

UNIVERSIDADE FEDERAL DO PARANÁ

JADER MAIKOL CALDONAZZO GARBELINI

APRENDIZADO NÃO SUPERVISIONADO EM GENÔMICA FUNCIONAL:
CONSTRUINDO MODELOS PROBABILÍSTICOS ATRAVÉS DA EXPLORAÇÃO DO
ESPAÇO DE K-MERS EMPREGANDO OTIMIZAÇÃO DETERMINÍSTICA PARA
DESCOBERTA DE MOTIFS BIOLÓGICOS EM LARGA ESCALA

CURITIBA PR

2024

JADER MAIKOL CALDONAZZO GARBELINI

APRENDIZADO NÃO SUPERVISIONADO EM GENÔMICA FUNCIONAL:
CONSTRUINDO MODELOS PROBABILÍSTICOS ATRAVÉS DA EXPLORAÇÃO DO
ESPAÇO DE K-MERS EMPREGANDO OTIMIZAÇÃO DETERMINÍSTICA PARA
DESCOBERTA DE MOTIFS BIOLÓGICOS EM LARGA ESCALA

Tese apresentada como requisito parcial à obtenção do grau de Doutor em Ciência da Computação no Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Aurora Trinidad Ramirez Pozo.

Coorientador: Danilo Sipoli Sanches.

CURITIBA PR

2024

DADOS INTERNACIONAIS DE CATALOGAÇÃO NA PUBLICAÇÃO (CIP)
UNIVERSIDADE FEDERAL DO PARANÁ
SISTEMA DE BIBLIOTECAS – BIBLIOTECA DE CIÊNCIA E TECNOLOGIA

Garbelini, Jader M. Caldonazzo

Aprendizado não supervisionado em genômica funcional: construindo modelos probabilísticos através da exploração do espaço de K-MERS empregando otimização determinística para descoberta de motifs biológicos em larga escala / Jader M. Caldonazzo Garbelini. – Curitiba, 2024.

1 recurso on-line : PDF.

Tese (Doutorado) - Universidade Federal do Paraná, Setor de Ciências Exatas, Programa de Pós-Graduação em Informática.

Orientador: Aurora Trinidad Ramirez Pozo

Coorientador: Danilo Sipoli Sanches

. I. Universidade Federal do Paraná. II. Programa de Pós-Graduação em Informática. III. Ramirez Pozo, Aurora Trinidad. IV. Sanches, Danilo Sipoli. V. Título.

Bibliotecário: Leticia Priscila Azevedo de Sousa CRB-9/2029

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação INFORMÁTICA da Universidade Federal do Paraná foram convocados para realizar a arguição da tese de Doutorado de **JADER MAIKOL CALDONAZZO GARBELINI** intitulada: **APRENDIZADO NÃO SUPERVISIONADO EM GENÔMICA FUNCIONAL: CONSTRUINDO MODELOS PROBABILÍSTICOS ATRAVÉS DA EXPLORAÇÃO DO ESPAÇO DE K-MERS EMPREGANDO OTIMIZAÇÃO DETERMINÍSTICA PARA DESCOBERTA DE MOTIFS BIOLÓGICOS EM LARGA ESCALA**, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.

A outorga do título de doutor está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

CURITIBA, 19 de Fevereiro de 2024.

Assinatura Eletrônica

15/03/2024 14:55:46.0

AURORA TRINIDAD RAMIREZ POZO

Presidente da Banca Examinadora

Assinatura Eletrônica

15/03/2024 13:31:39.0

ROBERTO SANTANA HERMIDA

Avaliador Externo (UNIVERSIDAD DEL PAÍS VASCO)

Assinatura Eletrônica

15/03/2024 09:51:39.0

LUCAS FERRARI DE OLIVEIRA

Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

Assinatura Eletrônica

15/03/2024 08:41:21.0

RENATO TINÓS

Avaliador Externo (UNIVERSIDADE DE SÃO PAULO)

Assinatura Eletrônica

15/03/2024 09:32:31.0

DANILO SIPOLI SANCHES

Coordenador(a) (UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ)

Dedicado a Galileu Galilei, cuja coragem de explorar o desconhecido quase culminou em sua morte. “E pur si muove.”

AGRADECIMENTOS

Em primeiro lugar, expresso minha mais profunda gratidão à minha orientadora, Aurora Trinidad Ramirez Pozo, por sua sabedoria, paciência e dedicação incansável. Seu apoio guiou este trabalho e também me ensinou a ser um pesquisador melhor e uma pessoa mais resiliente. Suas lições transcenderam o acadêmico, influenciando minha forma de ver o mundo e de enfrentar os desafios. Por tudo isso, minha admiração e meu mais sincero obrigado. Ao meu co-orientador, Professor Danilo Sipoli Sanches, agradeço por ter me indicado a este programa, abrindo a porta para uma jornada de crescimento e descobertas. Um agradecimento muito especial ao professor Elias Procópio Duarte Júnior. Suas aulas e orientações enriqueceram meu conhecimento e também moldaram a maneira como vejo a educação e a transmissão do saber. Sua paixão, dedicação e excelência no ensino são fontes de inspiração para mim, e espero um dia poder transmitir meus conhecimentos com a mesma maestria e entusiasmo. Agradeço também ao Professor André Yoshiaki Kashiwabara, a quem considero um grande amigo e colega de pesquisa. Nossas conversas foram sempre muito produtivas e se materializaram em ideias e conceitos fundamentais para meu trabalho. Estendo meus agradecimentos a todos os professores e colegas do programa. Cada um de vocês contribuiu, de maneira única, para a minha formação e para o sucesso deste trabalho. As discussões, os conselhos e os momentos compartilhados serão sempre lembrados com grande estima. Um agradecimento especial a minha esposa Rose Marye Endo, que me acompanhou nessa jornada. Seu apoio incondicional, compreensão e incentivo foram fundamentais nos momentos de dúvida e exaustão. Por fim, agradeço àqueles que, mesmo indiretamente, contribuíram para a realização deste trabalho. Cada conversa, cada gesto de incentivo e cada crítica construtiva foram peças fundamentais que me ajudaram a chegar até aqui.

RESUMO

A descoberta de padrões conservados em sequências biológicas é um grande desafio na biologia computacional e na ciência da computação. Dentro do genoma dos mais variados tipos de organismos, existem pequenas estruturas preservadas com importante função celular, conhecidas como MOTIFS. Por meio da análise dessas regiões, é possível identificar fragmentos específicos, conhecidos como sítios de ligação de proteínas. A identificação precisa desses fragmentos possibilita o desenvolvimento de medicamentos direcionados, capazes de ativar ou desativar funções específicas, o que enfraquece o agente invasor, reduzindo ou prevenindo a infecção no organismo hospedeiro. Identificar a localização destes fragmentos é um desafio complexo e ainda não resolvido na totalidade. Apesar de apresentarem certo grau de conservação, a variabilidade entre membros de uma mesma família de MOTIFS é significativa, tornando a elaboração de modelos que os descrevam de maneira precisa uma tarefa complexa. Adicionado a isso, o advento de sequenciadores de nova geração acelerou a extração de genomas inteiros, destacando-se o protocolo CHIP-SEQ neste contexto. Esta tecnologia une a imunoprecipitação da cromatina ao sequenciamento de alto rendimento, produzindo uma vasta quantidade de dados enriquecidos. Entretanto, o comprimento total dos fragmentos obtidos por esta técnica ainda supera em muito o tamanho médio das regiões conservadas. Assim, surge a necessidade de desenvolver novos algoritmos computacionais para efetuar a análise e determinar a localização destas estruturas. Neste trabalho, introduzimos o BIOMAPP::CHIP, uma arcabouço projetado para detectar fragmentos conservados em dados de CHIP-SEQ. O núcleo do BIOMAPP::CHIP é a SMT, uma estrutura de dados baseada em árvores de sufixos criada para contagem eficiente de KMERS. Além disso, o BIOMAPP::CHIP executa uma versão otimizada do algoritmo EM para aperfeiçoar e otimizar os modelos iniciais gerados pela SMT. A análise dos dados coletados a partir de uma variedade de experimentos revelou que o BIOMAPP::CHIP superou significativamente os algoritmos considerados estado da arte. Especificamente, em comparação com os principais modelos de referência do repositório JASPAR, o desempenho do BIOMAPP::CHIP foi superior, apresentando uma melhoria de aproximadamente 42% na métrica de distância EUCLIDIANA e de 51% na métrica de distância HELLINGER em relação ao segundo colocado.

Palavras-chave: K-mers. Motifs Biológicos. Descoberta de Motifs. Otimização.

ABSTRACT

The discovery of conserved patterns in biological sequences is a major challenge in computational biology and computer science. Within the genome of the most varied types of organisms, there are small preserved structures with important cellular functions, known as MOTIFS. By analyzing these regions, it is possible to identify specific fragments, known as protein binding sites. The precise identification of these fragments enables the development of targeted medicines, capable of activating or deactivating specific functions, which weakens the invading agent, reducing or preventing infection in the host organism. Identifying the location of these fragments is a complex challenge that has not yet been fully resolved. Despite showing a certain degree of conservation, the variability between members of the same family of MOTIFS is significant, making the development of models that accurately describe them a complex task. Added to this, the advent of new generation sequencers has accelerated the extraction of entire genomes, highlighting the CHIP-SEQ protocol in this context. This technology combines chromatin immunoprecipitation with high-throughput sequencing, producing a vast amount of enriched data. However, the total length of the fragments obtained by this technique still far exceeds the average size of the conserved regions. Thus, there is a need to develop new computational algorithms to perform the analysis and determine the location of these structures. In this work, we introduce BIOMAPP::CHIP, a framework designed to detect conserved fragments in CHIP-SEQ data. The core of BIOMAPP::CHIP is SMT, a suffix tree-based data structure designed to efficiently count KMERS. Additionally, BIOMAPP::CHIP runs an optimized version of the EM algorithm to refine and optimize the initial models generated by SMT. Analysis of data collected from a variety of experiments revealed that BIOMAPP::CHIP significantly outperformed state-of-the-art algorithms. Specifically, compared to the main reference models from the JASPAR repository, BIOMAPP::CHIP performed better, showing an improvement of approximately 42% in the EUCLIDEAN distance metric and of 51% in the HELLINGER distance metric in relation to second place.

Keywords: K-mers. Biological Motifs. Motif Discovery. Optimization.

LISTA DE FIGURAS

1.1	Dogma Central da Biologia Molecular na visão de James Watson. Este trabalho estuda a etapa de transcrição, destacada em verde. Fonte: o autor.	20
2.1	(a) Apresenta um exemplo com sete sequências cada uma com 32 pares de base. (b) Exibe as mesmas sequências, com o fator de transcrição CREB1 (MA0018.4) do organismo <i>H. sapiens</i> implantado em posições aleatórias. (c) Mostra o alinhamento do conjunto de dados, tomando como base os <i>motifs</i> implantados. (d) Apresenta as sub-sequências de interesse isoladamente; (e) Exibe a sequência consenso extraída do alinhamento. (f) Matriz de frequência absoluta (PFM) com pseudocontadores adicionados. (g) Matriz de frequência relativa (PPM). (h) Matriz PWM. Fonte: o autor.	30
2.2	<i>Logo</i> criada a partir do alinhamento de sequências do fator de transcrição CREB1. O eixo x representa cada posição do alinhamento e o eixo y é o valor da entropia relativa em <i>bits</i> , levando consideração uma distribuição de fundo com probabilidades uniformemente distribuídas. Fonte: o autor.	32
2.3	Representação abstrata dos métodos de melhor alinhamento e da <i>string</i> média. As duas abordagens são exponenciais. (a) O método do melhor alinhamento precisa gerar M^N combinações. (b) O método da <i>string</i> média precisa gera 4^k combinações. Fonte: o autor.	36
2.4	Numero de buscas vs número de comparações. Em vermelho temos o crescimento da função que representa a busca linear (nm) e em verde o crescimento da função que representa a busca utilizando KT's ($n + m$). Para compactar os valores, as funções de custo foram colocadas em escala log. O Gráfico foi gerado com, $m = 100$, com isso, para $n > 1.01$, as KT's passam a ser mais eficientes. Fonte: o autor.	37
2.5	(a) Dataset S contendo cinco sequências de mesma largura. (b) <i>Keyword Tree KT</i> construída a partir de S . (c) <i>Suffix Tree ST</i> construída a partir de KT . Fonte: o autor.	37
2.6	(a) Gráfico de contorno do <i>motif</i> GATA2 criado a partir do dataset MA0036.2 pertencente ao organismo <i>H. sapiens</i> . Este dataset possui largura de $t = 105$ pares de base e <i>motifs</i> com tamanho $k = 14$. Os eixos x e y representam as posições que variam de 1 até m , sendo que $m = 105 - k + 1 = 101$. A função de entropia relativa (Equação 2.4) foi empregada para calcular o score de todos os pares ordenados (x, y) , tal que $x, y \in \{1, 2, 3, \dots, m\}$. Além disso, uma distribuição de fundo uniforme foi utilizada. O ótimo global está localizado na posição $x = 51, y = 51$ e pode ser visualizado como o ponto vermelho mais escuro próximo ao centro da figura. (b) Gráfico de superfície do mesmo dataset. É interessante destacar sua alta rugosidade bem como observar seus vários ótimos locais. Fonte: o autor. . .	38
2.7	Árvore de busca para $T = 2$. Os níveis 1, 2, 3 e 4 são representados respectivamente pelas cores azul, vermelho, verde e preto. Fonte: o autor.	39

2.8	Árvore de busca para $T = 2$. Nenhum nó abaixo que possui como prefixo $(2, 1, 0, 0)$ precisará ser gerado, o que implicará em uma busca mais performática. Fonte: o autor.	40
2.9	(a) Busca bem sucedida do padrão <i>TACCA</i> . (b) Busca mal sucedida do padrão <i>GGATT</i> . Fonte: o autor.	43
2.10	Grafo totalmente conectado entre os nucleotídeos <i>A, C, G, T</i> . Fonte: (DURBIN et al., 1998).	45
2.11	Representação gráfica simplificada do algoritmo EM. As etapas <i>E-STEP</i> e <i>M-STEP</i> são executadas alternadamente até que a convergência seja atingida. Fonte: o autor.	56
2.12	Principais etapas do protocolo ChIP-seq. (a) <i>Crosslinking</i> . (b) Isolamento. (c) Imunoprecipitação. (d) Purificação. (e) <i>Crosslinking</i> reverso. (f) Sequenciamento. Fonte: o autor.	61
3.1	Exemplo da <i>String</i> de busca empregada para recuperar o rol de artigos utilizados neste trabalho. Neste caso particular, a <i>string</i> exibida está com a sintaxe formatada para a busca no banco de dados PubMed, porém, pode ser facilmente modificada para outros tipos de bancos de dados.. . . .	68
3.2	Principais abordagens empregadas no problema de descoberta de <i>motifs</i> biológicos. Os valores posicionados logo abaixo dos rótulos representam a frequência absoluta de artigos publicados na literatura levando em consideração os estudos recuperados pela <i>string</i> de busca. Fonte: o autor.	70
3.3	Fluxograma das atividades executadas na revisão sistemática. Fonte: o autor. . .	71
3.4	Taxonomia das áreas de pesquisa em <i>motifs</i> biológicos. O escopo deste trabalho reside nas áreas destacadas em vermelho ligadas pelas setas azuis. Fonte: o autor. . .	73
4.1	O BIOMAPP::CHIP é estruturado em diversos módulos que trabalham de forma interconectada. Cada módulo é responsável por uma parte específica da funcionalidade geral do pacote. O módulo UTILS contém uma coleção de funções utilitárias que auxiliam diversos outros módulos. O módulo SMT se destina ao manejo da SPARSE MOTIF TREE, incluindo sua criação e manipulação. Os módulos OOPS, ZOOPS e ANR abrigam a implementação do algoritmo EM adaptado para cada uma das respectivas distribuições de MOTIFS. Finalmente, o módulo BIOMAPP::CHIP atua como o núcleo do arcabouço, integrando todos os módulos anteriores e implementando a estrutura final do pacote. Fonte: o autor.	94
4.2	Problema de entorno. (a) Representação de uma sequência com dois kmers, κ_1 e κ_2 . A área em verde representa a área de influência de κ_1 . (b) Tabela de frequência de κ_1 , κ_2 e dos fragmentos do entorno imediato de κ_1 , com $h = 2$. É possível verificar que os fragmentos que representam o entorno de κ_1 aparecem mais vezes do que o fragmento κ_2 . Fonte: o autor.	97

