Geração de Números Pseudo-Aleatórios

Carlos M. Fonseca
Departamento de Engenharia Informática
Universidade de Coimbra

MEI 2020/2021

Bibliografia

- Ronald T. Kneusel, *Random Numbers and Computers*, Springer, 2018.
- Pierre L'Ecuyer, "Efficient and portable combined random number generators," *Communications of the ACM*, vol. 31, no. 6, June 1988. https://doi.org/10.1145/62959.62969

Aleatoriedade

- Muitos sistemas exibem comportamentos imprevisíveis
  - Por exemplo, o tempo de serviço numa cantina
- Simular esses sistemas obriga à modelação desses comportamentos
  - Por exemplo, para prever o tempo de espera numa fila
- Modelos probabilísticos (ou estocásticos)

Sequências Aleatórias

- O que é uma experiência aleatória?
  - Experiência cujo resultado é imprevisível
- O que é uma sequência aleatória?
  - “Sequência de números numa dada gama de valores tal que não é possível prever o próximo valor com base nos valores que o precedem”
  - Normalmente considera-se uma distribuição *uniforme*.

Sequências Aleatórias

- Geração de sequências aleatórias
  - Atirar uma moeda ao ar
  - Lançar um dado (justo)
  - Decaimento radioativo
  - “Estática” apanhada por um recetor de rádio ou televisão não sintonizado
  - Outros?

Sequências Aleatórias

- Moeda ao ar
  1. T T T T H T H T T T H H T H H T H T T T T H H H T H
  2. H T T T H H H T T H T T T H H H T T H T H H T H H T
- 1ª sequência – 11 caras (H) e 15 coroas (T) (p = 0.2786)
- 2ª sequência – 13 caras (H) e 13 coroas (T) (p = 1.0)

Sequências Aleatórias

- Gerar números de 0 a 7
  - T T T T H T H T T T H H T H H T H T T T T H H H T H
  - 0 0 0 0 1 0 1 0 0 0 1 1 0 1 1 0 1 0 0 0 0 1 1 1 0 1
  - 000 010 100 011 011 010 000 111 01
  - 0 2 4 3 3 2 0 7 ...

Sequências Aleatórias

- A aleatoriedade dos humanos não é em geral muito boa…
- Experiência
  - Pedir a cada estudante da turma para escolher um número à sorte entre 0 e 255
  - O que deveria acontecer?
  - O que se pode esperar que aconteça?

Sequências Pseudo-Aleatórias

- O que é uma sequência pseudo-aleatória?
  - “Sequência de números gerada deterministicamente que é indistinguível de uma verdadeira sequência de números aleatórios”
- Indistinguível?
  - Qualquer procedimento aritmético *nunca* pode gerar números aleatórios!

Sequências Pseudo-Aleatórias

- Gerador de números pseudo-aleatórios
  - Geração de bytes pseudo-aleatórios (p.ex.)
  - Agrupando esses bytes, pode-se obter inteiros de qualquer tamanho, ou mesmo números de vírgula flutuante (p.ex., usando apenas a mantissa)
  - Dado um valor $r \in [0,1[$, um inteiro em $[a,b[$ pode ser facilmente obtido como $i = \mathrm{floor}(r \cdot (b-a)) + a$

Sequências Pseudo-Aleatórias

- O método “Middle Square”
  - Tomar um inteiro de 32 bits
  - Elevá-lo ao quadrado e guardá-lo como um inteiro de 64 bits
  - Retornar os 32 bits do meio…
  - Usar esse resultado como valor inicial para gerar o próximo número

Método “Middle Square”

```
def middle_square(seed=0xfedcb2ed):
    while True:
        seed *= seed
        seed = (seed >> 16) & 0xffffffff
        yield seed # 32 bits

rng = middle_square()

x = next(rng)
```

Método “Middle Square”

- Foi uma das primeiras tentativas de gerar uma sequência de números pseudo-aleatórios (von Neumann, 1951)
- Ilustra bem a ideia por trás destes geradores
- Não é um bom método
  - Uma vez encontrado um zero, a sequência passa a ser zero daí em diante
- A ideia foi recentemente retomada com modificações simples mas importantes

