MAC 110 - 2013

GERADOR DE NÚMEROS PSEUDO-ALEATÓRIOS

(Relatório)

Nome: Lucas Sung Jun Hong

➢ Índice

1) Gerador e semente

Foi utilizada a operação: x=frac(128.7361+mod|534.187*sen(x)-sen^2(x)|), que será chamada de Gerador G, em que a semente aplicada para o gerador foi: 0.8124329. Como comparação, foram escolhidos 3 outros geradores:

Gerador A) x=frac(714.723*mod|127.387+sen(x)|);

Gerador B) x=frac(95.3313+mod|12.9*sen(x)|);

Gerador C) x=frac(7.2*mod(1+seno(x));

```
double      generator    (double ngn)
{
    double  ngn1;
    ngn1=frac(128.7361+modulo(534.187*seno_x(ngn)-pot(seno_x(ngn),2)));
    return  ngn1;
}

int main ()
{
    double  a,seed;
    int     n,cont;

    seed=0.8124329;

    printf("Indique n > 0 para gerar n numeros aleatorios: ");
    scanf("%d",&n);

    for (cont=0;cont<n;cont++)
    {
        a=generator(seed);
        printf("%lf\n",a);
        seed=a;
    }
    return 0;
}
```

```
double      generator    (double ngn)
{
    double  ngn1;
    ngn1=frac(714.723*modulo(127.387+seno_x(ngn)));
    return  ngn1;
}
```

```
double      generator    (double ngn)
{
    double  ngn1;
    ngn1=frac(95.3313+modulo(12.9*seno_x(ngn)));
    return  ngn1;
}
```

```
double      generator    (double ngn)
{
    double  ngn1;
    ngn1=frac(7.2*modulo(1+seno_x(ngn)));
    return  ngn1;
}
```

2) Testes: verificando qual o melhor gerador

i) Teste baseado no jogo “Bau cua ca cop”:

Cada face, numerada de 1 a 6, deve possuir aproximadamente 16,666% de aparecer em cada jogada. Um gerador que apresenta valores próximos ao esperado significa um considerável bom gerador.

Cada teste foram 100.000 números gerados e foi calculado o número total de vezes que cada face apareceu dividido pelo número total de testes realizados, neste caso, 100.000.

i.a) Gerador A: *x=frac(714.723*mod|127.387+sen(x)|);*

face=1 tem 16.741%
face=2 tem 16.767%
face=3 tem 16.662%
face=4 tem 16.714%
face=5 tem 16.623%
face=6 tem 16.493%

i.b) Gerador B: *x=frac(95.3313+mod|12.9*sen(x)|);*

face=1 tem 17.246%
face=2 tem 15.285%
face=3 tem 16.613%
face=4 tem 16.988%
face=5 tem 16.844%
face=6 tem 17.024%

i.c) Gerador C: *x=frac(7.2*mod(1+seno(x));*

face=1 tem 17.054%
face=2 tem 17.434%
face=3 tem 15.631%
face=4 tem 16.065%
face=5 tem 16.826%
face=6 tem 16.990%

i.d) Gerador G: *x=frac(128.7361+mod|534.187*sen(x)-sen*^{2}*(x)|)*

face=1 tem 16.682%
face=2 tem 16.668%
face=3 tem 16.691%
face=4 tem 16.521%
face=5 tem 16.788%
face=6 tem 16.650%

ii-1) Teste do Montecarlo:

Para a função f(x)= x^4, sua aréa vale 0.2. Considerando tal valor, os números gerados que se posicionarem dentro da sua função dividindo pelo total de pontos gerados devem possuir o valor aproximado de $\int_0^1 x^4\,dx = \frac{1}{5}$.