4.3	Pipeline geral do BIOMAPP::CHIP. O processo começa com o pré-processamento, onde as regiões de pico são obtidas e os dados são limpos e normalizados. Em seguida, a fase de inicialização carrega o dataset principal pré-processado e as sequências de controle. Um modelo probabilístico de fundo é gerado a partir dos dados de controle, e o valor de k é estimado. A SMT é construída a partir das regiões de pico, e os k -MERS enriquecidos são extraídos da SMT. Modelos iniciais são então gerados a partir dos dados enriquecidos. Finalmente, na fase de otimização, o EM é empregado para otimizar os modelos iniciais. Fonte: o autor..	99
4.4	Estrutura de dados SMT. (a) 4-MERS empregados para construir a árvore. (b) SMT com profundidade 4. (c) SMT ^{13×6} . (d) Usuário realizando pesquisas na SMT. Fonte: o autor..	102
4.5	Exemplo de execução do D-DFS para $d = 1$ e $s = CACT$. O algoritmo encontrou a sequência sombreada em azul e alterou o estado da variável RESP para TRUE. Fonte: o autor..	106
4.6	(a) Tempo teórico de execução do KDIVE para $n = 1000$ strings e $d = 0, 1, 2, 3, \dots, 20$ mismatches. (b) Tempo teórico de execução do D-DFS para $d = 5$ mismatches e $n = 0, 1, 2, 3, \dots, 1000$ strings. Fonte: o autor..	107
4.7	Funcionamento do algoritmo IUPAC SEARCH. A Figura (a) exibe o padrão no formato IUPAC, no qual R e Y significam purina e pirimidina respectivamente. (b) Em verde, temos o caminho realizado pelo algoritmo na SMT para recuperar todos os padrões que correspondentes. Fonte: o autor..	111
4.8	Representação abstrata dos modelos OOPS, ZOOPS e ANR, no qual os círculos vermelhos representam o MOTIF conservado e os círculos pretos representam a distribuição de fundo. (a) No modelo OOPS, é possível observar que em cada sequência, existe exatamente uma cópia do círculo vermelho. (b) No modelo ZOOPS, é possível observar que em cada sequência, existe zero ou uma cópia do círculo vermelho. (c) No modelo ANR, é possível observar que em cada sequência, existe um número arbitrário de cópias do círculo vermelho. Fonte: o autor..	113
4.9	Representação gráfica do modelo OOPS. No painel (a), postulamos que, a priori, cada posição ρ tem a capacidade de contribuir de maneira equitativa para a presença ou ausência do fragmento conservado. No painel (b), apresentamos uma modelagem gráfica elucidativa, demonstrando a abordagem do modelo em lidar com um fragmento situado em uma posição específica, denotada como ρ . Fonte: o autor..	115
4.10	Representação gráfica do modelo ZOOPS. Na parte superior temos duas “moedas” simbolizando as probabilidades w_1 e $w_2 = 1 - w_1$. Essas probabilidades representam a plausibilidade de cada modelo (OOPS e ZOOPS) e são parâmetros que, além do modelo α , também devem ser estimados pelo algoritmo. No painel (a) temos o conhecido modelo OOPS e no painel (b) o modelo ZOOPS: <i>Zero Occurrences per Sequence</i> . Fonte: o autor..	119

4.11	Representação gráfica do modelo ANR. Na parte superior da figura temos w_α e w_β que simbolizam as probabilidades a priori das componentes da mistura, sendo que ρ_α (retângulo vermelho) representa a distribuição de interesse e ρ_β (retângulo azul) a distribuição de controle. Ambas devem ser estimadas através do algoritmo. (a) Modelo positivo que representa a distribuição de interesse α que também deve ser estimada. (b) Modelo negativo que representa a distribuição β , construída a partir da extração de sequências genômicas sem o fragmento de interesse ou através embaralhamento do conjunto de dados principal. Fonte: o autor.	121
4.12	Pipeline do algoritmo de otimização do BIOMAPP-CHIP. Na etapa (A), as sementes iniciais fornecidas pela SMT são transformadas em modelos iniciais pré-otimizados que por sua vez são submetidos na etapa (B) ao EM para serem aperfeiçoados. Fonte: o autor.. . . .	122
5.1	Comparação do desempenho dos modelos RANDOM, RANDOM+EM, SMT e SMT+EM em relação ao modelo de referência. Cada BOXPLOT representa a distribuição das distâncias obtidas por cada algoritmo, com a linha central da caixa indicando a mediana. Os bigodes estendem-se até o ponto mais distante dentro do intervalo interquartil, e os pontos representam outliers. Os resultados evidenciam a superioridade dos modelos SMT e SMT+EM, com medianas significativamente mais baixas e menor dispersão em comparação aos modelos RANDOM e RANDOM+EM, indicando uma maior precisão e consistência na aproximação do modelo de referência. Fonte: o autor.	124
5.2	Distribuição da distância ao modelo de referência para os modelos RANDOM, RANDOM+EM, SMT e SMT+EM em relação à quantidade de bases, segmentada por níveis de mutação (0.1, 0.15, 0.2, 0.25, 0.3) e tamanho de k (indicado pela coloração dos pontos, variando de azul para vermelho conforme o incremento do tamanho de k). Cada subfigura corresponde a um nível de mutação específico, demonstrando como cada algoritmo se comporta em diferentes contextos de complexidade e perturbação. Os modelos SMT e SMT+EM mostram uma tendência de menores distâncias ao modelo de referência em todos os níveis de mutação. A partir do nível de mutação de 0.2, observa-se uma elevação na dispersão para conjuntos de dados menores e valores de k maiores, refletindo o aumento na dificuldade de otimização sob essas condições. Fonte: o autor.	125
5.3	Análise comparativa entre os algoritmos SMT, JELLYFISH e KANALYZE em relação ao consumo de RAM (em megabytes) e tempo de execução (em segundos). (a) Consumo médio de RAM. (b) Tempo médio de execução. (c) Distribuição do consumo de RAM, com mediana e outliers. (d) Distribuição do tempo de execução, com mediana e outliers. SMT prova ser mais eficiente em ambos os aspectos em comparação com os outros dois algoritmos. Fonte: o autor.	129
5.4	Análise comparativa entre os algoritmos SMT, JELLYFISH e KANALYZE em relação ao consumo de RAM (em megabytes) e tempo de execução (em segundos) para vários tamanhos de k . Fonte: o autor.	130
5.5	Análise comparativa entre os gráficos de boxplot dos algoritmos SMT, JELLYFISH e KANALYZE em relação ao consumo de RAM (em megabytes), agrupados por valores de k . Fonte: o autor.	131

5.6	Análise comparativa entre os gráficos de boxplot dos algoritmos SMT, JELLYFISH e KANALYZE quanto ao consumo de tempo (em segundos), agrupados por valores de k . Fonte: o autor.	132
5.7	Comparação de desempenho entre os algoritmos BIOMAPP-CHIP, MEME, HOMER e PROSAMPLER, com foco em tempo (medido em segundos) e memória (medida em Mbytes) consumo. MEME se destacou pelo uso eficiente de memória, enquanto BIOMAPP-CHIP foi o mais rápido em termos de tempo de execução. Fonte: o autor.	139
5.8	Variação do tempo de execução em relação ao k para os algoritmos BIOMAPP::CHIP, MEME, HOMER e PROSAMPLER. Observa-se que o algoritmo BIOMAPP::CHIP apresenta vantagem em termos de eficiência de tempo, seguido pelo algoritmo MEME. Fonte: o autor.. . . .	139
5.9	Variação do consumo de memória RAM em relação ao k para os algoritmos BIOMAPP::CHIP, MEME, HOMER e PROSAMPLER. Observa-se que o algoritmo MEME possui vantagem em termos de eficiência espacial, seguido pelo algoritmo BIOMAPP::CHIP. Fonte: o autor.	140
5.10	Comparação do desempenho entre algoritmos BIOMAPP, MEME, HOMER e PROSAMPLER em termos de tempo de execução e consumo de memória RAM. Dados sintéticos foram utilizados para avaliar a eficiência e a escalabilidade dos algoritmos. Os pontos representam a inter-relação entre a quantidade de RAM utilizada (eixo x) e o tempo necessário para completar a execução (eixo y), com cada cor correspondendo a um algoritmo específico. Fonte: o autor.	141
5.11	Comparação da média geral de memória RAM (medida em Mbytes) e consumo de tempo de execução (medido em segundos) entre diferentes algoritmos aplicados a conjuntos de dados reais. Observa-se que BIOMAPP:CHIP lidera em consumo de tempo e ocupa a segunda posição em consumo de RAM. MEME apresenta o menor consumo de RAM, enquanto PROSAMPLER, apesar de seu desempenho inferior em testes sintéticos, apresenta uma melhoria significativa, ficando em segundo lugar no consumo de memória. O algoritmo HOMER não se destaca em nenhum dos critérios. Fonte: o autor.	142
5.12	Comparação entre o desempenho médio dos algoritmos em relação ao consumo de tempo agrupado por valor de k . É possível verificar que BIOMAPP::CHIP, PROSAMPLER e MEME apresentam os tempos mais curtos enquanto HOMER apresentou tempos de execução significativamente maiores. Fonte: o autor.	142
5.13	Comparação entre o desempenho médio dos algoritmos em relação ao consumo de memória RAM agrupado pelo valor k . É possível verificar que BIOMAPP::CHIP, PROSAMPLER e MEME apresentam os tempos mais curtos enquanto HOMER apresentou tempos de execução significativamente maiores. Fonte: o autor.	144
5.14	Comparação do desempenho entre algoritmos BIOMAPP, MEME, HOMER e PROSAMPLER em termos de tempo de execução e consumo de memória RAM. Dados reais extraídos do repositório JASPAR foram empregados para avaliar a eficiência e a escalabilidade dos algoritmos. Os pontos representam a inter-relação entre a quantidade de RAM utilizada (eixo x) e o tempo necessário para completar a execução (eixo y), com cada cor correspondendo a um algoritmo específico. Fonte: o autor.. . . .	144

5.15	Análise comparativa dos algoritmos BIOMAPP::CHIP, MEME, PROSAMPLER e HOMER usando diversas métricas de distância e coeficientes de correlação. As métricas empregadas para comparação incluem distância EUCLIDIANA, distância MANHATTAN, distância HELLINGER, coeficiente de correlação PEARSON, coeficiente SANDELIN-WASSERMAN e BHATTACHARYYA coeficiente. Nota-se que, ao considerar a média geral, o algoritmo BIOMAPP apresentou desempenho superior em relação aos demais algoritmos avaliados. Fonte: o autor.	145
5.16	Análise comparativa de algoritmos baseada na evolução das métricas em função da variável k. Fonte: o autor.	146
5.17	Comparação do desempenho dos quatro algoritmos em termos de consumo de tempo vs precisão, medida pela distância EUCLIDIANA do modelo de referência. Os pontos representam a performance individual de cada execução dos algoritmos BIOMAPP (azul), HOMER (amarelo), MEME (vermelho) e PROSAMPLER (verde). Fonte: o autor.. . . .	147
5.18	Comparação do desempenho dos quatro algoritmos em termos de consumo de RAM vs precisão, medida pela distância EUCLIDIANA do modelo de referência. Os pontos representam a performance individual de cada execução dos algoritmos BIOMAPP (azul), HOMER (amarelo), MEME (vermelho) e PROSAMPLER (verde).Fonte: o autor.	148
5.19	Os resultados dos P-VALUES derivados do teste NEMENYI indicam a presença de diferenças estatísticas significativas entre o programa BIOMAPP::CHIP e as demais abordagens testadas em todas as métricas avaliadas. Adicionalmente, nota-se que MEME não apresenta diferenças estatísticas significativas em relação a HOMER exceto na métrica PCC. Neste último, tanto MEME quanto PROSAMPLER foram estatisticamente semelhantes. Fonte: o autor.	149
5.20	Comparação da distância crítica entre diferentes abordagens. A análise revela que o algoritmo BIOMAPP::CHIP se destaca, apresentando desempenho superior em relação às demais metodologias avaliadas. Fonte: o autor.. . . .	150

LISTA DE TABELAS

2.1	(a) Modelo de Markov de ordem 1 construído a partir de dados extraídos de ilhas CpG do organismo <i>H.sapiens</i> . (b) Modelo de Markov de ordem 1 construído a partir de dados extraídos de regiões aleatórias do organismo <i>H.sapiens</i>	45
2.2	Matriz <i>log-odds ratio</i> construída a partir do cálculo do logaritmo resultante da divisão entre a matriz positiva e a matriz negativa.	46
2.3	(a) Matriz de frequência absoluta. (b) Matriz de frequência relativa normalizada por coluna. (c) Modelo PWM.	49
2.4	(a) Cálculo de $S(x = ACCACTCAAAA)$. (b) Cálculo de $S(y = CTGCGGGCGTG)$. Os caminhos em azul e vermelho mostram as células selecionadas para realizar os dois cálculos respectivamente.. . . .	49
2.5	Cálculo das probabilidades utilizando o modelo representado pelas Equações 2.45, 2.46 e 2.47.	57
2.6	Cálculo das probabilidades utilizando o modelo representado pelas Equações 2.55, 2.56, 2.57 e 2.58.	59
3.1	Artigos utilizados como <i>baseline</i> para composição da <i>string</i> de busca.	68
3.2	Quantidade de resultados encontrados por cada fonte de pesquisa.	69
3.3	Crerios de inclusão e exclusão aplicados na pesquisa bibliográfica.	70
3.4	Artigos utilizados na revisão sistemática ordenados do mais antigo para o mais recente.	72
5.1	Resumo dos resultados do teste de FRIEDMAN e do teste post-hoc de NEMENYI para diferentes nveis de mutação.	127
5.2	Análise comparativa do tempo médio e consumo de memória RAM agrupados por valores de k	133
5.3	Resultados do teste de FRIEDMAN com post-hoc de NEMENYI para a variável tempo agrupada por k , Legenda: (+) SMT ganhou, (-) SMT perdeu, (=) não houve diferença significativa entre o primeiro colocado e os demais.. . . .	134
5.4	Resultados do teste de FRIEDMAN com post-hoc de NEMENYI para a variável RAM agrupada por valores k , Legenda: (+) SMT ganhou, (-) SMT perdeu, (=) não houve diferença significativa entre o primeiro colocado e os demais.. . . .	135
5.5	Quantidade de regiões de pico extraídas de diversos experimentos disponibilizados no JASPAR 2022. Nota-se de maneira clara que o protocolo CHIP-SEQ é responsável pela produção do volume mais expressivo de dados.	137
5.6	Parâmetros configurados para os algoritmos utilizados na pesquisa. Cada algoritmo é listado junto aos seus respectivos parâmetros para garantir a reprodutibilidade dos experimentos.	137
5.7	Comparação do consumo de memória (medido em Mbytes) e tempo (medido em segundos) entre os algoritmos BIOMAPP::CHIP, HOMER, MEME e PROSAMPLER. . . .	141

5.8	Comparação do consumo de memória (medido em Mbytes) e tempo (medido em segundos) entre os algoritmos BIOMAPP::CHIP, HOMER, MEME e PROSAMPLER agrupados por k.	143
5.9	Análise comparativa entre os algoritmos em diversas métricas de distância e correlação. A linha amarela destaca o vencedor de cada métrica em termos de média.	146
5.10	Resultados dos testes FRIEDMAN e NEMENYI para avaliação de algoritmos de descoberta de MOTIVOS, Cada linha representa uma métrica avaliada, As colunas χ^2 e p – value correspondem aos resultados do teste de FRIEDMAN. As colunas relativas a HOMER, MEME e PROSAMPLER exibem os P-VALUES obtidos pelo teste NEMENYI, todos comparados com o programa BIOMAPP::CHIP. A última coluna, chamada RESULTADO, indica se o BIOMAPP teve um desempenho melhor (+), inferior (-) em comparação com as outras abordagens ou (=) não estatisticamente significativo.. . . .	149

