

数学编程概率论泊松分布

关注者156

被浏览11,358

如何编程产生泊松分布的随机数？

我是一名大二学生，概率论只学了一半，刚学完连续型随机变量联合分布函数。我们VBA老师留了一项作业：生成泊松分布的随机数，手头工具只能使用Excel自带...显示全部

关注问题

写回答

添加评论

分享

邀请回答

7 个回答

默认排序

知乎用户

是时候了/你在说啥？

60 人赞同了该回答

你说的这个问题叫做Poisson过程，其产生的思路如下：

- 1, 从 $Uniform(1)$ 的均匀分布出发，用Inverse CDF 方法产生一系列独立的指数分布（参数为 λ ）随机数 $X_i \sim exp(\lambda)$ ；
- 2, 记 $Y = X_1 + X_2 + \dots + X_k$ 。如果 $Y > t$ ，则停止，输出 $k - 1$ ；若否，则继续生成 X_{k+1} ，直到 $Y > t$ 为止；
- 3, 重复过程2。

容易证明，输出的一系列整数 k 就满足服从参数为 $\mu = \lambda t$ 的Poisson分布。

编辑于 2015-12-04

60

19 条评论

分享

收藏

感谢

胡宏伟

游戏策划

10 人赞同了该回答

提供一个生成离散型随机数的一般思路，以泊松分布 $P(X = k) = \frac{\lambda^k}{k!} e^{-\lambda}, k = 0, 1, \dots$ 为例：
非常感谢 @王赞 Maigo 的提醒，我把原来的废话都删了，思路很简洁。

只要生成一个0到1之间的随机数，然后看泊松分布的前几项和刚好大于这个随机数就行了。

下面是vba代码，这个可以扩展到其他离散型随机变量，但运行效率不理想：

```
Function PoissonRand(lambda As Double) As Integer
    Dim rand As Single
    Dim k As Integer
    Randomize
    rand = Rnd
    k = 0
    Do While rand > WorksheetFunction.POISSON(k, lambda, True)
        k = k + 1
    Loop
    PoissonRand = k
End Function
```

这里直接用了POISSON函数可以直接获得累积概率。

根据 @王赞 Maigo 优化版，去除阶乘计算提升效率，可读性稍受影响，且其他离散型随机变量很难复用：



下载知乎客户端
与世界分享知识、经验和见解

相关问题

- 如何徒手生成一组随机数？ 16 个回答
- 如何生成总和固定的若干个随机数？ 17 个回答
- 泊松分布的现实意义是什么，为什么现实生活多数服从于泊松分布？ 25 个回答
- xorshift算法生成随机数的原理是什么？ 2 个回答
- 是否有方法产生 $(-\infty, +\infty)$ 的随机数？ 29 个回答

相关推荐

- 激发孩子热爱奥数 and 编程的兴趣

★★★★★ 343 人参与
- 业余自学，如何入门金融和编程？

★★★★★ 10115 人参与
- scikit-learn 机器学习：常用算法原理及编程实战

513 人读过 阅读

刘看山 · 知乎指南 · 知乎协议 · 隐私政策

应用 · 工作 · 申请开通知乎机构号

侵权举报 · 网上有害信息举报专区

违法和不良信息举报：010-82716601

儿童色情信息举报专区

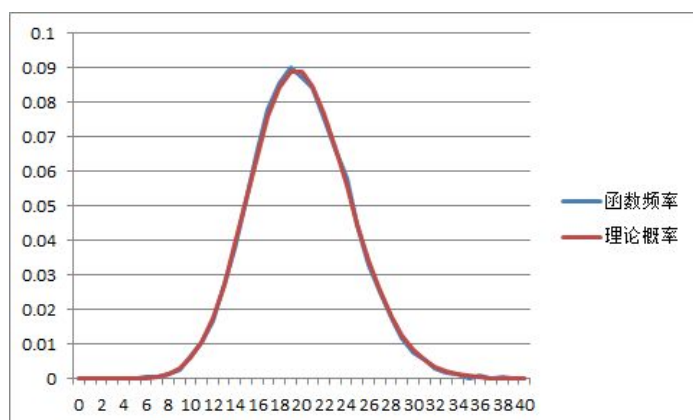
联系我们 © 2018 知乎

```
Function PoissonRand(lambda As Double) As Integer
    Dim rand As Single
    Dim k As Integer
    Dim p As Single
    Dim sump As Single

    Randomize
    rand = Rnd
    k = 0
    'p(0) 化简
    p = 1 / Exp(lambda)
    Do While rand > sump
        k = k + 1
        '从p(k) 转换到p(k+1)
        p = p * lambda / k
        sump = sump + p
    Loop
    PoissonRand = k
End Function
```

另外，由于浮点数精度、rnd伪随机性和试验次数较少等原因，随机数和理论值存在偏差。

我随机了10000组PoissonRand(20):



