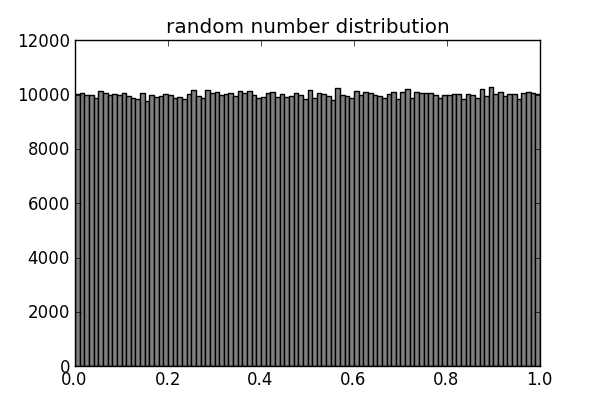
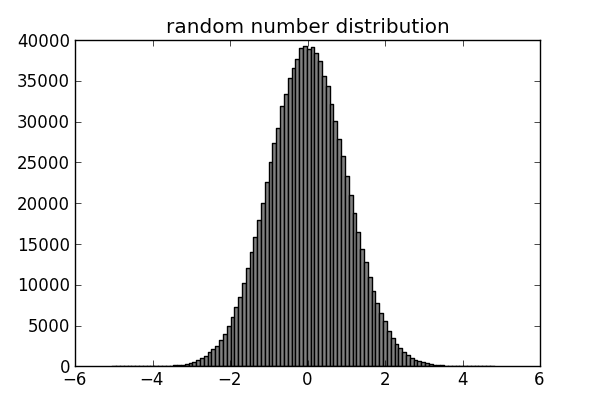
今年某公司的笔试题目还蛮有意思的，原题不还没见到，不过经过一系列变化之后，可以等价地表述为如下：