LISTA DE ACRÔNIMOS

AA	Aminoácido (<i>Amino Acid</i>)
ANR	Qualquer Número de Repetições (<i>Any Number of Repetitions</i>)
APM	Correspondência de Padrões Aproximada (<i>Approximate Pattern Matching</i>)
ATAC-SEQ	Sequenciamento de Cromatina Acessível por Transposase (<i>Transposase-Accessible Chromatin Sequencing</i>)
BLAST	Ferramenta de Busca de Alinhamento Local Básico (<i>Basic Local Alignment Search Tool</i>)
BW	Algoritmo Backward (<i>Backward Algorithm</i>)
CHIP-SEQ	Sequenciamento de Imunoprecipitação de Cromatina (<i>Chromatin Immunoprecipitation Sequencing</i>)
CREB1	Proteína de Resposta ao Elemento de Ligação ao AMP Cíclico 1 (<i>cAMP Response Element-Binding Protein 1</i>)
CRM	Módulo Regulatório Cis (<i>Cis-Regulatory Module</i>)
CRP	Proteína Receptora de cAMP (<i>cAMP Receptor Protein</i>)
CPU	Unidade Central de Processamento (<i>Central Processing Unit</i>)
DH	Distância de Hamming (<i>Hamming Distance</i>)
DKL	Divergência de Kullback-Leibler (<i>Kullback-Leibler Divergence</i>)
DNA	Ácido Desoxirribonucleico (<i>Deoxyribonucleic Acid</i>)
E-STEP	Etapa E (<i>E-Step</i>)
EM	Expectativa-Maximização (<i>Expectation-Maximization</i>)
ENA	Arquivo Europeu de Nucleotídeos (<i>European Nucleotide Archive</i>)
EPM	Correspondência de Padrões Exata (<i>Exact Pattern Matching</i>)
EUC	Distância Euclidiana (<i>Euclidian Distance</i>)
FW	Algoritmo Forward (<i>Forward Algorithm</i>)
GATA2	Fator de Transcrição GATA2 (<i>GATA binding protein 2</i>)
GPU	Unidade de Processamento Gráfico (<i>Graphics Processing Unit</i>)
HELL	Distância de Hellinger (<i>Hellinger Distance</i>)
H3K4ME1	Histona H3 Lisina 4 Monometilação (<i>Histone H3 Lysine 4 Mono-methylation</i>)
HINDII	Enzima de Restrição HindII (<i>HindIII restriction enzyme</i>)
HIV	Vírus da Imunodeficiência Humana (<i>Human Immunodeficiency Virus</i>)
HT-SELEX	SELEX de Alto Desempenho (<i>High-Throughput SELEX</i>)
IC	Conteúdo de Informação (<i>Information Content</i>)

IEEE	Instituto de Engenheiros Elétricos e Eletrônicos (<i>Institute of Electrical and Electronics Engineers</i>)
KL	Divergência de Kullback-Leibler (<i>Kullback-Leibler Divergence</i>)
KTREE	Árvore de Palavras-Chave (<i>Keyword Tree</i>)
M-STEP	Etapa M (<i>M-Step</i>)
MAN	Distância Manhattan (<i>Manhattan Distance</i>)
MCMC	Cadeias de Markov Monte Carlo (<i>Markov Chain Monte Carlo</i>)
MFM	Modelos Finitos de Mistura (<i>Mixture Finite Models</i>)
MRNA	RNA Mensageiro (<i>Messenger RNA</i>)
NCBI	Centro Nacional de Informação Biotecnológica (<i>National Center for Biotechnology Information</i>)
NF-KB	Fator Nuclear Kappa B (<i>Nuclear Factor Kappa B</i>)
NKX2-5	Fator de Transcrição NK2 Homeobox 5 (<i>NK2 Homeobox 5 Transcription Factor</i>)
NLM	Biblioteca Nacional de Medicina (<i>National Library of Medicine</i>)
NP	Tempo Polinomial Não-Determinístico (<i>Non-Deterministic Polynomial-Time</i>)
OOPS	Uma ocorrência Por Sequência (<i>One Occurrence Per Sequence</i>)
PCC	Coeficiente de Correlação de Pearson (<i>Pearson Correlation Coefficient</i>)
PFM	Matriz de Frequência de Posição (<i>Position Frequency Matrix</i>)
PMC	Centro de Publicações PubMed (<i>PubMed Central</i>)
PPM	Matriz de Probabilidade de Posição (<i>Position Probability Matrix</i>)
PSSM	Matriz de Pontuação de Sequência de Posição Específica (<i>Position Specific Scoring Matrix</i>)
PWM	Matriz de Peso de Posição (<i>Position Weight Matrix</i>)
POI	Proteína de Interesse (<i>Protein of Interest</i>)
RAM	Memória de Acesso Aleatório (<i>Random Access Memory</i>)
RNA	Ácido Ribonucleico (<i>Ribonucleic Acid</i>)
SAB	Anticorpo Específico (<i>Specific Antibody</i>)
SELEX	Seleção Sistemática de Ligantes por Enriquecimento Exponencial (<i>Systematic Evolution of Ligands by Exponential Enrichment</i>)
SMT	Árvore de Motif Esparsa (<i>Sparse Motif Tree</i>)
STREE	Árvore de Sufixo (<i>Suffix Tree</i>)
SW	Algoritmo de Sandelin-Wasserman (<i>Sandelin-Wasserman Algorithm</i>)
TCM	Dois Componentes de Misturas (<i>Two Component Mixtures</i>)
TF	Fator de Transcrição (<i>Transcription Factor</i>)

TFBS	Sítio de Ligação do Fator de Transcrição (<i>Transcription Factor Binding Site</i>)
WAM	Modelo de matriz de peso (<i>Weight Array Model</i>)
ZOOPS	Zero ou uma ocorrência por sequência (<i>Zero or One Occurrence Per Sequence</i>)

SUMÁRIO

1	INTRODUÇÃO	20
1.1	MOTIVAÇÃO	22
1.2	PROBLEMA DE PESQUISA	23
1.3	HIPÓTESE	25
1.4	OBJETIVO	25
1.5	CONTRIBUIÇÕES	25
1.6	ORGANIZAÇÃO DO TEXTO	26
2	FUNDAMENTOS	27
2.1	O QUE SÃO MOTIFS	27
2.2	DEFINIÇÃO DO PROBLEMA	29
2.3	DIFERENTES ABORDAGENS	36
2.3.1	Algoritmos enumerativos	38
2.3.2	Algoritmos probabilísticos	44
2.4	CHIP-SEQ	60
2.5	CONCLUSÃO	65
3	REVISÃO BIBLIOGRÁFICA	67
3.1	METODOLOGIA UTILIZADA	67
3.1.1	Grupo controle e <i>string</i> de busca	67
3.1.2	Fontes de pesquisa	68
3.1.3	Critérios de seleção	69
3.2	TRABALHOS RELACIONADOS	71
3.2.1	Algoritmos enumerativos	75
3.2.2	Algoritmos probabilísticos	77
3.2.3	Algoritmos inspirados na natureza	81
3.2.4	Algoritmos híbridos	84
3.3	ANÁLISE CRÍTICA DOS TRABALHOS RELACIONADOS	86
3.4	DESAFIOS	88
3.5	CONCLUSÃO	90
4	MATERIAL E MÉTODOS	92
4.1	ESTRATÉGIAS PARA SUPERAR OS DESAFIOS IDENTIFICADOS NO CAPÍTULO ANTERIOR	92
4.1.1	MOTIFS sutis e o problema (K,D)-MOTIF	92
4.1.2	Análise de dados derivados do protocolo CHIP-SEQ	93
4.1.3	Soluções sub-ótimas	93

4.2	BIOMAPP-CHIP	94
4.2.1	Inicialização	99
4.2.2	Otimização	113
4.3	CONCLUSÃO	122
5	EXPERIMENTOS E RESULTADOS	123
5.1	RESPONDENDO A HIPÓTESE INICIAL	123
5.1.1	Análise estatística	126
5.2	SMT	128
5.2.1	Análise estatística	133
5.3	BIOMAPP::CHIP	136
5.3.1	Dados sintéticos	136
5.3.2	Dados reais	136
5.3.3	Parâmetros	137
5.3.4	Biomapp::chip em dados sintéticos	138
5.3.5	Biomapp::chip em dados reais	140
5.3.6	Análise estatística	148
5.4	CONCLUSÃO	150
6	CONCLUSÃO E CONSIDERAÇÕES FINAIS	152
6.1	CONCLUSÕES DE PESQUISA	152
6.2	PUBLICAÇÕES DECORRENTES DO DOUTORADO	153
	REFERÊNCIAS	154

1 INTRODUÇÃO

O controle da expressão gênica desempenha um papel fundamental nos processos biológicos. A busca por um entendimento mais profundo das redes que governam esse mecanismo complexo e outros correlatos está apenas no início. Compreender esse intricado processo é fundamental para o desenvolvimento de tratamentos efetivos para diversas disfunções, tais como cânceres, doenças autoimunes e diversas outras enfermidades (Alon, 2007).

Devido à sua importância, os mecanismos subjacentes à regulação gênica vêm sendo largamente estudados ao longo dos últimos 40 anos (Ptashne e Hopkins, 1968; Jacob e Monod, 1961). De acordo com o Dogma Central da Biologia Molecular, postulado por Crick (1970) e aceito até os dias atuais, a informação flui do DNA para a proteína e não pode retornar. Já para James Watson (Watson, 2012), o Dogma Central se resume na transferência de informação no seguinte sentido: $\text{DNA} \rightarrow \text{RNA} \rightarrow \text{proteína}$. Embora existam algumas inconsistências na declaração feita por Watson, ela é útil para compreendermos como as proteínas são sintetizadas e, consequentemente, como a descrição fenotípica de um organismo é formada. A Figura 1.1 descreve o Dogma Central através da visão de Watson (2012). Contudo, é importante notar que, com os avanços recentes em epigenética, alguns pesquisadores começaram a questionar este dogma. Os mecanismos epigenéticos, incluindo modificações do DNA e histonas, bem como a regulação por RNA não codificante, demonstram que a expressão gênica pode ser influenciada por fatores que não alteram a sequência do DNA, sugerindo um fluxo de informação mais complexo e dinâmico do que o previamente entendido. Essa nova compreensão desafia a visão unidirecional do Dogma Central, indicando que a regulação gênica envolve não apenas a transferência linear de informação, mas também processos reversíveis e interações multidirecionais que modulam a atividade gênica sem alterar a sequência de DNA subjacente.

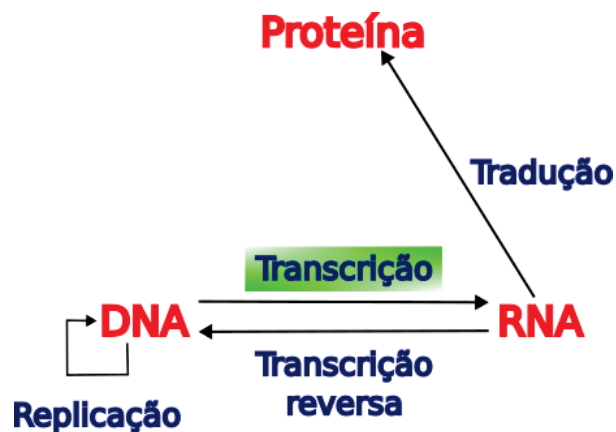


Figura 1.1: Dogma Central da Biologia Molecular na visão de James Watson. Este trabalho estuda a etapa de transcrição, destacada em verde. Fonte: o autor.

A Figura 1.1 exibe as várias etapas presentes na expressão gênica. Uma delas se chama transcrição e está destacada em verde. A compreensão desta etapa é de extrema importância para o desenvolvimento de algoritmos para análise e descoberta de MOTIFS sendo largamente abordada no decorrer deste trabalho. Basicamente, a transcrição pode ser entendida como a cópia de uma parte específica do DNA chamada de gene para uma molécula de RNA. Para que essa cópia seja iniciada, proteínas reguladoras se ligam a locais específicos chamados sítios de ligação dos fatores de transcrição (TFBS, do inglês *Transcription Factors Binding Sites*). Os TFBS's são

pequenas regiões conservadas, localizadas na região promotora do gene no qual se presume ter algum significado biológico.

É importante destacar que a representação tradicional do Dogma Central da biologia molecular, consiste na transcrição da sequência de DNA para RNA, e na tradução do RNA para proteína, sem considerar a transcrição reversa de RNA para DNA. No entanto, com a descoberta da transcriptase reversa na década de 1970, nosso entendimento do fluxo de informação genética expandiu-se para incluir também esse mecanismo. A transcriptase reversa, encontrada em retrovírus como o HIV, catalisa a formação de DNA a partir de um molde de RNA. Assim, a presença da transcriptase reversa na Figura 1.1 reflete a compreensão moderna do fluxo de informação genética, no qual inclui tanto a sequência de transcrição e tradução tradicionalmente aceita, quanto a transcrição reversa mediada pela transcriptase reversa. Embora a transcrição reversa seja uma exceção ao Dogma Central original, seu papel biológico fundamental em certos contextos, como a replicação viral, é inequívoco (Temin et al., 1970).

Os processos regulatórios que envolvem a transcrição gênica afetam diretamente o sistema biológico dos organismos através da tradução ou repressão de diversas proteínas com as mais variadas funções. A título de exemplo, organismos eucariotos superiores possuem um sofisticado sistema regulatório que depende principalmente da arquitetura de suas redes de regulação e não da quantidade de proteínas expressas. Um caso clássico é o do nematódeo *C. elegans*, que possui apenas 959 células somáticas e cerca de 20.000 genes. Enquanto isso, o organismo *H. sapiens* contém por volta de trilhões de células somáticas com “apenas” o dobro da quantidade de genes de seu parceiro menos evoluído, pouco mais de 40.000 (Zvelebil e Baum, 2007).

Para que seja possível detectar MOTIFS com precisão, é necessário empregar recursos pertencentes aos domínios da ciência da computação, biologia, matemática e estatística. Abordagens puramente biológicas, sem associação com serviços de tecnologia e algoritmos, não conseguem dar uma resposta satisfatória para o problema. Da mesma forma, a criação de algoritmos utilizando modelos matemáticos sofisticados com um conhecimento marginal dos conceitos biológicos subjacentes e das tecnologias de sequenciamento envolvidas também são insuficientes.

Por um lado, abordagens puramente biológicas podem identificar padrões recorrentes em sequências genéticas ou proteicas, mas sem a capacidade de análise e modelagem computacional avançada, a identificação de MOTIFS pode ser extremamente desafiadora, especialmente em larga escala. A biologia fornece o contexto essencial e a validação funcional, mas sem o suporte de algoritmos sofisticados e processamento de dados em grande escala, o potencial para descoberta fica limitado.

Por outro lado, a aplicação de modelos matemáticos e algoritmos de computação sem uma compreensão profunda dos conceitos biológicos e das nuances dos dados de sequenciamento pode levar a interpretações incorretas ou a uma relevância biológica questionável das descobertas. A matemática e a estatística oferecem as ferramentas para analisar e modelar os dados, enquanto a ciência da computação possibilita o processamento e a análise de grandes volumes de informação genômica. No entanto, sem a integração do conhecimento biológico, essas análises podem não alcançar seu pleno potencial na identificação de padrões significativos e na compreensão de sua função.

Portanto, a sinergia entre estas disciplinas não é apenas benéfica, mas essencial para o avanço na descoberta de MOTIFS biológicos. A colaboração interdisciplinar permite o desenvolvimento de métodos computacionais que são tanto computacionalmente eficientes quanto biologicamente significativos, garantindo que os padrões identificados tenham relevância funcional e contribuam para nossa compreensão dos processos biológicos. Esse enfoque

integrado é fundamental para enfrentar os desafios da biologia moderna e para impulsionar novas descobertas no campo da genômica e da biologia de sistemas.

A interação entre os fatores de transcrição (TF, do inglês *Transcription Factors*) e os seus respectivos sítios de ligação são temporárias, o que dificulta a sua detecção. Além disso, *in vivo*, um mesmo TF pode se ligar a vários locais ou a um mesmo sítio sob diferentes condições, o que leva a ativação ou repressão do gene regulado. Por essa razão, uma combinação entre experimentos biológicos e computacionais é essencial para entender a regulação transcricional (Liang et al., 2007).