Método “Middle Weyl”

- Modificação do método Middle Square
- Simples e muito rápido
  - A implementação em C compila para apenas 4 instruções máquina
- Período maior ou igual a 2^{64} (bastante respeitável)
- Distribuição uniforme (discreta)
- Escolha da *seed* permite gerar sequências diferentes (mesmo em paralelo!)
- Excelentes resultados em testes de aleatoriedade

Método “Middle Weyl”

```
# Ver também https://arxiv.org/abs/1704.00358

def middle_weyl(seed=0xb5ad4eceda1ce2a9):
    x = w = 0
    while True:
        x = x * x
        w = w + seed % (2**64) # 64 bits
        x = (x + w) % (2**64) # 64 bits
        x = ((x >> 32) | (x << 32)) % (2**64)
        yield x % (2**32) # yield only 32 bits

rng = middle_weyl()

x = next(rng)
```

Geração de números pseudo-aleatórios

- Dado um valor r ∈ [0,1[, um inteiro em [a,b[pode ser facilmente obtido como
  $i = \text{floor}(r \cdot (b - a)) + a$
- Do mesmo modo, dado um valor inteiro P ∈ [0,m[, um valor real w ∈ [0,1[pode ser facilmente obtido como $w = P \,/\, m$

Gerador Congruencial Linear

- Gerador congruencial linear (LCG) misto

  $$x_{n+1} = (a\, x_n + c) \bmod m$$

  onde a é o multiplicador ($0 < a < m$), c é o incremento ($0 \leq c < m$) e m é o módulo.
- Qualidade da sequência depende da escolha dos parâmetros a, c e m.
  - Período necessariamente menor ou igual a m

Gerador Congruencial Linear

```
# Exemplo simples

def lcg_10(seed=7):
    m = 10
    a = 3
    c = 9
    while True:
        seed = (a * seed + c) % m
        yield seed
```

Gerador Congruencial Linear

- Diferentes escolhas de *a* e *c*, bem como do valor inicial, influenciam o período do gerador
  - No exemplo anterior, o período só será máximo (10) se $a = 1$ e $c \in \{1,3,7,9\}$.
- No entanto, não basta o período ser longo
  - Para $a = c = 1$, a sequência é "0 1 2 3 4 5 6 7 8 9"

Gerador Congruencial Linear

- Regras para a escolha de *m*, *a* e $c \neq 0$:
  - *m* e *c* serem primos entre si
  - $a-1$ ser divisível pelo fatores primos de *m*
  - $a-1$ ser divisível por 4 se *m* for divisível por 4
- Se todas a condições forem satisfeitas, o período será igual a *m* para todos os valores iniciais (seed).
- *m* deve ser tão grande quanto possível. É comum ser uma potência de 2.

Gerador Congruencial Linear

- Para $c = 0$ (LCG multiplicativo), m é normalmente um primo ou uma potência de 2 e a escolha de a segue outras regras. O valor inicial (seed) tem que ser diferente de zero.
- Exemplos de LCGs "clássicos"
  1) MINSTD Apple CarbonLib C++ ($a = 16807$, $c = 0$, $m = 2^{31}-1$)
  2) MINSTD C++ ($a = 48271$, $c = 0$, $m = 2^{31}-1$)
  3) RANDU ($a = 65539$, $c = 0$, $m = 2^{31}$) – NÃO USAR!!!
- Período igual a m ($\approx 10^9$) é hoje em dia insuficiente

Gerador Congruencial Linear

- Análise do gerador RANDU

$$x_{n+1} = ((2^{16}+3)\, x_n) \bmod 2^{31}$$

$$x_{n+2} = ((2^{16}+3)^2\, x_n) \bmod 2^{31}$$

$$= ((2^{32} + 6\,(2^{16}) + 9)\, x_n) \bmod 2^{31}$$

$$= (6\,(2^{16} + 3) - 9)\, x_n) \bmod 2^{31}$$

$$= (6\, x_{n+1} - 9\, x_n) \bmod 2^{31}$$

- É fácil ver que cada valor está relacionado linearmente com os dois valores anteriores