如何利用一个能够返回[0,1]平均随机点的函数，等概率地生成一个单位圆中的点，使得生成地点在圆内的分布概率尽量平均，即在面积上平均分布。

首先，要弄明白[0,1]之间的平均随机是指什么；其次，还需要搞清楚在面积上平均分布是指什么。

下面两个图分别是平均随机和正态随机的分布情形：

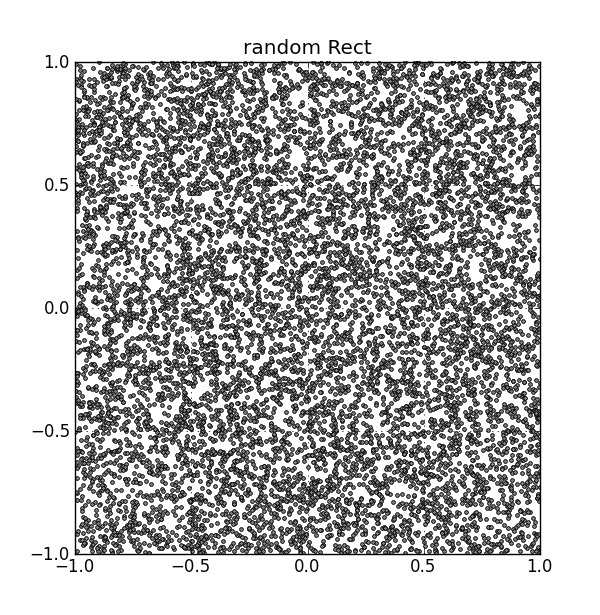




左图中的随机算法是平均随机的，就是说生成的数字在区间内的每个区间的个数基本相同；

而右图中的随机算法是正态随机，看这个图形这个是就是以N(\mu,\sigma)正态分布，\mu, \sigma分别是均值与方差。

下图是一个依靠[0,1]平均分布产生的正方形分布，可以看出，这些点还是比较均匀的分布在正方形当中：

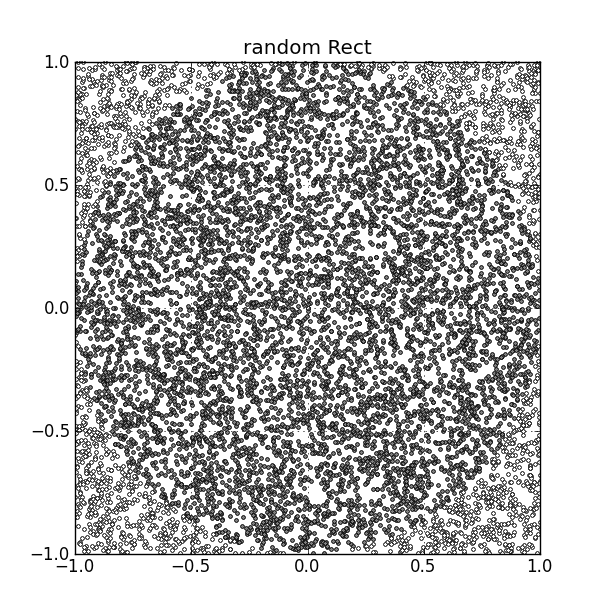


好了，下面来分析如果才能在圆内产生像正方形这样的随机分布，

聪明的孩子一般都会很快的想出算法1：

1. 随机产生(x,y) -1 \le x \le 1, -1 \le y \le 1的矩形
2. 如果 x^2 + y^2 \le 1，返回(x,y)；否则，返回步骤1

这个算法其实就是从平均分布的正方形中扣出一个圆来做为返回（如下图），不过由于正方形的面积比较大，每次取点时，只有 \frac{\pi R^2}{(2R)^2} = \frac{\pi}{4}的概率产生圆上的点（R为圆的半径），所以运气差的时候，我们有可能会碰很多次尝试都无法得到一个圆上的点，这不确定性也使得这个算法看上去不那么可靠。



但其实，相对于运气，我们应该相信科学，其实连续3次取不到圆上的点的概率为(1-\pi/4)^3 = 0.00988，所以我们3次尝试可以取到圆上的点概率已经高达99\%以上。总的说，这个方法虽不是最好，但无错，效率也不低。