Como exemplo, podemos citar a imunoprecipitação da cromatina (CHIP, do inglês *Chromatin Immuno Precipitation*), tecnologia desenvolvida em meados dos anos 2000 que se mostrou muito bem sucedida para localizar os sítios de ligação dos fatores de transcrição *in vivo*. Esta técnica, empregada em conjunto com o sequenciamento de nova geração (*next-generation sequencing*), como *Solexa/Illumina* (Quail et al., 2008; Illumina, 2015), recebe o nome de CHIP-SEQ (vide Capítulo 2).

Nesta abordagem, dezenas de milhões de sequências chamadas de READS, são mapeadas no genoma. É necessário corrigir o ruído de fundo e normalizar as distribuições de densidade das READS para destacar as regiões de picos, que são consideravelmente enriquecidas e que abrangem os locais de ligação dos TF's. Essas regiões possuem entre 100-300bp¹, uma resolução muito maior do que o tamanho médio de um MOTIF, que fica entre 5-25bp. Portanto, é necessário que técnicas computacionais sejam utilizadas para encontrar da forma mais precisa possível os locais de ligação dentro das milhões de regiões de pico excessivamente largas retornadas pelo experimento de CHIP-SEQ.

1.1 MOTIVAÇÃO

No contexto da Ciência da Computação, a criação de algoritmos especializados na identificação de MOTIFS biológicos representa uma contribuição significativa para o campo da biologia computacional e também introduz desafios e oportunidades únicas para a área de algoritmos e teoria da computação. Abordagens algorítmicas eficazes para este problema podem, por exemplo, influenciar o desenvolvimento de métodos computacionais em outras disciplinas, promovendo um impacto transdisciplinar. Além disso, os desafios intrínsecos, como a necessidade de manipular grandes volumes de dados e abordar a variabilidade biológica, oferecem novas questões de pesquisa em otimização, aprendizado de máquina e análise de dados, tais como:

- Avanços em técnicas de otimização: a busca por fragmentos conservados em grandes conjuntos de dados é um problema complexo de otimização. Desenvolver algoritmos eficientes para identificar MOTIFS pode levar a melhorias nas técnicas de otimização, como algoritmos evolutivos, meta-heurísticas e aprendizado de máquina, que têm aplicações em uma ampla gama de problemas na ciência da computação e em outros campos.
- Desenvolvimento de algoritmos de aprendizado de máquina e inteligência artificial: a descoberta de MOTIFS biológicos pode ser enquadrada como um problema de aprendizado supervisionado e/ou não supervisionado. Desenvolver algoritmos eficientes e precisos para esse problema pode contribuir para avanços em algoritmos de aprendizado de máquina e técnicas de inteligência artificial.

¹Diferente da área de comunicação de dados, na bioinformática bp não significa bits por segundo e sim pares de base.

- Lidar com ruído e incerteza nos dados: os dados biológicos são frequentemente ruidosos e incompletos, o que apresenta desafios para o desenvolvimento de algoritmos robustos. Criar algoritmos capazes de lidar com esses desafios pode ter implicações em outras áreas da ciência da computação que também lidam com dados ruidosos e incertos.
- Análise de dados em grande escala e computação de alto desempenho: a análise de grandes conjuntos de dados biológicos requer algoritmos eficientes e escaláveis que possam tirar proveito de recursos computacionais avançados, como clusters de computadores e GPUs. Desenvolver esses algoritmos pode levar a avanços no campo da computação de alto desempenho e no desenvolvimento de técnicas para processamento e análise de grandes volumes de dados.
- Integração interdisciplinar: o desenvolvimento de algoritmos para a descoberta de MOTIFS biológicos incentiva a colaboração entre cientistas da computação e biólogos, levando a uma melhor compreensão mútua e integração de conhecimentos entre as disciplinas. Essa integração pode levar a novas ideias e soluções inovadoras para problemas em biologia e também em outros campos que enfrentam desafios semelhantes.

1.2 PROBLEMA DE PESQUISA

A análise computacional dos sítios de ligação invariavelmente começa com a obtenção de dados experimentais. A qualidade dos dados e a natureza da análise subsequente dependem tanto do experimento conduzido quanto da tecnologia utilizada em sua extração, que podem ser: i) CHIP-SEQ ([Robison et al., 1998](#)); ii) evolução sistemática de ligando por enriquecimento exponencial (SELEX, do inglês *Systematic Evolution of Ligands by Exponential enrichment*) ([Liu e Stormo, 2005](#)); iii) *protein-binding microarrays* ([Berger e Bulyk, 2009](#)); iv) genes co-expressados ([Stormo e Hartzell, 1989](#)); e v) *dnase footprint* ([Sabo et al., 2004](#)). Neste trabalho, optamos por validar nosso algoritmo com dados originados de experimentos CHIP-SEQ, pois este é o método mais utilizado atualmente nos laboratórios científicos de alta performance ao redor do mundo para captura de elementos regulatórios ([Nakato e Sakata, 2021](#)).

Embora confiáveis, os dados oriundos de experimentos CHIP-SEQ ainda apresentam uma quantidade considerável de ruído. Além disso, as READS² possuem baixa resolução e diversos erros sistêmicos. Abordagens computacionais podem ajudar na elucidação dos problemas acima, diminuindo significativamente a quantidade de falsos positivos e negativos. Não obstante a grande quantidade de abordagens publicadas na literatura para lidar com a descoberta de MOTIFS, existem muitas diferenças entre elas. A maneira como os algoritmos são implementados e as metodologias que cada um apresenta, faz com que os resultados alcançados por cada abordagem no mesmo conjunto de dados seja diferente. Além do mais, na sua representação mais simples, já foi provado que o problema de descoberta de MOTIFS é NP-COMPLETO ([Wang e Jiang, 1994](#); [Li et al., 2002](#); [Evans et al., 2003](#)). Isso por si só já é um fator encorajante para que novas abordagens sejam desenvolvidas.

Outrossim, a complexidade de tempo e espaço são fatores que devem ser considerados. Muitas abordagens existentes na literatura que possuem excelente desempenho em datasets pequenos e médios. Porém, quando o volume de dados aumenta, elas não são capazes de lidar com o problema em tempo hábil³.

²READS são os fragmentos de sequência bruta gerados como resultado direto da aplicação da técnica CHIP-SEQ.

³Neste contexto, tempo hábil significa um intervalo de tempo em que a resolução do problema ainda resguarda algum significado.

É importante destacar que os métodos computacionais tornaram-se fundamentais para ajudar a responder inúmeras questões na biologia moderna. Essas abordagens oferecem diversas vantagens em comparação com as técnicas experimentais tradicionais. Primeiramente, os métodos computacionais são substancialmente mais rápidos, permitindo analisar e processar grandes volumes de dados em um curto espaço de tempo. Em segundo lugar, os resultados obtidos em muitos experimentos atuais são extremamente complexos e, portanto, requerem análise computacional avançada para interpretação adequada e extração de informações biologicamente relevantes. Dentro deste espectro, os *k*-mers emergem como elementos importantes, permitindo a análise eficiente de sequências genéticas e também facilitando a compreensão de padrões biológicos complexos.

K-mers são subseqüências de comprimento *k* extraídas de uma sequência maior de DNA, RNA ou proteínas. Eles são fundamentais para o estudo de genômica e bioinformática, servindo como os blocos construtores básicos para diversas análises, incluindo montagem de genomas, busca por padrões genéticos, identificação de espécies e compreensão da diversidade genética. A escolha do valor de *k* é crítica, influenciando diretamente a sensibilidade e especificidade da análise. *K*-mers permitem a decomposição de sequências complexas em unidades menores e manejáveis, facilitando o processamento computacional e a comparação entre sequências biológicas.

Neste contexto, a contagem e o agrupamento de *k*-mers semelhantes apresentam um potencial significativo para contribuir positivamente na criação de modelos probabilísticos, especialmente na análise de dados de *CHIP-SEQ*. Essa abordagem pode efetivamente identificar padrões recorrentes e estruturas importantes nos dados, facilitando a modelagem estatística. Ao agrupar *k*-mers com características semelhantes, é possível reduzir a complexidade dos dados, melhorando assim a eficiência e precisão dos modelos probabilísticos. Esta técnica pode ser particularmente útil no contexto de grandes volumes de dados, onde a identificação precisa de *MOTIFS* biológicos é essencial, mas desafiadora devido à vasta quantidade de informações a serem analisadas.

Para acompanhar a evolução das tecnologias de sequenciamento, novos algoritmos precisam ser concebidos, sempre buscando melhorar os resultados obtidos pelas abordagens anteriores. Desta forma, levando em consideração as dificuldades apresentadas, nosso problema de pesquisa se resume na seguinte questão: *de que maneira o agrupamento e a contagem de k-mers poderia contribuir para a melhoria do desempenho de algoritmos na detecção de MOTIFS biológicos em dados de CHIP-SEQ, levando em conta os desafios impostos pela complexidade e pelo grande volume de dados produzidos por este protocolo?*

Para responder a esta e outras questões bem como obter alguns *insights*, foi realizada uma detalhada revisão bibliográfica do estado da arte que pode ser encontrada no Capítulo 3, no qual foi possível identificar os principais trabalhos publicados na literatura. Nela, detectamos que os algoritmos mais rápidos são os que empregam modelos enumerativos em conjunto com árvores de sufixo ou alguma estrutura correlata. Entretanto, estas abordagens não são capazes de encontrar fragmentos com alto grau de degeneração, perdendo precisão. Outra limitação das técnicas enumerativas é a dificuldade em lidar eficientemente com fragmentos maiores e conjuntos de dados volumosos. Além disso, identificamos que os modelos probabilísticos são mais adequados para lidar com fragmentos degenerados e extensos. No entanto, essas abordagens tendem a ser mais lentas e podem facilmente ficar presas em ótimos locais. Desta forma, percebemos uma lacuna entre desempenho e precisão.

1.3 HIPÓTESE

Levando em consideração o problema de pesquisa apresentado na seção anterior, nossa hipótese teórica postula que a *contagem de k-mers desempenha um papel importante na formulação de modelos probabilísticos iniciais*, facilitando a identificação de regiões genéticas similares que apresentam baixo grau de mutações, e este processo contribui significativamente com o incremento da pontuação dos fragmentos detectados. A hipótese também sustenta que tais modelos estarão próximos a ótimos locais e, potencialmente, ao ótimo global; e que, conseqüentemente, ao serem processados por um algoritmo de otimização *que assegura a localização de um ótimo local*, espera-se que essa abordagem identifique regiões altamente promissoras no espaço de busca.

Esperamos que a análise de (K,D)-MERS (sequências de símbolos de comprimento k com até d mutações espalhadas uniformemente ao longo do fragmento) possa fornecer insights importantes para a formação de distribuições a priori, fundamentadas em dados empíricos, que são essenciais na modelagem estatística e computacional avançada. Presumimos que essa abordagem, ao ser incorporada em algoritmos de otimização, potencializará a construção de modelos mais sofisticados, aumentando consideravelmente as chances de alcançar ótimos globais. Além disso, supomos que a utilização de (K,D)-MERS como base empírica para os modelos a priori possa *reduzir consideravelmente o tempo de convergência dos algoritmos de otimização*, tornando o processo mais eficiente. Por fim, acreditamos que essa integração entre a genômica computacional e métodos estatísticos avançados abrirá novas possibilidades para a descoberta de padrões biológicos complexos em grandes volumes de dados.

Para validar essa hipótese, desenvolvemos um novo arcabouço eficiente para manipulação e otimização de k-mers e descoberta de MOTIFS capaz de lidar com grandes volumes de dados, especialmente os produzidos pelo protocolo CHIP-SEQ. Também conduzimos uma análise comparativa rigorosa entre nosso método e as abordagens consideradas estado da arte, abrangendo métricas como tempo de execução, uso de memória e acurácia. Definimos a hipótese nula (H_0) como a ausência de diferenças estatisticamente relevantes em todas essas métricas, em comparação com os métodos de referência. Em contrapartida, a hipótese alternativa (H_a) postula que nosso algoritmo demonstra um desempenho significativamente superior em pelo menos uma dessas métricas. Para avaliação estatística, optamos pelo teste de FRIEDMAN seguido de análise post-hoc de NEMENYI com a correção de BONFERRONI. Essa escolha metodológica é justificada pela potencial não-normalidade dos conjuntos de dados e a viabilidade de heterocedasticidade entre os grupos analisados. O nível de confiança adotado para todas as análises será de 95%, correspondendo a um valor de α de 0,05.

1.4 OBJETIVO

Desenvolver novos métodos computacionais para análise e descoberta de MOTIFS biológicos em dados de CHIP-SEQ, capazes de lidar com grandes volumes de dados, complexidade e ruído experimental, e que possam ser empregados por biólogos, bioinformatas e demais pesquisadores da área para realizar investigações científicas e aprofundar a compreensão dos mecanismos moleculares e da regulação genética.

1.5 CONTRIBUIÇÕES

As principais contribuições deste trabalho se concentram na revisão bibliográfica do estado da arte e no desenvolvimento de uma nova ferramenta computacional para análise e

descoberta de MOTIFS. A principal contribuição da revisão sistemática consiste na apresentação de vários aspectos que podem ser levados em consideração na construção de novos algoritmos e modelos. Além disso, através dela foi possível identificar os principais problemas em aberto da área. Essas informações podem ser utilizadas por pesquisadores como uma trilha na identificação dos esforços inerentes ao desenvolvimento de métodos mais efetivos para análise e localização de MOTIFS biológicos.

Nós também realizamos o desenvolvimento de um novo arcabouço para a descoberta de MOTIFS levando em consideração: i) os dados pesquisados na revisão bibliográfica; ii) os problemas em aberto identificados; iii) as necessidades atuais dos biólogos e demais pesquisadores da área. Assim, nosso arcabouço se propõe a ser um algoritmo robusto, tendo como principais características a maximização das métricas de desempenho e a minimização do tempo de resposta e da quantidade de espaço empregado na resolução do problema. Além do mais, esperamos que o BIOMAPP-CHIP possa contribuir com o desenvolvimento científico sendo uma ferramenta capaz de auxiliar na pesquisa e no estudo de MOTIFS biológicos.

1.6 ORGANIZAÇÃO DO TEXTO

Antes dos trabalhos serem efetivamente iniciados, identificamos a priori as seguintes etapas como um conjunto essencial de procedimentos necessários para a elaboração desta pesquisa. São elas: i) entender o problema; ii) planejar e coletar dados; iii) explorar os dados; iv) postular um modelo; v) ajustar o modelo; vi) testar o modelo; e vii) usar o modelo.

Sendo assim, esse estudo apresenta a seguinte organização. O Capítulo 2 visa identificar as questões ligadas aos passos (i) e (ii) e detalha os fundamentos dos métodos de descoberta de MOTIFS e o funcionamento dos elementos cis-regulatórios. Este capítulo também apresenta os mecanismos que regem a obtenção de dados através do protocolo CHIP-SEQ e os principais repositórios deste tipo de experimento. O Capítulo 3 está relacionado com a etapa (i) e apresenta a metodologia empregada na busca pelos principais estudos relacionados à análise e descoberta de MOTIFS. Este capítulo ainda apresenta o resumo das principais publicações da área, a classificação e as questões em aberto relacionadas aos estudos encontrados. O Capítulo 4 aborda assuntos relativos as etapas (iii), (iv) e (v) e detalha a proposta deste trabalho bem como as técnicas e os algoritmos utilizados na construção do arcabouço. O Capítulo 5 está associado as etapas (vi) e (vii) e apresenta os resultados obtidos e a análise dos dados. Por fim, o Capítulo 6 apresenta a conclusão e os trabalhos futuros.