Gerador Congruencial Linear

- Em estudos de simulação, é muitas vezes necessário gerar sequências pseudo-aleatórias independentes
- Dividir o período de um LCG em várias partes disjuntas *sem* ter que iterar
- No caso dos LCGs multiplicativos ($c = 0$)

$$x_{i+j} = (a^j x_i) \bmod m = ((a^j \bmod m)\, x_i) \bmod m$$

onde ($a^j \bmod m$) pode ser facilmente pré-calculado

Implementação de LCGs

- Supondo que a e x_n são ambos inteiros de 32 bits, o seu produto irá ocupar até 64 bits
- A decomposição de Schrage permite calcular o produto ax sem overflow

$$(a\ x) \bmod m = A(x) + B(x)\ m$$

onde, usando sempre divisão inteira,

$$A(x) = a\ (x \bmod q) - r\ (x / q) \quad \text{com} \quad q = m / a \quad \text{e} \quad r = m \bmod a$$

$$B(x) = 1 \text{ se } A(x) < 0 \text{ e } 0 \text{ caso contrário}$$

- Notar que $m = a\ q + r$ e que $x < m$.

Combinação de Geradores

- Geradores diferentes podem ser combinados de modo a obter novos geradores com período mais longo

  1) Sejam $W_1,\ldots,W_l$ variáveis aleatórias independentes (inteiras) tais que W_1 é uniformemente distribuída entre 0 e $d-1$, onde d é um inteiro positivo. Então, $W = (W_1 + \ldots + W_l) \bmod d$ também segue uma distribuição uniforme entre 0 e $d-1$.

  2) Dada uma família de l geradores com períodos p_j, $j = 1,\ldots,l$, o período p da sequência $\{x_i = (x_{i,1},\ldots, x_{i,l}),\ i = 1, 2,\ldots\}$ é o menor múltiplo comum de $p_1,\ldots,p_l$.

Combinação de Geradores

- Então, dados l geradores

  $$x_{j,i} = f_j(x_{j,i-1})$$

  com período p_j, se $x_{1,i}$ mod p_1 for uniformemente distribuída entre 0 e p_1–1,

  $$Z_i = (x_{1,i} - x_{2,i} + x_{3,i} - \ldots + (-1)^{l-1} x_{l,i}) \bmod p_1$$

  é uma sequência pseudo-aleatória uniformemente distribuída entre 0 e $p_1 - 1$
- Para que o período seja o mais longo possível, os valores de p_j devem ser primos entre si. No caso de geradores congruenciais lineares multiplicativos, onde m_j é primo e portanto o período $p_j = (m_j - 1)$ é par, os valores de $p_j / 2$ devem ser primos entre si

Teste de Geradores Pseudo-Aleatórios

Carlos M. Fonseca
Departamento de Engenharia Informática
Universidade de Coimbra

MEI 2020/2021

Bibliografia

- R. T. Kneusel, *Random Numbers and Computers*, Springer, 2018.
- J. Banks and J. S. Carson, *Discrete-Event System Simulation*, Prentice-Hall International, 1984.
- A. M. Law and W. D. Kelton, *Simulation Modeling and Analysis*, McGraw Hill Book Company, 3rd edition, 2000.
- A. Rukhin *et al*., *A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*, NIST Special Publication 800-22, Revision 1a, April 2010.
- E. L. Lehman and J. P. Romano, *Testing Statistical Hypothesis*, Springer, 2005.

Testes de aleatoriedade

- Boas sequências pseudo-aleatórias são “indistinguíveis” de verdadeiras sequências aleatórias (apesar de determinísticas)
  - Uniformidade
  - Independência
- A verificação destas propriedades é feita com recurso a testes empíricos

Testes empíricos

- Gerar uma sequência de valores U_i, $i = 1, \ldots, n$, no intervalo [0,1] (ou [0,1[, ou]0,1[, conforme o caso)
- Verificar se esses valores parecem seguir uma distribuição uniforme U(0,1)
- Testes de ajustamento
  - Teste de $\chi2$, teste de Kolmogorov-Smirnov, ...
- Testes de independência
  - Testes de correlação, auto-correlação, ...

