Exercício Programa 5 MAP-2212

Luiz Guilherme De Padua Sanches
NUSP: 13686431

Daniel Vertamatti NUSP
NUSP: 2370859

Victor de Faro Quadros
NUSP: 12556632

Junho, 2023

1 Objetivos e considerações

Foi solicitada no enunciado a aproximação da seguinte função, denominada função verdade $W(v)$, através de uma função $U(v)$ com erro de até 0,05%:

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

A região de integração T(v) é definida por:

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

Em que v é um nível preestabelecido, θ é um vetor de parâmetros no *simplex* m-dimensional e $f(\theta|x, y)$ é a uma densidade definida como:

$$f(\theta|x, y) = \frac{1}{B(x + y)} \prod_{i=1}^{m} \theta_i^{x+y-1}$$

Essa função densidade é conhecida como função densidade de probabilidade de Dirichlet, em que x é um vetor de observações e y um vetor de informações a priori, ambos com dimensão m. Nesse exercício, a FDP Dirichlet deve ser gerada utilizando-se o método MCMC, diferentemente do exercício anterior, no qual utilizou-se a FDP gerada pelo Python com a biblioteca NumPy.

Nesse projeto, foi solicitada dimensão igual a três, ou seja, $m = 3$. Os vetores x e y são solicitados ao usuário e o vetor de parâmetros θ será gerado através do método MCMC para simularmos a distribuição Dirichlet.

Foi discutido com o monitor a diferença entre o método MCMC comum e o método apresentado em sala de aula pelo Professor. Os dois métodos foram considerados válidos para a execução desse exercício programa.

A função $f(\theta|x, y)$, durante a maior parte do programa, não utilizará a constante de normalização (recebendo o nome de função potencial). Essa constante será utilizada apenas quando informado o valor v, em que esse valor será multiplicado pela constante. Dessa forma, o valor de v refere-se ao valor da função densidade.

Foi requisitado no exercício que sejam feitas K partições, em que a quantidade de pontos dentro dessas partições seja aproximadamente igual, e que essas partições sejam utilizadas como uma aproximação de uma partição da função verdade.

2 Funcionamento do gerador de vetores θ

2.1 Matriz de Covariância

Para a futura geração dos candidatos no gerador, é necessário gerar uma matriz de covariância (2x2, pois um dos elementos é gerado por $1 - (\theta_1 + \theta_2)$). Como a matriz de covariância de uma Distribuição Dirichlet é conhecida, utilizou-se essa informação para fixar essa matriz, de modo que não seja necessário ajustar a matriz durante a execução do algoritmo.

2.2 Geração dos passos

Os passos são gerados pelo gerador de normal multivariada do NumPy. Para esse gerador é necessária a matriz de covariância previamente explicada. Esse gerador, como foi visto em aula, permite que a aceitação/rejeição dos candidatos seja feita de maneira mais simples.

Esse gerador normal multivariado gera um vetor inicial com tamanho $2n$, em que n é a quantidade necessária de pontos para que a precisão seja alcançada. Caso ainda seja preciso mais candidatos a serem aceitos, podem ser gerados mais vetores de tamanho n.

Empiricamente, $2n$ se mostrou mais do que o suficiente.

2.3 Aceitação dos candidatos

Inicia-se com um candidato conhecido da distribuição, no caso inicial $[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$, e calcula-se seu potencial (potencial atual). Em seguida soma-se o passo gerado pela normal multivariada, e calcula-se a função potencial dessa soma.

Verifica-se se os valores do vetor a ser avaliado são maiores que 0.

Divide-se a potencial da soma com o potencial atual. Se essa divisão for maior ou igual a 1, o candidato é aceito automaticamente. Caso essa divisão seja menor que 1, é sorteado um valor uniforme entre [0,1] e se o valor da divisão for maior que o valor da uniforme o candidato é aceito, caso não, é rejeitado.

Quando o candidato é aceito, o valor da soma é aceito como o próximo valor da distribuição (novo valor atual), seu valor potencial é armazenado e soma-se 1 ao contador.

Quando o candidato é rejeitado, mantém-se no valor atual e testa-se o próximo passo.