2 FUNDAMENTOS

Este capítulo tem como objetivo fornecer um sólido alicerce para o desenvolvimento do trabalho, apresentando os conceitos, teorias e métodos relevantes para a área de pesquisa proposta. Iremos abordar os principais tópicos relacionados ao nosso tema estudo, estabelecendo assim o contexto e as bases teóricas que sustentarão a investigação. Inicialmente, serão apresentadas as definições e conceitos fundamentais, proporcionando um entendimento claro e preciso do assunto. Além disso, será apresentada uma discussão sobre as principais teorias que servem de base para essa pesquisa, destacando as suas contribuições e limitações. Por fim, serão estabelecidos os fundamentos para os capítulos subsequentes, que se aprofundarão na metodologia, análise de dados e discussão dos resultados. Ao fornecer um entendimento concreto dos conceitos e teorias fundamentais, este capítulo permitirá uma melhor compreensão das contribuições e implicações do estudo proposto, bem como sua relevância no contexto mais amplo da área de pesquisa. Este capítulo está dividido como segue. Na Seção 2.1, os MOTIFS são explicados tanto em termos biológicos quanto computacionais, destacando sua relevância e aplicação nestes contextos. Na Seção 2.2, a definição do problema é apresentada, abordando diversas perspectivas que podem ser adotadas para enfrentá-lo e compreendê-lo. Na Seção 2.3, são apresentadas as principais técnicas e algoritmos utilizados no desenvolvimento do nosso arcabouço, destacando seus aspectos fundamentais e a maneira como foram aplicados. Por fim, a Seção 2.4 aborda o funcionamento do protocolo CHIP-SEQ, que é essencial para a geração dos dados analisados nesta pesquisa, detalhando suas etapas e relevância no contexto deste estudo.

2.1 O QUE SÃO MOTIFS

Conforme apresentado brevemente no capítulo anterior, MOTIFS são pequenos padrões conservados e recorrentes de DNA, RNA ou aminoácidos no qual se presume ter alguma função biológica bem definida. Frequentemente eles definem os locais de ligação de diversas proteínas, tais como nucleases, histonas e fatores de transcrição (TF), sendo este fenômeno responsável pelo início da transcrição ou repressão gênica. Em outros casos, os MOTIFS estão envolvidos em processos importantes no nível do RNA, incluindo ligação de ribossomos, processamento de mRNA (*splicing*, edição, *poliadenilação*) e terminação da transcrição (Weirauch et al., 2014).

A grande maioria dos fatores de transcrição realiza a regulação gênica em apenas um sentido: eles ligam ou desligam a expressão dos genes a montante ou a jusante de sua localização, mas não em ambas as direções. Em outros casos, alguns TFs são ativados em níveis baixos quando o organismo encontra-se em condições normais e aumentam a concentração quando o organismo sofre algum tipo de condição estressante.

A evolução dos MOTIFS biológicos, particularmente os sítios de ligação dos fatores de transcrição (TFBS's), é um processo complexo que reflete a interação entre pressão seletiva e deriva genética. Embora estes elementos possuam um alto nível de conservação devido à sua importância funcional, eles também estão sujeitos a mutações pontuais, que podem ser influenciadas tanto por seleção natural quanto por eventos aleatórios de neutralidade. A pressão seletiva tende a preservar variantes de MOTIFS que mantêm ou melhoram a funcionalidade biológica, enquanto a deriva genética pode introduzir variabilidade neutra, permitindo uma certa tolerância a substituições de bases sem comprometer significativamente a função do elemento. Esta dualidade sugere que a variabilidade observada nos TFBS's não pode ser adequadamente capturada por uma única sequência de nucleotídeos, exigindo mecanismos mais sofisticados

para sua representação e análise. Assim, a compreensão da evolução dos MOTIFS e dos TFBS's requer uma abordagem integrada que considere tanto a seleção natural que favorece adaptações funcionais quanto a deriva genética que promove a diversidade genética por meio de mudanças neutras (Neil et al., 2018).

Determinadas condições podem ser ativadas (ou desativadas) nos organismos dependendo do contexto. Por exemplo, a mosca da fruta (*Drosophila melanogaster*), assim como os humanos, são suscetíveis a infecções por bactérias e outros patógenos. Embora não tenham um sistema imunológico tão sofisticado quanto o nosso, elas possuem um pequeno conjunto de genes de defesa, que são ativados quando o organismo é infectado por um agente estranho. Quando isso ocorre, são produzidas proteínas que destroem o invasor, geralmente curando a infecção (Jones e Pevzner, 2004). Esse mecanismo de ativação e desativação de certas funções possui como principal característica a ligação de uma proteína reguladora a uma região específica, normalmente localizada a montante dos genes, chamada de região promotora.

É bastante comum a realização de experimentos controlados, no qual é possível infectar certos organismos com algum tipo de patógeno e depois medir quais genes foram ativados (ou suprimidos) como uma resposta imune. A partir desse conjunto de genes, é possível determinar, por exemplo, o que desencadeou a sua ativação. É interessante notar que muitos genes de imunidade no genoma da *Drosophila melanogaster* têm cadeias que se assemelham a TCGGGGATTTC localizadas a montante do início dos genes. Essas cadeias curtas, chamadas sítios de ligação *NF- κ B*, são exemplos importantes de MOTIFS reguladores que ativam, além da imunidade, outros genes auxiliares. Proteínas conhecidas como fatores de transcrição se ligam a esses sítios, incentivando a RNA polimerase a transcrever os genes a jusante. A descoberta de MOTIFS é a tarefa responsável por encontrar tais elementos, sem nenhum conhecimento prévio de sua estrutura, distribuição ou quaisquer outras características. A única informação disponível é o conjunto de dados¹(D'haeseleer, 2006a).

Uma forma bastante comum de descrever MOTIFS é utilizar a sequência CONSENSO (D'haeseleer, 2006b; Cui et al., 1995). Ela pode ser considerada um resumo de um alinhamento arbitrário e é empregada para sintetizá-los através de uma única sequência. Posteriormente, esse padrão pode ser comparado com outros tipos de sequências de mesmo tamanho em diferentes datasets, a fim de se obter possíveis correspondências. Além disso, a sequência CONSENSO também é muito utilizada por algoritmos enumerativos na busca por MOTIFS biológicos (vide Capítulo 3).

Frequentemente os códigos IUPAC são empregados em conjunto com algum tipo de limiar para identificar cada base. Por exemplo, se o limiar for de 25%, então, se dois ou mais nucleotídeos ultrapassarem esse limite, a sequência CONSENSO de saída deverá apresentar um símbolo especial, que captura este comportamento. Como exemplo, considere um alinhamento arbitrário, no qual as pirimidinas *C* e *T* possuem respectivamente 30% e 35% de cobertura cada uma. Neste caso, a sequência CONSENSO seria *Y* (*pYrimidine*). Isso captura corretamente a frequência relativa de cada base no alinhamento.

Além da sequência CONSENSO, podemos definir um alinhamento de sequências através de uma matriz de probabilidades. Stormo (2000) desenvolveu um modelo probabilístico chamado de Matriz de ESCORE de Posição Específica (PSSM, do inglês *Position Specific Scoring Matrix*) também conhecida como PWM (*Position Weight Matrix*). A matriz PWM é construída através da contagem dos símbolos que aparecem em cada coluna do alinhamento. A matriz de contagem é chamada de matriz de frequência (PFM, do inglês *Position Frequency Matrix*). Essa matriz possui

¹O termo conjunto de dados se refere às sequências de entrada que são obtidas através de experimento biológicos, como o CHIP-SEQ por exemplo.

um total de k linhas (sendo k a largura dos MOTIFS) e 4 colunas, uma para cada nucleotídeo². Desta forma, se l_i é a i -ésima linha, então $l_i \sim \text{Multinomial}(n, \alpha_{i,a}, \alpha_{i,c}, \dots, \alpha_{i,t})$, considerando que estamos lidando com sequências de DNA. Para outros alfabetos, basta substituir $\{A, C, G, T\}$ pelos símbolos correspondentes. Desta forma, cada coluna do alinhamento possui uma distribuição diferente.

Para estimar os parâmetros $\alpha_{i,a}, \alpha_{i,c}, \dots, \alpha_{i,t}$, basta dividir as contagens de cada símbolo por n , sendo n a quantidade de sequências pertencentes ao alinhamento que pode ser diretamente calculado por $\sum_{symbol=\{A,C,G,T\}} \alpha_{i,symbol}$, para $i = 1, 2, 3, \dots, k$, sendo que k é a largura a alinhamento. Isso resulta em uma nova estrutura chamada Matriz de Posição de Probabilidade (PPM, do inglês *Position Probability Matrix*). A matriz PPM é um modelo de MARKOV não homogêneo de ordem 0. A etapa final consiste em multiplicar cada entrada da matriz PPM pela divisão do logaritmo da mesma entrada pela probabilidade de fundo do genoma. Essa etapa calcula basicamente a divergência de Kullback-Leibler entre a distribuição dos MOTIFS e a distribuição de fundo $DKL(A||B) = Pr_A(M) \times \log(\frac{Pr_A(M)}{Pr_B(M)})$. Se a base do logaritmo é 2, então essa distância computa a esperança da diferença entre o logaritmo da distribuição dos MOTIFS (A) e a distribuição de fundo (B), que é o mesmo que $\mathbb{E}(\log(Pr_A(M)) - \log(Pr_B(M)))$. Quanto maior for essa diferença, mais distante uma distribuição estará da outra o que implica maior conservação das bases. Essa última etapa possui como saída a matriz PWM.

A Figura 2.1 apresenta todas as fases para a criação da matriz PWM. Primeiramente temos na Figura 2.1(a) um conjunto de dados sem nenhuma sub-sequência de interesse, i.e, formada apenas com sequências de fundo. A Figura 2.1(b) mostra o mesmo dataset, porém com os MOTIFS implantados em posições aleatórias. Na natureza, esse é normalmente o comportamento esperado quando vários genes são co-expressados, cada um deles compartilhando com os demais, sítios de ativação (ou repressão) que ligam ou desligam o mecanismo de transcrição, formando uma espécie de rede de regulação gênica. A Figura 2.1(c) mostra o alinhamento global das sequências levando em consideração as regiões conservadas. Para melhorar a visualização, na Figura 2.1(d), os sítios de interesse foram separados do restante das sequências. A Figura 2.1(e), a sequência CONSENSO foi calculada utilizando o esquema de codificação IUPAC. É interessante notar que nas posições 1, 2, 3, 12 e 13 o valor computado foi Y, B, S, S e R respectivamente³. Isso significa que existe nestas colunas uma cobertura maior que 25% de dois ou mais nucleotídeos. Por fim, nas Figuras 2.1(f), 2.1(g) e 2.1(h) temos as matrizes PFM, PPM e PWM respectivamente. Essas três matrizes são a forma mais usual para a representação de MOTIFS, pois elas conseguem capturar as variações que existem nas sequências individuais.

2.2 DEFINIÇÃO DO PROBLEMA

A descoberta de MOTIFS envolve frequentemente a análise de alinhamentos locais de sequências, sendo que seus principais objetivos são: descoberta, modelagem e o reconhecimento de padrões, conforme pode ser visto abaixo.

- Descoberta: consiste na identificação de uma ou mais sub-sequências conservadas em datasets de sequências. Em *machine learning*, esta etapa está fortemente associada ao aprendizado não-supervisionado e a criação de *clusters*;

²É possível encontrar na literatura uma variante dessa matriz com 4 linhas e k colunas.

³Y = C ou T; B = C ou G ou T; S = G ou C; R = A ou G. Para mais códigos, consulte: <https://www.bioinformatics.org/sms/iupac.html>

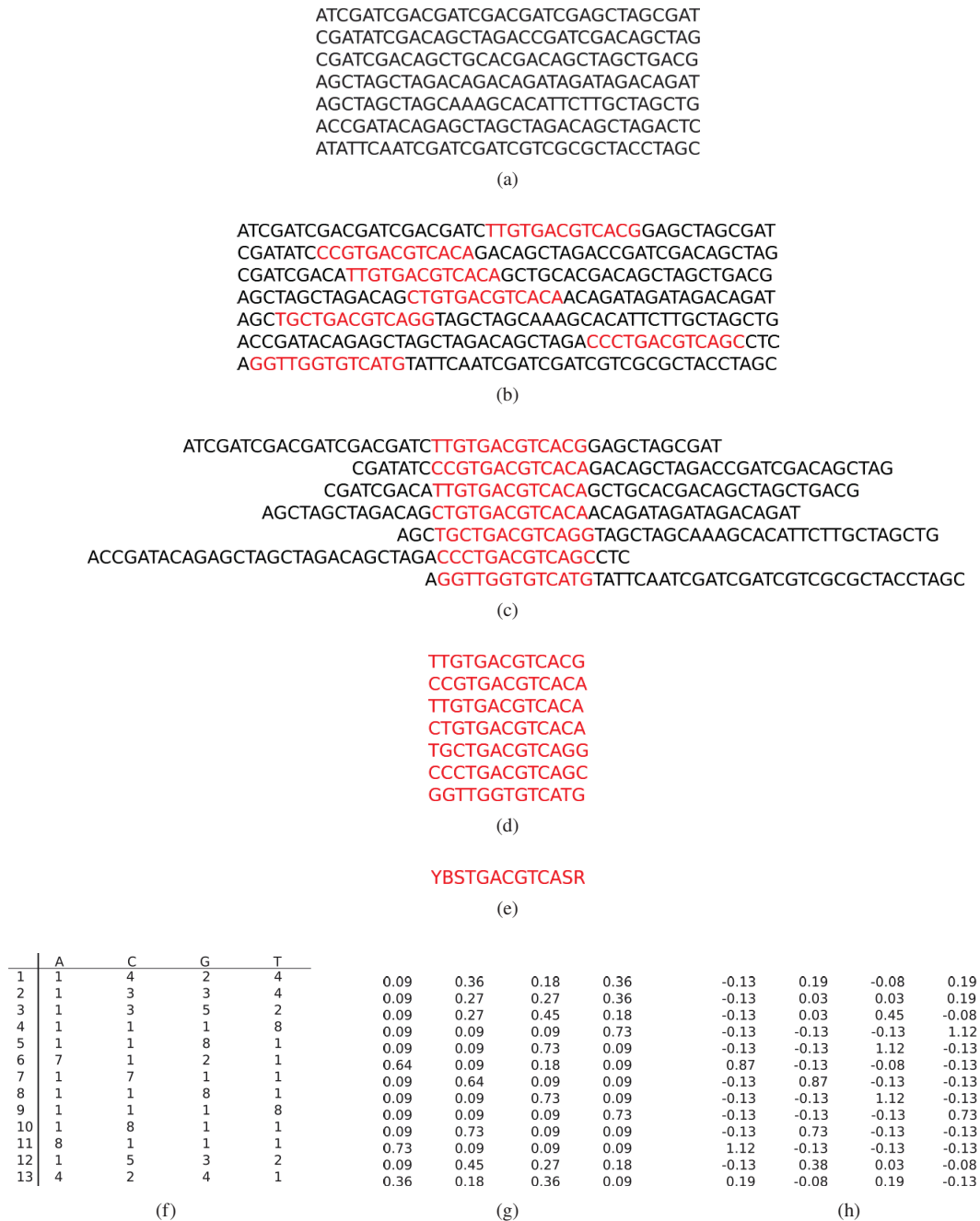


Figura 2.1: (a) Apresenta um exemplo com sete sequências cada uma com 32 pares de base. (b) Exibe as mesmas sequências, com o fator de transcrição CREB1 (MA0018.4) do organismo *H. sapiens* implantado em posições aleatórias. (c) Mostra o alinhamento do conjunto de dados, tomando como base os *motifs* implantados. (d) Apresenta as sub-sequências de interesse isoladamente; (e) Exibe a sequência consenso extraída do alinhamento. (f) Matriz de frequência absoluta (PFM) com **pseudocontadores** adicionados. (g) Matriz de frequência relativa (PPM). (h) Matriz PWM. Fonte: o autor.

- Modelagem: a partir de um alinhamento local, essa etapa compreende a construção de modelos que capturam as principais propriedades deste alinhamento. Esta fase está intimamente ligada com a postulação e estimação de modelos probabilísticos;
- Reconhecimento: esta etapa emprega os modelos estimados no reconhecimento de novos MOTIFS em sequências que ainda não foram rotuladas. Esta fase também é conhecida em *machine learning* como classificação ou aprendizado supervisionado.