Foram gerados 500 e 50.000 pontos para cada gerador.

ii.a) Gerador A: *x=frac(714.723*mod|127.387+sen(x)|);*

500 pontos: aproximacao = 0.186000;

50.000 pontos: aproximacao = 0.200920;

ii.b) Gerador B: *x=frac(95.3313+mod|12.9*sen(x)|);*

500 pontos: aproximacao = 0.202000;

50.000 pontos: aproximacao = 0.228320;

ii.c) Gerador C: *x=frac(7.2*mod(1+seno(x));*

500 pontos: aproximacao = 0.200000;

50.000 pontos: aproximacao = 0.184320;

ii.d) Gerador G: *x=frac(128.7361+mod|534.187*sen(x)-sen²(x)|)*

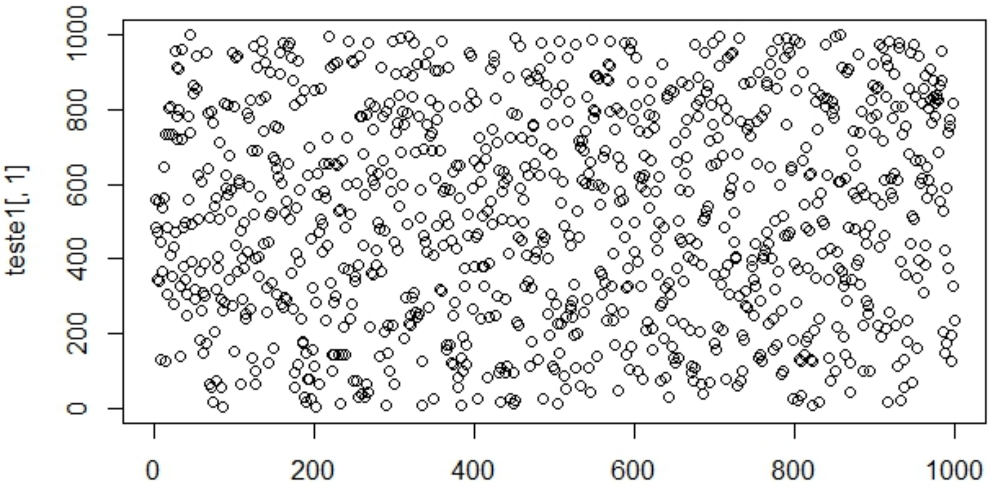
500 pontos: aproximacao = 0.214000;

50.000 pontos: aproximacao = 0.205200;

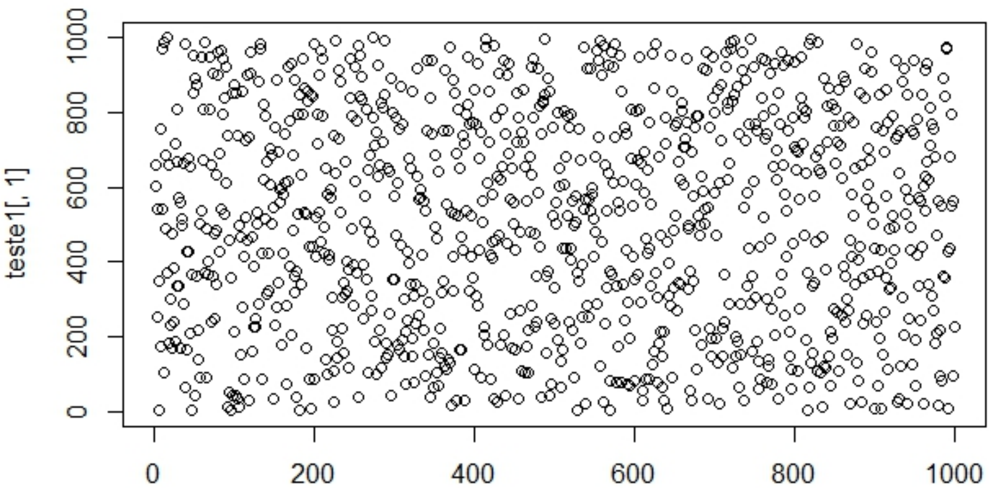
ii-2) Usando o software R para o teste Montecarlo

Com 1000 números gerados, foram formados pontos para verificar se tais pontos estavam dentro da função f(x)= x^4 , processo para o teste Montecarlo. Usando o software *R commander*, a imagem do gráfico foi desenhada para verificar a distribuição dos pontos visualmente.

