Lucas Lui Motta
R.A: 192701

Segundo Trabalho: Processos Estocásticos
Teste de Aleatoriedade de Sequências

Campinas
09 de outubro de 2017

1 Teste de Aleatoriedade de Sequências

1.1 Introdução

O Professor pediu aos alunos que gerassem e enviam-se a ele uma sequência aleatória de 0's e 1's com 100 dígitos, porém, sem utilizar nenhum algoritmo ou *software*. Dado a quantidade de alunos na disciplina, foram criadas 35 sequências. Em seguida, o professor gerou através do Matlab mais 35 sequências aleatórias, formando um total de 70 sequências geradas. As sequências foram então ordenadas e salvas em um arquivo como sendo as primeiras dos alunos e as restantes pelo professor. A problemática é, dado esse arquivo de 70 sequências pseudo aleatórias, saber quais delas foram geradas por alunos e quais pelo Matlab e, para isso, o professor nos indicou dois métodos diferentes listados abaixo.

- Método baseado no Número de Corridas.
- Método baseado no Agrupamento de Bits.

Esses métodos são utilizados para decidir se uma sequência é aleatória ou não. Em geral, as sequências que foram feitas manualmente pelos alunos não são verdadeiramente aleatórias devido a tendência a "psique"humana a padrões. Enquanto que, as feitas pelo *software* através de algum algoritmo de aleatoriedade, chegam o mais perto da verdadeira aleatoriedade. O trabalho então é implementar e comparar ambos os métodos de teste de aleatoriedade sabendo a priori que as 35 primeiras sequências no arquivo foram feitas por alunos e as 35 últimas pelo professor e *software* Matlab.

1.2 Método baseado no Número de Corridas (Wald-Wolfowitz)

Primeiramente, para entender esse método é preciso saber oque seria o número de corridas de uma sequência. Dado uma sequência qualquer, é considerado como sendo uma corrida cada trecho com 1 ou mais dígitos de mesmo valor, por exemplo, seja a sequência **1**00**11**00**11**0 analisada, o número de corridas é **6** devido ter 3 corridas de 1's e três de 0's.

No mundo de probabilidade a aparição de sequências de mesmo dígito interrompidas pela aparição de outro valor é traduzida como uma distribuição geométrica e a combinação de vários eventos desse tipo nos apresenta uma distribuição binomial a qual é a base para esse método [1]. No entanto, essa análise mais profunda do método baseado nas distribuições não será realizado nesse trabalho. Contudo, oque pode ser dito do método agora é que ele é baseado em um modelo de inferência estatística, ou seja, dado um conjunto de amostras do experimento (número de corrida) é realizado proposições sobre alguma hipótese do universo do experimento, nesse caso:

- ***Ho*** - Sequência foi produzida de maneira aleatória.
- ***Ha*** - Sequência não foi produzida de maneira aleatória.

Nesse modelo, o parâmetro teste estatístico **Z** (utilizado em inferência estatística) é calculado a partir do número de corridas **R** da sequência analisada, do valor esperado de corridas $\overline{R}$ e desvio padrão σ_R conforme equação 1.1 (qual pode ser interpretada como uma normalização de **R**) abaixo.

$$Z = \frac{R - \overline{R}}{\sigma_R} \tag{1.1}$$

O valor esperado e o desvio padrão do número de corridas são calculados através das equações 1.2 e 1.3 (quais foram retiradas da referência[2] e são baseados na real natureza do evento número de corrida das sequências) a seguir.

$$\overline{R} = \frac{2n_1n_2}{n_1 + n_2} + 1 \tag{1.2}$$

$$\sigma_R = \sqrt{\frac{2n_1n_2(2n_1n_2 - n_1 - n_2)}{(n_1 + n_2)^2(n_1 + n_2 - 1)}} \tag{1.3}$$

As constantes $\mathbf{n}_1$ e $\mathbf{n}_2$ são as quantidades de 0's e 1's da sequência analisada. Para valores altos dessas constantes, ou seja, sequências com um conjunto grande de bits, o teste estatístico **Z** tem a probabilidade aproximada a uma distribuição normal qual pode ser calculada com uma tabela **Z**.

Nesse tipo de teste é necessário assumir um nível de significância α que nesse trabalho é de **10%** e que implica em um intervalo de credibilidade com **90%** de probabilidade de não conter a hipótese nula ***Ho***. Sendo assim, se o valor associado a essa probabilidade da tabela da distribuição normal (no caso, 1,645) for menor que o teste estatístico **Z** da sequência atual analisada, a mesma é enfim dita ser não-aleatória (aluno). A implementação desse método pode ser vista no Listing 1.1.

Listing 1.1 – Algoritmo do método baseado no Número de Corridas.

```
% Abrindo arquivo de dados com a lista de sequencias geradas.
file = load('BitSequences.mat');
list_seq = file.bits;
% Nivel de significancia para o teste de hipotese.
alfa = 0.1;
% Encontrando o numero de sequencias da lista de sequencias.
num_seq = length(list_seq(:,1));
cont_alunos = 0; % Contador de seq. feitas por alunos.
cont_matlab = 0; % Contador de seq. feitas pelo Matlab.
% Inic. vetor de resultados do teste de cada seq.
```

```
vect_result = zeros(1,tam_seq);
% Loop itera todas as seq. realizando o teste de aleatoriedade.
for i = 1:tam_seq
    % Calculando o numero de digitos da sequencia atual.
    tam_seq = length(list_seq(i,:));
    % Encontrando a quantidade de 0's e 1's na sequencia.
    n = sum(list_seq(i,:)); % Quantidade de 1's.
    m = (tam_seq - n);      % Quantidade de 0's.
    % Valor esperado, var. e desvio padrao do num. de corridas.
    R_med = (2*n*m/(n+m))+1;
    VarR = (2*n*m*((2*n*m)-n-m))/((n+m)^2*(n+m-1));
    SigmaR = sqrt(VarR);
    % Contando o numero de corridas da sequencia atual.
    R = 1;
    for k=2:length(list_seq(i,:))
        if list_seq(i,k)~=list_seq(i,k-1)
            R = R + 1;
        end
    end
    % Calculando fator de teste estatistico:
    Z = (R-R_med)/SigmaR;
    % Calculando fator de aceitacao do teste de hipotese.
    Z_crit = norminv(1-alfa/2);
    % Tomando decisao baseado nos valores de teste e margem.
    if abs(Z) > Z_crit
        cont_alunos = cont_alunos + 1;
        vect_result(i) = 0;
    else
        cont_matlab = cont_matlab + 1;
        vect_result(i) = 1;
    end
end
```

O código varreu todas as 70 sequências do arquivo dado e os resultados encontrados dessa análise podem ser vistas na Figura 1.

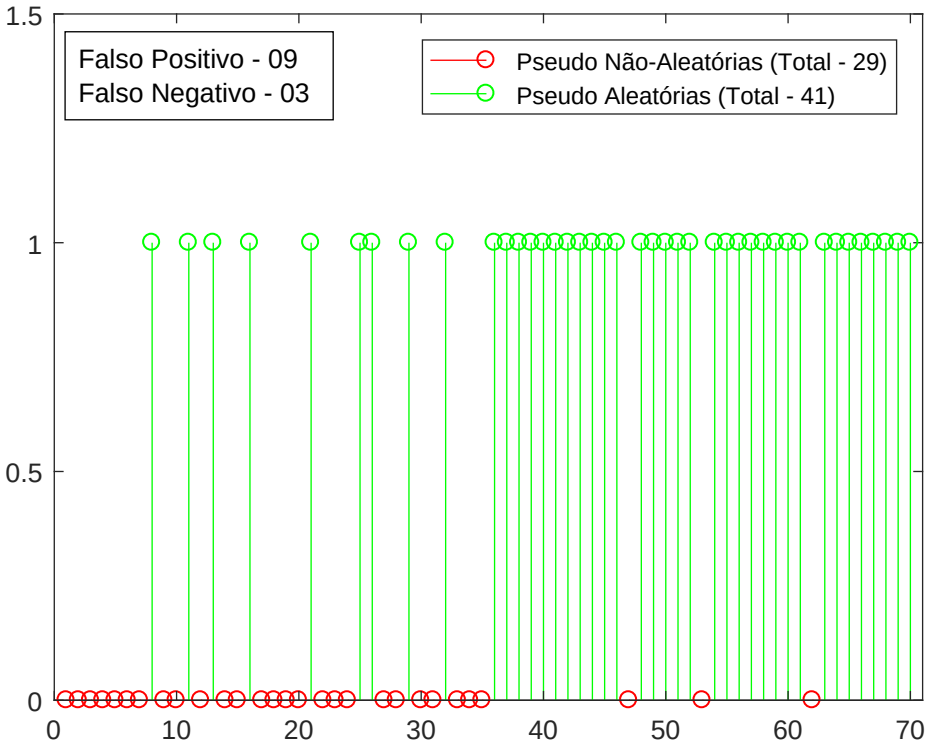

Figura 1 – Resultados encontrados no método baseado no Número de Corridas.

1.3 Método baseado no Agrupamento de Bits

O método baseado no agrupamento de bits, como o próprio nome indica, consiste em separar a sequência analisada em grupos de 2, 3, ou mais bits, dependendo do tamanho da sequência analisada. Conforme a escolha do número de bits do agrupamento, é formado o número de combinações possíveis de serem encontradas na sequência. Por fim, cada possível combinação do agrupamento deve aparecer o mesmo número de vezes na sequência analisada para que o teste seja dado positivo para aleatória.

No entanto, esse método é muito sensível ao tamanho da sequência analisada e o número de bits do agrupamento. Basicamente, para uma sequência pequena não devemos fazer grupos grandes de bits pois, existirão muitas combinações de valores e consequentemente não haverá igual incidência de cada. Nesse trabalho, para analisarmos a aleatoriedade das sequências dadas com 100 bits consideramos o agrupamento em 2 em 2 bits devido os melhores resultados.

Dado esse agrupamento de 2, aparecerão $100/2 = \mathbf{50}$ pares de bits onde os possíveis valores desses pares são: '**00**', '**01**', '**10**' e '**11**' e cada possível valor deverá aparecer a mesma quantidade de vezes, ou seja, $50/4 = \mathbf{12{,}5}$ vezes. No entanto, além de não ser possível aparecer 1/2 vez de qualquer valor para completar 12,5, abrimos uma faixa de valores de vezes em torno desse ideal ao qual cada combinação pode pertencer e ainda ser considerado a sequência como aleatória. Essas considerações podem ser vistas na Figura 2.

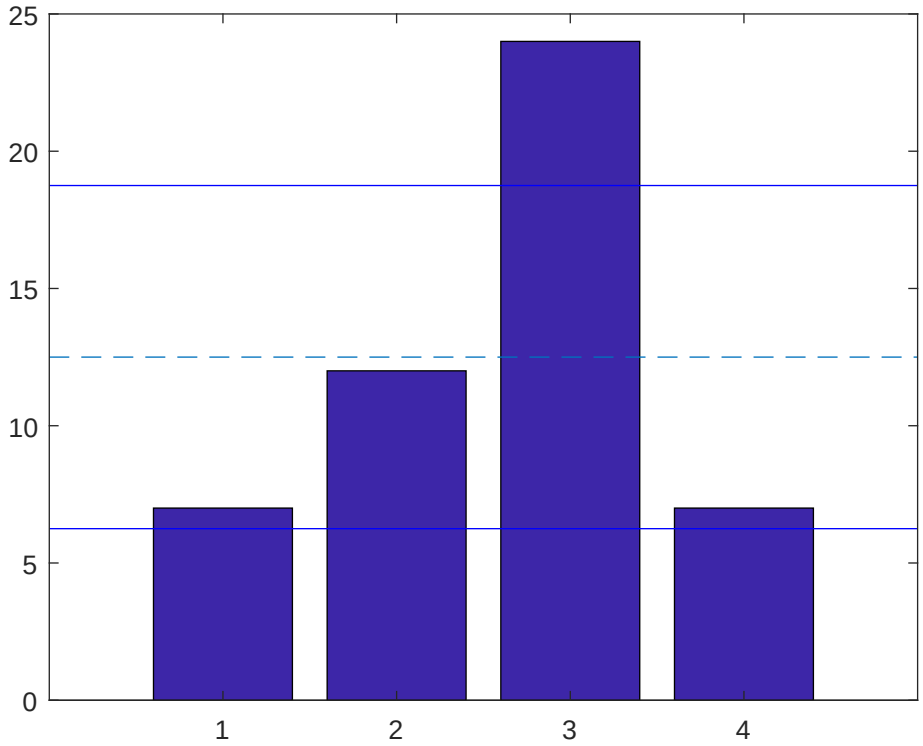

Figura 2 – Critério utilizado para seleção das sequências aleatórias.

A faixa foi construída com uma margem de **50%** em torno do valor ideal, tal valor de margem foi escolhido empiricamente devido ao bom resultado. Enfim, o método está implementado no código mostrado no Listing 1.2

Listing 1.2 – Algoritmo do método baseado no Agrupamento de Bits.

```
n_bits = 2; % Definindo o numero de bits do agrupamento.
cont_alunos_met2 = 0; % Contador de seq. feitas por alunos.
cont_matlab_met2 = 0; % Contador de seq. feitas pelo Matlab.
% Inic. vetor que armazena o resultado do teste de cada seq.
vect_result_met2 = zeros(1,tam_seq);
% Calculando o numero de digitos das sequencias.
tam_seq = length(list_seq(1,:));
% Inic. matriz que armazena as contagem das comb. possiveis.
mat_cont = zeros(tam_seq, 2^n_bits);
% Sabendo que todas comb. devem ter mesma quant. de aparicoes.
cont_ideal = (100/n_bits)/2^n_bits;
% Margem sobre o ideal para o teste de hipotese.
margem = 0.5*cont_ideal;
% Loop itera todas as seq. realizando o teste de aleatoriedade.
for i = 1:num_seq
    % Varre a seq. contando o num. de aparicoes de cada comb.
    for j = 1:n_bits:tam_seq
```

```
        capture = bi2de(list_seq(i,j:j+n_bits-1), 'left-msb');
        mat_cont(i,capture+1) = mat_cont(i,capture+1)+1;
    end
    % Contabiliza o num. de comb. dentro da faixa de aceitacao.
    num_comb_dentro = 0;
    for k = 1:length(mat_cont(i,:))
        if (mat_cont(i,k)<cont_ideal+margem)&&(mat_cont(i,k)>
            cont_ideal-margem)
             num_comb_dentro = num_comb_dentro + 1;
        end
    end
    % Decisao conforme o num. de comb. dentro da faixa.
    if num_comb_dentro <= 3
        cont_alunos_met2 = cont_alunos_met2 + 1;
        vect_result_met2(i) = 0;
    else
        cont_matlab_met2 = cont_matlab_met2 + 1;
        vect_result_met2(i) = 1;
    end
end
```

Após a execução do algoritmo, o método implicou nos resultados vistos na Figura 3.

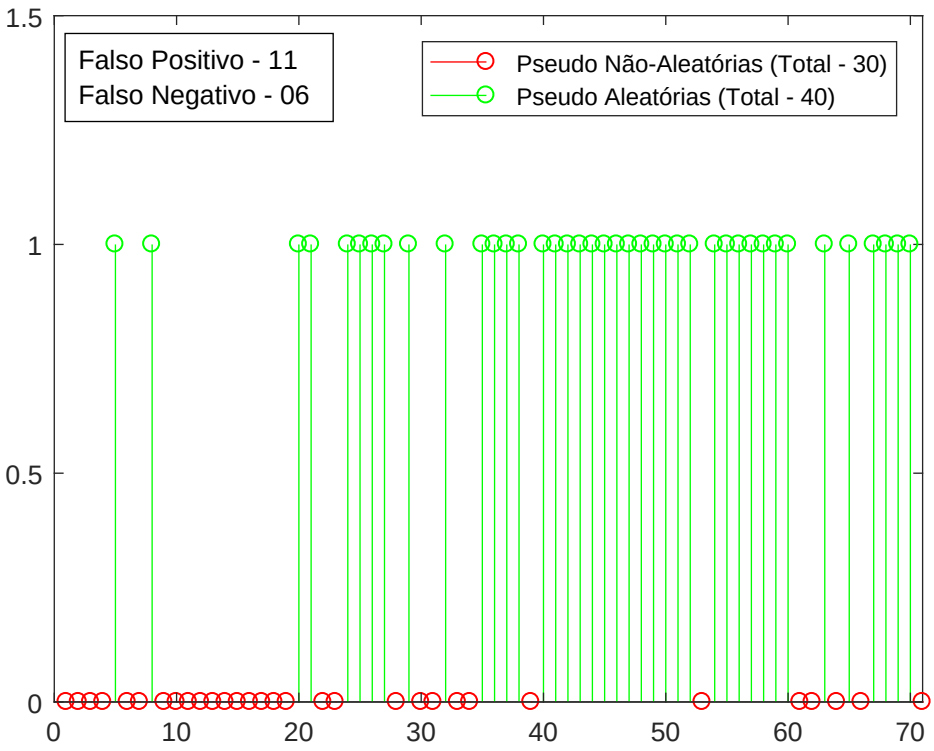

Figura 3 – Resultados encontrados no método baseado no Agrupamento de Bits.

1.4 Conclusão

Visto os resultados encontrados nos dois métodos podemos primeiramente afirmar que nenhum deles conseguiu distinguir completamente as 35 sequências feitas por alunos e as 35 pelo professor/*software*. Podemos pensar que esse fato se deu devido a boa aleatoriedade que alguns alunos geraram ou então, devido ao tamanho da sequência não ser suficientemente grande para que o *sofware* prova-se sua aleatoriedade nos critérios apresentados pelos métodos. Além disso, depois de várias simulação, foi encontrado um trade-off nos parâmetros dos métodos, quando diminuímos/aumentamos o nível de significância ou faixa de critério (respectivamente, primeiro e segundo método) temos a aparição de falsos negativos/positivos o qual também não queremos. Por fim, esses são só dois métodos de avaliar aleatoriedade de sequências e acredito que existem diversos outros e que a aproximação ao verdadeiro pode estar na interseção de vários desses métodos.

Referências

[1] Rick Wicklin, How to tell whether a sequence of heads and tails is random, http://blogs.sas.com/content/iml/2013/10/09/how-to-tell-if-a-sequence-is-random.html, 08 10 2017.

[2] NIST/SEMATECH e-Handbook of Statistical Methods, Runs Test for Detecting Non-randomness, http://www.itl.nist.gov/div898/handbook/eda/section3/eda35d.htm, 08 10 2017.

[3] Dan Meyer, Can You Recognize Random?, http://blog.mrmeyer.com/2011/can-you-recognize-random/, 08 10 2017.