MAP2212 - EP4 Estimativa de uma integral com base em geradores randômicos não-uniformes

Vinícius da Costa Collaço - 11811012 Nikolas Lukin - 5381328

28 de maio de 2022

Instituto de Matemática e Estatística

Universidade de São Paulo

Maio de 2022

Sumário

1 Introdução

Esse relatório visa apresentar uma solução para o quarto exercício programa (EP4) proposto na matéria MAP2212/2022 (Laboratório de Computação e Simulação) do curso de bacharelado de matemática aplicada e computacional (BMAC) do instituito IME-USP.
O objetivo deste EP é estimar a função verdade definida por,

$$W(v) = \int_{T(v)} f(\theta \mid x, y) d\theta \tag{1}$$

através de uma função *U(v)* obtida por integral condensada da massa de probabilidade[3] de $f(\theta|x, y)$ no domínio *T(v)* e que não ultrapassa um nível v pré-estabelecido, ou seja,

$$T(v) = \{\theta \in \Theta \mid f(\theta \mid x, y) \leq v\} \tag{2}$$

A função f é a função de densidade de probabilidade posterior de *Dirichlet*, que representa um modelo estatístico m-dimensional multinomial, dado por:

$$f(\theta \mid x, y) = \frac{1}{B(x+y)} \prod_{i=1}^{m} \theta_i^{x_i + y_i - 1} \tag{3}$$

onde x é um vetor de observações, y é um vetor de observações a priori, θ é um vetor simplex de probabilidades, $m = 3$ é a dimensão e B representa a distribuição Beta. Observe que,

$$x, y \in \aleph^m, \theta \in \Theta = S_m = \{\theta \in \Re_m^+ \mid f(\theta \mid x, y) \leq v\} \tag{4}$$

Nas próximas seções demonstraremos como estimamos a função verdade usando integral condensada e a variante de geradores randômicos não-uniformes, Utilizando a linguagem *Python* e bibliotecas adequadas[1, 6, 2, 5],

2 Definindo o tamanho do n

Para a definição do valor do n, iremos utilizar a aproximação assintótica de uma distribuição Bernoulli para determinar a quantidade de pontos dentro de um bin necessários para satisfazer a resolução e o erro máximo tolerável definido em $\epsilon = 0.05\%$. A quantidade k de bins deve ser tal a garantir uma resolução que seja maior que o erro ϵ,

$$W(v_j) - W(v_{j-1}) = \frac{1}{k} \geq \epsilon \tag{5}$$

Supondo que o tamanho da amostra seja relativamente grande, pelo teorema do limite central, podemos aproximar a Bernoulli por uma normal, tendo:

$$P(|\hat{p} - p| \leq \varepsilon) \geq \gamma$$

[4]

$$P(-\varepsilon \leq \hat{p} - p \leq \varepsilon) = P\left(\frac{-\sqrt{n}\varepsilon}{\sigma} \leq Z \leq \frac{\sqrt{n}\varepsilon}{\sigma}\right) \approx \gamma$$

obtendo finalmente

$$n = \frac{\sigma^2 {Z_\gamma}^2}{\varepsilon^2} \tag{6}$$

esse resultado obtido em 6 será utilizado para definir o número de pontos necessários para resultar no erro mínimo exigido.

2.1 Número de bins (k)

O número de bins da distribuição discreta de probabilidades deve ser tal que permita uma resolução maior que o erro estipulado, definido em 7

$$k \geq \frac{1}{\epsilon} \implies k \geq 2000 \tag{7}$$

Para uma resolução melhorada no programa foi utilizado

$$k = 4000$$

2.2 Intervalo de confiança

Para o problema será utilizado um intervalo de confiança $\gamma = 95\%$, escolhido arbitrariamente obtendo assim o Z_γ da $N(0, 1)$, portanto $Z_\gamma = 1,96$.

2.3 Erro (ε)

Para o cálculo do n será utilizado o valor do erro relativo exigido entre o valor real de $W(u)$ e sua estimativa, portanto $\varepsilon = 0,0005$

2.4 Variância amostral

Para o cálculo da variância, considera-se que os pontos se distribuem segundo uma distribuição de Bernoulli dentro de um bin com probabilidade igual à resolução do bin. Com o valor de $k = 4000$ variância pode ser determinada por,

$$\sigma^2 = \frac{\epsilon}{2}(1 - \frac{\epsilon}{2}) \approx \frac{\epsilon}{2} \tag{8}$$

