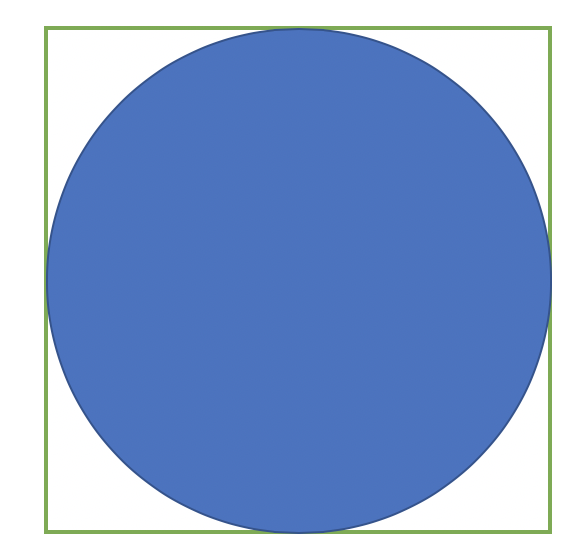
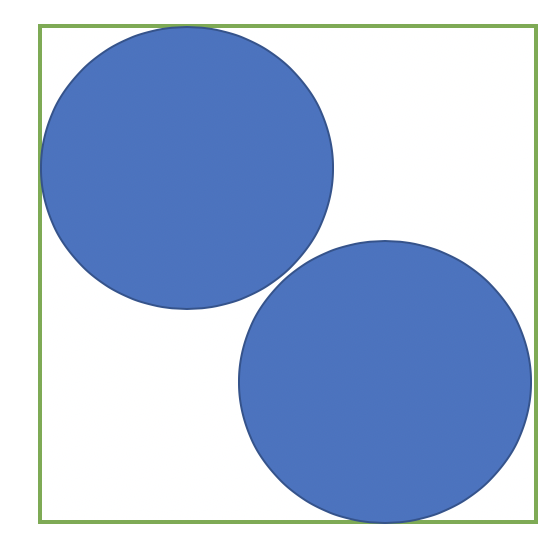
**第三周实验报告**

思路：

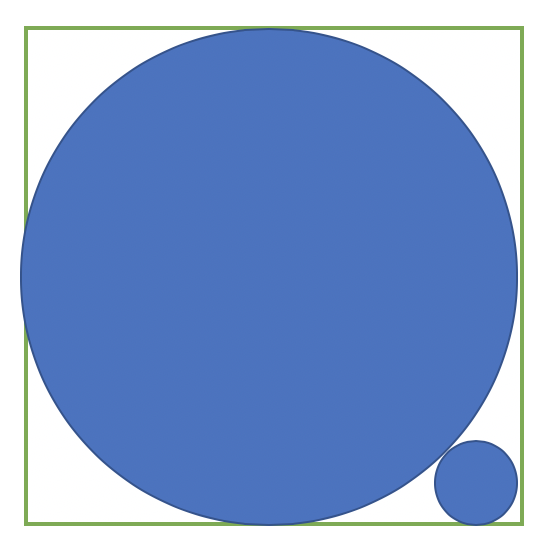
1、首先考虑只有一个圆的情况：该圆与四条边相切的时候面积最大→此时有



2、再考虑框中塞下两个半径相等的圆并使其面积最大，有圆面积之和为。



因此增大其中一个圆的面积使其与四条边相切，此时两个圆的半径平方之和显然大于只有一个圆时的



3、猜测：在一个圆的基础上，每次不断找到面积最大的空隙，然后向其中填入最大的内切圆，就能得到最大的

4、证明：考虑m的情况，此时：

1.若每次packing的圆面积相等，可看作此时每个圆的无穷小，记为，

2.若在一个最大圆的基础上packing，则可看作无数个半径的小圆packing成了大圆。因此推出，要达到相同的，在一个最大圆的基础上packing的方法需要的圆m的个数小得多。即随着m增大 ，一个最大圆的基础上packing的方法的增长较快。

