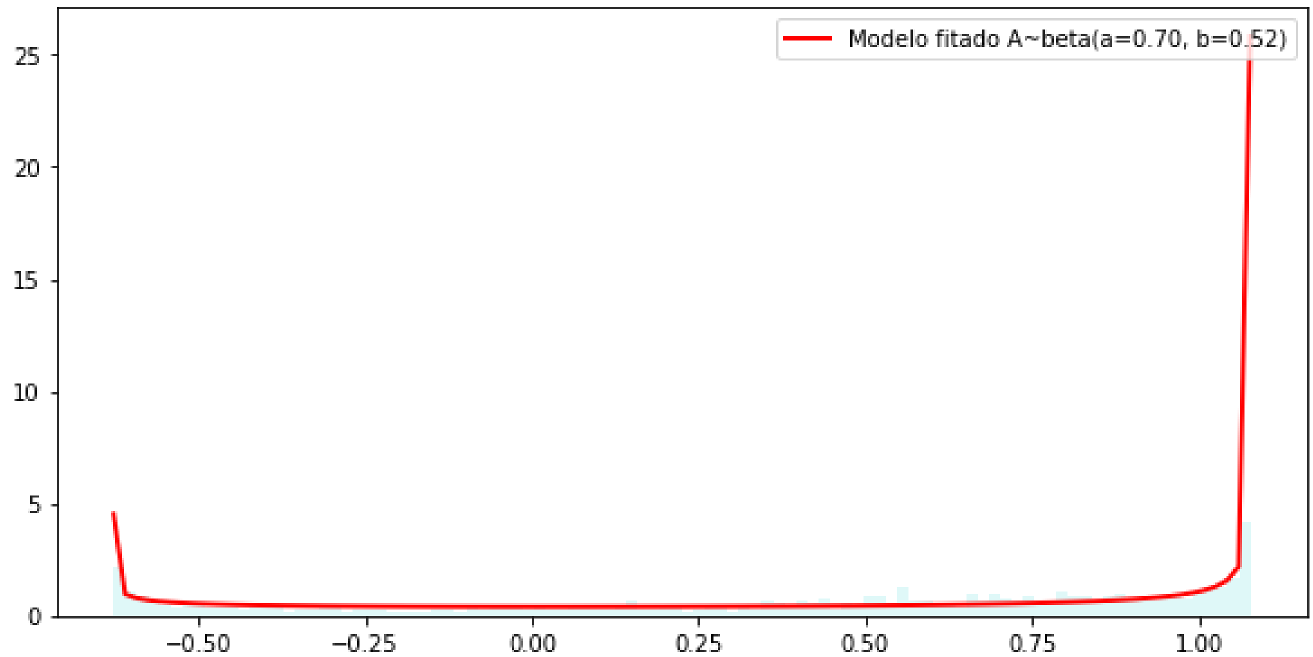
Out[9]:



1.7 Fitando melhor distribuição segundo método de Cullen-Frey

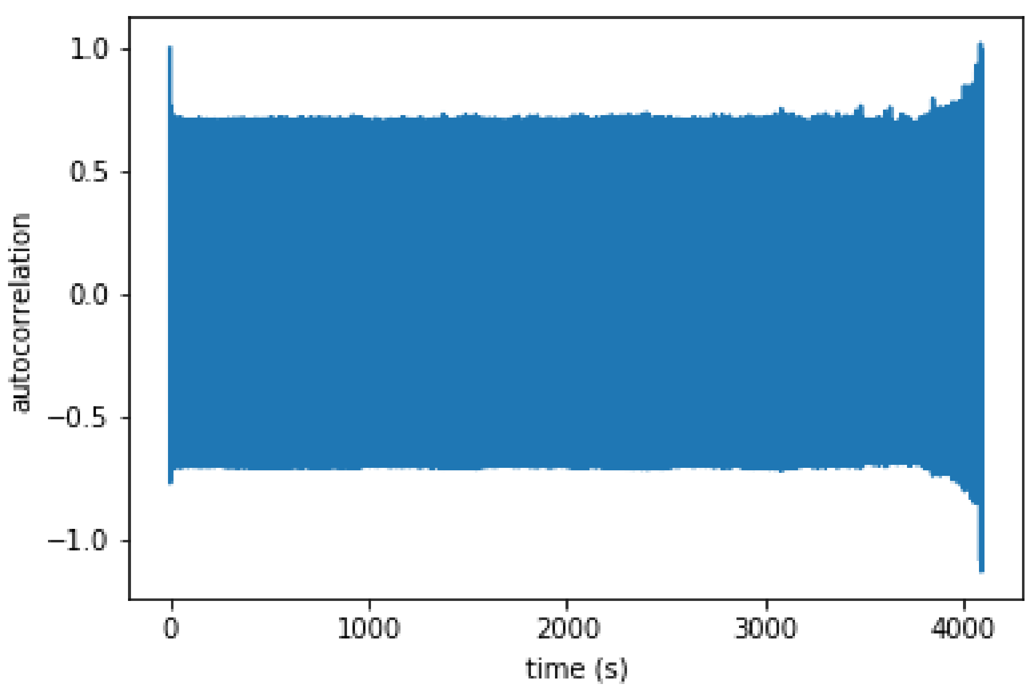
```
In [10]: aux.fitting_beta_distribution(A)
```

```
parametros de fitting:  (0.69925125508493824, 0.52059146221823038, -0.62740380167110077, 1.7050908987200
        Fitado                         Original
mean :  0.35000659235069675          0.392404431686
var  :  0.3204021993428364          0.305329438832
skew :  -0.2740426246032963          -0.5151147944193247
kurt :  -1.3358999816152202          -1.0564337466562803
```