Teste de ajustamento de χ^2

- Dividir o intervalo [0,1] em k sub-intervalos de igual comprimento e gerar uma sequência de valores U_i, $i = 1, \ldots, n$
  - Regra geral, devemos ter $k \geq 100$ e $n / k \geq 5$
- Contar quantos valores caem em cada sub-intervalo, e designar esses números f_j, $j = 1,\ldots,k$
- Para uma distribuição uniforme, a probabilidade de cada valor cair em cada sub-intervalo é $p_j = 1 / k$ e o número esperado de valores em cada intervalo é $n\, p_j = n / k$

Teste de ajustamento de χ^2

- Calcular
$$\chi^2 = \sum_{j=1}^{k} \frac{(f_j - n p_j)^2}{n p_j} = \frac{k}{n} \sum_{j=1}^{k} \left(f_j - \frac{n}{k} \right)^2$$
- Rejeitar a hipótese nula de uniformidade dos U_i se $\chi^2 > \chi^2_{k-1,1-\alpha}$, onde $\chi^2_{k-1,1-\alpha}$ representa o quantil $1-\alpha$ da distribuição de χ^2 com $k-1$ graus de liberdade
- Valores típicos de α incluem 0.05, 0.10, ou mesmo valores maiores
- Alguns autores consideram que $\chi^2 < \chi^2_{k-1,\alpha}$ também é indicativo de não aleatoriedade

