Illustration of the Central Limit Theorem

The average of several independently distributed random varibles tends to a Gaussian distribution, no matter the underlying distributions.

Illustration of the Central Limit Theorem

Let's sample 10000 points from a uniform distribution in $[-1, 1]$ and plot the empirical density.

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt

numpoints = 10000
samples = np.random.uniform(-1,1,numpoints)

fig, ax = plt.subplots()
N, bins, _ = ax.hist(samples,30,density=True)
ax.set(ylabel='Probability Density',xlabel='Value')
fig.tight_layout(); plt.grid(); plt.show()
```

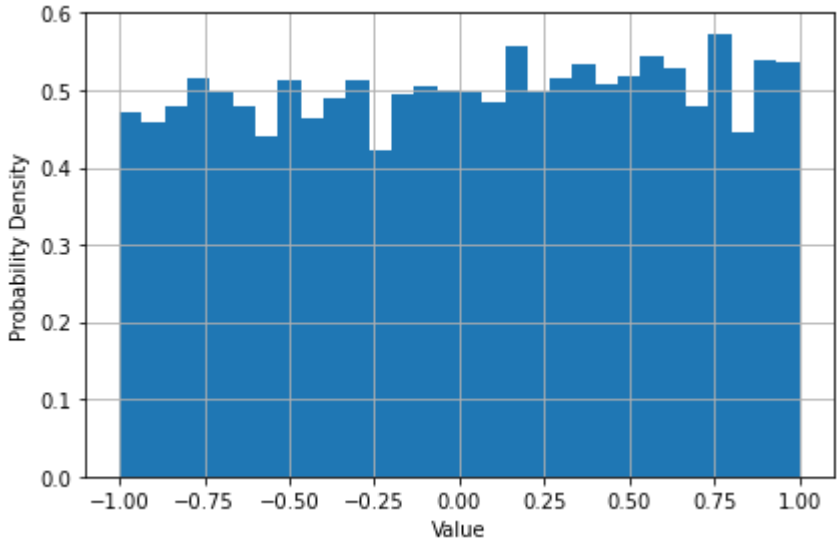


Illustration of the Central Limit Theorem

Let's sample 10000 points created by averaging n points, each sampled from a uniform distribution in $[-1, 1]$, and plot the empirical density.

In [9]:
```
numpoints = 10000
n=2 # increase me and watch the magic happen!
```

```
samples = np.mean(np.random.uniform(-1,1,(numpoints,n)),axis=1)

fig, ax = plt.subplots()
N, bins, _ = ax.hist(samples,30,density=True)
ax.set(ylabel='Probability Density',xlabel='Value')
fig.tight_layout(); plt.grid(); plt.show()
```

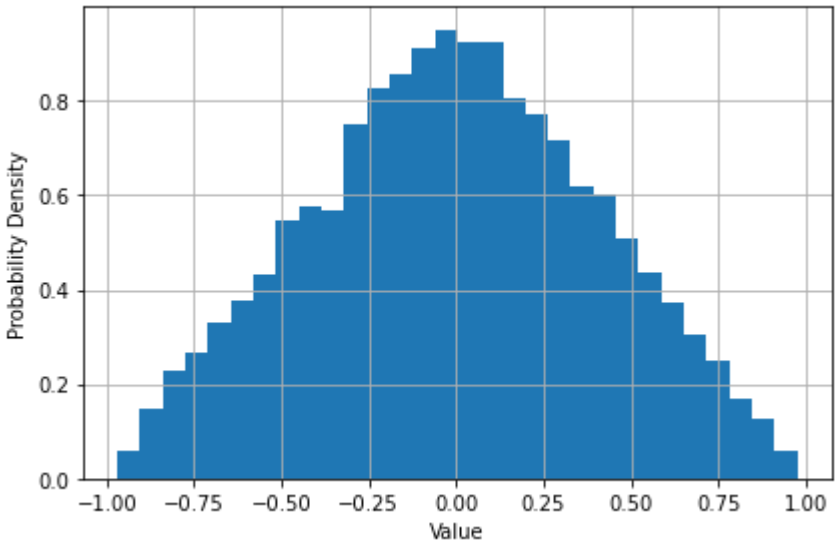


Illustration of the Central Limit Theorem

Let's sample 10000 points created by averaging n points, each sampled from an exponential distribution with parameter $\beta = 1/\lambda$, and plot the empirical density.

In [13]:

```
numpoints = 10000
n = 2 # increase me and watch the magic happen!
lamb = 1
samples = np.mean(np.random.exponential(1/lamb,(numpoints,n)),axis=1)

fig, ax = plt.subplots()
N, bins, _ = ax.hist(samples,30,density=True)
ax.set(ylabel='Probability Density',xlabel='Value')
fig.tight_layout(); plt.grid(); plt.show()
```

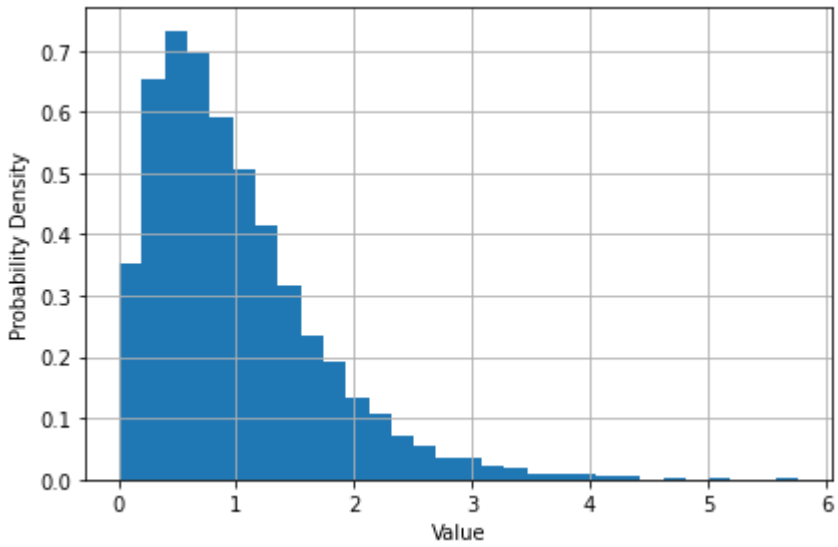


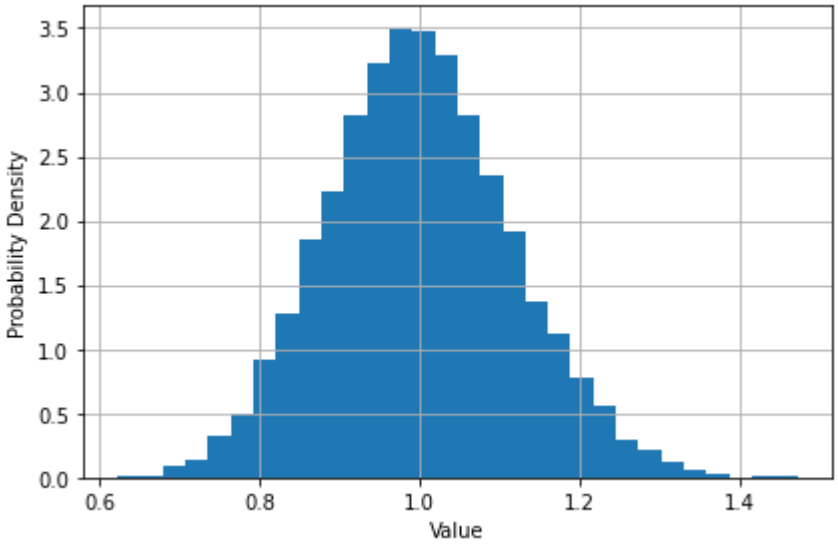
Illustration of the Central Limit Theorem

Let's sample 10000 points created by averaging n points, half of which are sampled from an exponential distribution with parameter $\beta = 1/\lambda$, and the other half sampled from a uniform distribution in $[-1, 1]$, and plot the empirical density.

In [14]:

```
numpoints = 10000
n = 200 # increase me (even) and watch the magic happen!
lamb = 1
samples = np.mean(np.random.uniform(-1,1,(numpoints,int(n/2))) + \
                  np.random.exponential(1/lamb,(numpoints,int(n/2))),axis=1)

fig, ax = plt.subplots()
N, bins, _ = ax.hist(samples,30,density=True)
ax.set(ylabel='Probability Density',xlabel='Value')
fig.tight_layout(); plt.grid(); plt.show()
```



In [ ]: