In [1]:

```
%matplotlib inline
import numpy as np
from scipy import stats
from scipy.stats import norm
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('white')
sns.set_context('talk')
```

Sampling: Inverse transform method ¶

Python provides random sampling for some commonly used distributions, like uniform distribution, normal distribution, etc. Now, we will draw sampling from a distribution with its CDF given by F. The next propostion is the theoretical basis for the inverse transform method.

Propostion

If F is a strictly increasing CDF and $U \sim U(0, 1)$, then the r.v. given by $X = F^{-1}(U)$ has its CDF F.

[Proof] ...

In [2]:

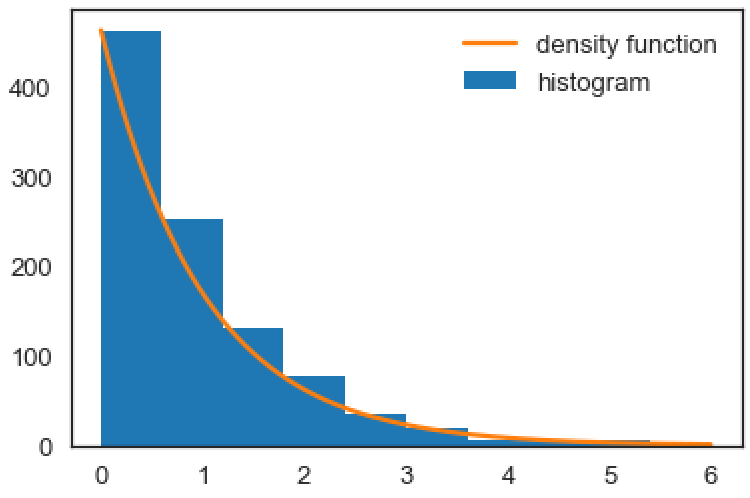
```
#Inverse Transform to generate random numbers,
#Input
#F_inv: inverse of the given distribution
#size: number of random numbers
#Output
#A numpy array
def InverseTransform(F_inv, size):
    R = np.random.uniform(0, 1, size)
    return F_inv(R)
```

Ex

Draw from the distribution $f(x) \sim \exp{(-x)}$.

In [3]:

```
# probability distribution we're trying to calculate
phi = lambda x: np.exp(-x)

# CDF of p
Phi = lambda x: 1-np.exp(-x)

# invert the CDF
Phi_inv = lambda x: -np.log(1-x)

#generate r.v.s
size = 1000
X = InverseTransform(Phi_inv, size)

histX = plt.hist(X, label='histogram')

x_cod = np.linspace(0,X.max(),1000)
y_cod = phi(x_cod)*histX[0][0]
plt.plot(x_cod,y_cod, label='density function');

plt.legend();
```


density function
histogram