Estatística Computacional

Patrícia de Siqueira Ramos

PPGEAB
UNIFAL-MG

18 de Abril de 2018

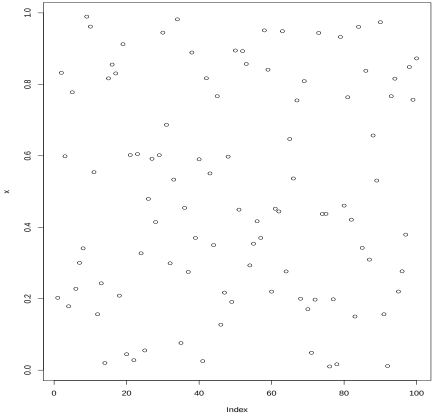
Diagnóstico em distribuições

Relembrando: gerar realizações de v.a.s uniformes

- Para gerar números de uma f.d.p. uniforme podemos usar as funções vistas com o método congruencial (`congruencial`, `gna0`) ou o algoritmo Mersenne-Twister implementado no `R` (função `runif(n)`)
- Uma representação gráfica simples da amostra gerada é o gráfico de dispersão dos índices de x *versus* valores de x, usando

```
x = runif(100)
x
plot(x)
```

Questão: $\sim U(0,1)$?



Questão: $\sim U(0,1)$?

- Essa imagem representa números aleatórios? Ela representa realizações independentes de uma v.a. distribuída uniformemente entre 0 e 1?
- Gere várias sequências de números e obtenha os gráficos. Qual a sua conclusão?
- Dica: use

  ```
  par(mfrow = c(2, 4))
  ```

  por exemplo, que mostra oito gráficos ao mesmo tempo, organizados de forma 2×4

Questão: $\sim U(0,1)$?

- Essa imagem representa números aleatórios? Ela representa realizações independentes de uma v.a. distribuída uniformemente entre 0 e 1?
- Gere várias sequências de números e obtenha os gráficos. Qual a sua conclusão?
- Dica: use

  ```
  par(mfrow = c(2, 4))
  ```

  por exemplo, que mostra oito gráficos ao mesmo tempo, organizados de forma 2×4
- Gere também os histogramas correspondentes.

Desafio

Já sabemos que os números gerados são pseudoaleatórios, mas

- Conseguimos perceber que eles não são números independentes? Se conseguirmos, podemos usar um gerador melhor
- Somos capazes de perceber que a sequência gerada é determinística?

Desafio

Já sabemos que os números gerados são pseudoaleatórios, mas

- Conseguimos perceber que eles não são números independentes? Se conseguirmos, podemos usar um gerador melhor
- Somos capazes de perceber que a sequência gerada é determinística?
- Sabemos que `runif()` gera uma sequência determinística usando uma função recursiva (método da congruência) que depende da escolha da semente adequada e de outros valores. Sua forma mais simples é:

$$U_{i+1} = (aU_i + c) mod \quad m,$$

em que a, c e m são números inteiros adequadamente escolhidos para que o período da sequência seja $\leq m$.

Padrões

O R, além de gerar números pseudoaleatórios, consegue gerar vários tipos de sequências (já vistas):

```
# :
1:10
10.1:1.2
# c() - combina argumentos num vetor
# c(...)
c(3, 5, 9)
c("a", "b", "c")
# seq() - gera sequências gerais
seq(from = 3, to = 5, by = 0.2)
seq(5, 20, length = 4)
# rep() - repete um argumento
x = c(5, 9, 2)
rep(x, 3)
rep(1:3, c(2, 3, 1))
```

Padrões

Exercício: Use

```
plot(sin(1:100))
```

para gerar um gráfico da função seno de forma discreta. Essa função é uma sequência aleatória?
Dica: use

```
plot(sin(1:100), type = "l")
```

para conectar os pontos.

Diagnóstico sobre as distribuições

- Devemos adotar estratégias para verificar se há padrões nos dados gerados
- Vamos assumir que a sequência consiste de números aleatórios independentes com uma distribuição de probabilidade comum
- Como poderíamos checar se essa distribuição é uma uniforme?
- Primeiramente vamos ignorar os valores (a, b) e adotar $(0, 1)$
- Realizações de v.a.s não nos permitem reconhecer distribuições diretamente

Diagnóstico sobre as distribuições - `QQ-plots`

- Gráficos muito utilizados para verificar se um conjunto de dados provém de uma determinada distribuição de probabilidade são os `QQ-plots`
- Neles são plotados os quantis esperados da distribuição $\times$ quantis observados
- Se o diagrama de dispersão formar uma linha reta significa que os dados seguem a distribuição testada

Diagnóstico sobre as distribuições - `QQ-plots`

- Gráficos muito utilizados para verificar se um conjunto de dados provém de uma determinada distribuição de probabilidade são os `QQ-plots`
- Neles são plotados os quantis esperados da distribuição $\times$ quantis observados
- Se o diagrama de dispersão formar uma linha reta significa que os dados seguem a distribuição testada

Como seria a implementação dessa estratégia para a distribuição $U(0,1)$?

