MODELAGEM EM *PYTHON* PARA A ANÁLISE DA EVOLUÇÃO GENÉTICA DO VÍRUS SARS-COV-2 UTILIZANDO MODELOS DE MARKOV DE PARTIÇÃO

Palavras-Chave: Modelos de Markov com Partição, Python, Critério de Informação Bayesiano

Autores(as):

BELUCCI LEITÃO BERNARDINO, IMECC – UNICAMP

Prof(ª). Dr(ª). V.A. GONZÁLEZ-LÓPEZ, IMECC - UNICAMP

INTRODUÇÃO:

Na era contemporânea, caracterizada por uma explosão de dados digitais, a capacidade de modelar tais dados torna-se essencial em domínios tão diversos como a bioinformática e a linguística computacional. Estes modelos, desde a transcrição de um genoma viral até os padrões de navegação num website ou rede social são frequentemente governados por estados estocásticos. A emergência da pandemia de COVID-19, causada pelo vírus SARS-CoV-2, mostrou ao mundo a necessidade de ferramentas computacionais avançadas para analisar a evolução de sequências genômicas virais. Uma ferramenta clássica para decifrar esta dinâmica é a cadeia de Markov. Conforme estabelecido em textos fundamentais como o de **[2]**, a sua principal característica é a propriedade de Markov. Esta propriedade estipula que, a distribuição de probabilidade de estados futuros depende de um lapso de estados anteriores.

No entanto, a aplicação de cadeias de Markov de forma direta possui certos desafios, um deles é: o número de parâmetros, pois cresce exponencialmente com a memória do modelo. Para superar esta limitação, os **Modelos de Markov com Partição (PMM)** oferecem uma solução parcimoniosa. Como formalizado no artigo **[1]**, os **PMM** identificam os estados que são "estocasticamente equivalentes", ou seja, que partilham probabilidades de transição iguais, agrupando-os. Os autores demonstraram a consistência estatística do **Critério de Informação Bayesiano - (BIC)** para identificar a partição “mínima”, isto é, que melhor descreve o processo com o menor número.

Ademais, a principal contribuição do presente trabalho reside na tradução do conjunto de ferramentas de análise de **PMM**, originalmente desenvolvido em linguagem *R* pelo grupo de pesquisa **[3]**, para um *pipeline* computacional em *Python*. O objetivo é a implementação do código em outra linguagem, bem como a análise comparativa do **SARS-CoV-2** e suas variantes, ajudando a identificar as "regras gramaticais" estocásticas que foram conservadas ao longo de sua evolução.

METODOLOGIA:

A abordagem metodológica deste estudo consiste na implementação computacional de um *pipeline* em *Python* para a análise comparativa de genomas virais, utilizando a teoria dos **PMM**. A implementação é uma tradução funcional do conjunto de ferramentas originalmente desenvolvido em *R* pelo grupo de pesquisa **[3]**, cujo arcabouço teórico está disponível em **[1],** além de adicionar bibliotecas úteis em *Python*, como: *Numpy, itertools, pandas e matplotlib*. Dessa forma, o objetivo é criar um framework robusto que não só aprenda a "gramática" estocástica de sequências biológicas individuais, mas permite a geração de novas sequências por meio do modelo aprendido, bem como permite realizar comparações entre diferentes variantes virais, utilizando-se de todo o vasto repertório de bibliotecas existentes em *Python*.

O modelo se baseia numa busca por equivalências probabilísticas que agrupa estados que partilham o mesmo vetor de probabilidades de transição para todos os elementos do alfabeto onde a cadeia de Markov opera. Esta relação gera uma partição $L = \{L_1, L_2, \dots, L_k\}$ no espaço de estados A^M. O número de parâmetros do modelo, probabilidade de transição, é assim reduzido de $|A|^M(|A|-1)$ para $|L|(|A|-1)$. A seleção da partição ótima ou mínima é guiada pelo **Critério de Informação Bayesiano - (BIC)**:

$$BIC(L, x_1^n) = \ln\big(ML(L, x_1^n)\big) - \frac{(|A|-1)|L|}{2}\ln(n),$$

$ML(L, x_1^n)$ é a máxima verossimilhança dado uma partição L e uma amostra x_1^n

A decisão de fundir duas partes candidatas L_i e L_j é tomada através de uma métrica $d_L(i, j)$, cuja derivação e relação com o **BIC** estão detalhadas no **Corolário 2** do artigo **[1]**. A importância desta metodologia reside na sua capacidade de transformar um problema computacionalmente complexo de manipular devido ao volume de possibilidades e encontrar a "melhor" partição em um procedimento algorítmico e estatisticamente fundamentado. A ligação direta entre a função distância *dentrop* implementada no algoritmo e o critério **BIC** garante que o processo de agrupamento de estados seja um método de inferência consistente, que converge para a estrutura verdadeira do processo, para tamanhos de amostras suficientemente grandes.