Portanto, podemos verificar que o problema de descoberta de MOTIFS possui relação com mais de uma área do aprendizado de máquina. Formalmente, a descoberta de MOTIFS pode ser definida de várias maneiras. A mais comum é considerando um vetor $S = \{s_1, s_2, s_3, \dots, s_n\}$, sendo que s_i é a posição inicial estimada do MOTIF na i -ésima sequência. Se t_i é o comprimento da i -ésima sequência e k é o comprimento esperado dos MOTIFS, então $1 \leq s_i \leq t_i - k + 1$. Frequentemente temos que $t_i = t_j, \forall i \neq j, 1 \leq i, j \leq n$, no qual n é a quantidade de sequências presente no conjunto de dados. Por exemplo, se $t = 32, k = 13$ e $n = 7$, então, $S = \{s_1 = 22, s_2 = 8, s_3 = 10, s_4 = 15, s_5 = 4, s_6 = 30, s_7 = 2\}$ ⁴ é uma solução válida. Podemos calcular o ESCORE de S através da Equação 2.1:

$$escore(PFM) = \sum_{i=1}^k \max(PFM_i) \quad (2.1)$$

No qual $\max(PFM_i)$ denota o maior valor da i -ésima linha da matriz PFM. A Equação 2.1 também é conhecida como CONSENSUS ESCORE.

Por exemplo, se aplicarmos a Equação 2.1 na matriz PFM da Figura 2.1(f), então teremos: $escore(PFM) = 4 + 4 + 5 + \dots + 4 = 84$. Se a matriz PFM apresentar uma distribuição uniforme, então o ESCORE será menor. Considere que a matriz PFM da Figura 2.1(f), se ela não fosse conservada, apresentaria uma distribuição uniforme com média, $\frac{11}{4} = 2,75$ nucleotídeos de cada tipo em cada posição do alinhamento. Neste caso, $escore(PFM) = 13 \times 2,75 = 35,75$.

A Equação 2.1 não leva em consideração as sequências de fundo do genoma e por essa razão não é a melhor opção para avaliar MOTIFS, principalmente os que possuem menor nível de conservação. Uma maneira mais apropriada, que leva em consideração as sequências de fundo, é apresentada na Equação 2.2:

$$IC = \sum_i^k \sum_j^4 PPM_{i,j} \times \log_2 \frac{PPM_{i,j}}{B_j} = \sum_i^k \sum_j^4 PWM_{i,j} \quad (2.2)$$

A Equação 2.2 calcula o Conteúdo da Informação (IC, do inglês *Information Content*) do alinhamento de MOTIFS. O IC é a diferença média da quantidade de bits entre a distribuição de interesse e a distribuição de fundo do genoma. As matrizes PPM e PWM são as mesmas mostradas na Figura 2.1(g) e 2.1(h).

A Equação 2.2 também é utilizada para calcular de forma independente o nível de conservação de cada coluna do alinhamento de MOTIFS. Ela é derivada da teoria da informação através da Equação 2.3:

$$I_i = 2 + \sum_{b \in \{A,C,G,T\}} PPM_{b,i}^\top \times \log_2(PPM_{b,i}^\top) \quad (2.3)$$

No qual I_i é o nível de conservação de cada coluna do alinhamento. Se a coluna for totalmente conservada, então $I_i = 2$ bits de informação (pois, $\log_2(1) = 0$), se duas das quatro bases possuírem 50% de representação cada, então $I_i = 1$ bit e se todas as bases possuírem 25% de representação cada, então $I_i = 0$ bits (sem informação). $B = \{A, C, G, T\}$.

A Equação 2.3 permite representar o alinhamento de MOTIFS visualmente através de gráficos especiais chamados de *logos*. Para isso, basta calcular I_i individualmente para cada nucleotídeo. O valor resultante será a altura em bits do símbolo correspondente. A Figura 2.2 mostra a *logo* do fator de transcrição CREB1 (Figura 2.1) construída pela ferramenta online *WebLogo* (Wang et al., 2019; Crooks et al., 2004) disponível em: <https://weblogo.berkeley.edu/logo.cgi>.

⁴O vetor S corresponde as posições extraídas da Figura 2.1



Figura 2.2: Logo criada a partir do alinhamento de seqüências do fator de transcrição CREB1. O eixo x representa cada posição do alinhamento e o eixo y é o valor da entropia relativa em *bits*, levando consideração uma distribuição de fundo com probabilidades uniformemente distribuídas. Fonte: o autor.

Na Figura 2.2, é possível observar que o eixo x é formado pelas posições correspondentes ao número de colunas que o alinhamento possui. O eixo y é a quantidade de informação em bits que cada símbolo contribuiu em sua respectiva coluna. Não é difícil verificar que a região mais conservada está entre as posições 4 e 11. Além do mais, nota-se que esta sub-sequência - *TGACGTCA* - é palíndroma, i.e., é igual ao seu complemento reverso.

Utilizando a Equação 2.3, podemos calcular o conteúdo da informação total de um alinhamento somando todos os I_i , para $i = 1, 2, 3, \dots, k$. Por exemplo, a sequência CONSENSO do fator de transcrição CREB1 exibido na Figura 2.1(e) é *YBSTGACGTCASR*. Aplicando a Equação 2.3, o conteúdo da informação desta sequência é: $13 \times 2 + 4 \log_2(1/2) + \log_2(1/4) + 8 \log_2(1) = 20$ bits. Isso ocorre porque temos 4 bases com degeneração dupla (*YSSR*), 1 base com degeneração tripla (*B*) e 8 bases totalmente conservadas (*TGACGTCA*). É possível realizar os cálculos acima para qualquer sequência. Por exemplo, considere a sequência CONSENSO da enzima de restrição *HindIII* que é *CTYRAC*. aplicando a mesma equação temos: $2 \times 6 + 4 \log_2(1) + 2 \log_2(1/2) = 10$ bits. A interpretação deste resultado sugere que o valor esperado da diferença entre o logaritmo da distribuição de interesse e o logaritmo da distribuição de fundo é 20 bits no primeiro exemplo e 10 bits no segundo, considerando que a probabilidade de fundo seja uniformemente distribuída entre as bases nucleicas.

Entretanto, a premissa de que todas as bases tenham probabilidades iguais não é a melhor escolha na maioria dos casos, pois existem diversos organismos que possuem valores viesados, devido a diversos fatores, como metilação, por exemplo. Em organismos como *H. sapiens* e *E. coli*, cuja quantidade de CG é de 42% e 51% respectivamente, assumir uma distribuição uniforme é uma aproximação adequada. Porém, para outros organismos como *S. cerevisiae* (38%), *C. elegans* (36%), *P. falciparum* (19%) e *S. coelicolor* (72%) essa aproximação não irá prover bons resultados. (D'haeseleer, 2006b).

Desta forma, um fator de correção deve ser adicionado a Equação 2.3. Schneider e Stephens (1990) argumentaram que seria necessário substituir o “2” da Equação 2.3 por um valor proporcional ao conteúdo CG das seqüências de fundo do DNA. Uma abordagem mais genérica é utilizar a distância de Kullback-Leibler, definida pela Equação 2.4:

$$I_i = - \sum_b^B f_{b,i}^\top \times \log_2 \left(\frac{f_{b,i}^\top}{p_b} \right) \quad (2.4)$$

No qual $f_{b,i}^\top$ é a probabilidade do símbolo b aparecer na i -ésima coluna do alinhamento e p_b é a probabilidade do símbolo b aparecer ao acaso no genoma do organismo.

As Equações 2.4 e 2.2 são muito semelhantes. O cálculo de $\sum_{i=1}^k I_i$ resulta exatamente no valor de IC , discutido anteriormente. Porém, a Equação 2.4 é um pouco mais flexível e pode ser empregada em contextos diferentes. Por exemplo, com ela é possível utilizar modelos

estatísticos de ordem superior mais robustos para descrever as distribuições de interesse e de fundo, desde que esse modelo seja capaz de calcular a probabilidade da sequência $Pr(S|M)$. Esta equação é equivalente ao teste G , utilizado em casos no qual o teste de qui-quadrado falha (Teixeira e Queirós, 2016; Thijs et al., 2001a).

A complexidade inerente à descoberta de MOTIFS em sequências biológicas exige que abordagens multidisciplinares sejam adotadas para uma compreensão completa e eficaz do problema. Enquanto as metodologias matemáticas e computacionais fornecem as ferramentas necessárias para a análise e identificação de padrões dentro de grandes conjuntos de dados, uma visão que transcenda os limites estritos dessas disciplinas pode revelar insights adicionais significativos. A inclusão de perspectivas derivadas de outras áreas do conhecimento, como a física termodinâmica, permite uma interpretação mais rica e integrada dos fenômenos biológicos subjacentes. Esta abordagem holística é particularmente relevante no estudo de MOTIFS, onde as interações entre moléculas biológicas não são apenas determinadas por sequências específicas, mas também por suas propriedades físico-químicas e o contexto ambiental em que ocorrem. Assim, ao expandir o escopo da análise para incluir teorias e conceitos de outras disciplinas, é possível obter uma compreensão mais profunda e abrangente da natureza e da função dos MOTIFS biológicos.

Desta forma, com um viés mais voltado para a física termodinâmica, D’haeseleer (2006a) apresentou uma última interpretação do que seria o conteúdo da informação no contexto da descoberta de MOTIFS. Ele argumentou que o IC pode ser descrito como o nível energia existente entre os sítios de ligação e seus respectivos fatores de transcrição, desde de que cada símbolo contribua independentemente para essa ligação. Segundo ele, o IC pode ser entendido como a média da energia de ligação estimada entre todos os sítios conhecidos e sua respectiva sequência CONSENSO quando comparado a mesma ligação realizada randomicamente no DNA genômico. Em outras palavras, nas Equações 2.3 e 2.4, quanto maior for a diferença entre as distribuições de interesse e de fundo, menor será a sua entropia, porém a diferença entrópica será grande. O equilíbrio termodinâmico ocorre justamente quando as distribuições de interesse e de fundo são aproximadamente iguais. Isso pode ser interpretado como a diferença entre a maior entropia possível (neste caso 2) e o valor de entropia estimada através da distribuição de interesse. Considere que p é a distribuição de interesse e q é a distribuição de fundo. Então $\log_2(\frac{p}{q}) = \log_2(p) - \log_2(q)$, se $q = \frac{1}{4}$, então $\log_2(p) - \log_2(\frac{1}{4}) = \log_2(4) + \log_2(p)$ o que resulta em $2 + \log_2(p)$.

É interessante notar que a entropia de uma coluna não pode ser maior que 2 nem menor que 0, se a distribuição uniforme for empregada para representar a probabilidade da distribuição de fundo. Para ver isso, considere que a i -ésima coluna é totalmente conservada, i.e., sua entropia é mínima. Isso pode ser resumido através das Equações 2.5 até 2.10. Na Equação 2.5, a informação I_i está sendo calculada como a soma dos produtos de cada probabilidade p_i pelo logaritmo base 2 dessa probabilidade dividida por 1/4. Isso representa a medida de informação em cada probabilidade. Na Equação 2.6, a expressão é reorganizada para demonstrar a multiplicação explícita da probabilidade p_i pelo logaritmo base 2 de p_i subtraído de 2, para cada i . Na Equação 2.7, essa reorganização é continuada, distribuindo as probabilidades p_i nas duas partes da expressão. Na Equação 2.8, a soma das probabilidades é destacada.

Como a soma das probabilidades p_i é 1 (como representado na anotação sob a soma), a primeira parte da equação se simplifica para 2. A segunda parte da equação é a negativa da entropia de Shannon. Na Equação 2.9, a soma das probabilidades p_i é substituída pela entropia de Shannon. Se a coluna (referente às probabilidades) for totalmente conservada (ou seja, se uma das probabilidades for 1 e as demais forem 0), a entropia de Shannon será 0. Finalmente,

na Equação 2.10, se a entropia de Shannon for 0, a informação mútua será 2, indicando que a informação é perfeitamente preservada (totalmente conservada).

$$I_i = \sum_{i=1}^4 p_i \log_2 \left[\frac{p_i}{1/4} \right] \quad (2.5)$$

$$I_i = p_1(\log_2 p_1 - 2) + \dots + p_4(\log_2 p_4 - 2) \quad (2.6)$$

$$I_i = p_1 \log_2 p_1 - 2p_1 + \dots + p_4 \log_2 p_4 - 2p_4 \quad (2.7)$$

$$I_i = 2 \underbrace{(p_1 + p_2 + p_3 + p_4)}_{\text{soma 1}} - \underbrace{(p_1 \log_2 p_1 + \dots + p_4 \log_2 p_4)}_{\text{entropia de Shannon}} \quad (2.8)$$

$$I_i = 2 - \underbrace{\sum_{i=1}^4 p_i \log_2 p_i}_{\text{0, se coluna for totalmente conservada}} \quad (2.9)$$

$$I_i = 2 - 0 = 2 \quad (2.10)$$

No caso em que todas as probabilidades p_i são iguais a $1/4$, isso indica que todos os eventos são igualmente prováveis e não há preferência ou conservação de um determinado evento sobre os outros. Esta é uma situação de máxima incerteza, pois não há qualquer informação que favoreça um evento em detrimento dos demais. Ao substituir $p_i = 1/4$ nas equações, começamos com a Equação 2.5.

A medida de informação de cada probabilidade se torna zero, já que $\log_2(1)$ é zero. Isso indica que não há informação adicional obtida da observação de qualquer evento específico. Na Equação 2.9, a entropia de Shannon se torna máxima, atingindo o valor de 2. Isso ocorre porque a entropia de Shannon mede a quantidade de incerteza ou de informação “faltante” em um conjunto de probabilidades, e nesse caso, a incerteza é máxima. Finalmente, na Equação 2.10, a informação mútua se torna 0, pois a entropia de Shannon é 2. Isso indica que não há informação preservada, pois todos os eventos são igualmente prováveis, evidenciando a falta de conservação ou preferência.

Torna-se evidente que o conteúdo de informação de uma coluna, como demonstrado através das Equações 2.5 . . . 2.10, apresenta um limite inferior e superior bem definidos. O valor mínimo que o conteúdo da informação pode assumir é 0, uma situação que ocorre quando todos os eventos na coluna são igualmente prováveis, resultando na máxima incerteza. Por outro lado, o valor máximo que o conteúdo da informação pode atingir é 2, que é alcançado quando existe uma total conservação de um evento, i.e., quando um evento é completamente predominante sobre os outros. Estes limites demonstram que o conteúdo da informação de uma coluna serve como uma medida confiável e delimitada de conservação e variabilidade dentro dos dados biológicos estudados.

Uma maneira alternativa para a definição do problema é empregando o conceito de *string* media. Neste caso, o objetivo é encontrar uma *string* v que satisfaça a equação $d_{hm}(v, s) = \min d_h(v, s_i)$ para $s_i \in s$ e $i = 1 \dots m$ e $m = t - k + 1$, onde $d_h(v, s_i) = \sum_{j=1}^k \mathbb{1}\{v_{ij} \neq s_j\}$ é a

distância de *hamming* entre as *strings* v e s_i e d_{hm} é a menor distância de *hamming* entre a *string* v e todas as m sub-strings de s . O maior desafio aqui é encontrar v , pois realizar o cálculo de d_{hm} é trivial e computacionalmente simples, bastando apenas calcular a função d_h para todas as sequências, i.e., $d_h(v, s_1), d_h(v, s_2), d_h(v, s_3), \dots, d_h(v, s_m)$.

As duas formas anteriores são equivalentes em complexidade e guardam relação entre si. Se $A = \sum_{i=1}^n \sum_{j=1}^m d_{hm}(v, s_j)$ e $B = \sum_{i=1}^k \max(PFM_i)$, então $A = kn - B$. Para ver que isso é verdade, considere que kn é o maior *SCORE* possível. É fácil mostrar isso, pois k é o comprimento do *MOTIF* e n é a quantidade de sequências. Ocorre que, se o alinhamento for totalmente conservado, seu *SCORE* será kn . Portanto, para encontrar A , calculamos a diferença entre o maior *SCORE* kn e o *SCORE* do alinhamento B . Quando B tende à kn , A tende a zero. Portanto, B pode ser visto como sendo a representação da sequência *CONSENSO*. Reescrevendo, temos $A = kn - \text{consenso}$, o que mostra que isso é de fato a distância total de *Hamming*. Podemos escrever a equação anterior da seguinte forma, $B = kn - A$. Quando A tende à kn , então B tende zero. Porém, B representa a sequência *CONSENSO*, sendo que A então representa o erro. Reescrevendo após a mudança de parâmetro, $B = kn - \text{erro}$. Se o erro é baixo, o *SCORE* da sequência *CONSENSO* é alto. Mais especificamente, se o erro é zero, o *SCORE* é kn , o que termina a prova.