QQ-plot da $U(0,1)$

```
n = 100
q = seq(0, 1, length.out=n)
x1 = runif(n)
xsort1 = sort(x1)
plot(q, xsort1, col = "cadetblue3")
abline(0, 1, col = "deepskyblue4")

x2 = runif(n)
xsort2 = sort(x2)
plot(q, xsort2, col = "cadetblue3")
abline(0, 1, col = "deepskyblue4")
```

Dica de como escolher cores no R:

http://www.stat.columbia.edu/ tzheng/files/Rcolor.pdf

Distribuição empírica

Distribuição empírica

- Precisamos caracterizar as distribuições de forma que possamos analisá-las empiricamente
- Para n observações $X_1, \cdots, X_n$ podemos definir a distribuição empírica P_n como

$$P_n = \frac{1}{n}\sum^{n} \delta_{X_i},$$

em que δ_{X_i} é a medida em X_i.

- Então,

$$P_n(A) = \frac{1}{n}\#\{i : X_i \in A\} = \frac{1}{n}\sum^{n} I_{X_i \in A}.$$

Distribuição empírica

Problemas:

- P_n de um conjunto de observações independentes de uma distribuição comum P difere bastante da distribuição original
- distribuições empíricas são sempre discretas (independente da distribuição original)
- Estratégia: restringirmos a uma família de testes que sejam tratáveis empiricamente.

Ex. 1: Função distribuição

- No lugar da distribuição P usaremos sua função distribuição $F = F_p$, em que

$$F(x) = P(X \leq x).$$

- Para uma distribuição empírica P_n de n observações $X_1, \cdots, X_n$, a função de distribuição empírica correspondente é

$$F_n(x) = \frac{1}{n}\#\{i : X_i \leq x\}.$$

- Essa função distribuição empírica é útil em simulações, testes de permutação, reamostragem etc. (as inferências são feitas usando essa distribuição no lugar da teórica)

Ex. 1: Função distribuição

- Para testar se uma sequência aleatória tem uma função distribuição F comparamos F com a função distribuição empírica F_n
- Vamos usar a $U(0,1)$: neste caso temos $F(x) = F_{unif}(x) = x$ para $0 \leq x \leq 1$
- Temos que obter F_n e F

Ex. 1: Função distribuição

- Para testar se uma sequência aleatória tem uma função distribuição F comparamos F com a função distribuição empírica F_n
- Vamos usar a $U(0,1)$: neste caso temos $F(x) = F_{unif}(x) = x$ para $0 \leq x \leq 1$
- Temos que obter F_n e F
- Teremos uma imagem completa de F_n se avaliarmos F_n nas obs. X_i, $i = 1, \ldots, n$.
- Se $X_{(i)}$ denota a i-ésima estatística de ordem, temos que $F_n(X_{(i)}) = i/n$ se $X_{(1)} < \cdots < X_{(n)}$
- Temos, então, que comparar $F_n(X_{(i)})$ com o valor teórico $F(X_{(i)}) = X_{(i)}$.

Ex. 1: Função distribuição - no R

```
n = 100
x = runif(n)
xsort = sort(x)
i = (1:n)
y = i / n
plot(xsort, y)
# como seria uma linha com valores teóricos?
# usar abline(intercepto, inclinação)
```

Ex. 1: Função distribuição - no R - mais compacta

```
# versão compacta
x = runif(100)
plot(sort(x), 1 : length(x) / length(x))
abline(0, 1)
```

Ex. 1: Função distribuição - no R - mais compacta

```
# versão compacta
x = runif(100)
plot(sort(x), 1 : length(x) / length(x))
abline(0, 1)
```

- Limitação: implementação só funciona para a uniforme (a função distribuição teórica é uma diagonal, facilmente comparável). E se forem outras distribuições?