O núcleo da abordagem, adquire as sequências de interesse a partir de bancos de dados como o **National Center for Biotechnology Information – NCBI** no formato **FASTA** por meio da função *obter_genoma* como consta na Figura 1, a qual acessa o site, extrai o código **FASTA** de cada vírus e filtra qualquer caractere que não pertença ao formato em questão. Em seguida, cada genoma é modelado

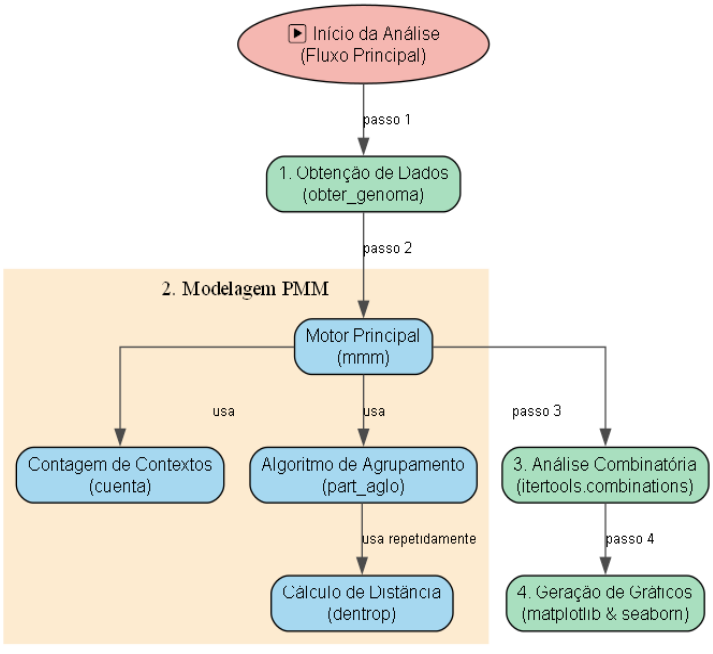

Figura 1: Fluxograma do pipeline de análise

individualmente por meio da modelagem **PMM** até que finalmente no terceiro passo, as sequências são comparadas entre si, por meio da biblioteca *itertools,* que combina todas as possibilidades entre si. Os números de acesso da *GenBank* para cada variante estarão detalhados na seção de referências.

Modelagem *PMM*: Esta é a fase central da análise, é com essa função que a estrutura estatística de cada genoma é aprendida. O processo é gerido pela função *MMM*, que orquestra um conjunto de sub-rotinas:

- **Contagem de Estados (*cuenta*):** A sequência de entrada é processada para contar a frequência de todos os estados de uma ordem definida, bem como as frequências de transição para o nucleotídeo seguinte. No caso, o alfabeto são os caracteres usados no formato **FASTA**, logo o alfabeto A = {a,c,g,t}
- **Cálculo de Distância (*dentrop*):** Esta função implementa a métrica de $d_L(i,j)$, conforme derivado no **Corolário 2** do artigo **[1]**.
- **Algoritmo de Agrupamento (*part_aglo*):** Esta função executa a estratégia de agrupamento aglomerativo hierárquico. Em cada passo, ela invoca repetidamente a *dentrop* para calcular a distância entre todos os pares de partes existentes e une o par com a menor distância global. O processo continua até que nenhuma outra fusão seja probabilisticamente vantajosa.
- **Dissimilaridade (*DBS*):** Esta função identifica o subconjunto de estados comuns a ambas as sequências e calcula uma medida de distância entre as suas distribuições de transição subsequentes, utilizando-se de *dentrop* e *cuenta*. A dissimilaridade total é definida como a média destas distâncias. Além disso, quanto mais próximo de zero mais similaridades existentes na “gramática” das sequências avaliadas.

RESULTADOS E DISCUSSÃO:

A aplicação do *pipeline* **PMM** em *Python* ocorreu em sete variantes do **SARS-CoV-2**, um conjunto de resultados que revelam estruturas estocásticas quantificáveis. Esta seção detalha os achados, focando na aplicação metodológica do **PMM**, na descoberta de regras gramaticais conservadas ao longo da evolução do vírus.

1. Aprendizagem do Modelo e Redução de Complexidade

O primeiro passo da análise consistiu na aplicação da função *MMM* a cada uma das sete sequências virais. Para cada variante, partiu-se de um espaço de estados inicial de 64 estados possíveis (correspondendo a uma memória de ordem 3, isto é, M=3. O algoritmo *part_aglo*, motor do processo de aprendizagem, reduziu drasticamente essa complexidade. Assim, os 64 estados iniciais foram agrupados em um menor número de partes finais: 9 partições para as variantes Wuhan, Alpha, Beta, Gamma, Delta e Omicron BA.1, e 8 partes para a Omicron BA.2.

Este resultado mostra a versatilidade da implementação **PMM** em *Python*, por meio da busca pela parcimônia através da identificação de equivalência estocástica **[1]** e o uso de bibliotecas consolidadas em *Python*. O agrupamento não é arbitrário; ele é governado pela métrica *dentrop*, que é uma implementação direta da teoria estatística que relaciona a fusão de estados ao **BIC**.

2. Descoberta de uma Regra Gramatical Universal

A análise comparativa entre os modelos aprendidos revelou a existência de algumas estruturas estocásticas conservadas. Através de uma análise a qual todas as sete variantes são comparadas entre si, isto é, em duplas, trios, quartetos até comparar todas as sete, de modo que são 120 subgrupos possíveis, diante todos os 120 subgrupos possíveis de variantes, foi possível identificar as "regras gramaticais", ou seja, as partes que são partilhadas entre diferentes linhagens. A Figura 2 quantifica a tal conservação. Diante disso, o resultado mais interessante é a descoberta de uma parte universalmente presente em todas as 120 combinações analisadas: o agrupamento dos estados {TAA, TAG, TAT, TCG, TGA, TGC, TGG, TTG}. A presença desta partição em todas as variantes e em todas as suas combinações significa que, do ponto de vista estocástico, estes oito estados são tratados como funcionalmente idênticos pela linguagem viral, independentemente das mutações que definem cada variante, isto é, os estados supracitados estão em uma mesma linha da Matriz Markoviana em qualquer variante do vírus.

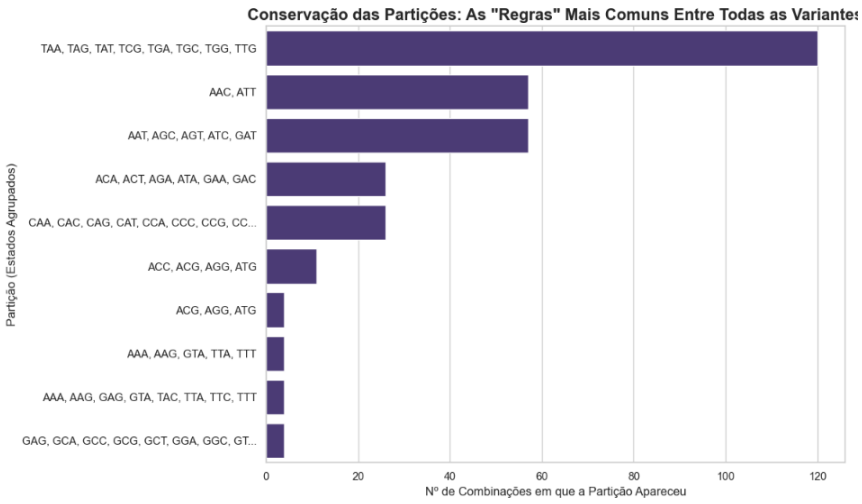

Figura 2: Frequência das partições mais comuns entre todas as variantes.

3. Quantificação da Proximidade Estocástica entre Variantes

Além de identificar regras comuns, o *pipeline* implementado permite quantificar a "distância gramatical" entre quaisquer duas sequências através da função ***DBS***. Para isso, foi colocado para comparar dois a dois, isto é, $\binom{7}{2}$ maneiras de se configurar esta análise, isto mais uma vez mostra a versatilidade da implementação em *Python*, visto que a facilidade de se fazer tais iterações advém da biblioteca *itertools*. Diante disso, a métrica de comparação entre sequências distintas é fundamentada na função *dentrop*, calculando-se depois a média. Os resultados

```python
import numpy as np
import requests
import re
import itertools

# ==========================================================================
# Bloco 1: Funções do Algoritmo PMM (Tradução do R)
# ==========================================================================

def simula(n, a, order, L, P): ...

def dentrop(nlia, nlja, n, sizeA): ...

def cuenta(x, orden, minimo): ...

def part_aglo(N, dmin, NC, Part, NL, NLA, sizeA): ...

def mmm(x, orden, minimo, dmin, method="aglo"): ...

def dbs(x, y, orden, minimo): ...

def obter_genoma(accession_id, nome_virus): ...


# ==========================================================================
# Bloco 2: Análise entre sete variantes
# ==========================================================================

if __name__ == "__main__":
    # --- PASSO 1: OBTER AS SEQUÊNCIAS DAS VARIANTES VIRAIS ---
    variantes = [
        {"nome": "Wuhan", "id": "NC_045512.2"},
        {"nome": "Alpha", "id": "OL689430.1"},
        # ... (e outras variantes)
    ]

    sequencias = {}
    for v in variantes:
        sequencias[v["nome"]] = obter_genoma(v["id"], f"SARS-CoV-2 ({v['nome']})")

    if all(sequencias.values()):
        # --- PASSO 2: APRENDER OS MODELOS PARA CADA VARIANTE ---
        modelos = {}
        for v in variantes:
            modelos[v["nome"]] = mmm(sequencias[v["nome"]], orden=3, minimo=5,
dmin=1.0)
        if all(modelos.values()):
            # --- PASSO 3: ANÁLISE DE PROXIMIDADE ENTRE VARIANTES REAIS ---
            print("\n--- ANÁLISE DE PROXIMIDADE (DBS) ---")
            # ...

            # --- PASSO 4: ANÁLISE DE VALIDAÇÃO DOS MODELOS GENERATIVOS ---
            print("\n--- ANÁLISE DE VALIDAÇÃO DO MODELO ---")
            # ...
```

Figura 3 – Versão resumida pipeline utilizado para a análise

desta análise de proximidade são os seguintes:

- **Par Mais Próximo:** Alpha e Delta (Distância: 0.000899)
- **Par Mais Distante:** Wuhan e Omicron BA.2 (Distância: 0.003228)

Uma distância baixa, como a observada entre Alpha e Delta, indica uma alta similaridade em suas gramáticas estocásticas, indicando uma relação evolutiva próxima em termos "gramaticais". Por outro lado, a distância entre a cepa original de Wuhan **[4]** e a variante Omicron BA.2 **[5]** quantifica o quão significativamente a "linguagem" estocástica do vírus se alterou ao longo do tempo.

CONCLUSÕES:

A ferramenta **PMM** oferece uma estratégia com alto potencial em futuras implementações em *Python*. Além disso, a "similaridade estocástica" ou o "parentesco gramatical" possíveis despertam o interesse em criar métricas e novos algoritmos, buscando sempre o aperfeiçoamento do modelo. Ademais, O número de partes partilhadas entre duas variantes pode ser interpretado como uma medida da sua afinidade evolutiva, isto é, um par de variantes com um grande conjunto de partes em comum opera sob um conjunto de regras probabilísticas mais semelhante, sugerindo uma relação evolutiva mais próxima, ou seja, um parentesco filogenético, sobre essa perspectiva uma abordagem distinta poderia ser tomada, como analisar sequências genômicas de grupos filogenéticos distintos para saber se possuem "parentesco gramatical" mínimo ou não.

BIBLIOGRAFIA

[1]. García, J. E., & González-López, V. A. (2017). Consistent estimation of partition Markov models. *Entropy*, *19*(4), 160.

[2.] Brémaud, P. (2020). *Markov chains: Gibbs fields, Monte Carlo simulation and queues*. Springer.

[3] Garcia, J. E., & González-López, V. A. (2016). *PMM.r*. Universidade Estadual de Campinas, Instituto de Matemática, Estatística e Ciência da Computação, Grupo de Pesquisa Dependência Estocástica, Teoria e Aplicações. http://dgp.cnpq.br/dgp/espelhogrupo/1815992558330177.

[4] Wu, F., Zhao, S., Yu, B., Chen, Y. M., Wang, W., Song, Z. G., Hu, Y., Tao, Z. W., Tian, J. H., Pei, Y. Y., Yuan, M. L., Zhang, Y. L., Dai, F. H., Liu, Y., Wang, Q. M., Zheng, J. J., Xu, L., Holmes, E. C., & Zhang, Y. Z. (2020). *Severe acute respiratory syndrome coronavirus 2 isolate Wuhan-Hu-1, complete genome*. NCBI Nucleotide Database. Accession No. NC_045512.2. https://www.ncbi.nlm.nih.gov/nuccore/NC_045512.2

[5] Xia, S., Yan, S., & Quan, L. (2025). *Severe acute respiratory syndrome coronavirus 2 isolate SARS-CoV-2/human/CHN/JMS-01/2022, complete genome*. NCBI Nucleotide Database. Accession No. PV570236.1. https://www.ncbi.nlm.nih.gov/nuccore/PV570236.1