É interessante notar que embora os dois problemas sejam assintoticamente equivalentes, pode ser mais simples explorar o espaço de busca da *string* média. Isso ocorre porque a complexidade de tempo e espaço para gerarmos todas as *strings* de tamanho k é b^k , sendo b o tamanho do alfabeto, enquanto que para produzirmos todas as posições iniciais das quais os *MOTIFS* podem se originar precisamos de $(t - k + 1)^n$, sendo que t é a largura de cada sequência e n é a quantidade de sequências. Embora sejam assintoticamente exponenciais, a segunda opção possui base e expoente maiores.

Como as duas abordagens anteriores precisam gerar todas as combinações possíveis, conforme mostra a Figura 2.3, elas são denominadas algoritmos de força bruta. Ambas não podem ser empregadas na prática, pois demorariam tempo exponencial para encontrar soluções viáveis. Além disso, o tempo de execução e a quantidade de espaço necessários para lidar com o problema crescem muito rapidamente em função de k , t e n . Existem várias técnicas que podem ser aplicadas para minimizar o tempo de execução e diminuir o espaço utilizado, como por exemplo o algoritmo *BRANCH AND BOUND* e a geração de soluções de forma preguiçosa (*Lazy*). Entretanto, mesmo com essas otimizações, o problema ainda continua sendo intratável no pior caso.

Árvores de sufixo (*STREE*, do inglês *Suffix tree*) são estruturas de dados construídas através de conjuntos de *strings* que também podem ser empregadas para representar sequências biológicas. O resultado final é uma árvore invertida, no qual cada aresta representa um ou mais caracteres. Uma estrutura similar são as *Keywords Trees* (*KTREES*), no qual cada aresta representa exatamente um caractere. Sua utilização é vantajosa, pois permite a rápida execução de muitas funções importantes que operam sobre strings (Fischer et al., 2018; Aho e Hopcroft, 1974).

Por exemplo, a busca exata de um *KMERS* em uma *KTREE* pode ser realizada em tempo $O(k)$, sendo k a largura do *MOTIF*. Sua construção pode ser executada em tempo $O(mk)$, sendo $m = t - k + 1$ e t a largura de cada sequência do dataset de entrada. Sendo assim, uma busca demoraria tempo $O(k) + O(mk)$. Se consideramos a busca de apenas um único padrão, então a utilização de *KTREES* ou *STREES* não seria indicada, pois podemos fazer isso com busca linear em tempo $O(mk)$. Porém, se mais de uma busca for realizada⁵, a busca linear torna-se pouco performática. Para ver isso, considere que iremos executar a busca n vezes. Queremos encontrar n que satisfaça $nmk \leq nk + mk$. Dividindo tudo por k , temos $nm \leq m + n$. Rearranjados os

⁵Que é exatamente o que ocorre na descoberta de *MOTIFS*.

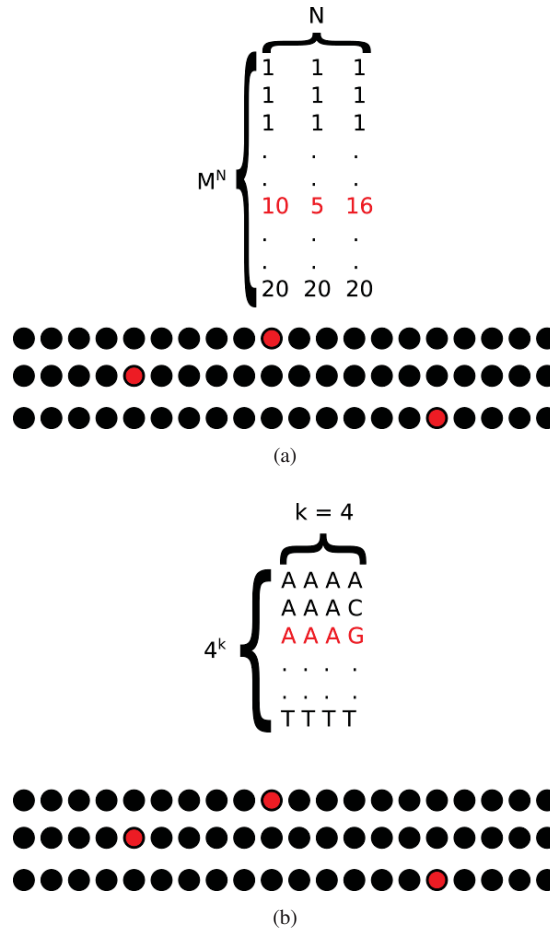


Figura 2.3: Representação abstrata dos métodos de melhor alinhamento e da *string* média. As duas abordagens são exponenciais. (a) O método do melhor alinhamento precisa gerar M^N combinações. (b) O método da *string* média precisa gerar 4^k combinações. Fonte: o autor.

termos, temos que $nm - n \leq m = n(m - 1) \leq m = n \leq \frac{m}{m-1} = n \leq \frac{m}{m-1} = n \leq \frac{1}{1-\frac{1}{m}}$. Quando m tende ao infinito, a relação $\frac{1}{m}$ diminui, tendendo a zero. Com isso, no limite, $n \leq \frac{1}{1-0} = 1$. Por exemplo, se $m = 100$, então $n \leq \frac{1}{1-\frac{1}{100}} 1,010101$. Se $m = 1000$, então $n \leq 1,001001$. Desta forma, a busca linear só será vantajosa para busca de um único padrão. Para dois ou mais, o uso de uma *KTREE* será mais performática, como podemos observar no gráfico da Figura 2.4. A Figura 2.5 mostra o exemplo de uma *STREE* construída a partir de uma *KTREE*.

2.3 DIFERENTES ABORDAGENS

Diversas técnicas podem ser empregadas na descoberta de *MOTIFS*. Elas se dividem basicamente em duas grandes classes: i) algoritmos probabilísticos e ii) algoritmos enumerativos. A diferença básica entre ambas consiste no tipo de função na qual se pretende otimizar. Enquanto o primeiro grupo tenta maximizar a função de entropia relativa ou alguma outra medida estatística capaz de calcular a distância entre duas distribuições de probabilidade, o segundo tenta minimizar a distância total de *Hamming* ou algum outro tipo de medida capaz de computar a semelhança (ou disparidade) entre duas *strings*.

Os algoritmos probabilísticos são divididos em duas sub-categorias: i) determinísticos e ii) não-determinísticos. As abordagens determinísticas frequentemente executam modelos de mistura finita em conjunto com algum algoritmo de otimização (normalmente o EM). Uma

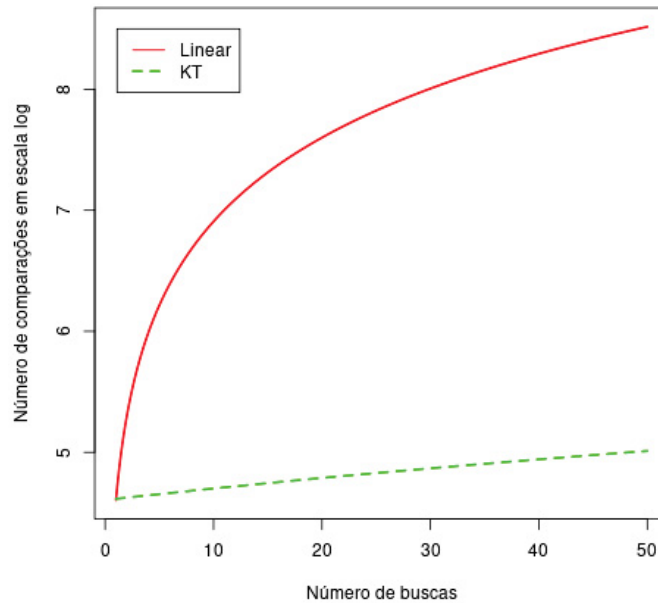


Figura 2.4: Numero de buscas vs número de comparações. Em vermelho temos o crescimento da função que representa a busca linear (nm) e em verde o crescimento da função que representa a busca utilizando KTs ($n + m$). Para compactar os valores, as funções de custo foram colocadas em escala log. O Gráfico foi gerado com, $m = 100$, com isso, para $n > 1.01$, as KTs passam a ser mais eficientes. Fonte: o autor.

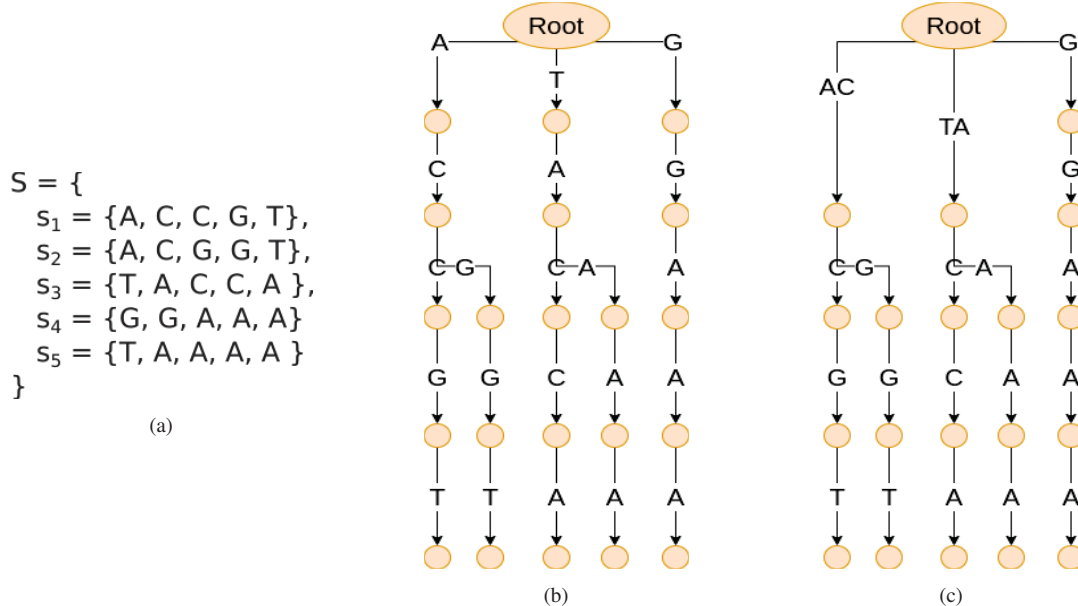


Figura 2.5: (a) Dataset S contendo cinco sequências de mesma largura. (b) *Keyword Tree* KT construída a partir de S . (c) *Suffix Tree* ST construída a partir de KT . Fonte: o autor.

característica dos algoritmos determinísticos é que dada uma entrada eles sempre retornam a mesma saída. Por essa razão, se a superfície de busca for muito acidentada, conforme mostra a Figura 2.6, esse tipo de abordagem sempre ficará aprisionada no mesmo ótimo local. Desta forma, é bastante comum que técnicas determinísticas utilizem algum mecanismo secundário, como por exemplo reinicialização automática, para conseguir explorar de maneira mais abrangente o espaço de busca.

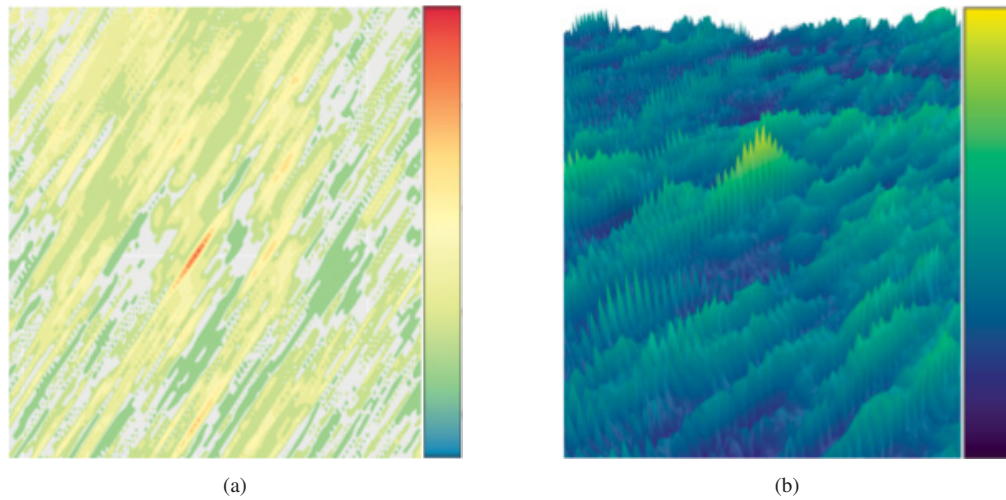


Figura 2.6: (a) Gráfico de contorno do *motif* GATA2 criado a partir do dataset MA0036.2 pertencente ao organismo *H. sapiens*. Este dataset possui largura de $t = 105$ pares de base e *motifs* com tamanho $k = 14$. Os eixos x e y representam as posições que variam de 1 até m , sendo que $m = 105 - k + 1 = 101$. A função de entropia relativa (Equação 2.4) foi empregada para calcular o score de todos os pares ordenados (x, y) , tal que $x, y \in \{1, 2, 3, \dots, m\}$. Além disso, uma distribuição de fundo uniforme foi utilizada. O ótimo global está localizado na posição $x = 51, y = 51$ e pode ser visualizado como o ponto vermelho mais escuro próximo ao centro da figura. (b) Gráfico de superfície do mesmo dataset. É interessante destacar sua alta rugosidade bem como observar seus vários ótimos locais. Fonte: o autor.

As abordagens não-determinísticas costumam empregar o algoritmo de amostragem de *Gibbs* através de uma Cadeia de MARKOV DE MONTE CARLO (MCMC, do inglês *Markov Chain Monte Carlo*) para modelar a busca. Ao contrário dos algoritmos determinísticos, que produzem o mesmo resultado para um dado conjunto de entradas, os não-determinísticos podem retornar saídas diferentes para a mesma entrada, devido ao seu comportamento estocástico. Esta característica permite que eles explorem o espaço de busca de maneira mais ampla, reduzindo o risco de ficarem consistentemente aprisionados no mesmo ótimo local. No entanto, essa exploração também pode levar à obtenção de soluções de variada qualidade, incluindo soluções subótimas.

Os algoritmos enumerativos também podem ser divididos em duas sub-classes: i) *pattern driven* e ii) *sample driven*. A diferença entre ambas está relacionada à forma como cada uma realiza a busca pela string média. No primeiro caso, todos os KMERS possíveis são gerados, enquanto no segundo, apenas os KMERS presentes no conjunto de dados são utilizados. Por utilizar força bruta, a desvantagem da primeira sub-classe em relação à segunda é o tempo de execução, pois todas as combinações são consideradas.

Em outras palavras, conforme o tamanho de k cresce, a técnica baseada em *pattern driven* se torna inviável. Sua vantagem consiste na obtenção de métricas de desempenho melhores do que os algoritmos baseados em *sample driven*, que por sua vez oferecem tempos de execução menores e podem ser empregados em uma gama maior de problemas. Em geral, a principal vantagem dos algoritmos enumerativos em relação às demais abordagens consiste no seu bom desempenho para encontrar padrões pequenos e bem conservados, porém, para padrões grandes e com baixo nível de conservação, as abordagens probabilísticas são mais indicadas.

2.3.1 Algoritmos enumerativos

O funcionamento básico dos algoritmos enumerativos se dá através da contagem da frequência dos padrões conservados. Isso pode ser realizado através de várias abordagens no

qual podem ser empregadas diversas estruturas de dados secundárias com o objetivo de auxiliar a contagem.

