ETC algorithm

Explore-then-commit in Python

June 22, 2022

Algorithm

Explore-then-commit

1. **Input** m.
2. In round t choose action

$$A_t = \begin{cases} (t \bmod k) + 1 & \text{if } t \leqslant mk \\ \operatorname{argmax}_i \hat{\mu}_i(mk) & \text{if } t > mk \end{cases}$$

Code

```python
def get_reward(rew_avg) -> np.ndarray:
    # Add epsilon (sub-gaussian noise) to reward.
    mean = np.zeros(rew_avg.size)
    cov = np.eye(rew_avg.size)
    epsilon = np.random.multivariate_normal(mean, cov)
    reward = rew_avg + epsilon

    return reward
```

均值 $E(X_i) = 0, \quad \forall 1 \leqslant i \leqslant n$

协方差 $cov(X, Y) = 0$

方差 $D(X_i) = 1, \quad \forall 1 \leqslant i \leqslant n$

随机生成服从 1-subgaussian 的 epsilon，reward 也服从 1-subgaussian

Code

在 *run_algo* 函数中一些变量初始化：

```
regret = np.zeros((num_trial, num_iter))
k = rew_avg.size
max_arm = np.argmax(rew_avg)
```

对于每一次 trial 中变量初始化

```
means = np.zeros(rew_avg.size)
num = np.zeros(rew_avg.size)
cum = [0]
```

means 记录每一个 arm 的均值，num 记录每一个 arm 被选择的次数（方便算均值），cum 记录每一次增加后 regret 的累积量

Code

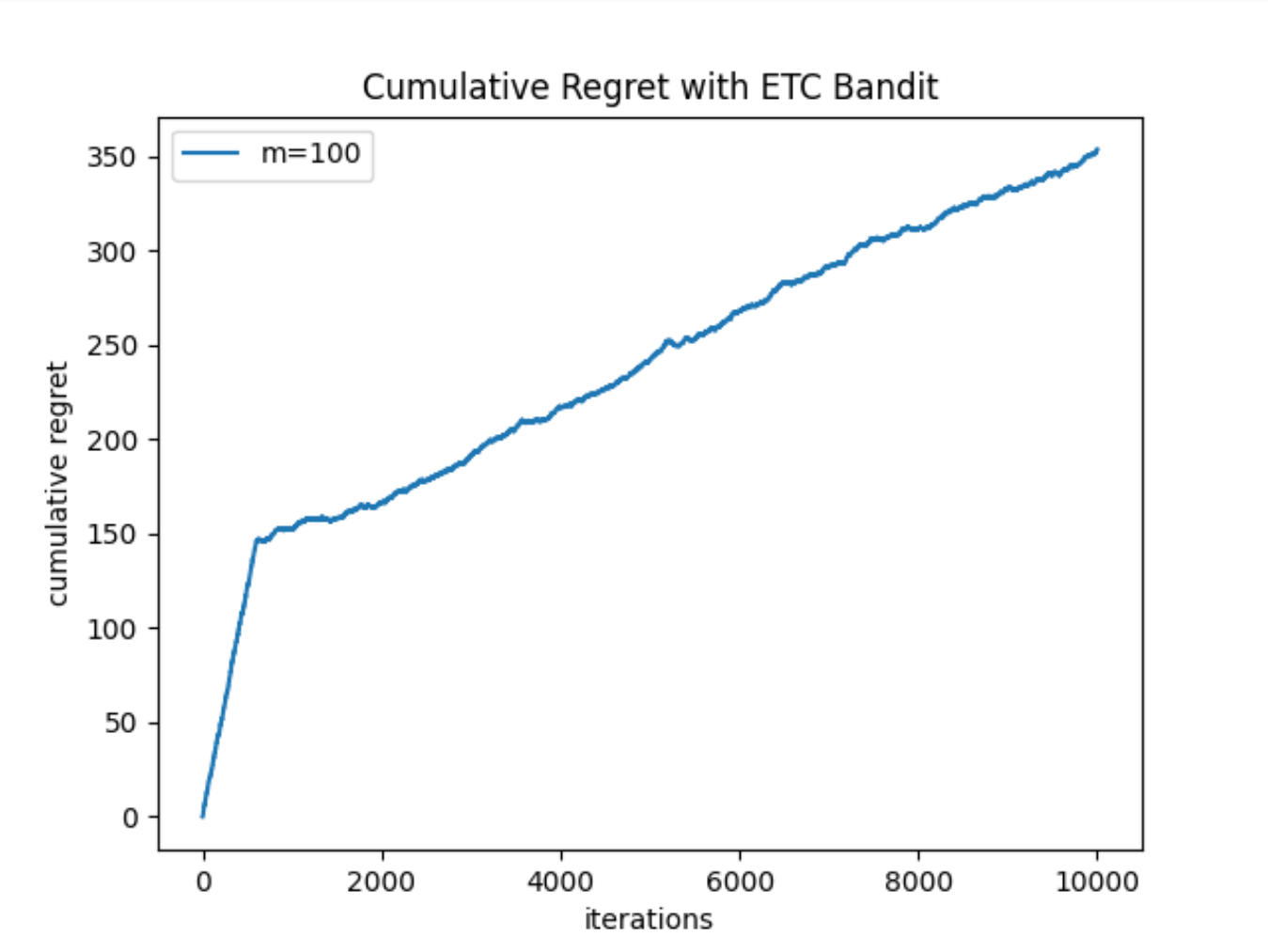
```
for i in range(num_trial):
        for t in range(num_iter - 1):
        rew = get_reward(rew_avg)
        if t <= m * k:
            chosen_arm = t % k
            num[chosen_arm] += 1
            means[chosen_arm] += (rew[chosen_arm] - means[
                chosen_arm]) / num[chosen_arm]
        else:
            chosen_arm = np.argmax(means)

        reg = rew[max_arm] - rew[chosen_arm]
        reg += cum[-1]
        cum.append(reg)
        regret[i, :] = np.asarray(cum)
return regret
```

Code

绘制 cumulative regret 随着轮数增加的变化趋势的折线图

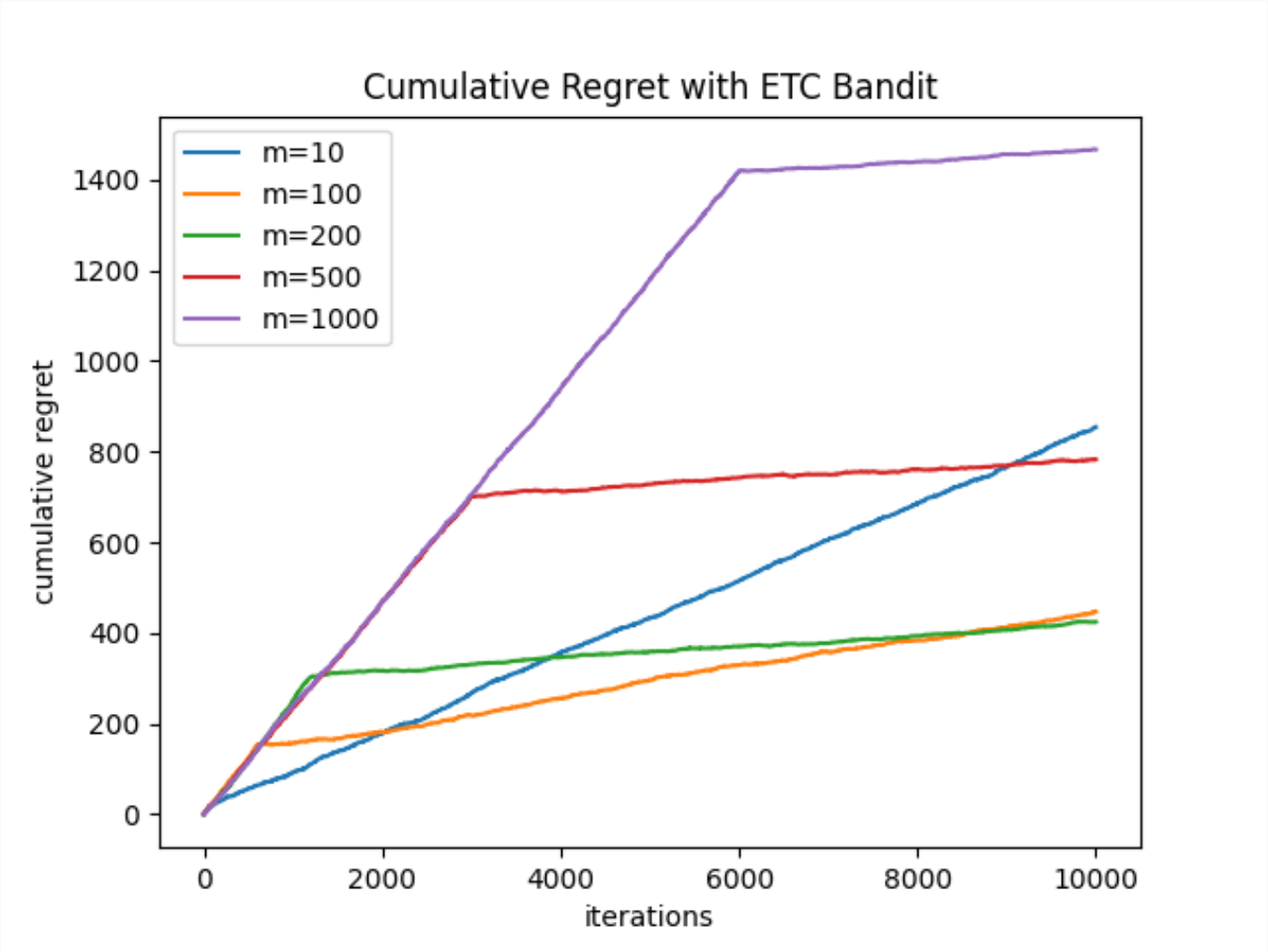
```
for m_val, reg in zip(m, regrets):
    plt.plot(reg, label="m=" + str(m_val))

plt.xlabel('iterations')
plt.ylabel('cumulative regret')
plt.title('Cumulative Regret with ETC Bandit')
plt.legend()
plt.show()
```

Results



在前 mk 轮次中寻找收益最大的 arm，在 mk 轮次后 regret 的增加量趋于稳定 (始终选择收益最大的 arm)。

Results



在 m 增加的过程中，能够找到收益最大的 arm 的几率更高，在前 mk 轮次中所付出的代价会越高，但在 mk 轮次后所付出的代价会降低。

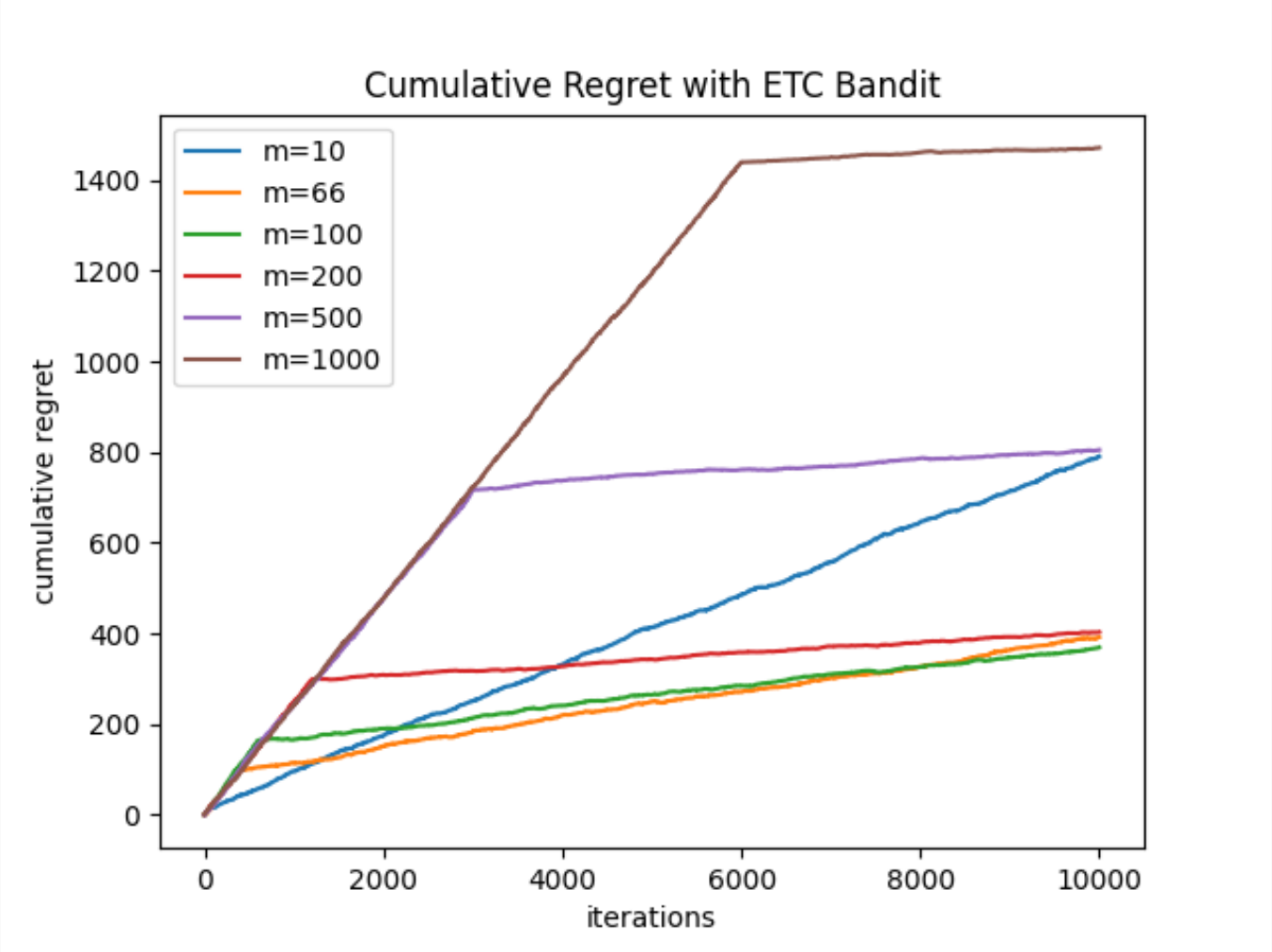
Algorithm

如何达到一个平衡？选取合适的 m，

Choose m

$$m = \max\left\{1, \left\lceil \frac{4}{\Delta^2} \log\left(\frac{n\Delta^2}{4}\right)\right\rceil\right\}$$

Results


Cumulative Regret with ETC Bandit
m=10
m=66
m=100
m=200
m=500
m=1000
cumulative regret
0
200
400
600
800
1000
1200
1400
0
2000
4000
6000
8000
10000
iterations