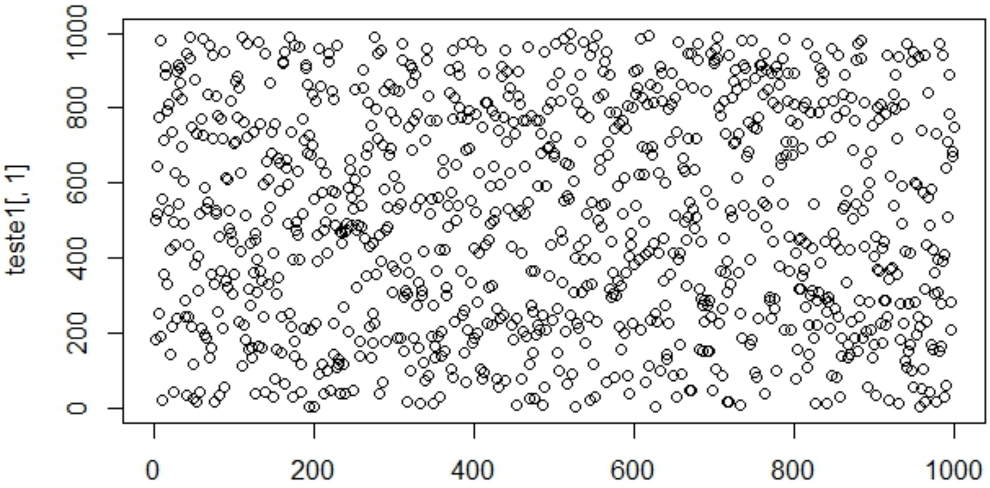
a) Gerador A:



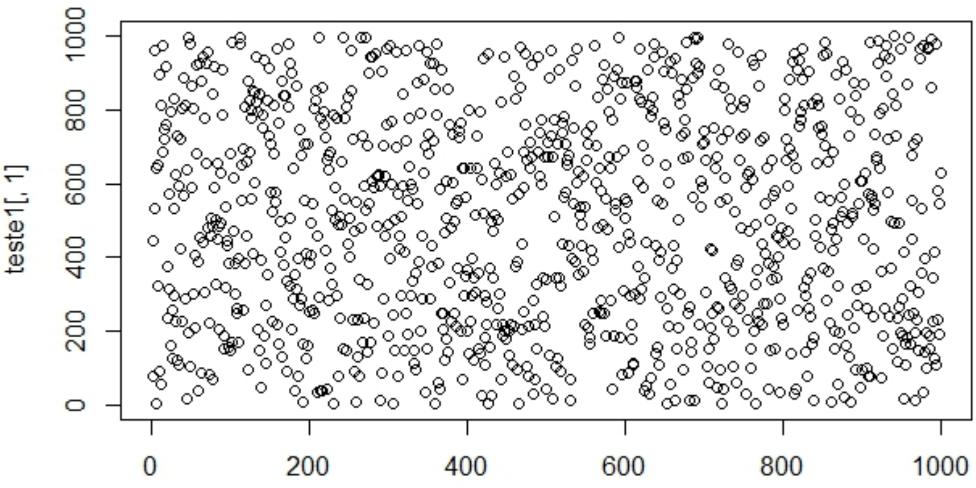
b) Gerador B:



c) Gerador C:



d) Gerador G:



3) Conclusão

O gerador A obteve bons resultados no teste baseado no jogo, no entanto, o mesmo não pode ser dito quando foi feito o teste Montecarlo. No gerador B acontece o inverso. Já o gerador C não teve bons resultados em ambos os teste. Ou seja, não foram considerados bons geradores pois seus valores não possuíram valores próximos aos esperados.

Finalmente, o gerador G obteve valores interessantes. Para 100.000 números gerados, no teste baseado no jogo, houve uma variação de 16.521% a 16.788%, já no teste Montecarlo, a área aproximada calculada foi 0.205200, um resultado próximo ao valor esperado: 0.2.

