Números pseudoaleatorios

Física de los Sistemas Complejos 4º Física, UGR

Febrero 2018

Número aleatorio

Resultado de una variable al azar proveniente de una función de distribución.

Distribución uniforme
$$\rightarrow U(a,b)$$

Distribución normal
$$o N(\mu, \sigma)$$

- ¿Por qué son necesarios?
 - Simulaciones
 - Análisis numérico
 - Criptografía
- Generación de número (pseudo)aleatorios (secuencias sin significado físico). Un ordenador es una máquina determinista.

Condiciones

- La sucesión de valores debe resultar en una variable U(0,1)
- ► La ciencia debe ser reproducible!! Los resultados deben ser reproducibles desde una semilla inicial
- ► La sucesión de valores debe tener un ciclo tan largo como sea posible

El generador debe ser rápido y ocupar poca memoria



Generador congruencial

Ejemplo

Generador de enteros dado por

$$X_{n+1} = aX_n + c \pmod{m}$$

donde a y c son constantes, que en este caso consideraremos a=5, c=1 y m=16, con semilla $X_0=1$.

Ejercicio

Genera una serie de números aleatorios congruenciales dada por

$$X_{n+1} = mod(cX_n + a_0, N_{max})$$

con parámetros, c=65539, $a_0=0$, $x_0=1$ y $N_{max}=2^{31}-1$. Localiza un error al generarlos. $(\frac{X_n}{N_{max}}\Rightarrow U(0,1))$.

Distribuciones de probabilidad

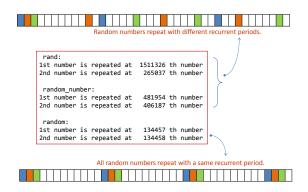
- Se pueden generar secuencias de enteros y distribuciones uniformes.
- ¿Cómo se genera una distribución de probabilidad? Distribución acumulada

$$F(x)=u=U(0,1)$$

Ejemplo: Distribución gaussiana

Generación de aleatorios Fortran/C

Funciones rand, random_number, random



Gnu Scientific Library



ABOUT GNU PHILOSOPHY LICENSES EDUCATION SOFTWARE

GSL - GNU Scientific Library

https://www.gnu.org/software/gsl/

FGSL: A Fortran interface to the GNU Scientific Library

This web page provides the source code for a portable, object-based Fortran interface to the GNU of numerical routines for scientific computing. The package is licensed under the GPL. Please be at package is at your own risk.

https://www.lrz.de/services/software/mathematik/gsl/ fortran/

Generación de aleatorios Fortran/C

```
#include <stdio.h>
#include <asl/asl rna.h>
main (void)
  int i, n = 10;
  const gsl rng type * T;
  gsl_rng * r;
  qsl rnq env setup(): //Inicializacion de las qsl
  qsl rnq default seed=0: //Permite cambiar la semilla por defecto
  T = qsl rnq mt19937; //qsl rnq default; // Generador a utilizar
  r = qsl rnq alloc (T); //Comprueba si hay suficiente memoria para emplear el generador
  gsl rng set(r,35); // Otra forma de cambiar la semilla, pero despues de llamar al generador. Elegir este caso o el anterior
  for (i = 0: i < n: i++)
      double u = asl rng uniform (r);
      unsigned long int h = gsl_rng_uniform_int (r,100);
      printf ("Uniforme (0-1): %.5f
                                               Entero entre 1 y 100: %lu \n", u, h);
  asl rng free (r): //Libera toda la memoria que hava empleado el generador
  return 0:
```

```
1 program random
2 use fgsl
3
4 implicit none
5 integer*8 :: seed
7 type(fgsl_rng) :: r
8 type(fgsl_rng_type) :: ini_fgsl
9
10 !Inicializo aleatorios con las gsl.
11 seed=21
12 ini fgsl = fgsl_rng env setup()
13 fgsl rng default seed=seed
14 ini fgsl = fgsl_rng default
15 r = fgsl_rng_alloc(ini_fgsl)
16
17 write(*,*) fgsl_rng_uniform(r)
18
19 stop
20 end
```

Caminante aleatorio

Random walk Considerad el siguiente proceso estocástico

$$x(t+1) = x(t) + \eta(t)$$

donde $\eta\left(t\right)$ es un ruido blanco gaussiano con $\langle\eta_{i}(t)\eta_{j}(t')\rangle=\delta\left(t-t'\right)\delta_{ij}$. Generando muchos caminantes, representad la posición media del caminante aleatorio y la evolución de su varianza en función del tiempo.