1.8 Calculando autocorrelação

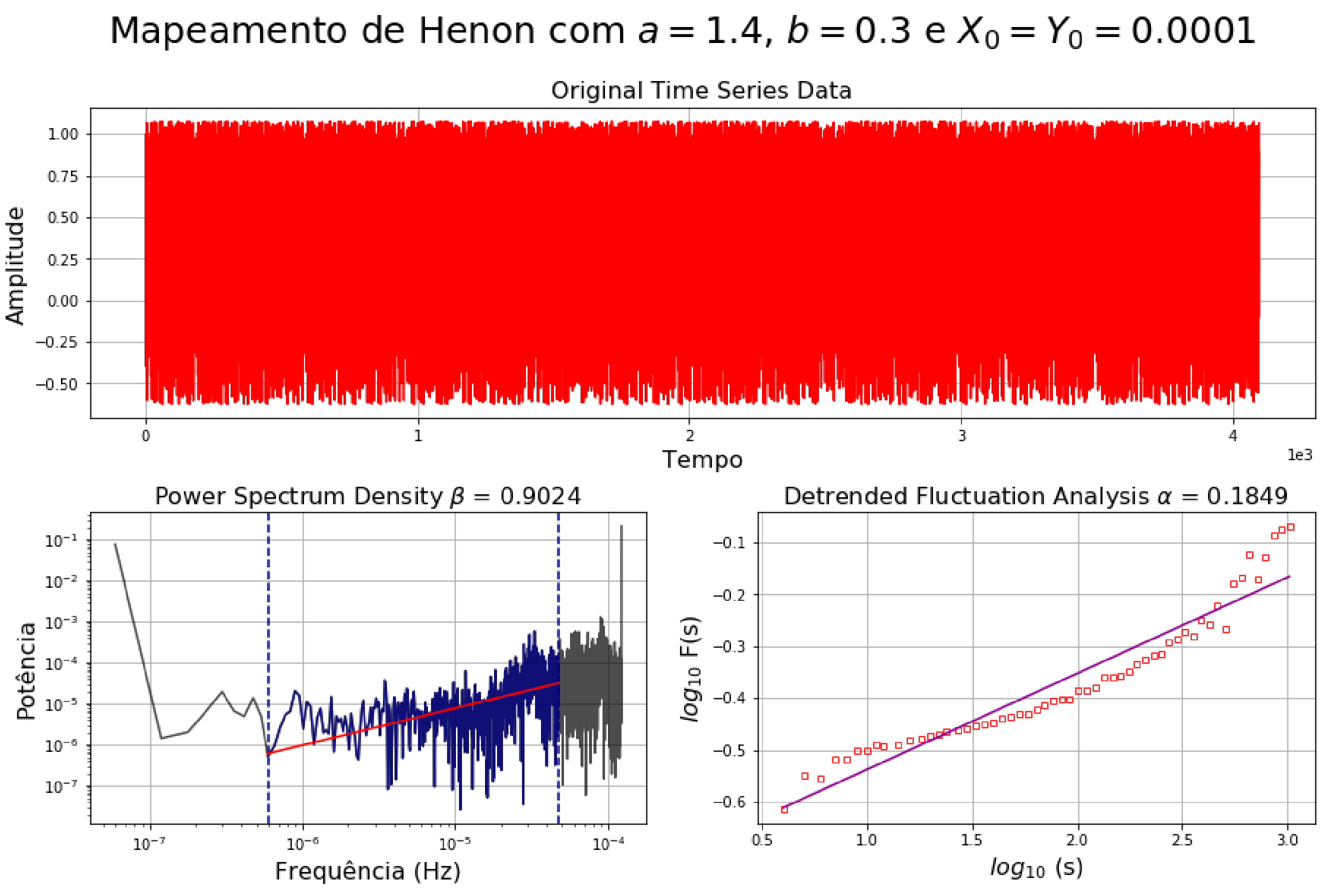
```
In [11]: aux.plot_estimated_autocorrelation(t, A, 0, len(A))
```



1.9 Plotando DFA e PSD

```
In [12]: aux.plot_psd_dfa(A, 'Mapeamento de Henon com $a=1.4$, $b=0.3$ e $X_0=Y_0=0.0001$')
```

```
Original time series data (4096 points):

First 10 points: [  1.00000000e-04   1.00002999e+00  -4.00074962e-01   8.65918733e-01
  -8.57481005e-02   1.06763885e+00  -6.03511142e-01   5.86171518e-01
   4.64648129e-01   7.50498400e-01]


1. Plotting time series data...
2. Plotting Power Spectrum Density...
3. Plotting Detrended Fluctuation Analysis...
```



2 Analise dos primeiros 1024 pontos

```
In [13]: A = save_A[1024:]
         name = "A.ex:1.2.a"
         savetxt(name + ".txt", A)
```

2.1 Calculando os momentos do ensemble

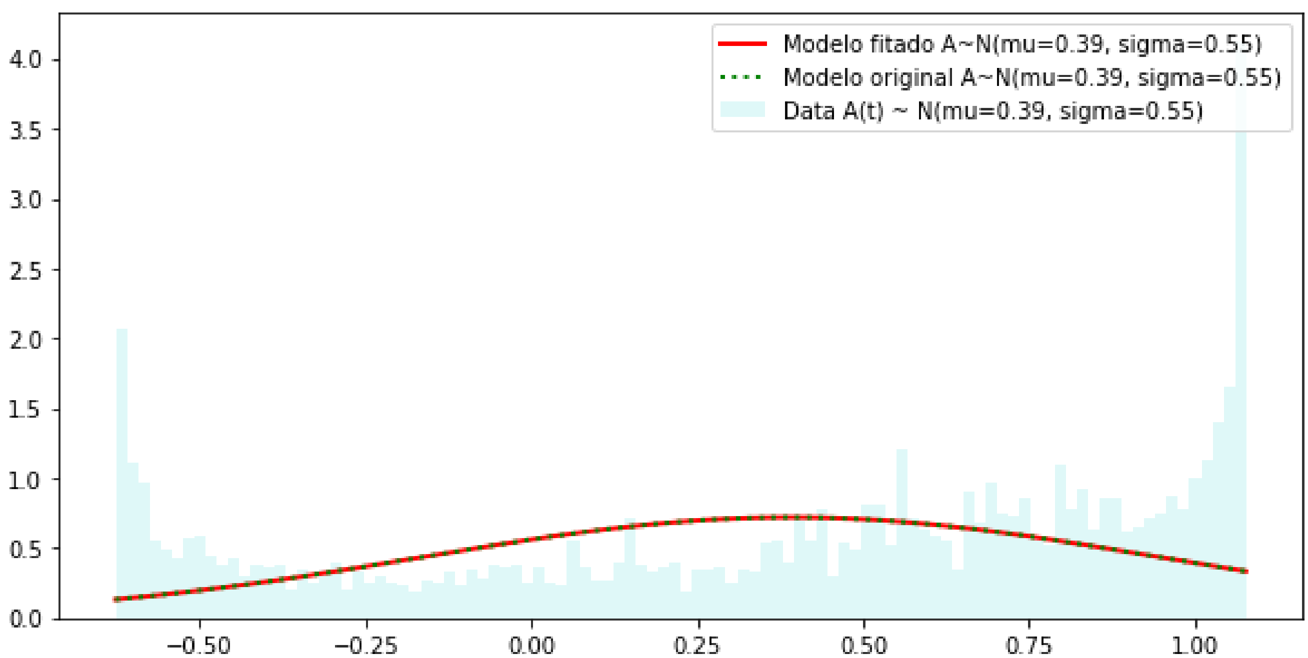
```
In [14]: A_mean, A_var, A_skew, A_kurtosis = aux.calcMoments(A)

         print("mean : ", A_mean)
         print("var  : ", A_var)
         print("skew : ", A_skew)
```

```
        print("kurt : ", A_kurtosis)

        A_Q1 = scoreatpercentile(A, 25)
        A_Q3 = scoreatpercentile(A, 75)

        print("Q1   : ", A_Q1)
        print("Q3   : ", A_Q3)
```

```
mean :  0.39145590788
var  :  0.306851648031
skew :  -0.506537615529
kurt :  -1.07626322527
Q1   :  -0.0533241662663
Q3   :  0.870887826789
```

