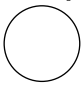
Three ways to generate poisson data

Poisson data distribution generated by Metapost (poissondeviate function by Anthony Phan + Metapost improvement's by Hans Hagen)



5 7 7 1 4 6 4 1 4 5 2 5 5 4 8 6 1 4 2 4 1 3 5 4 4 4 3 2 2 4 4 4 3 2 6 5 3 2 4 1 3 3 4 4
5 7 1 2 6 8 2 2 5 2 5 3 2 3 1 5 6 2 2 2 4 3 4 2 8 4 3 4 3 3 4 8 3 4 3 3 6 4 1 6 1 3 6 3
2 5 2 4 5 4 4 4 3 6 6 1
Poisson data distribution generated with Lua (local function)
3, 3, 7, 3, 4, 2, 4, 5, 2, 2, 0, 2, 11, 1, 5, 5, 2, 2, 5, 3, 5, 3, 6, 5, 5, 5, 3, 6, 0, 3, 4, 9,
2, 3, 1, 1, 7, 2, 5, 5, 5, 6, 3, 2, 3, 4, 3, 2, 2, 7, 1, 2, 4, 9, 7, 3, 3, 6, 4, 3, 3, 5, 1, 6, 5,
2, 4, 1, 6, 4, 4, 1, 4, 1, 3, 6, 5, 1, 5, 6, 3, 7, 6, 2, 2, 3, 8, 7, 5, 1, 8, 3, 4, 4, 4, 3, 9, 2,
4, 2,
Poisson data distribution generated with GSL library**
4, 8, 3, 2, 6, 3, 4, 2, 2, 2, 2, 3, 4, 3, 4, 4, 3, 6, 4, 4, 1, 2, 7, 1, 2, 2, 6, 5, 7, 3, 3, 4, 4,
6, 5, 5, 1, 1, 1, 8, 2, 4, 6, 1, 3, 3, 4, 4, 3, 12, 4, 5, 5, 3, 2, 6, 3, 5, 2, 5, 1, 6, 3, 0, 9,
2, 4, 4, 3, 5, 2, 1, 8, 6, 5, 4, 6, 3, 2, 7, 2, 2, 6, 4, 5, 6, 6, 5, 7, 8, 6, 5, 3, 2, 1, 7, 5, 5,
4, 8,
**It's need to build the randist.so library (GSL+Swig : after suggestion of Alan Braslou). In my MacOSX 10.8.2 I do:

- randist.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>


void rpoisson(int* arr, int len,int lambda)
{
  const gsl_rng_type * T;
  gsl_rng * r;

  int i;
/*  create a generator chosen by the environment variable GSL_RNG_TYPE */

  gsl_rng_env_setup();
  T = gsl_rng_default;
  r = gsl_rng_alloc (T);

  /* print n random variates chosen from
     the poisson distribution with mean
     parameter mu */

  for (i = 0; i < len; i++)
    {
      unsigned int k = gsl_ran_poisson (r, lambda);
      arr[i]=k;
/*       printf (" %u", k);  */
    }

 /* printf ("\n"); return 0;*/
  gsl_rng_free (r);
}
```

- randist.i (interface for Swig)

```
/* File : randist.i */
%module randist
 %{
 /* Put header files here or function declarations like below */
#include <stdio.h>
#include <stdlib.h>
#include <gsl/gsl_rng.h>
#include <gsl/gsl_randist.h>

 %}

// this way uses the SWIG-Lua typemaps to do the conversion for us
// the %apply command states to apply this wherever the argument signature matches
%include <typemaps.i>
%apply (int *INOUT,int,int) {(int* arr,int len,int lambda)};
extern void rpoisson(int* arr, int len, int lambda);


%include <carrays.i>    // array helpers
// this declares a batch of function for manipulating C integer arrays
%array_functions(int,int)
```

```
%inline %{
extern void rpoisson(int* arr, int len, int lambda);
%}
```

- Build the dynamic library (randist.so)

```
swig -lua randist.i
gcc -I/usr/include/lua -c randist_wrap.c -o randist_wrap.o
gcc -c randist.c -o randist.o
gcc -fpic -bundle -undefined dynamic_lookup -llua -lgsl -lgslcblas -lm
-Wall -I/usr/local/include -L/usr/local/lib randist_wrap.o randist.o -o randist.so
```

- sudo cp randist.so /usr/local/lib/lua/5.2
- change LUAINPUTS