蒙特卡洛算法实验报告

吴孟周　2100013053

一、实验要求

（1）阅读《基于蒙特卡洛的路径规划-实验指导书》，尝试运行并理解蒙特卡洛算法在冰湖路径规划问题上的示例代码。
（2）在示例代码的基础上，尝试在策略评估中使用不同数值的迭代次数，比较策略收敛所需的迭代次数（由于随机性建议多次实验），分析策略评估的迭代次数对总的算法效率的影响。
（3）在示例代码的基础上，尝试实现不同的 epsilon 衰减策略，比较策略收敛所需的迭代次数（由于随机性建议多次实验），分析不同 epsilon 衰减策略对算法效率的影响。

二、代码修改思路

首先给 policy_iteration_MC 加入了额外的返回值，迭代次数 iteration 和消耗时间 time_consumed。

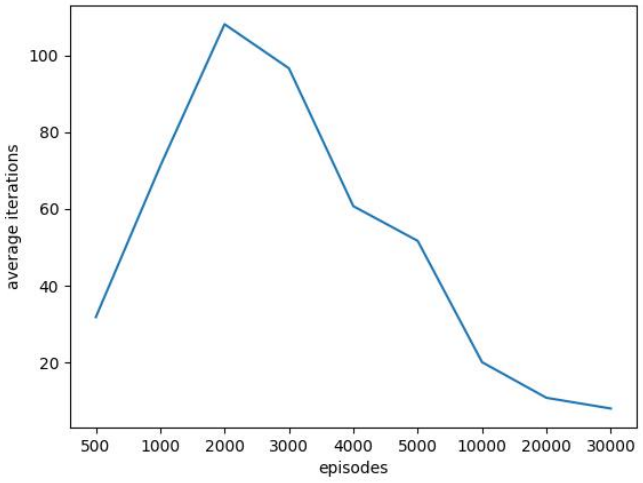
因为具有随机性，需要多次实验，实现了函数 PI_average，这个函数将相同的参数进行 100 次实验，并将迭代次数，消耗时间和策略成功率打印在 log 里。

为了实现不同的 epsilon 衰减策略并比较，为 policy_iteration_MC 加入参数 fn，这个参数需要提供一个函数，接受 eps0, decay, iteration 作为参数，返回当前的 epsilon 值。接着实现了四种不同的衰减策略，指数，线性，倒数（也就是默认的）和固定 eps 值。

```python
def exp_decay(eps0, decay, iteration):
    return eps0 * (decay ** iteration)

def linear_decay(eps0, decay, iteration):
    eps = max(eps0 - decay * iteration, 0.05)
    return eps

def inverse_decay(eps0, decay, iteration):
    return eps0 / (1 + decay * iteration)

def fix_decay(eps0, decay, iteration):
    return eps0
```

三、实验结果和分析

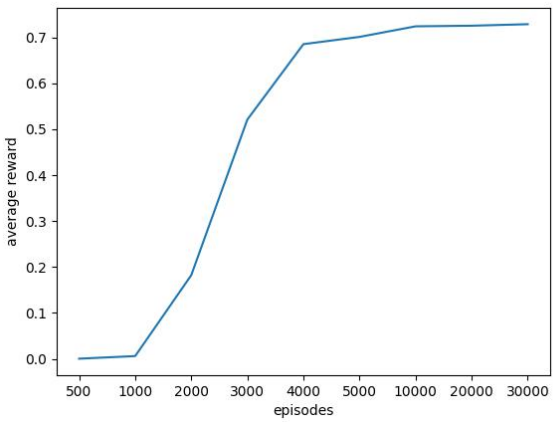
首先研究了修改 episodes 参数，即策略评估的采样轮数对策略迭代轮数的影响。对每个参数进行了 100 次实验，取算术平均值见右图。可以发现在采样轮数取 2000 时迭代轮数达到最高，向左和向右分别递减。



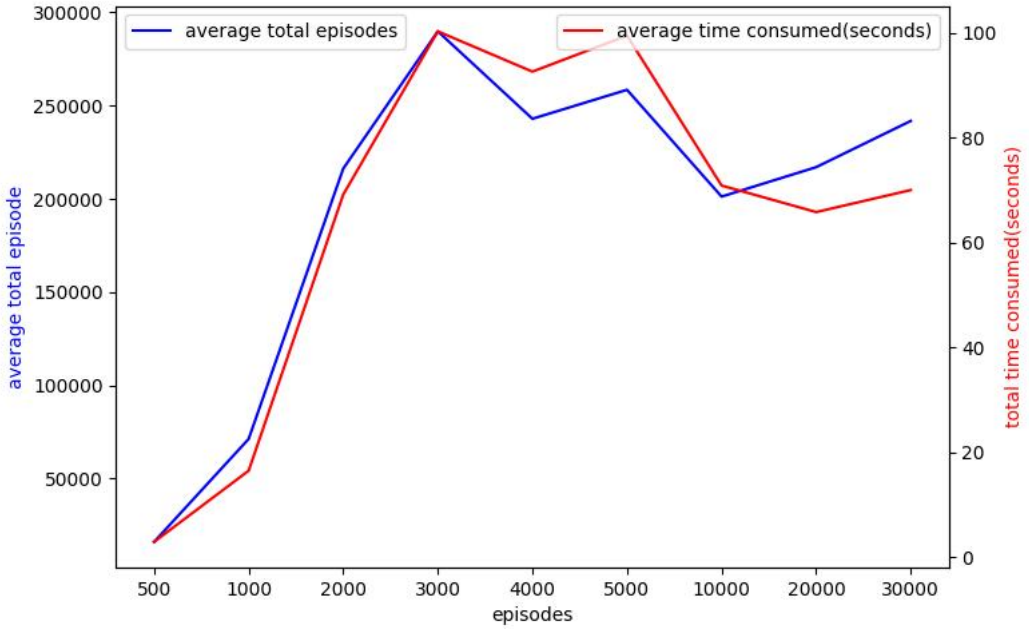
向右递减是容易理解的，采样轮数更高的情况下，策略评估的更准，那么一定可以通过更少的迭代轮数达到收敛。

向左的递减主要是因为策略根本没有达到收敛，就因为随机因素提前终止（这是因为在本次实验中判断收敛的方法为两次之间最优动作没有改变）。针对这个问题，我进行了进一步的实验以测量策略是否真正的收

敛。通过观测 100 次实验中每次策略的成功率的平均值，绘制右图。可以发现对于 episodes 较小的情况，最终策略的回报很低，即没有找到一个好的策略。



进一步的，我们关心总的效率，下图中 average total episodes 表示平均总共进行了多少轮采样。average time consumed 表示平均每次实验花费了多少时间，可以发现这两个指标是类似的。另外除了在 episodes 较小时因为策略未真正收敛而耗时较少之外，随着 episodes 的增大，因为策略评估更准了，总的耗时有一定程度的降低。



针对不同的 epsilon 衰减策略，同样进行了 100 次实验并取平均值。因为 episodes 固定取 5000，策略迭代轮数大概相当于算法效率，因此我们在下面两张图分别展示四种衰减策略的平均迭代轮数和平均回报。其中 eps0 均取 0.5，为了控制迭代轮数接近以比较平均回报，exp decay 的 decay 取 0.965，linear decay 的 decay 取 0.01，inverse decay 的 decay 取 0.1。如果不考虑实验误差，可以发现 exp decay 的表现最差，linear decay 的表现最优。