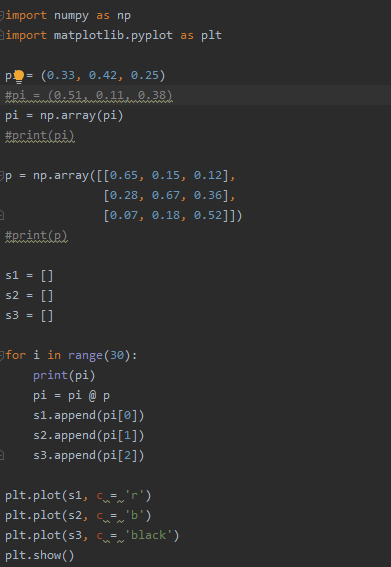
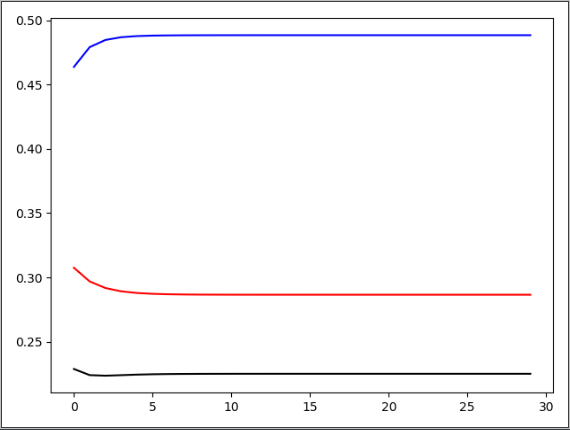
取pi = [0.33，0.42，0.25],代码如下：



所得的结果图形如下:



程序经过30次迭代的结果如下：

[0.33 0.42 0.25]

[0.3075 0.4638 0.2287]

[0.296889 0.479178 0.223933]

[0.29172651 0.48479406 0.22347943]

[0.28915887 0.48694804 0.22389309]

[0.28786264 0.48782118 0.22431617]

[0.28720184 0.48819555 0.22460261]

[0.28686284 0.48836448 0.22477268]

[0.28668824 0.48844396 0.2248678 ]

[0.28659809 0.48848257 0.22491935]

[0.28655146 0.48850175 0.22494679]

[0.28652733 0.48851143 0.22496125]

[0.28651483 0.48851636 0.22496882]

[0.28650835 0.48851888 0.22497277]

[0.28650499 0.48852019 0.22497482]

[0.28650325 0.48852086 0.22497589]

[0.28650235 0.48852121 0.22497645]

[0.28650188 0.48852139 0.22497673]

[0.28650164 0.48852148 0.22497688]

[0.28650151 0.48852153 0.22497696]

[0.28650145 0.48852155 0.224977 ]

[0.28650141 0.48852157 0.22497702]

[0.2865014 0.48852157 0.22497703]

[0.28650139 0.48852158 0.22497704]

[0.28650138 0.48852158 0.22497704]

[0.28650138 0.48852158 0.22497704]

[0.28650138 0.48852158 0.22497704]

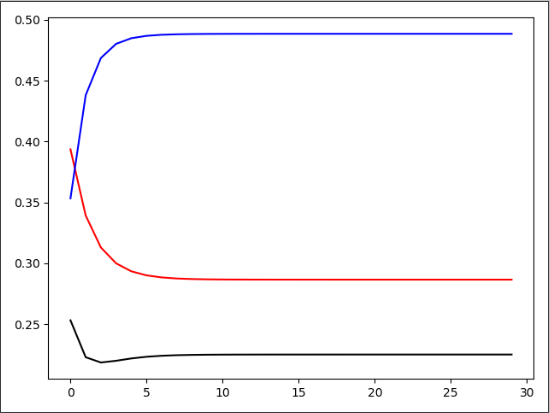
[0.28650138 0.48852158 0.22497704]

[0.28650138 0.48852158 0.22497704]

[0.28650138 0.48852158 0.22497704]

可以看到最终收敛于[0.29,0.49,0.22]。

接下来将pi的初始值改为pi = [0.51,0.11,0.38]，所得的图形如下：



[0.51 0.11 0.38]

[0.3936 0.3533 0.2531]

[0.339207 0.438035 0.222758]

[0.31292076 0.46865429 0.21842495]

[0.29990763 0.48024917 0.2198432 ]

[0.29335852 0.48488463 0.22175685]

[0.29002655 0.48684555 0.22312789]

[0.28831944 0.48772 0.22396056]

[0.2874409 0.48812764 0.22443145]

[0.28698751 0.4883243 0.22468819]

[0.28675311 0.48842153 0.22482536]

[0.28663179 0.48847043 0.22489778]

[0.28656896 0.48849529 0.22493575]

[0.28653641 0.48850802 0.22495557]

[0.28651954 0.48851457 0.22496589]

[0.28651079 0.48851796 0.22497125]

[0.28650626 0.4885197 0.22497404]

[0.28650391 0.48852061 0.22497548]

[0.28650269 0.48852108 0.22497623]

[0.28650206 0.48852132 0.22497662]

[0.28650173 0.48852144 0.22497683]

[0.28650156 0.48852151 0.22497693]

[0.28650147 0.48852154 0.22497698]

[0.28650143 0.48852156 0.22497701]

[0.2865014 0.48852157 0.22497703]

[0.28650139 0.48852157 0.22497704]

[0.28650138 0.48852158 0.22497704]

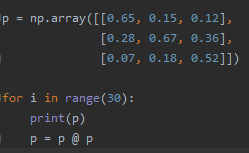
[0.28650138 0.48852158 0.22497704]

[0.28650138 0.48852158 0.22497704]

[0.28650138 0.48852158 0.22497704]

可以看到最终收敛于[0.29,0.49,0.22]。

总结：通过取不同的初始值，两次模拟的结果收敛于相同的值，这说明马尔可夫链的收敛和初始状态无关，只与转移状态矩阵有关。当我们只迭代计算转移状态的矩阵时，有如下结果：



[[0.65 0.15 0.12]

[0.28 0.67 0.36]

[0.07 0.18 0.52]]

[[0.4729 0.2196 0.1944]

[0.3948 0.5557 0.462 ]

[0.1323 0.2247 0.3436]]

[[0.33605161 0.26956224 0.2601828 ]

[0.46721388 0.49931197 0.49222572]

[0.19673451 0.23112579 0.24759148]]

[[0.29006084 0.28531743 0.28453936]

[0.48713125 0.48902172 0.4892061 ]

[0.22280791 0.22566085 0.22625454]]

…

[[0.28650138 0.28650138 0.28650138]

[0.48852159 0.48852159 0.48852159]

[0.22497705 0.22497705 0.22497705]]

[[0.28650138 0.28650138 0.28650138]

[0.48852159 0.48852159 0.48852159]

[0.22497705 0.22497705 0.22497705]]

可以看到转移矩阵P里的每一行的值最后均收敛于同一个值，而且每一列相加的概率和为1，这成为马氏链的平稳分布，所以这就是“阶级固化”的原因。

参考：https://blog.csdn.net/qq\_34652535/article/details/85343518