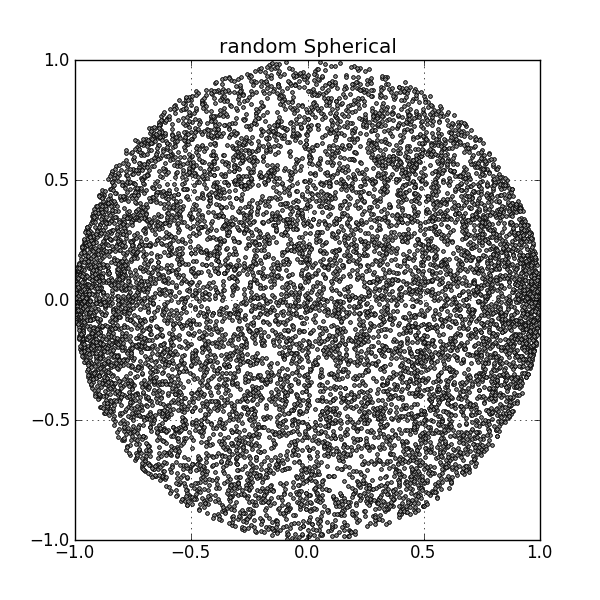
利用x^2 + y^2 \le 1的关系，不少同学会想到算法2：

1. 随机产生 [0,1] 的分布上的x，利用上述公式产生得到y的范围z = \sqrt{1-x^2}

2.随机产生一个 [0,1] 范围内的因子c，y = c* \sqrt{1-x^2}

这个(x,y)就是一个随机点

但是，这个算法却不尽如人意，效果如下图：当x靠近1时，由于y的范围变小了，y会变得很密集，不满足在圆的面积上平均分布的要求。

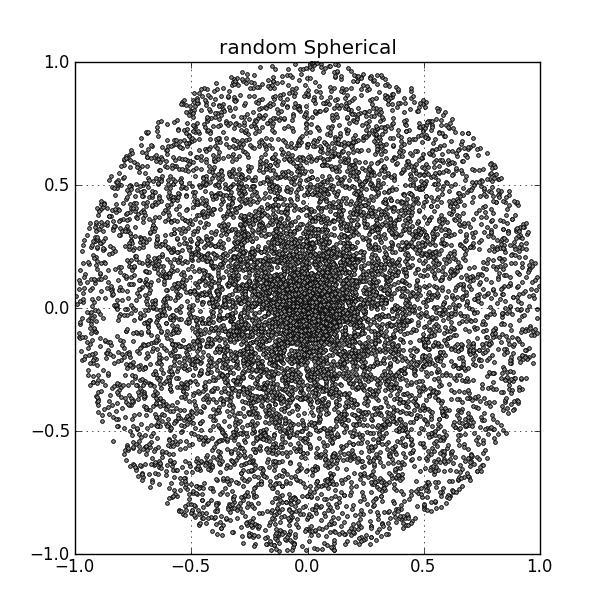


另一些人同学可能会想起极坐标的性质，一个角度加一个半径便可以唯一确认圆上的一点，于是便有了下面的算法3：

1. 随机产生[0,360]上的一个角度\theta
2. 随机产生[0,1]上的一个半径r，

这个点(\theta,r)就是一个随机的圆上的点

实际里，这种算法也有一些问题，效果如下，问题很明显，由于点的半径上的随机分布，会使得靠近圆点附近的点显得过于密（因为这里面积比较小的缘故）。

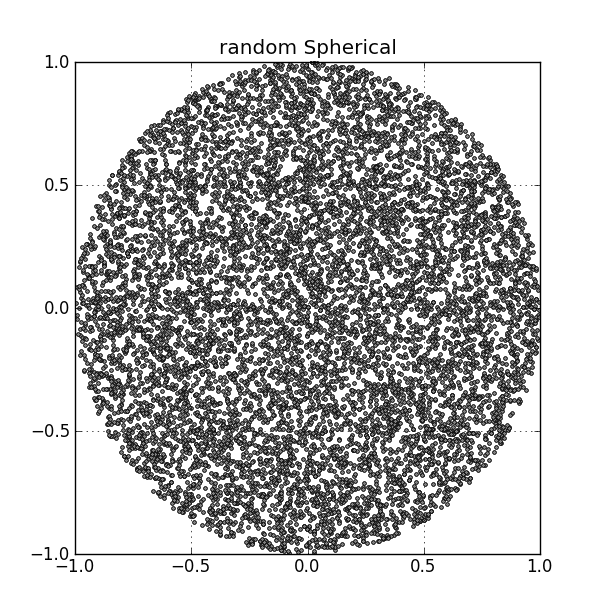


最后，就是正解算法4

1. 随机产生[0,360]上的一个角度\theta
2. 随机产生 [0,1] 上的一个半径的平方r^2，

这个点(\theta,r)就是一个随机的圆上的点

可以看出，算法4与算法3基本上是相同的，唯一的区别就是算法3产生的是r，而算法4中则是产生r^2。而这一点点不同，也使得r的分布在r较大的地方点变得较密，抵消了面积变大的因素，变得平均，当然这只是简单的分析，相信会有牛人给大家来证明一下哈~ 最后，贴效果图，done!



**update 这里补充某牛人对最后一种解法证明，感谢 Luliang同学 :)**

<http://wenku.baidu.com/view/de291225aaea998fcc220ef9.html>

参考

### [可汗学院公开课：统计学\_概率密度函数](http://www.google.com.hk/url?sa=t&rct=j&q=%E6%A6%82%E7%8E%87%E5%AF%86%E5%BA%A6&source=web&cd=8&ved=0CF4QtwIwBw&url=%68%74%74%70%3a%2f%2f%76%2e%31%36%33%2e%63%6f%6d%2f%6d%6f%76%69%65%2f%32%30%31%31%2f%36%2f%48%2f%30%2f%4d%38%32%49%43%36%47%51%55%5f%4d%38%33%4a%41%48%54%48%30%2e%68%74%6d%6c&ei=YIyDUZPeDMXNkwXA-oHoCg&usg=AFQjCNFaeyK1b8nOPOLqb1slYbBnCq2ymA&bvm=bv.45960087,d.dGI&cad=rjt)