Repete-se o processo até atingir o valor de repetições desejado.

Neste programa, os primeiros 5.000 valores são "queimados", para "aquecer"a matriz.

3 Estimativa do erro

3.1 Partições

Foi solicitado que $W(t_j) - W(t_{j-1})$ seja aproximadamente $\frac{1}{K}$. O erro máximo admitido dessa diferença é 0,05%.

Dessa forma:

$$\frac{1}{K} \leq 0,05\%$$

$$K \geq 2.000$$

Definiu-se $K = 2.500$ a partir desse critério.

3.2 Número de elementos gerados

Para isso, considera-se que a probabilidade de um ponto qualquer recair numa partição específica é uma variável Bernoulli.

A variância de uma variável aleatória Bernoulli é dada pela multiplicação desses dois valores ($\sigma^2 = p(1-p)$). Pode ser mostrado que a variável de Bernoulli possui uma variância máxima igual a $0,25$. Assim, essa variância foi escolhida para a estimativa de pontos necessários.

Utiliza-se o Teorema Central do Limite para aproximar-se a Bernoulli por uma Distribuição Normal.

Sabe-se que o erro em estimação por intervalo, em uma normal, é dado por:

$$\epsilon = z_{\gamma/2}\frac{\sigma}{\sqrt{n}}$$

Que manipulando chega-se a:

$$n = {z_{\gamma/2}}^2\frac{\sigma^2}{\epsilon^2}$$

Substituindo $z_{\gamma/2}$ por 1,96 (95% de taxa de acerto), o valor da variância e do erro chega-se ao valor para n:

$$n = 3.841.600$$

Adotou-se:

$$n = 3.850.000$$

Essa é a quantidade necessária para o programa ficar dentro da precisão requerida 95% das vezes.

4 Funcionamento do programa

Inicialmente solicita-se ao usuário a inserção de valores para os vetores X e Y, que serão utilizados para as futuras operações.

Geram-se 3.850.000 valores da função potencial, utilizando-se o gerador previamente explicado.

Os valores da função potencial são inseridos num vetor, chamado de vetor *valores*, que é posteriormente ordenado e separado nas 2.500 partes. São colocados em um segundo vetor, chamado *vetor fronteira*, os maiores pontos de cada parte. O último ponto do vetor fronteira (o maior dos números, ou teto) é mostrado ao usuário, conforme solicitado no enunciado.

É solicitado que o usuário insira um valor v maior ou igual a zero. Esse valor é multiplicado pela constante de normalização e é buscada sua posição no vetor fronteira. Essa posição é multiplicada por 1.540 e o valor dessa multiplicação alocado em uma variável.

A posição encontrada no vetor fronteira refere-se a uma partição específica. Dessa forma, também é realizada a busca da posição do valor v nessa partição. Esta posição é somada ao valor atual da variável.

A aproximação de $W(v)$ é dada pela divisão entre o valor da variável e o valor total de pontos gerados. Essa aproximação é arredonda para 4 casas decimais e apresentada ao usuário.

5 Conclusão

Os resultados do programa, para as entradas padrão fornecidas ($x = [4, 6, 4]$ e $y = [1, 2, 3]$) foram comparadas com sucesso aos resultados esperados.

O tempo de execução é principalmente devido à aceitação e rejeição, o que demandou uma maior refino na quantidade de pontos necessários para o programa funcionar adequadamente e num tempo razoável.

Tempo considerável foi reduzido ao produzir uma quantidade alta de passos de uma vez, ainda que ocupe mais memória. Foi discutida a validade dessa ideia na monitoria. O método comumente usado não possibilita que esse truque seja

realizado, mas o método que o professor passou em sala de aula e nos slides (passos normais $[0, \Sigma]$), sim. Ambas as formas foram aceitas pelo monitor.

Numa máquina mediana, o tempo de execução não leva mais que 80 segundos, o que foi considerado bastante razoável. O método sugerido pelo Professor, em utilizar a Função Potencial e não a FDP, ou seja, trabalhar sem a constante de normalização, aumenta a velocidade de execução, sem prejuízo dos resultados finais.