2.5 Valor final do n

O cálculo do n pode ser obtido através das equações em 7 6 8 resultando em,

$$n = k \cdot \frac{1,96^2 \cdot \frac{\epsilon}{2}}{\epsilon^2} \geq 15.366.400$$

portanto:

$$n_{min} = 15.366.400 \; pontos$$

3 Resultados e discussão

Para a simulação construiu-se um algoritmo que recebe como dados de entrada os vetores x e y, constrói e ordena crescentemente uma lista com n entradas de $f(\theta|x,y)$ obtidas de simulações aleatórias de θ_i advindas de um gerador gerador randômico não-uniforme (Gamma). Esta lista é condensada em uma lista menor com k bins, os quais cada um condensa a informação de n/k pontos. Finalmente, o algoritmo percorre esta última lista e localiza a posição i correspondente ao cut-off v. O valor calculado da integral condensada *U(v)* é obtido por,

$$\begin{cases} U(v) = 0, & v \leq min(f(\theta \mid x,y)) \\ U(v) = \frac{i}{n} & min(f(\theta \mid x,y)) \leq v \leq max(f(\theta \mid x,y)) \\ U(v) = 1, & v \geq max(f(\theta \mid x,y)) \end{cases} \tag{9}$$

O gráfico de *U(v)* está ilustrado na Figura 1 e a Tabela 1 resume alguns valores de *W(v)* fornecidos pelo monitor para validação dos resultados.

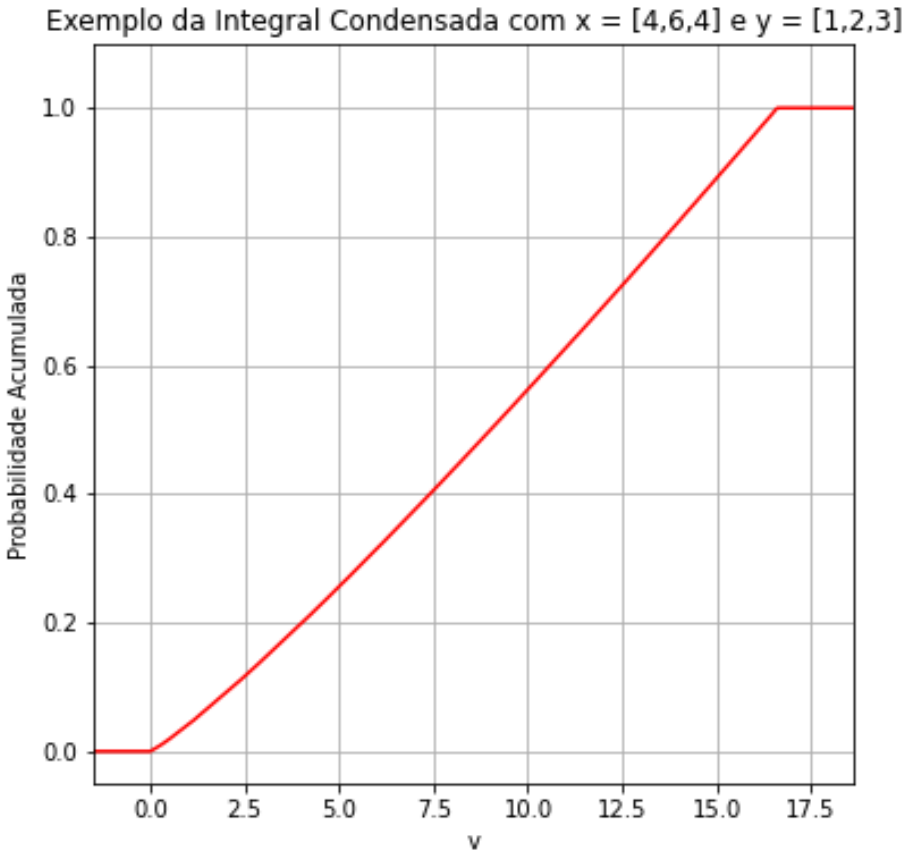

Figura 1: Gráfico de *U(v)*

v	W(v)	U(v)	Erro (%)
0.00	0.00	0.00	0
0.50	0.02	0.02	0
1.00	0.04	0.04	0
15.0	0.89	0.89	0
500	1.00	1.00	0

Tabela 1: Validação de *U(v)*

4 Conclusão

Com os resultados obtidos no trabalho, foi observado um método efetivo para o cálculo da integral da função verdade através da discretização da distribuição de *Dirichlet* usando geradores randômicos não-uniformes implementado em Python.

O algoritmo implementado mostrou-se eficiente para cálculos de diversos *U(v)* após a implementação da classe, cumprindo assim os requisitos exigidos pelo problema proposto.

Referências

[1] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020.

[2] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[3] Stan Kaplan and James C Lin. An improved condensation procedure in discrete probability distribution calculations. *Risk Analysis*, 7(1):15–19, 1987.

[4] Wilton de O. Bussab Pedro A. Morettin. *Estatística Básica,* 6^a *Edição*. 2010.

[5] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[6] Michael L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 2021.