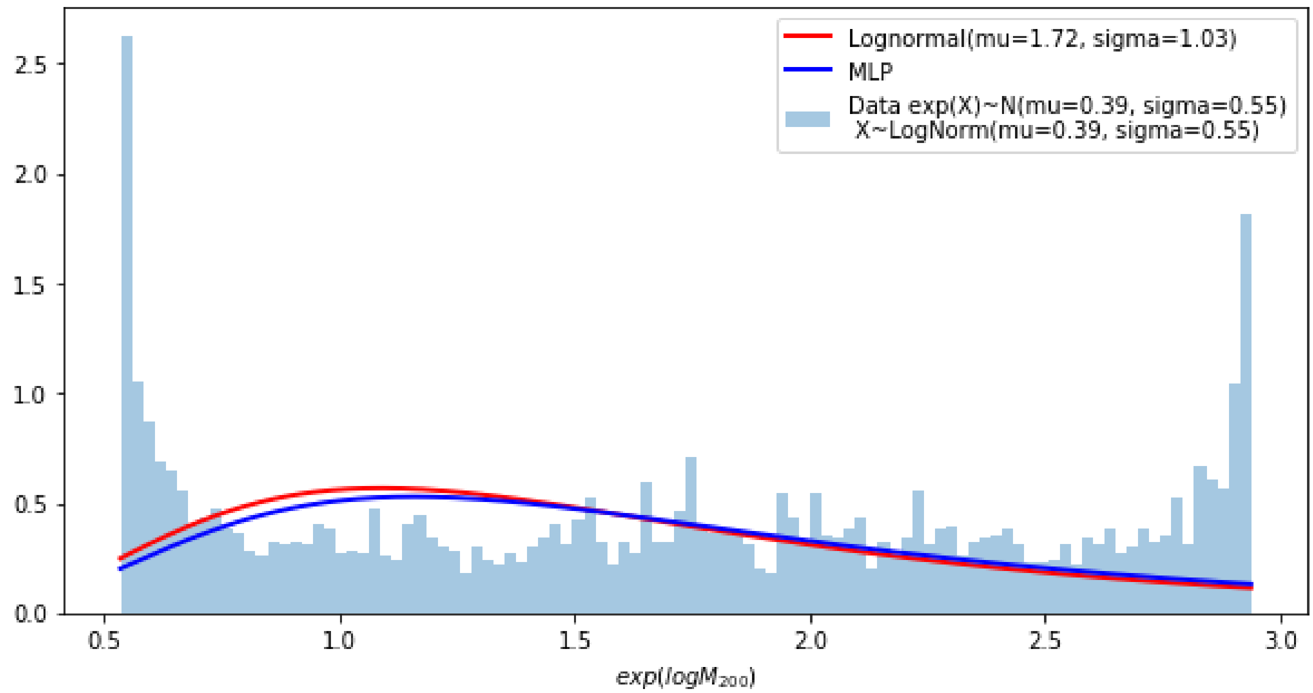
2.2 Fitando uma distribuição normal

In [15]: `aux.fitting_normal_distribution(A)`



2.3 Fitando uma distribuição lognormal

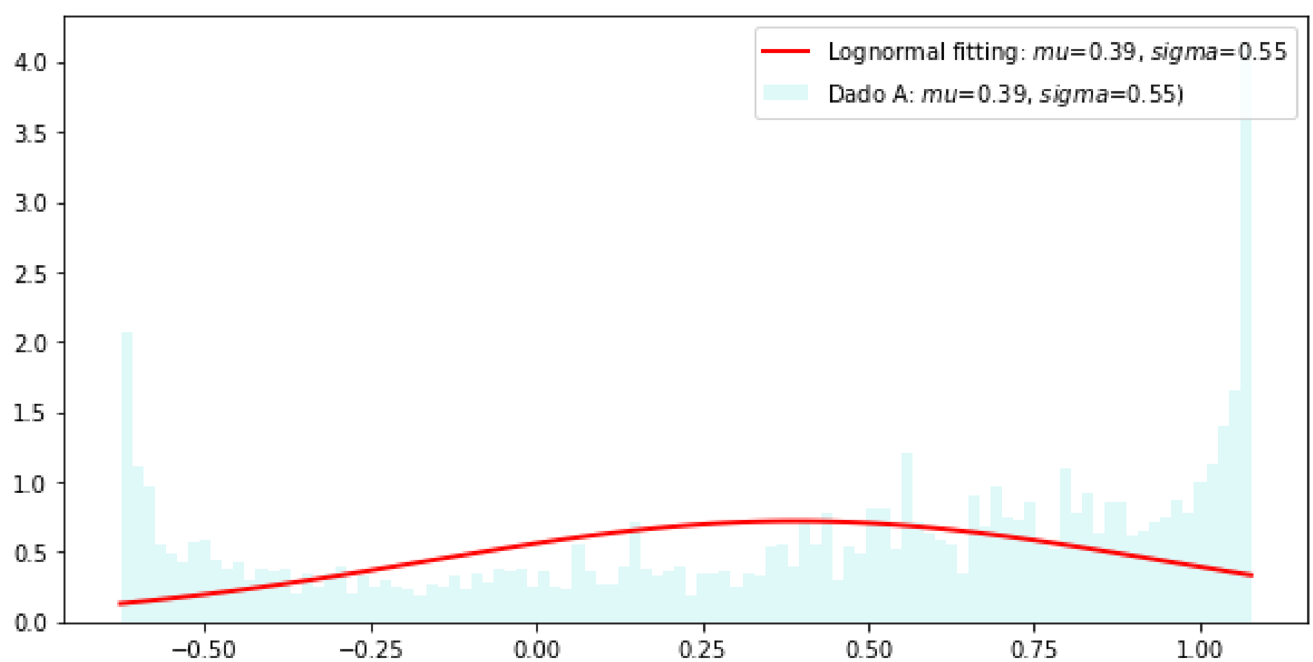
In [16]: `aux.fitting_lognormal_and_mlp_distribution(A)`



2.4 Fitando uma distribuição lognormal (utlizando minha implementação)

```
In [17]: aux.fitting_lognormal_distribution(A)
```

```
parametros de fitting:  (0.0027510480251955495, -200.88975934797566, 201.28205559418655)
        Fitado                         Original
mean :  0.3930579256441149          0.39145590788
var  :  0.3066277048342848          0.306851648031
skew :  0.008253180511857033          -0.5065376155291291
kurt :  0.00012109356120948433          -1.0762632252693016
```

