[ ]:

```
import numpy as np
import matplotlib
import matplotlib.pyplot as plt
```

[ ]:

```
mu = [0.2, 0.8]
# mu = [0.2, 0.4, 0.6, 0.8]

def SEA(alpha, T):
    t = 0
    A = [i for i in range(len(mu))]
    n = [0 for i in range(len(mu))]
    r = [0. for i in range(len(mu))]
    while t < T:
        for arm in A:
            r[arm] += mu[arm] + np.random.randn()
            n[arm] += 1
            t += 1
            if t >= T:
                return r, n
        if len(A) > 1:
            means = np.array([r[arm]/n[arm] for arm in A])
            bonus = np.array([alpha * np.sqrt(np.log(T)/n[arm]) for arm in A])
            upper = means + bonus
            lower_val = np.max(means - bonus)
            sign = upper > lower_val
            A = [A[i] for i in range(len(A)) if sign[i]]
    return r, n

def UCB(alpha, T):
    t = 0
    A = [i for i in range(len(mu))]
    n = [0 for i in range(len(mu))]
    r = [0. for i in range(len(mu))]
    while t < T:
        arm = np.argmax([r[arm]/(n[arm] + 1e-10) + \
                          alpha * np.sqrt(np.log(T)/(n[arm] + 1e-10)) for arm in␣
 ↪A])
        r[arm] += mu[arm] + np.random.randn()
        n[arm] += 1
        t += 1
    return r, n

def ThS(alpha, T):
    t = 0
    A = [i for i in range(len(mu))]
    n = [0 for i in range(len(mu))]
    r = [0. for i in range(len(mu))]
    sample = [0. for i in range(len(mu))]
    while t < T:
        for arm in A:
            mu_hat = (0 + r[arm]) / (n[arm] + alpha**2)
            sigma_hat = alpha / np.sqrt(n[arm] + alpha**2)
            sample[arm] = np.random.normal(loc=mu_hat, scale=sigma_hat)
        arm = np.argmax(sample)
        r[arm] += mu[arm] + np.random.randn()
        n[arm] += 1
        t += 1
    return r, n

def SEA_new(alpha, T):
    t = 0
    A = [i for i in range(len(mu))]
    n = [0 for i in range(len(mu))]
    r = [0. for i in range(len(mu))]
    while t < T:
        for arm in A:
            r[arm] += mu[arm] + np.random.randn()
            n[arm] += 1
            t += 1
            if t >= T:
                return r, n

        if len(A) > 1:
            means = np.array([r[arm]/n[arm] for arm in A])
            bonus = np.array([alpha * np.sqrt(T*np.log(T))/n[arm] for arm in A])
            upper = means + bonus
            lower_val = np.max(means - bonus)
            sign = upper > lower_val
            A = [A[i] for i in range(len(A)) if sign[i]]
    return r, n

def UCB_new(alpha, T):
    t = 0
    A = [i for i in range(len(mu))]
    n = [0 for i in range(len(mu))]
    r = [0. for i in range(len(mu))]
    while t < T:
        arm = np.argmax([r[arm]/(n[arm] + 1e-10) + \
                            alpha * np.sqrt(T*np.log(T))/(n[arm] + 1e-10) for arm␣
 ↪in A])
        r[arm] += mu[arm] + np.random.randn()
        n[arm] += 1
        t += 1
    return r, n
```

[ ]:

```
T = 500
N = 5000
```

[ ]:

```
sea_old = {}
ucb_old = {}
ths = {}
sea_new = {}
ucb_new = {}

eta_list = [0.1, 0.2, 0.4, 0.8]

for alpha in eta_list:
    print(alpha, end = ' ')

    print("sea_old", end=' ')
    sea_old[alpha] = [SEA(alpha, T)[0] for i in range(N)]
    sea_old[alpha] = np.sum(sea_old[alpha], axis=1)

    print("ucb_old", end=' ')
    ucb_old[alpha] = [UCB(alpha, T)[0] for i in range(N)]
    ucb_old[alpha] = np.sum(ucb_old[alpha], axis=1)

    print("ths", end=' ')
    ths[alpha] = [ThS(alpha, T)[0] for i in range(N)]
    ths[alpha] = np.sum(ths[alpha], axis=1)

    print("sea_new", end=' ')
    sea_new[alpha] = [SEA_new(alpha, T)[0] for i in range(N)]
    sea_new[alpha] = np.sum(sea_new[alpha], axis=1)

    print("ucb_new", end=' ')
    ucb_new[alpha] = [UCB_new(alpha, T)[0] for i in range(N)]
    ucb_new[alpha] = np.sum(ucb_new[alpha], axis=1)

    print("Completed.")
```

[ ]:

```
policy = [sea_old, ucb_old, ths, sea_new, ucb_new]
name = ['SE', 'UCB', 'TS', 'SE_new', 'UCB_new']
i = 0
for p in policy:
    print(name[i])
    i += 1
    for para in eta_list:
        print(para, np.mean(p[para]))
```

[ ]:

```
name = ['SE-', 'UCB-', 'TS-', 'SE_new-', 'UCB_new-']
fig = plt.figure(figsize=(16, 8))
```

```
policy = [sea_old, ucb_old, ths, sea_new, ucb_new]
font = {'size':12}

matplotlib.rc('font', **font)
for i in range(6):
    axx = fig.add_subplot(2, 3, i+1)
    arr = policy[i]
    for alph in [0.1, 0.2, 0.4, 0.8]:
        axx.hist(arr[alph], bins=200, range=(0, 1.02*T), alpha=0.3,
                  label=name[i]+str(alph), density=True)
    axx.legend(prop={'size':12}, loc='upper left')
    axx.set_ylim([0, 0.02])
    axx.get_yaxis().set_ticks([])
plt.subplots_adjust(wspace=0.1)
```