经过以上分析，该问题转化为每次迭代寻找最大空隙，向空隙中填入最大圆的问题。由于我的python中numpy等工具始终安装不上，只能用matlab代写。

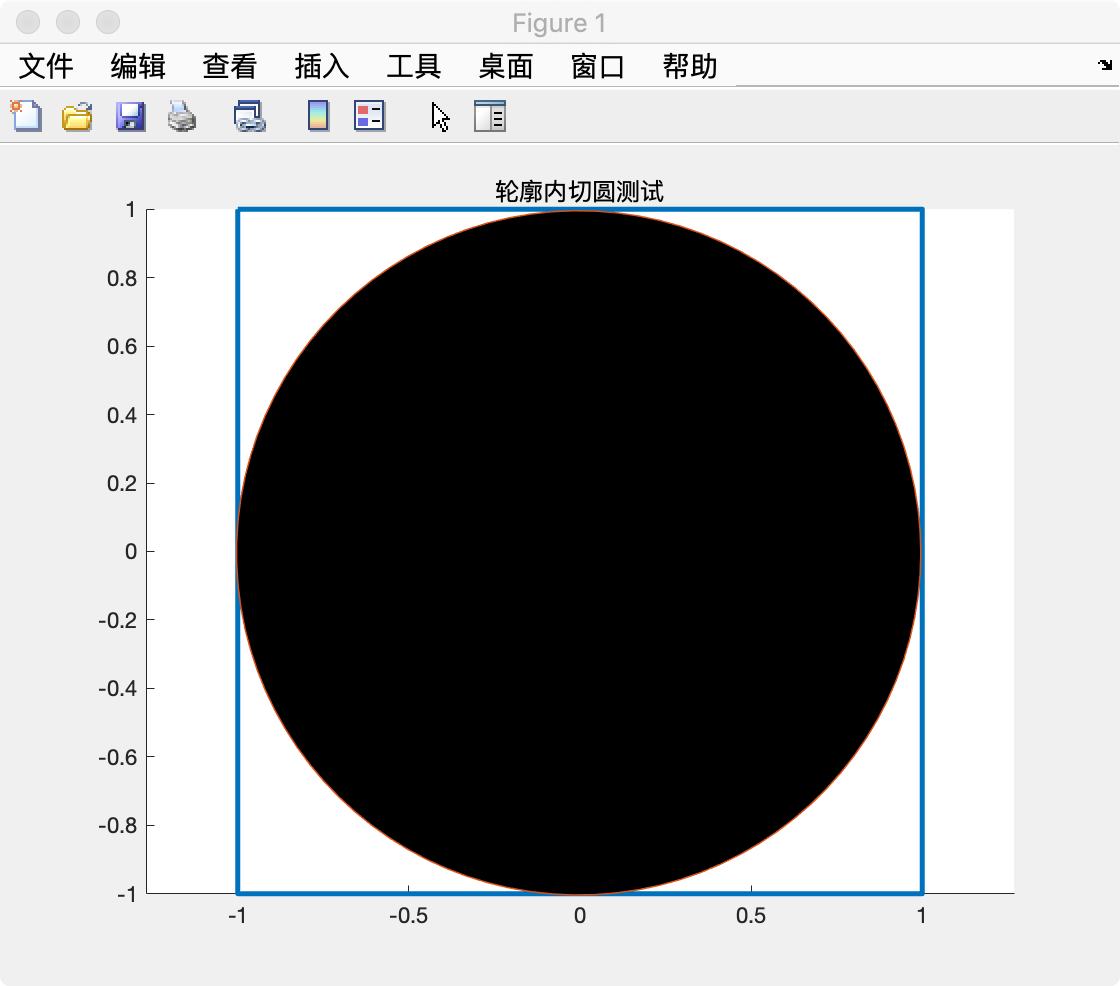
Matlab程序思路：

1、首先需要能在一定范围内找到最大内切圆的函数，查阅后学习了已有的程序思路[1]:即将图像转化为像素，遍历像素找到最大的内切圆。

2、把每次得到的新区域转换为像素，再迭代搜索最大的内切圆。

运行结果：

M=1时：



总结：

程序在m更多时迭代出现了问题，原因是每次像素上下边界没有很好的办法分离出来。导致搜索会跑到大圆的内部。这个问题暂时没有解决，考虑的解决方法是将每次生成的图案二值化后作为下一次迭代的输入，依靠像素的颜色分辨空隙；但是截至目前这部分程序bug没有解决：附件中的Matlab程序去掉了这一部分。

参考：[1] https://blog.csdn.net/qq\_34626136/article/details/93314025