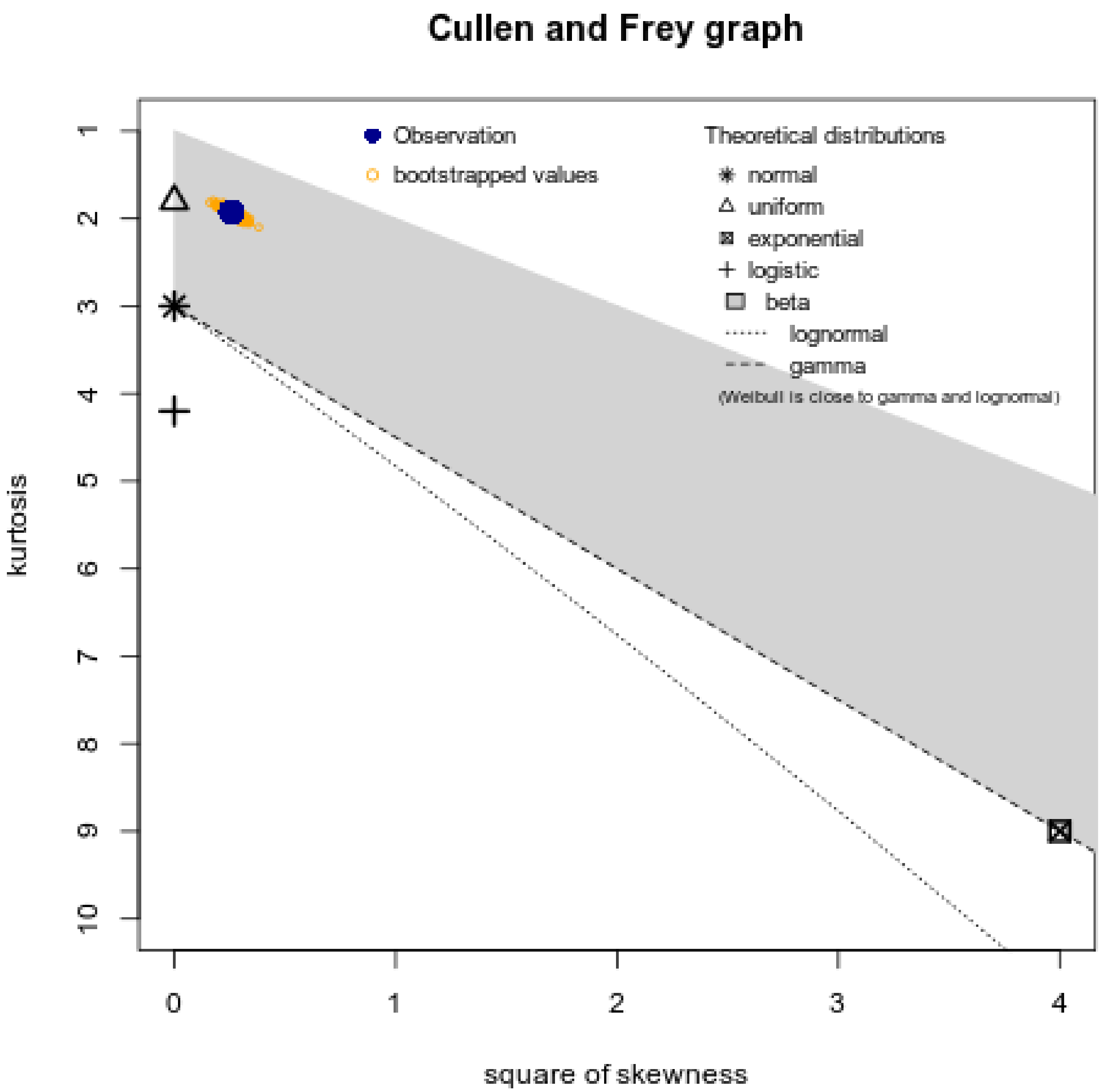


2.5 Plotando dados no espaço de Cullen-Frey

In [18]:
```
command = 'Rscript'
path_script = 'cullen_frey_script.R'

# define arguments
args = [name,]

# build subprocess command
cmd = [command, path_script] + args

x = subprocess.check_output(cmd, universal_newlines=True)
print(x)


Image(name+".png")
```

```
summary statistics
------
min:  -0.6273038   max:  1.077587
median:  0.5360758
mean:  0.3914559
estimated sd:  0.5540321
estimated skewness:  -0.5067851
estimated kurtosis:  1.923939
```

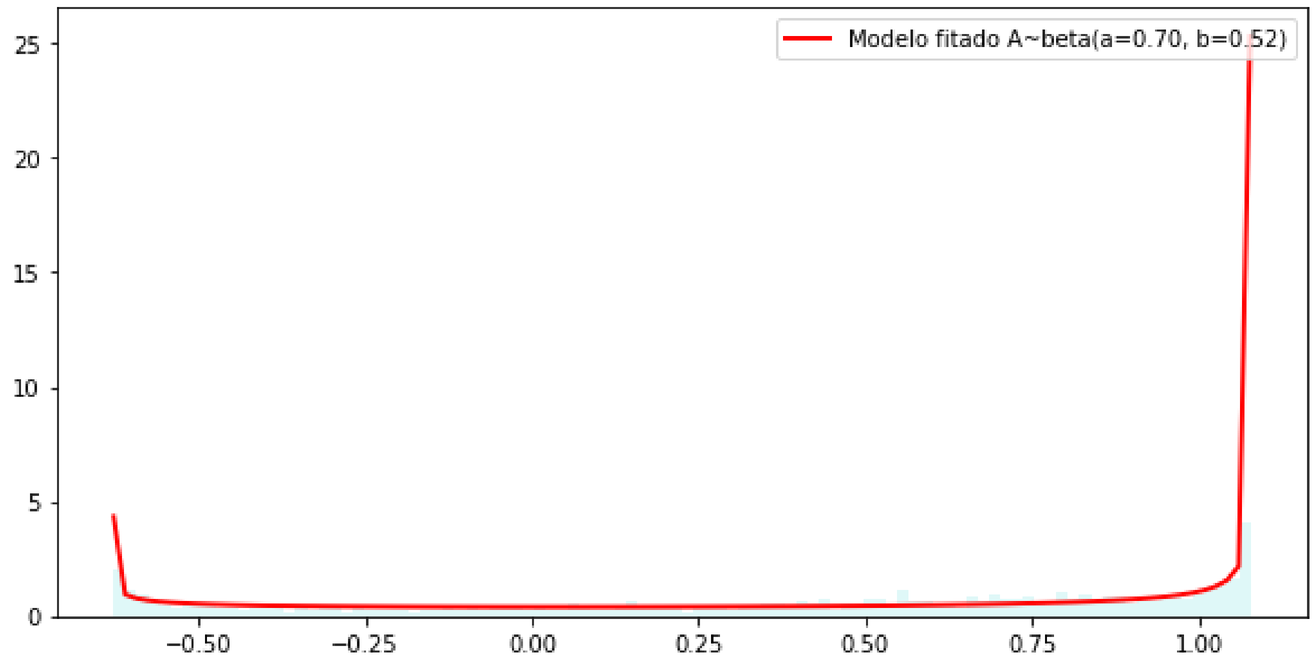
Out[18]:



2.6 Fitando melhor distribuição segundo método de Cullen-Frey

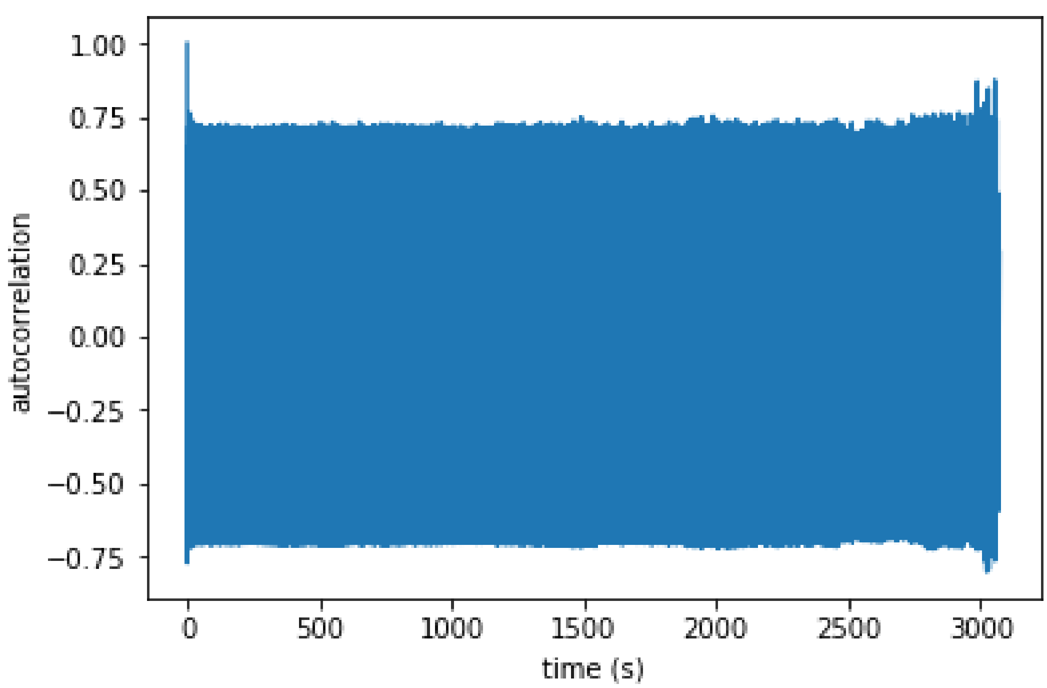
```
In [19]: aux.fitting_beta_distribution(A)
```

```
parametros de fitting:  (0.7045879231544325 3, 0.5237021715400384, -0.62740380167110077, 1.70509089872006
        Fitado                         Original
mean :  0.350692871344513           0.39145590788
var  :  0.31911044040277137           0.306851648031
skew :  -0.2753834781768166           -0.5065376155291291
kurt :  -1.3321623331460515           -1.0762632252693016
```



2.7 Calculando autocorrelação

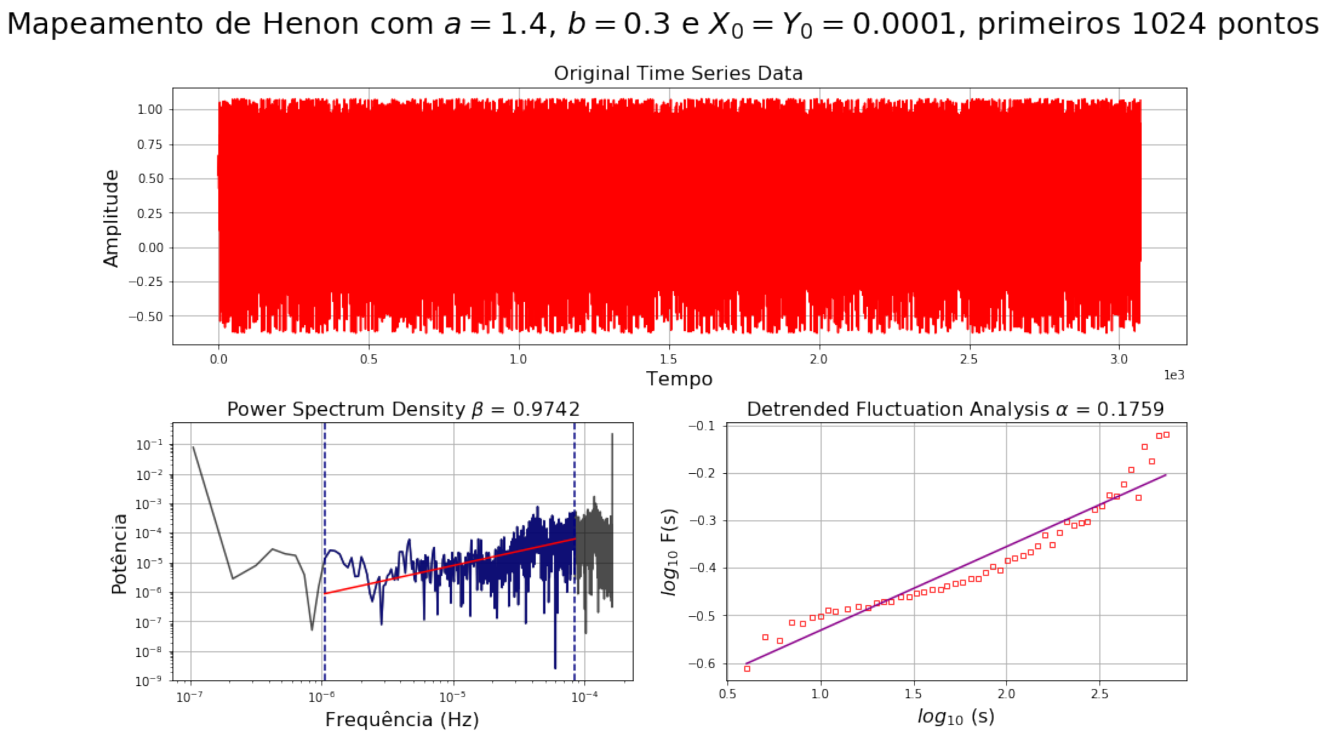
```
In [20]: aux.plot_estimated_autocorrelation(t, A, 0, len(A))
```



2.8 Plotando DFA e PSD

```
In [21]: aux.plot_psd_dfa(A, 'Mapeamento de Henon com $a=1.4$, $b=0.3$ e $X_0=Y_0=0.0001$, primeiros 102
```

```
Original time series data (3072 points):

First 10 points: [ 0.52222929  0.6675674   0.42309591  0.80946685  0.12074742  1.0524401
 -0.53981497  0.68675933  0.29112293  0.94315475]


1. Plotting time series data...
2. Plotting Power Spectrum Density...
3. Plotting Detrended Fluctuation Analysis...
```



3 Analise dos últimos 1024 pontos

```
In [22]: A = save_A[3*1024:4096]
         name = "A.ex:1.2.a"
         savetxt(name + ".txt", A)

In [23]: A.shape

Out[23]: (1024,)
```