数据源:

k	函数频率	理论概率
0	0	2.06115E-09
1	0	4.12231E-08
2	0	4.12231E-07
3	0	2.7482E-06
4	0	1.3741E-05
5	0	5.49641E-05
6	0.0003	0.000183214
7	0.0003	0.000523468
8	0.0011	0.001308669
9	0.0026	0.002908153
10	0.0065	0.005816307
11	0.0104	0.010575103
12	0.0168	0.017625171
13	0.0275	0.027115648
14	0.0375	0.03873664
15	0.0519	0.051648854
16	0.0665	0.064561067
17	0.0782	0.075954196
18	0.0854	0.084393552
19	0.0903	0.088835317
20	0.0874	0.088835317
21	0.0844	0.084605064
22	0.0755	0.076913695
23	0.0665	0.066881474
24	0.0582	0.055734561
25	0.0447	0.044587649
26	0.033	0.034298192
27	0.0252	0.025406068
28	0.0177	0.018147191
29	0.0116	0.012515304
30	0.0075	0.008343536
31	0.0055	0.005382927
32	0.0031	0.003364329
33	0.0019	0.002038987
34	0.0011	0.001199404
35	0.0001	0.000685374
36	0.0007	0.000380763
37	0.0002	0.000205818
38	0.0003	0.000108325
39	0	5.55514E-05
40	0.0001	2.77757E-05

编辑于 2015-12-04

▲ 10 ▼


● 5 条评论

➦ 分享

★ 收藏

❤ 感谢

收起 ^



王诗

不以物喜，不以己悲

9 人赞同了该回答



4.2 Generating a Poisson Random Variable

The random variable X is Poisson with mean λ if

$$p_i = P\{X = i\} = e^{-\lambda} \frac{\lambda^i}{i!} \quad i = 0, 1, \dots$$

The key to using the inverse transform method to generate such a random variable is the following identity (proved in Section 2.8 of Chapter 2):

$$p_{i+1} = \frac{\lambda}{i+1} p_i, \quad i \geq 0 \quad (4.1)$$

Upon using the above recursion to compute the Poisson probabilities as they become needed, the inverse transform algorithm for generating a Poisson random variable with mean λ can be expressed as follows. (The quantity i refers to the value presently under consideration; $p = p_i$ is the probability that X equals i , and $F = F(i)$ is the probability that X is less than or equal to i .)

- STEP 1: Generate a random number U .
- STEP 2: $i = 0, p = e^{-\lambda}, F = p$.
- STEP 3: If $U < F$, set $X = i$ and stop.
- STEP 4: $p = \lambda p / (i + 1), F = F + p, i = i + 1$.
- STEP 5: Go to Step 3.

Ross, Sheldon M.

Simulation

发布于 2015-12-04

▲ 9 ▼ 1 条评论 分享 ★ 收藏 ♥ 感谢



fumin

魔鬼在细节

8 人赞同了该回答

Poisson distribution

Wikipedia 上有 Knuth 给出的算法, 其中 λ 是参数

```
algorithm poisson random number (Knuth):
  init:
    Let  $L \leftarrow e^{-\lambda}$ ,  $k \leftarrow 0$  and  $p \leftarrow 1$ .
  do:
     $k \leftarrow k + 1$ .
    Generate uniform random number  $u$  in  $[0,1]$  and let  $p \leftarrow p \times u$ .
  while  $p > L$ .
  return  $k - 1$ .
```

不要问我为什么, 凡是Knuth说的都对!

如果将 L 都取一个对数, 就变成了

```
algorithm poisson random number (Knuth):
  init:
    Let  $L \leftarrow -\lambda$ ,  $k \leftarrow 0$  and  $p \leftarrow 0$ .
  do:
     $k \leftarrow k + 1$ .
    Generate uniform random number  $u$  in  $[0,1]$  and let  $p \leftarrow p + \ln(u)$ .
  while  $p > L$ .
  return  $k - 1$ .
```





而 $\ln(u)$ 正是产生指数分布。

当然，我们喜欢看正数一些，对 L 加个负号，就和顶楼的方法一样啦。

```
algorithm poisson random number (Knuth):
  init:
    Let  $L \leftarrow \lambda$ ,  $k \leftarrow 0$  and  $p \leftarrow 0$ .
  do:
     $k \leftarrow k + 1$ .
    Generate uniform random number  $u$  in  $[0,1]$  and let  $p \leftarrow p - \ln(u)$ .
  while  $p < L$ .
  return  $k - 1$ .
```

编辑于 2015-12-04

▲ 8 ▼ ● 1 条评论 ↗ 分享 ★ 收藏 ♥ 感谢

收起 ^



Vichare Wang

至少还爱好编程

3 人赞同了该回答

一个比较理想的办法是构造一个函数 $Y=f(X)$ ，如果输入的随机变量 X 服从一个分布(例如 $[0,1]$ 的均匀分布)，那么输出的随机变量 Y 服从我们需要的分布。但是这个问题里泊松分布是离散的，数学上这个函数比较难写出来，所以可以用另一个办法。

考虑泊松分布的物理意义，是一个泊松过程在一个定长时间内的事件发生次数。

假如说你在一个公交车站，一辆公交车到下一辆公交车之间的间隔如果是参数为 λ 的指数分布，那一个单位时间内到达公交车的数量服从泊松分布。

余下你就模拟一下就好了。

发布于 2015-12-04

▲ 3 ▼ ● 添加评论 ↗ 分享 ★ 收藏 ♥ 感谢