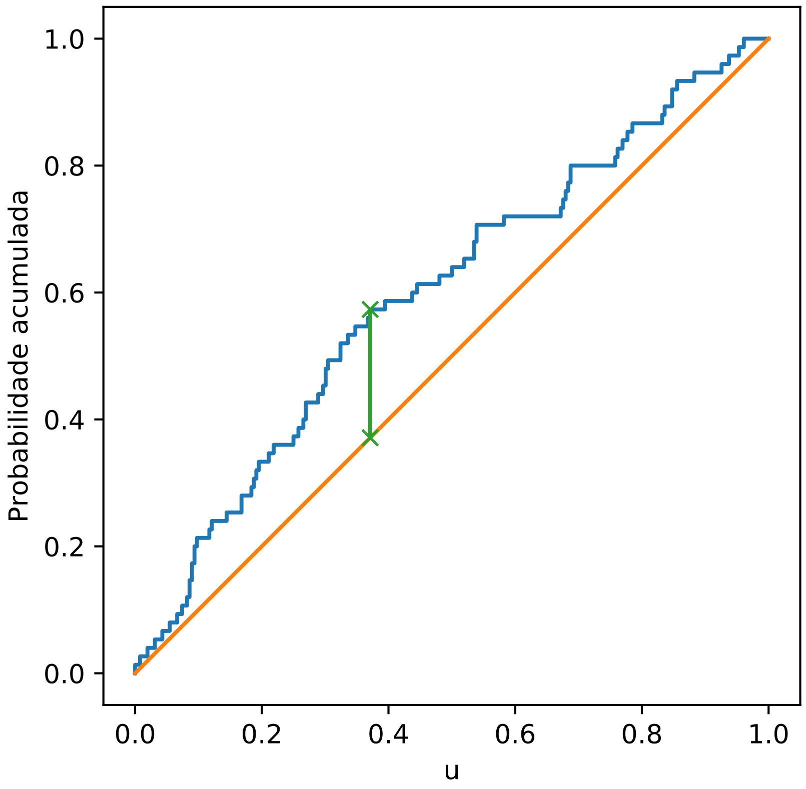
Teste de ajustamento de χ^2

- Exemplo:
  - x = (19, 172, 244, 47, 2, 22, 43, 175, 5, 76, 137, 240, 50, 138, 213, 123, 21, 77, 14, 137, 24, 128, 31, 23, 74, 138, 201, 149, 174, 37, 83, 22, 69, 246, 43, 133, 30, 95, 176, 114, 101, 94, 86, 226, 214, 8, 66, 217, 217, 23, 197, 0, 199, 112, 219, 173, 83, 77, 78, 194, 17, 25, 195, 68, 54, 237, 89, 56, 176, 49, 48, 64, 11, 24, 69)
  - $n = 75$, $U_i = x_i / 256$
  - k = 8 intervalos $[\,(j-1)/k,\ j/k\,[$, $j = 1,\ldots,k$
  - f = (18, 9, 16, 4, 7, 6, 10, 5)
  - $\chi^2 = 19.613 > \chi^2_{7,0.99} = 18.475 > \chi^2_{7,0.95} = 14.067$ (muito longe de ser uniforme)

Teste de Kolmogorov-Smirnov

- Compara a função de distribuição cumulativa $F_U(u) = u$, $0 \leq u \leq 1$, da distribuição U(0,1) com a função de distribuição cumulativa empírica $S_n(u)$ calculada a partir dos dados
- A estatística de teste é $D = \max |F_U(u) - S_n(u)|$
- Rejeitar a hipótese de uniformidade se D exceder o valor crítico do teste para o número de pontos n e nível de significância α considerados

Teste de Kolmogorov-Smirnov



Teste série

- Generalização do teste de χ^2 para mais dimensões, d
- Por exemplo, para $d = 2$, proceder como anteriormente, mas considerando a sequência de pares (U_1, U_2), (U_3, U_4), (U_5, U_6), …, e dividindo o quadrado unitário em k^2 sub-regiões
- Aplicar o teste de χ^2 tendo em conta que agora $p_j = 1 / k^2$, $j = 1,\ldots,k^2$, e que o número de graus de liberdade é k^2-1

Teste série

- Exemplo: sequência anterior
  - $\boldsymbol{X}$ = ((19, 172), (244, 47), (2, 22), (43, 175), (5, 76), (137, 240), (50, 138), (213, 123), (21, 77), (14, 137), (24, 128), (31, 23), (74, 138), (201, 149), (174, 37), (83, 22), (69, 246), (43, 133), (30, 95), (176, 114), (101, 94), (86, 226), (214, 8), (66, 217), (217, 23), (197, 0), (199, 112), (219, 173), (83, 77), (78, 194), (17, 25), (195, 68), (54, 237), (89, 56), (176, 49), (48, 64), (11, 24), (69, ...))
  - $n = 37$, $\boldsymbol{U}_i = \boldsymbol{x}_i / 256$
  - $k = 8$ daria origem a $k^2 = 64$ intervalos e a $n / k^2 = 0.578$ (muito menor que 5, dever-se-ia considerar um valor de k menor, como 3 ou mesmo 2)

Teste de separação (*gap test*)

- Em vez da frequência com que os valores (ou tuplos de valores) ocorrem numa sequência, este teste conta o comprimento das sub-sequências de valores fora de um dado intervalo
- Este teste requer uma sequência muito longa ($n > 10^8$ valores) para ser fiável
- Considerando novamente a distribuição U(0,1) e o intervalo [α, β[, é possível calcular as probabilidades de cada comprimento ocorrer.

Gap test

- Probabilidades de ocorrência de cada comprimento

  $p_0 = \beta - \alpha$

  $p_i = p_0\,(1 - p_0)^i$, para $0 < i < t$

  $p_{\geq t} = (1 - p_0)^t = 1 - p_0 - p_1 - \ldots - p_{t\text{-}1}$

- Aplicar o teste de χ^2 tendo em conta que agora há diferentes valores de p_j, $j = 0,\ldots, t$. O número de graus de liberdade é t, porque há $t + 1$ classes