3.1 Calculando os momentos do ensemble

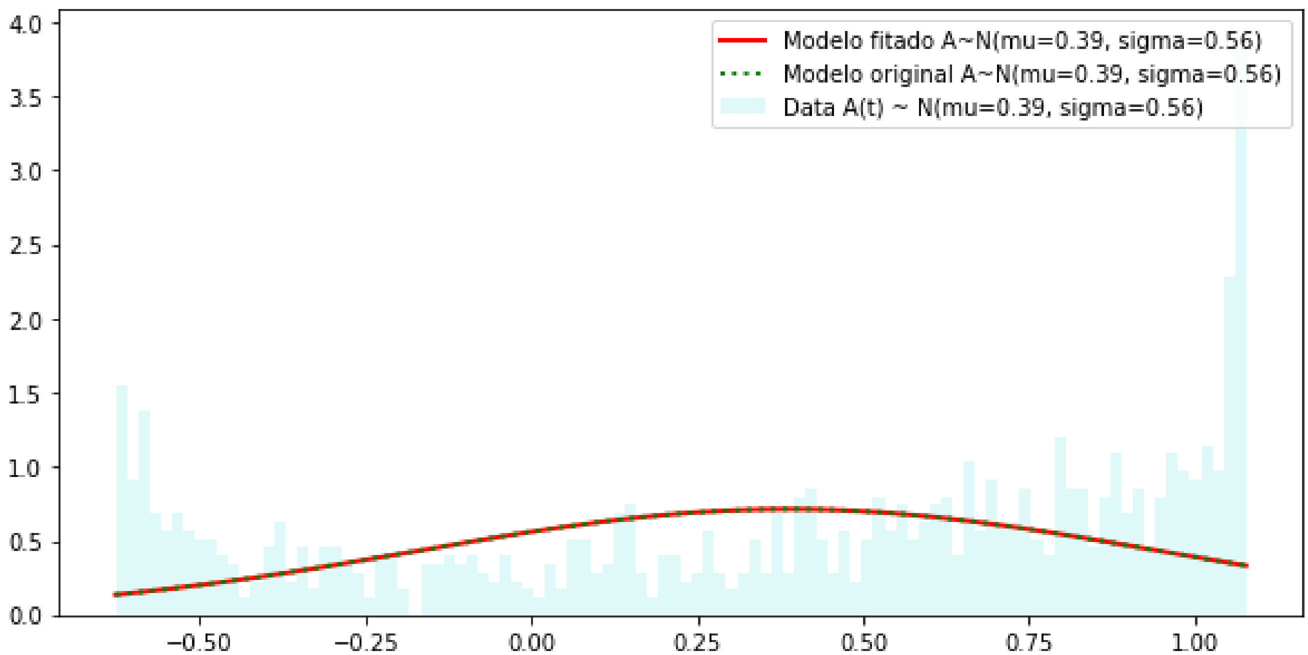
```
In [24]: A_mean, A_var, A_skew, A_kurtosis = aux.calcMoments(A)

         print("mean : ", A_mean)
         print("var  : ", A_var)
         print("skew : ", A_skew)
         print("kurt : ", A_kurtosis)
```

```
A_Q1 = scoreatpercentile(A, 25)
A_Q3 = scoreatpercentile(A, 75)

print("Q1   : ", A_Q1)
print("Q3   : ", A_Q3)
```

```
mean :  0.387808777575
var  :  0.311967663301
skew :  -0.478781069683
kurt :  -1.13642060736
Q1   :  -0.0887932860104
Q3   :  0.880448831948
```

