hw3

November 13, 2020

[2]:
```
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import scipy.stats

sns.set()
```

0.1 Task 1 (N1)

[75]:
```
def length1(X, z_alpha, n):
    ans = 2 * z_alpha / (np.sqrt(n) * (np.cumsum(X) / n))

    return ans

def length2(X, z_alpha, n):
    ans = (
        np.sqrt(2 * n / np.cumsum(X**2))
        * (np.sqrt(1 + z_alpha / np.sqrt(n)) - np.sqrt(1 - z_alpha / np.
 ↪sqrt(n)))
    )

    return ans
```

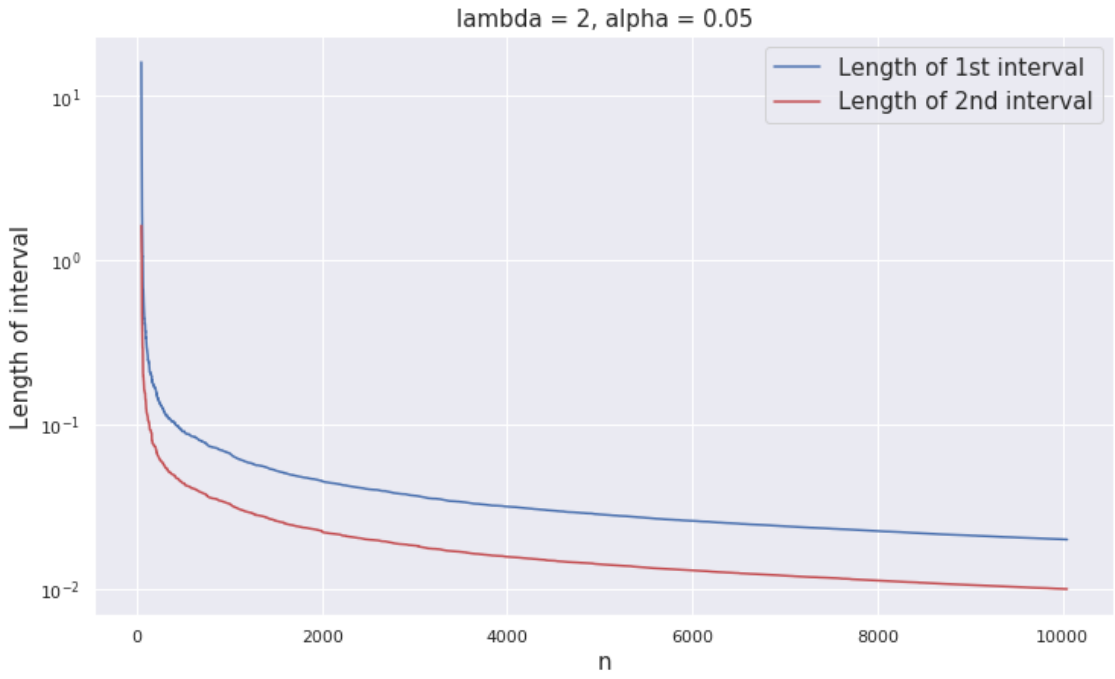
[76]:
```
alpha = 0.05
lam = 2
size = 10000
n = np.arange(50, size + 50)

z_alpha = scipy.stats.norm.ppf(1 - alpha/2)
X = np.random.exponential(lam, size)

l1 = length1(X, z_alpha, n)
l2 = length2(X, z_alpha, n)
```

[77]:
```
fig, ax = plt.subplots(figsize=(12, 7))
```

```
ax.plot(n, l1, c="b", label="Length of 1st interval")
ax.plot(n, l2, c="r", label="Length of 2nd interval")
ax.set_yscale("log")
ax.set_xlabel("n", fontsize=15)
ax.set_ylabel("Length of interval", fontsize=15)
ax.set_title("lambda = {}, alpha = {}".format(lam, alpha), fontsize=15)
ax.legend(fontsize=15)
fig.savefig("Images/1_1_2.png")
```



[80]:
```
lam = 2
n = 10000
alpha = np.linspace(0.01, 1 - 1e-8, n)

z_alpha = scipy.stats.norm.ppf(1 - alpha / 2)
X = np.random.exponential(lam, size)

l1 = length1(X, z_alpha, n)
l2 = length2(X, z_alpha, n)
```

[82]:
```
fig, ax = plt.subplots(figsize=(12, 7))

ax.plot(alpha, l1, c="b", label="Length of 1st interval")
ax.plot(alpha, l2, c="r", label="Length of 2nd interval")
ax.set_yscale("log")
ax.set_xlabel("alpha", fontsize=15)
```

```
ax.set_ylabel("Length of interval", fontsize=15)
ax.set_title("lambda = {}, n = {}".format(lam, size), fontsize=15)
ax.legend(fontsize=15)
fig.savefig("Images/1_2_1.png")
```