Gap test

- Exemplo: sequência anterior (valores arredondados por simplicidade de visualização, $\alpha = 0$, $\beta = 0.5$)
  - x = (0.07, 0.67, 0.95, 0.18, 0.01, 0.09, 0.17, 0.68, 0.02, 0.30, 0.54, 0.94, 0.20, 0.54, 0.83, 0.48, 0.08, 0.30, 0.05, 0.54, 0.09, 0.50, 0.12, 0.09, 0.29, 0.54, 0.79, 0.58, 0.68, 0.14, 0.32, 0.09, 0.27, 0.96, 0.17, 0.52, 0.12, 0.37, 0.69, 0.45, 0.39, 0.37, 0.34, 0.88, 0.84, 0.03, 0.26, 0.85, 0.85, 0.09, 0.77, 0.00, 0.78, 0.44, 0.86, 0.68, 0.32, 0.30, 0.30, 0.76, 0.07, 0.10, 0.76, 0.27, 0.21, 0.93, 0.35, 0.22, 0.69, 0.19, 0.19, 0.25, 0.04, 0.09, 0.27)

Teste do máximo de *t*

- Teste de frequência para o máximo de *t* amostras
- Dividir a sequência em sub-sequências de comprimento *t*, tomar o máximo de cada uma e elevar esse valor a *t*
- A distribuição dos valores resultantes deverá ser uniforme (porquê?)
- Aplicar o teste de χ^2 tendo em conta que agora o número de amostras é n / t. O número de graus de liberdade depende do número de intervalos usados para testar a uniformidade.

Teste do máximo de *t*

- Exemplo: sequência anterior (valores arredondados por simplicidade de visualização, $t = 5$)
  - X = ((0.07, 0.67, 0.95, 0.18, 0.01), (0.09, 0.17, 0.68, 0.02, 0.30), (0.54, 0.94, 0.20, 0.54, 0.83), (0.48, 0.08, 0.30, 0.05, 0.54), (0.09, 0.50, 0.12, 0.09, 0.29), (0.54, 0.79, 0.58, 0.68, 0.14), (0.32, 0.09, 0.27, 0.96, 0.17), (0.52, 0.12, 0.37, 0.69, 0.45), (0.39, 0.37, 0.34, 0.88, 0.84), (0.03, 0.26, 0.85, 0.85, 0.09), (0.77, 0.00, 0.78, 0.44, 0.86), (0.68, 0.32, 0.30, 0.30, 0.76), (0.07, 0.10, 0.76, 0.27, 0.21), (0.93, 0.35, 0.22, 0.69, 0.19), (0.19, 0.25, 0.04, 0.09, 0.27))
- n / t deverá ser pelo menos 10^6

Teste de correlação série

- Calcula o coeficiente de correlação entre cada valor na sequência e o valor seguinte

$$C=\frac{n\left(U_0U_1+U_1U_2+\cdots+U_{n-2}U_{n-1}+U_{n-1}U_0\right)-\left(U_0+U_1+\cdots+U_{n-1}\right)^2}{\left(U_0^2+U_1^2+\cdots+U_{n-1}^2\right)-\left(U_0+U_1+\cdots+U_{n-1}\right)^2}$$

- Para U_i uniformes, C deverá estar no intervalo [$m - 2\sigma$, $m + 2\sigma$], com $m = -1/(n-1)$ e $\sigma=\frac{1}{n-1}\sqrt{\frac{n(n-3)}{n+1}}$, para α ≈ 0.05.

Teste de permutação

- Dividir a sequência em sub-sequências de comprimento $t \leq 5$, e tomar as permutações que ordenariam cada sub-sequência
- Há $t!$ permutações possíveis
- Aplicar o teste de χ^2 notando que o número de amostras é n / t, e que o número de classes é $t!$

Teste de permutação

- Exemplo: sequência anterior (valores arredondados por simplicidade de visualização, $t = 5$)
  - X = ((0.07, 0.67, 0.95, 0.18, 0.01), (0.09, 0.17, 0.68, 0.02, 0.30), (0.54, 0.94, 0.20, 0.54, 0.83), (0.48, 0.08, 0.30, 0.05, 0.54), …)
  - P = ((5, 1, 4, 2, 3), (4, 1, 2, 5, 3), (3, 1, 4, 5, 2), (4, 2, 3, 1, 5), …)
- Para $t = 4$ ou $t = 5$, n deverá ser pelo menos 10^8