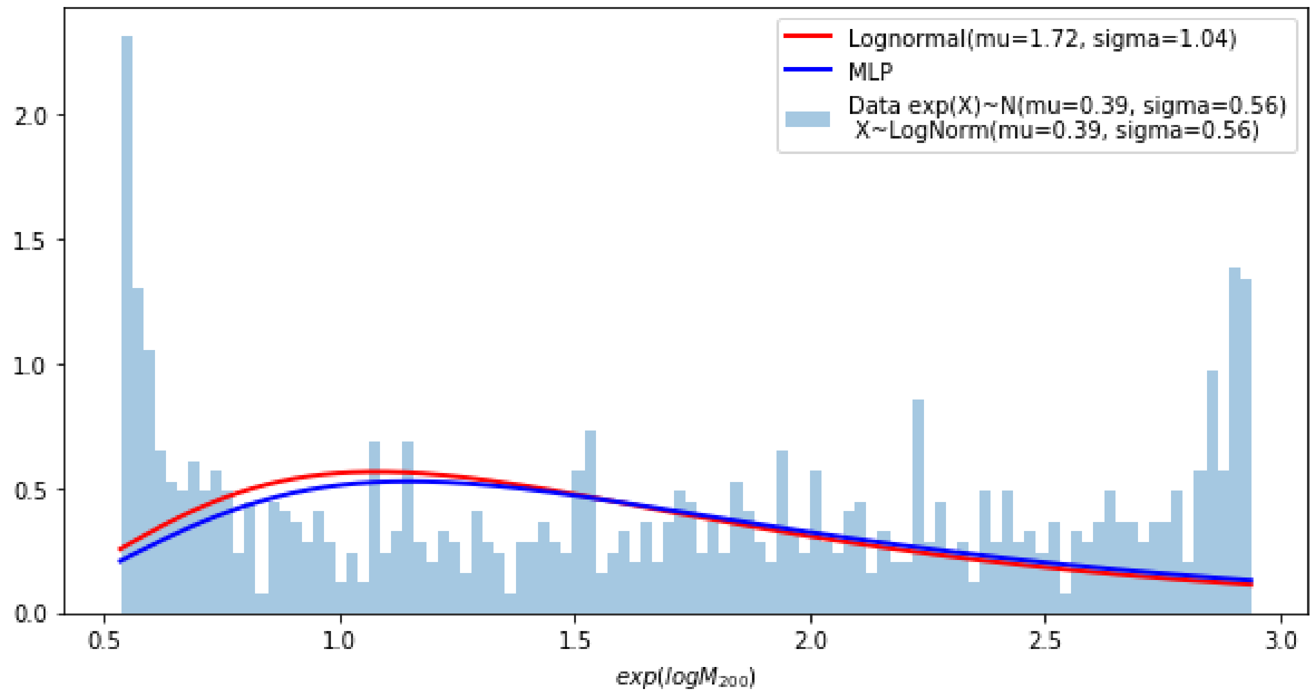
3.2 Fitando uma distribuição normal

In [25]: `aux.fitting_normal_distribution(A)`



3.3 Fitando uma distribuição lognormal

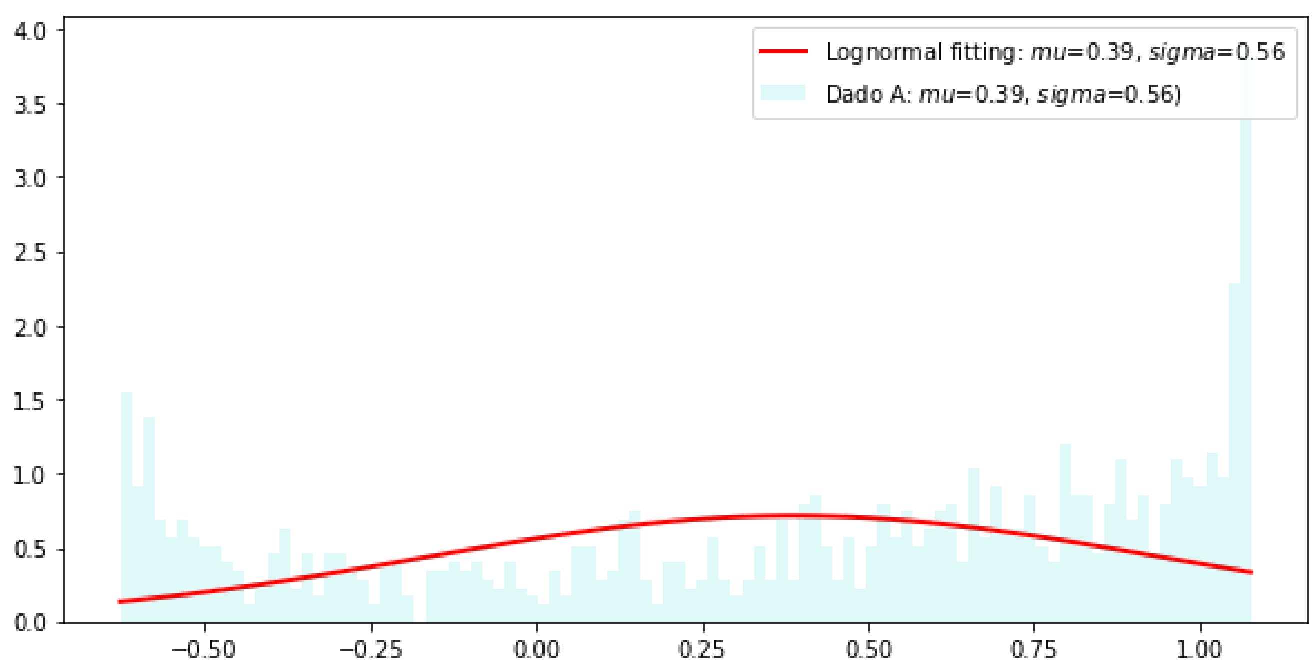
In [26]: `aux.fitting_lognormal_and_mlp_distribution(A)`



3.4 Fitando uma distribuição lognormal (utlizando minha implementação)

```
In [27]: aux.fitting_lognormal_distribution(A)
```

```
parametros de fitting:  (0.0035682809115648237, -155.91733277231816, 156.30727776795442)
        Fitado                        Original
mean :  0.39094010006647295          0.387808777575
var  :  0.31108908035130595          0.311967663301
skew :  0.010704922244023944          -0.478781069683446
kurt :  0.00020372578742655634          -1.1364206073647602
```

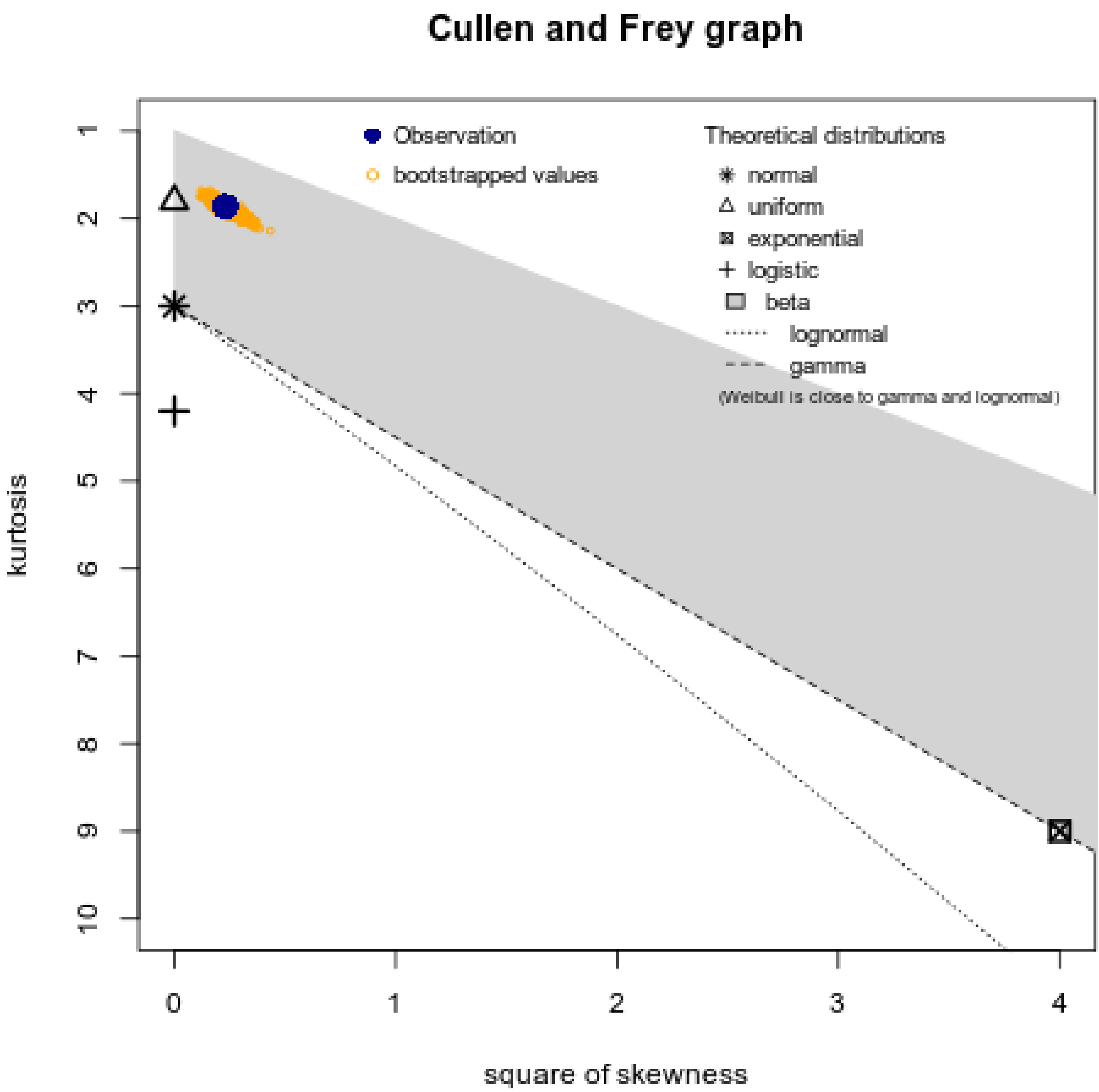


3.5 Plotando dados no espaço de Cullen-Frey

In [28]:
```
command = 'Rscript'
path_script = 'cullen_frey_script.R'

# define arguments
args = [name,]

# build subprocess command
cmd = [command, path_script] + args

x = subprocess.check_output(cmd, universal_newlines=True)
print(x)


Image(name+".png")
```

```
summary statistics
------
min:  -0.6272904   max:  1.077586
median:  0.5360689
mean:  0.3878088
estimated sd:  0.5588136
estimated skewness:  -0.4794837
estimated kurtosis:  1.863893
```

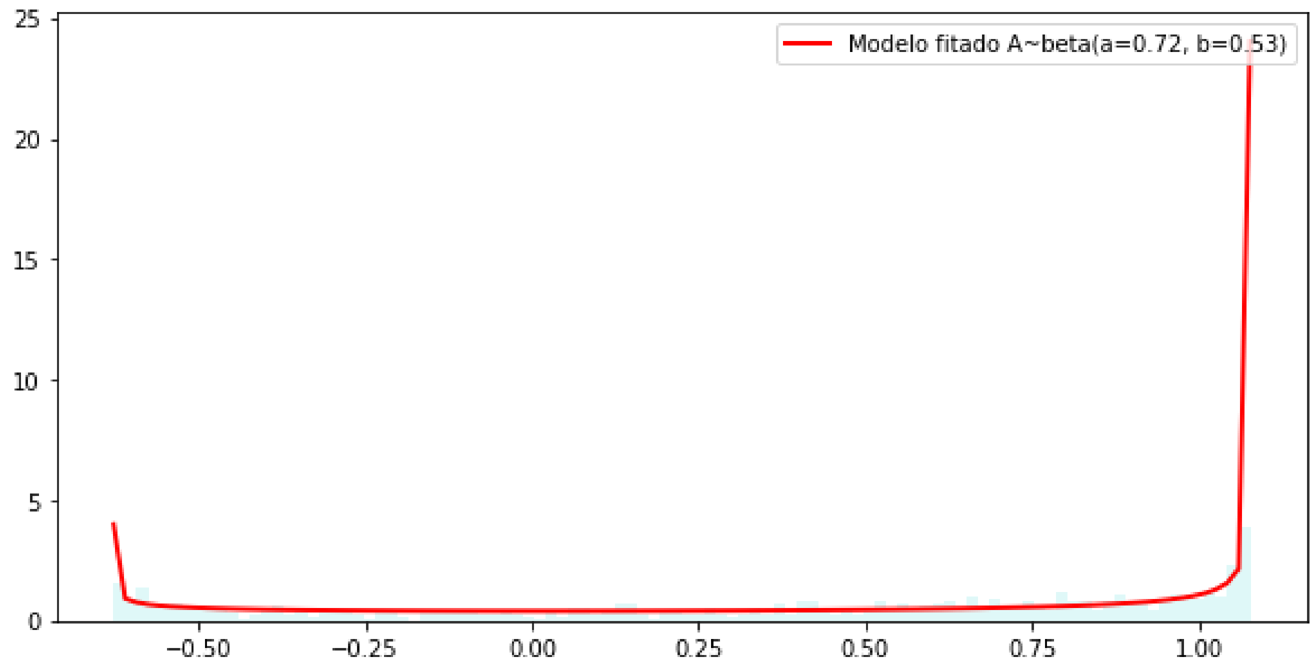
Out[28]:



3.6 Fitando melhor distribuição segundo método de Cullen-Frey

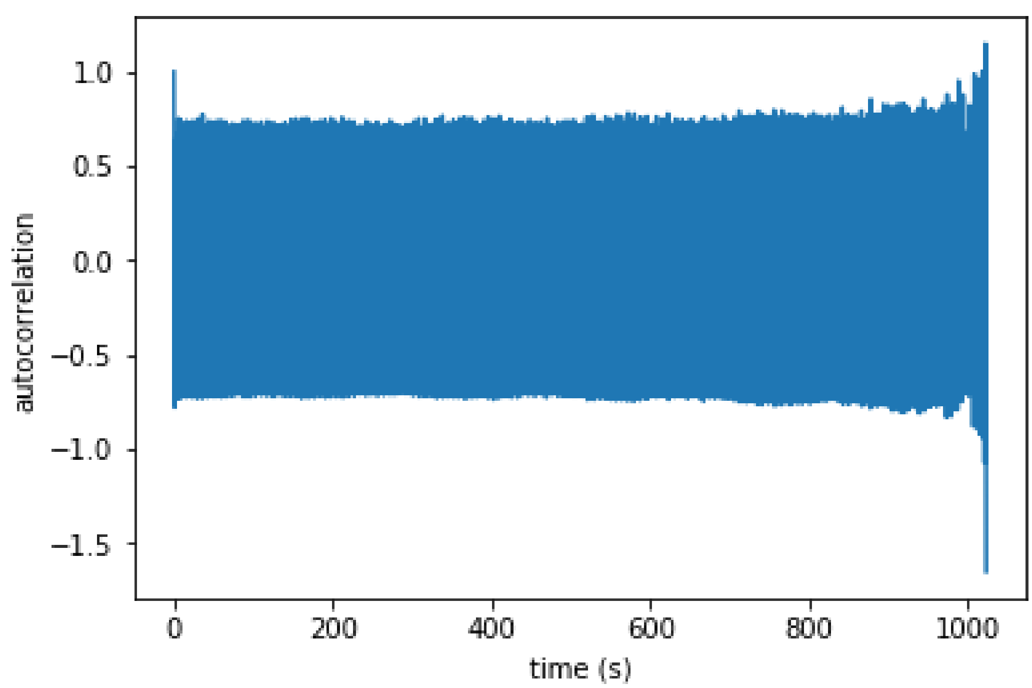
In [29]: `aux.fitting_beta_distribution(A)`

```
parametros de fitting:  (0.7155072288486714б, 0.53131054068642913, -0.62739044659952792, 1.7050762622687
        Fitado                         Original
mean :  0.35109608222696476          0.387808777575
var  :  0.3164289849390078          0.311967663301
skew :  -0.27583958765638605          -0.478781069683446
kurt :  -1.3255658805028345          -1.1364206073647602
```



3.7 Calculando autocorrelação

```
In [30]: aux.plot_estimated_autocorrelation(t, A, 0, len(A))
```



3.8 Plotando DFA e PSD

```
In [31]: aux.plot_psd_dfa(A, 'Mapeamento de Henon com $a=1.4$, $b=0.3$ e $X_0=Y_0=0.0001$, últimos 1024
```

```
Original time series data (1024 points):

First 10 points: [-0.3721241   0.89443379 -0.15350769  1.04750858 -0.54999962  0.67077635
  0.32058275  0.91648725 -0.14707599  1.05219997]


1. Plotting time series data...
2. Plotting Power Spectrum Density...
3. Plotting Detrended Fluctuation Analysis...
```