Ex. 1: Função distribuição - no R - mais elementos gráficos

```
# adicionando título - não recomendável
plot(sort(x), (1:length(x))/length(x),
main = "Função distribuição empírica\n X uniforme")

# incluindo nomes dos eixos
# ajuda para expressões matemáticas: help(plotmath)
x = runif(100)
plot(sort(x), (1:length(x))/length(x),
     xlab = "x", ylab = expression(F[n]),
     main = "Função distribuição empírica\n X uniforme")
# abline
```

Ex. 1: Função distribuição pronta no R

```
# classe ecdf para funções de distribuição empíricas
# forma automática do R
# uniforme(0,1)
plot(ecdf(runif(30)))
# normal padrão
plot(ecdf(rnorm(30)))
# exponencial com alpha = 2
plot(ecdf(rexp(30, 2)))
```

Exercício: gere os gráficos das funções distribuição ($n = 100$):

- Normal com $\mu = 0$ e $\sigma = 5$
- Beta com $a = 1$ e $b = 3$
- Cauchy com $\alpha = 0$ e $\beta = 1$
- Cauchy com $\alpha = 0$ e $\beta = 0,5$

Ex. 2: Histograma

- Selecionaremos conjuntos de teste A_j, $j = 1, \ldots J$ que cubram a amplitude de X. Por ex., para a distribuição $U(0,1)$ podemos escolher os intervalos

$$A_j = \left(\frac{j-1}{J}, \frac{j}{J}\right]$$

Ex. 2: Histograma

- Selecionaremos conjuntos de teste A_j, $j = 1, \ldots J$ que cubram a amplitude de X. Por ex., para a distribuição $U(0, 1)$ podemos escolher os intervalos

$$A_j = \left(\frac{j-1}{J}, \frac{j}{J} \right]$$

- No lugar de P consideraremos o vetor $(P(A_j))_{j=1,\cdots,J}$.
- Sua representação gráfica é o histograma
- Versão empírica: $(P_n(A_j))_{j=1,\cdots,J}$
- O histograma depende muito da escolha dos conjuntos de teste (uma discretização com uma escolha ruim desses conjuntos pode ser desastrosa)

Ex. 2: Histograma

- O vetor de probabilidades teóricas

$$(P(A_j))_{j=1,\cdots,J} = (1/J, 1/J, \ldots, 1/J)$$

será comparado com as frequências relativas observadas

$$\#i\{X_i \in A_j\}/n, j = 1, \ldots, J.$$

- Como escolher o número de classes J? Vamos utilizar o método automático do `R`

Ex. 2: Histograma no R

```
x = runif(100)
hist(x)
rug(x)  # adiciona dados originais ao histograma
```

Ex. 2: Histograma no R

```
x = runif(100)
hist(x)
rug(x)  # adiciona dados originais ao histograma
```

- Melhor:

```
# para que as escalas sejam comparáveis usar
# probability = T
x = runif(1000)
hist(x, probability = TRUE)
rug(x)
lines(density(x))   # estimação kernel
```

Ex. 2: Histograma no R

Particularidades:

- Histogramas: problema de discretização
- Estimadores de densidade Kernel: não apresentam o efeito de discretização, mas "borram" os dados

Ex. 2: Histograma no R

```
# inspecionando objeto gerado
x = runif(100)
histog = hist(x)
histog
```

- `counts` informa as contagens por célula (que são os números que estamos procurando)
- O objeto que chamamos de `histog` possui cinco componentes (e cada um é um vetor)
- usando, por exemplo,
  `histog$counts`
  vemos o vetor com as contagens das classes

Testes de uniformidade e independência

Teste de Kolmogorov-Smirnov

- Para variáveis independentes e identicamente distribuídas $(X_1, \ldots, X_n)$ com função de distribuição F, assumimos que

$$i/n = F_n(X_{(i)}) \approx F(X_{(i)})$$

- Para a $U(0,1)$, isso leva a

$$i/n = X_{(i)} \approx F(X_{(i)})$$

- Do ponto de vista estatístico, cada $X_{(i)}$ é uma v.a., então $F(X_{(i)})$ também é v.a. e podemos analisar sua distribuição

Teorema (Kolmogorov-Smirnov)

- Para uma função de distribuição contínua F, a distribuição de

$$\sup_x |F_n - F|(x)$$

é independente de F (em geral, dependerá de n).

- De forma prática, para uma função distribuição contínua, podemos usar a seguinte estratégia de decisão:
as observações $X_1, \ldots, X_n$ não seguem a hipótese de observações i.i.d. com distribuição F se $\sup |F_n - F|$ for muito grande, ou

$$\sup |F_n - F| > F_{crit}/\sqrt{(n)},$$

em que $F_{crit} = F_{Kolmogorov-Smirnov,1-\alpha}$ (quantil superior da função distribuição K-S)

Teste de Kolmogorov-Smirnov para a $U(0,1)$

- Para a distribuição $U(0,1)$

$$\sup_x |F_n - F|(x) = \max_{X(i)} |F_n - F|(X(i)) = \max_i |i/n - X(i)|.$$

- No R:

```
x = runif(100)
max(abs((1 : length(x)) / length(x) - sort(x)) )
```

que é a estatística procurada (a estatística e a função de distribuição já estão implementadas no R).