Aplicação de testes de aleatorieade

- Testes de hipóteses
  - Hipótese de investigação
  - Hipótese nula, H_0
  - Hipótese alternativa, H_1
  - Pressupostos subjacentes ao processo de amostragem
  - Estatística de teste
  - Distribuição da estatística de teste sob a hipótese nula

Aplicação de testes de aleatoriedade

- Aplicação de um teste de hipóteses
  - Nível de significância, α
  - Região crítica
  - Valor observado da estatística de teste
  - Decisão
    - Rejeitar H_0 em favor de H_1 quando o valor observado da estatística de teste pertence à região crítica
    - Caso contrário, *não rejeitar* H_0 por falta de evidência nesse sentido

Aplicação de testes de aleatoriedade

- Aplicação de um teste de hipóteses (em alternativa)
  - Nível de significância, α
  - Valor observado da estatística de teste
  - Cálculo do p-valor
  - Decisão
    - Rejeitar H_0 em favor de H_1 quando o p-valor é *menor ou igual* que o nível de significância α
    - Caso contrário, *não rejeitar* H_0 por falta de evidência nesse sentido

Aplicação de testes de aleatoriedade

- *p*-valor
  - É o menor valor de significância *α* que conduziria à rejeição da hipótese nula
  - Sugere quão fortemente os dados contradizem a hipótese nula
  - **Não é** a probabilidade de alguma das hipóteses estar correta ou incorreta!!!
  - Hoje em dia pode ser calculado (ou aproximado) facilmente para cada conjunto de observações (usando o computador)

Aplicação de testes de aleatoriedade

- Tipos de erro
  - Tipo I: Rejeitar H_0 quando ela é de facto verdadeira
  - Tipo II: Não rejeitar H_0 quando ela é de facto falsa
- A probabilidade de cometer um erro de Tipo I é menor ou igual a α
- A probabilidade de cometer um erro de Tipo II está relacionada com a potência do teste

Aplicação de testes de aleatoriedade

- Experiência 1
  - Gerar uma sequência pseudo-aleatória de um dado comprimento, *n*
  - Aplicar o teste de uniformidade de χ^2 considerando $\alpha = 0.05$
  - O *p*-valor é p = 0.032
  - O que é que isto diz sobre o gerador?

Aplicação de testes de aleatoriedade

- Experiência 2
  - Gerar 100 sequências pseudo-aleatórias de um dado comprimento, *n*
  - Aplicar o teste de uniformidade de χ^2 considerando $\alpha = 0.05$
  - Há 4 *p*-valores menores ou iguais a 0.05
  - O que é que isto diz sobre o gerador?

Aplicação de testes de aleatoriedade

- Experiência 3
  - Gerar 100 sequências pseudo-aleatórias de um dado comprimento, *n*
  - Aplicar todos os testes referidos a cada sequência
  - Como é que podemos decidir se o gerador é bom ou mau?

Aplicação de testes de aleatoriedade

- Recordar que todos os testes podem falhar mesmo que a sequência seja verdadeiramente aleatória
  - A probabilidade de um dado teste falhar nas condições da hipótese nula é menor ou igual que α
  - Uma possibilidade é comparar a proporção de testes falhados de cada tipo com o que acontece com (uma realização de) uma sequência verdadeiramente aleatória, enquanto estimativa das probabilidades de erro de Tipo I

Aplicação de testes de aleatoriedade

- Comparação de geradores
  - Calcular a proporção de testes falhados de cada tipo para cada gerador e para uma sequência verdadeiramente aleatória
  - Calcular a diferença absoluta entre essas proporções e considerar a norma do vetor resultante
  - Ordenar os diversos geradores em conformidade
  - Resultados são relativos, mas não absolutos

Aplicação de testes de aleatoriedade

- Observações finais
  - Considerar que um teste de aleatoriedade falha quando $p < 0.05$ ou $p > 0.95$ leva a que o teste deva falhar em média até 1/10 das vezes mesmo para um gerador ideal
  - Importa distinguir entre testes para geradores pseudo-aleatórios e os testes estatísticos subjacentes