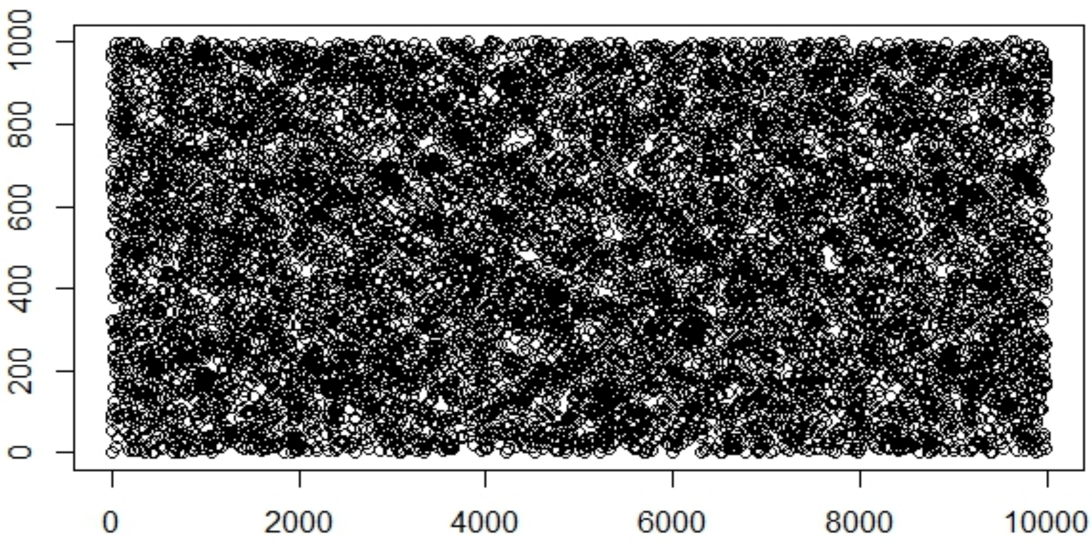


Figura 5: 5.000 pontos distribuídos com o gerador G.

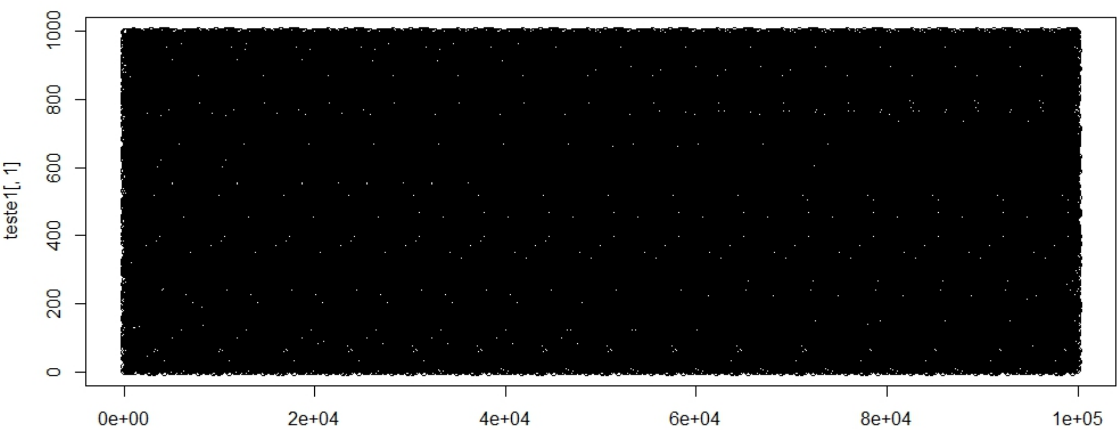


Figura 4: 500.000 pontos distribuídos com o gerador G.

Bibliografia:

http://www.rcommander.com/

http://www.codeblocks.org/

http://www.r-project.org/

https://developer.apple.com/xcode/

http://www.ime.usp.br/~cef/mac110-13/ep2.pdf

http://www.ime.usp.br/~pf/algoritmos/aulas/random.html

http://en.wikipedia.org/wiki/Random_number_generation

http://physinfo.ulb.ac.be/cit_courseware/pcprogram/c_008.htm

http://pt.wikipedia.org/wiki/Parte_inteira

http://es.wikipedia.org/wiki/Funci%C3%B3n_piso#Funci.C3.B3n_piso.2Fsuelo

http://www.math.utah.edu/~pa/Random/Random.html

http://en.wikipedia.org/wiki/Pseudo-random_number_generator

http://en.wikipedia.org/wiki/Mersenne_twister

http://en.wikipedia.org/wiki/Random_seed

http://www.wolframalpha.com/input/?i=integral+of+%28x%CB%864%29+from+0+to+1