Teste de Kolmogorov-Smirnov no R

- Use

  ```
  help(ks.test)
  ```

  para entender como usar a função `ks.test`
- Quais são os resultados esperados para os vetores:

  ```
  1:100
  runif(100)
  sin(1:100)
  rnorm(100)
  ```

(para os testes, use uma distribuição uniforme em um intervalo adaptado para os dados).

Estatísticas para histogramas e outros - χ^2

- Suponha que tenhamos escolhido A_j, $j = 1, \ldots, J$ de forma a cobrir a amplitude dos valores de X. O vetor $X_1, \ldots, X_n$ traduzido em contagens N_j é

$$N_j = (\# i : X_i \in A_j)$$

- Se $(X_i)_{i=1,\ldots,n}$ são independentes com distribuição idêntica P, $(N_j)_{j=1,\ldots,J}$ é um vetor aleatório que segue uma distribuição multinomial com parâmetros

$$n \qquad \mathrm{e} \qquad (p_j)_{j=1,\ldots,J},$$

em que $(p_j) = P(A_j)$

- Obs.: para $J = 2$ temos a distribuição binomial

Teorema

- Para $(p_j)_{j=1,\dots,J}$, $p_j > 0$ (com $n \to \infty$), a estatística

$$Q^2 = \sum_{j=1,\dots,J} \frac{(N_j - np_j)^2}{np_j} \sim \chi^2_{J-1}$$

- Regra de decisão: fixamos χ^2_{crit} e rejeitamos a hipótese de que as observações $(X_1, \dots, X_n)$ vêm de uma distribuição uniforme se o valor da estatística Q^2 ultrapassar o valor χ^2_{crit} (ou utilizar o valor-p)
- χ^2_{crit} é o quantil superior da distribuição χ^2_{J-1}

Teste χ^2 no R

- Função `chisq.test()`
- Aqui, o "assintótico" deve ser visto com cautela: não é suficiente que a amostra seja grande, mas que o tamanho amostral em cada classe do histograma o seja
- Há a possibilidade de se calcular o valor-p usando simulação Monte Carlo
- Os testes χ^2 foram feitos para tabelas de contingência (tabelas de dupla entrada) e só estamos usando um caso especial (em que temos um vetor com contagens)

Exercícios

1. Use `help(chisq.test)` para ver como usar a função do `R`.

Exercícios

1. Use `help(chisq.test)` para ver como usar a função do `R`.

a) Como a binomial é o caso mais simples da multinomial, testar a hipótese ($p_j = 1/J$), $J = 2$, ou seja, $p_1 = 0,5$ e $p_2 = 0,5$ para testar, por exemplo, se o lançamento de uma moeda 100 vezes pode ser considerada uma uniforme:

```
(48, 52)
(57, 43)
```

Exercícios

1. Use `help(chisq.test)` para ver como usar a função do `R`.

a) Como a binomial é o caso mais simples da multinomial, testar a hipótese ($p_j = 1/J$), $J = 2$, ou seja, $p_1 = 0,5$ e $p_2 = 0,5$ para testar, por exemplo, se o lançamento de uma moeda 100 vezes pode ser considerada uma uniforme:

`(48, 52)`

`(57, 43)`

b) Como você calcularia o valor-p para o caso `(48, 52)` usando a função apropriada do `R` para retornar probabilidades da χ^2?

Exercícios

1. Use `help(chisq.test)` para ver como usar a função do `R`.

a) Como a binomial é o caso mais simples da multinomial, testar a hipótese ($p_j = 1/J$), $J = 2$, ou seja, $p_1 = 0,5$ e $p_2 = 0,5$ para testar, por exemplo, se o lançamento de uma moeda 100 vezes pode ser considerada uma uniforme:

`(48, 52)`

`(57, 43)`

b) Como você calcularia o valor-p para o caso `(48, 52)` usando a função apropriada do `R` para retornar probabilidades da χ^2?

c) Faça o mesmo da letra (a), porém agora simule alguns acontecimentos de caras e coroas e teste se eles podem ser considerados como vindos de uma uniforme.

Exercícios

2. Aplique `help(chisq.test)` para testar a hipótese $(p_j = 1/J), J = 5$ classes aos seguintes vetores de contagens:

```
(30 30 30 30 30)
(15 25 40 22 18)
(0 0 90 0 60)
(0 0 9 0 6)
```

Exercícios

3. Quais são os resultados esperados se um teste χ^2 for usado para verificar se os seguintes vetores seguem uma distribuição uniforme?

```
1:100
runif(100)
sin(1:100)
rnorm(100)
```

Realize os testes e discuta.
Dica: a função `chisq.test()` recebe uma tabela de frequência como argumento. A função `table()` pode ser usada para gerar essa tabela. Mas é melhor usar a função `hist()` que retorna as contagens como um de seus resultados (objeto `counts`).