O algoritmo enumerativo mais comum é o de busca exaustiva (força bruta), que gera todas as combinações possíveis e utiliza uma função objetivo para avaliar o `ESCORE` de cada combinação. Independentemente do tipo de combinação gerada, essa abordagem tende a ser pouco eficiente em termos de desempenho. Por exemplo, se optarmos pela busca da *string* média, o número de combinações depende de b e k , sendo b a quantidade de símbolos no alfabeto e k a largura dos `MOTIFS`. Desta forma, teremos b^k combinações possíveis. Se optarmos pela otimização do conteúdo da informação, será preciso levar em consideração n e m , sendo n a quantidade de sequências no dataset, $m = t - k + 1$ o número de posições válidas e t a largura de cada sequência do dataset. Desta forma, o espaço de busca será n^m .

Gerar todas as combinações a priori não é uma boa estratégia em nenhum dos casos acima. Sendo assim, é possível utilizar alguns artifícios para melhorar o tempo de execução dos algoritmos de força bruta, como por exemplo a técnica `BRANCH AND BOUND`. A ideia intuitiva é pensar na resolução do problema de forma a colocar todas as combinações possíveis em uma árvore e podar (*Branch*) alguns galhos levando em consideração algum parâmetro (*Bound*). Por exemplo, considere a árvore de pesquisa da Figura 2.7. Cada nó p pertence a um nível e possui tamanho 4. Eles representam uma solução, sendo que neste caso $p_i \in \{1, 2\}$ indica a posição inicial dos `MOTIFS` na i -ésima sequência. Se $p_i = 0$, então nenhum valor de posição foi definido para a i -ésima sequência.

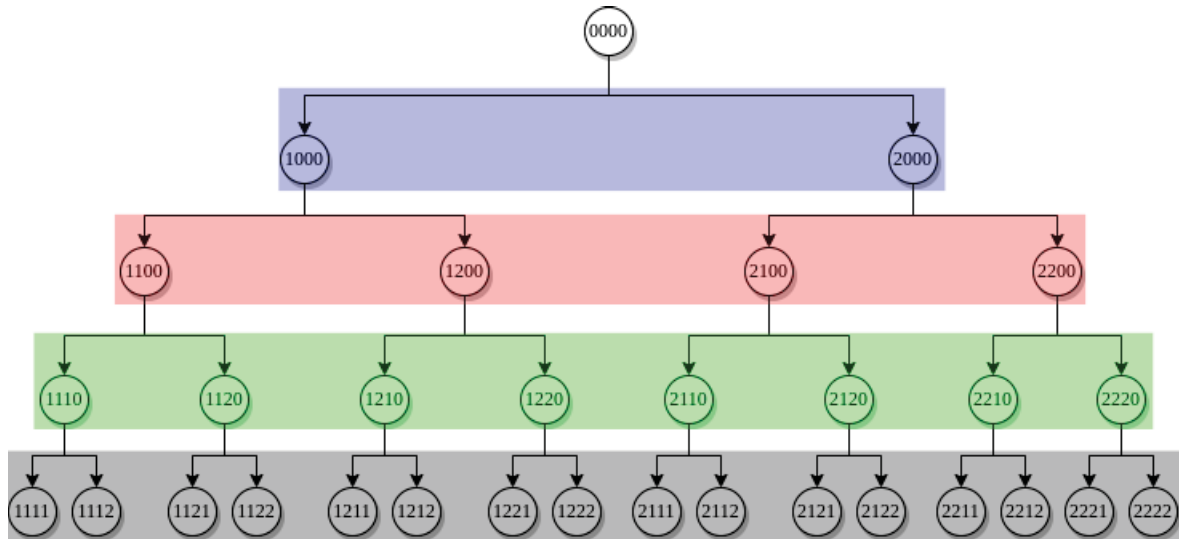


Figura 2.7: Árvore de busca para $T = 2$. Os níveis 1, 2, 3 e 4 são representados respectivamente pelas cores azul, vermelho, verde e preto. Fonte: o autor.

É interessante notar que neste exemplo, estamos modelando $n = 4$ sequências com $m = 2$. Se $m = t - k + 1$, então t e k podem variar livremente, desde que a restrição $m = 2$ se mantenha. Sem perda de generalidade, vamos considerar que $k = 10$ e $t = 11$. Desta forma, o menor `ESCORE` alcançável será $k = 10$ para $n \leq 4$ e $\lceil \frac{n}{4} \rceil k$ caso contrário. Isso ocorrerá se cada coluna do alinhamento possuir uma distribuição uniforme. O maior `ESCORE` atingível será nk e isso será verdade se e somente se o alinhamento for completamente conservado. Podemos utilizar essa informação para construir um algoritmo que seja capaz de podar a árvore sem a necessidade de gerar todos os nós.

Por exemplo, considere o nó $p = (1, 1, 1, 1)$ que pertence ao nível $i = 4$ da árvore de busca. De acordo com a definição acima, o `ESCORE` desta solução deverá ser algum número

natural entre 10 e 40. Sem perda de generalidade, suponha que seu *SCORE* seja $scop = 35$. Considere agora o nó $q = (2, 1, 0, 0)$ que pertence ao nível $i = 2$ da árvore de busca. O *SCORE* desta solução deverá estar situado entre 10 e 20. Suponha que seu *SCORE* seja $scoq = 10$. Desta forma, o maior *SCORE* alcançável será $opt_{scoq} = scoq + k(n - i)$, no qual opt_{scoq} é o *SCORE* mais otimista possível. Em particular, para este exemplo, $opt_{scoq} = 5 + 10(4 - 2) = 25$. Como $opt_{scoq} \leq scop$, então não existe a possibilidade de que outra solução que possua q como prefixo consiga obter um *SCORE* maior que $scop$. Assim, não será preciso gerar qualquer solução que esteja abaixo de q , conforme mostra a Figura 2.8. O Algoritmo 1 mostra um exemplo de implementação desta abordagem.

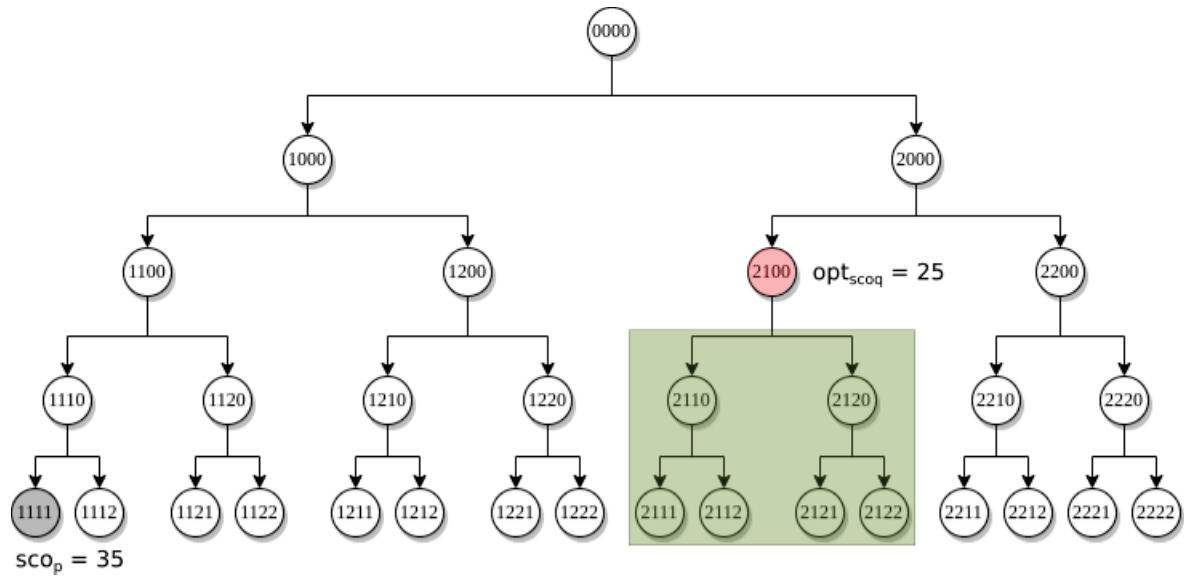


Figura 2.8: Árvore de busca para $T = 2$. Nenhum nó abaixo que possui como prefixo $(2, 1, 0, 0)$ precisará ser gerado, o que implicará em uma busca mais performática. Fonte: o autor.

Neste algoritmo, é importante destacar as linhas 6, 7, 8, 13 e 14. Na linha 6, é computado o *SCORE* otimista (através da Equação 2.1), que é o maior *SCORE* que uma solução arbitrária poderá atingir. Isso é realizado através da soma do *SCORE* da solução até aquele momento e o maior *SCORE* que esta solução poderia conquistar a partir daquele ponto. A linha 7 testa se o *SCORE* otimista é maior do que $best_score$. Se isso for verdade, então a linha 8 testará se $i \geq n$. Caso isso também seja verdade, então a solução possui n elementos e está completa, as variáveis $best_score$ e $best_motif$ são atualizadas e um novo vértice é gerado. A linha 13 será executada somente se a linha 8 retornar o valor lógico falso. Neste caso, o *SCORE* otimista é menor do que $best_score$. Se isso ocorrer, então não será necessário verificar outras soluções que estão abaixo deste nó na árvore. A função *bypass* é responsável por mudar o prefixo da solução. Por exemplo, se $s = (1, 1, 1, 0)$, então $bypass(s) = (1, 1, 2, 0)$. Vale ressaltar que é bastante simples adaptar o Algoritmo 1 para minimizar a *string* média. Devido ao menor espaço de busca da *string* média, essa modificação torna o algoritmo viável para trabalhar com tamanhos menores de *MOTIFS*.

Embora o algoritmo *BRANCH AND BOUND* represente uma melhoria significativa em relação aos algoritmos de força bruta em termos de desempenho, seu tempo de execução assintótico ainda é exponencial no pior caso. Por isso, tornou-se necessário desenvolver outras abordagens para lidar com problemas mais realistas que envolvem conjuntos de dados volumosos, compostos por milhares de sequências. A utilização de árvores de sufixo e suas variações pode oferecer uma solução parcial para essa questão.

Um problema bastante comum na análise de sequências biológicas é a correspondência exata de padrões (EPM, do inglês *Exact Pattern Matching*) que pode ser definida da seguinte

Algoritmo 1 BRANCH-AND-BOUND:

```

1:  $m \leftarrow t - k + 1$ 
2:  $best\_motif \leftarrow (0, 0, 0, \dots, 0)$ 
3:  $best\_score \leftarrow 0$ 
4:  $(s, i) \leftarrow next\_vertex(0, 0)$ 
5: while  $i > 0$  do
6:    $opt\_score = score(s, i) + k(n - i)$  // A etapa Bound é realizada aqui
7:   if  $opt\_score \geq best\_score$  then
8:     if  $i = n$  then
9:        $best\_score = opt\_score$ 
10:       $best\_motif = s$ 
11:    end if
12:     $(s, i) = next\_vertex(s, i)$ 
13:  else
14:     $(s, i) = bypass(s, i)$  // A etapa Branch é realizada aqui
15:  end if
16: end while
17: return  $(best\_score, best\_motif)$ 

```

maneira: seja um padrão $p = p_1, p_2, p_3, \dots, p_k$ e um texto longo $t = t_1, t_2, t_3, \dots, t_m$ o problema da correspondência exata de padrões consiste na identificação de todas as ocorrências do padrão p no texto t . Por exemplo, se $t = ATGGTCGGT$ e $p = GGT$, então o algoritmo EPM deve devolver como resultado as posições 3 e 7: AT GGT C GGT. O Algoritmo 2 mostra um exemplo simples de implementação do EPM.

Algoritmo 2 EXACT-PATTERN-MATCHING:

```

1:  $k \leftarrow |p|$ 
2:  $m \leftarrow |t| - k + 1$ 
3:  $pos \leftarrow \emptyset$ 
4: for  $i = 1$  to  $m$  do
5:    $t_i = t[i \dots (i + k - 1)]$ 
6:   if  $t_i = p$  then
7:      $pos.add(i)$ 
8:   end if
9: end for
10: return  $pos$ 

```

No Algoritmo 2, para cada posição i variando de 1 até m , o EPM irá comparar o padrão p com a janela deslizante $t[i \dots i + k - 1]$ e um *match* será encontrado se $p_1 = t_i, p_2 = t_{i+1}, p_3 = t_{i+2}, \dots, p_k = t_{i+k-1}$. O algoritmo gasta a maior parte do tempo descobrindo que o padrão p não está presente em t . No melhor caso, basta que $t_i \neq p_1$ para que o algoritmo desvie seu fluxo para o próximo padrão, t_{i+k} . Porém, no pior caso, o Algoritmo 2 é $O(mk)$. Um possível cenário seria $p = AAAAT$ e $t = AAAA \dots AAAAAA$. Neste caso, se m possuir digamos 1 milhão de pares de base, o EPM retornará como resultado que nenhuma posição foi encontrada não antes de realizar 5 milhões de operações.

Podemos calcular o desempenho do EPM em um texto aleatório. Considerando o alfabeto $\Sigma = \{A, C, G, T\}$, a probabilidade de $p_1 = t_i$ será $\frac{1}{4}$, portanto a probabilidade de serem diferentes será $1 - \frac{1}{4}$. Seguindo o mesmo raciocínio, a probabilidade de $p_1 = t_i$ e $p_2 = t_{i+1}$ será $\frac{1}{4^2}$ e a probabilidade de serem diferentes será $1 - \frac{1}{4^2}$. Portanto, a probabilidade

de que $p = t_{i\dots j}$ será $1 - \frac{1}{4^j}$. É fácil verificar que $\lim_{j \rightarrow \infty} \frac{1}{4^j} = 0$. Portanto, em um texto randômico, a probabilidade do algoritmo 2 realizar um número grande de comparações é bastante pequena. Considere a variável aleatória X que conta o número de comparações realizadas em um sequência qualquer. Se p é a probabilidade de um *mismatch*, então $X \sim GEOM(p)$. Desta forma, $\mathbb{E}(X) = \frac{1}{p} = \frac{1}{\frac{3}{4}} = 1,333 \dots \approx 1,33$. Isso significa que um texto constituído de m exemplares de *KMERS*, o número médio de comparações será $\frac{4}{3}m$. Como $\frac{4}{3}$ é uma constante, então a complexidade final do caso médio será $O(m)$.

Embora animadora, a análise acima é apenas sobre o caso médio e o EPM ainda será lento em datasets massivos. Além disso, se a busca por múltiplos padrões for realizada, a complexidade será $O(nm)$, no qual n é o número de padrões a serem testados. Como visto anteriormente, com uso de *Keywords Trees* (Aho e Corasick, 1975), é possível realizar a busca de correspondência exata no pior caso em tempo $O(k)$. Porém, o tempo para construir uma *KTREE* no pior caso é $O(m)$, sendo que a complexidade final será $O(m) + O(k)$. Para múltiplos padrões, uma *KTREE* apresenta complexidade de tempo $O(m) + O(km)$ que já foi mostrado ser melhor que $O(nm)$. O Algoritmo 3 mostra como a EPM é realizada em uma *KTREE*.

Algoritmo 3 *KTREE-EXACT-PATTERN-MATCHING*:

```

1: if  $i \geq k$  then
2:   return True
3: end if
4: if  $node.contains(p_i)$  then
5:   KTreeExactPatternMatching( $node.get(p_i), p, i + 1, k$ )
6: else
7:   return False
8: end if

```

O Algoritmo 3 foi definido de forma recursiva. Considerando um padrão p de tamanho $k = 10$, a chamada inicial será: *KTreeExactPatternMatching*(*root*, p , 1, 10). A linha 1 testa o caso base, i.e., se $i = k$ significa que $p_1 = t_i, p_2 = t_{i+1}, p_3 = t_{i+2}, \dots, p_k = t_{i+k-1}$ e portanto, o padrão foi encontrado. A linha 4 verifica se o nó atual contém o padrão p_i . Em caso positivo, a chamada recursiva é realizada. Caso contrário, o algoritmo retorna *False*, indicando que o padrão não pode ser encontrado. É fácil verificar através da linha 1 que o Algoritmo 3 fará no máximo k chamadas recursivas. A Figura 2.9 mostra um exemplo de uma busca bem e mal sucedida em uma *KTREE* arbitrária.

Além do EPM, outro problema comum que emerge da análise de sequências biológicas é o da correspondência aproximada de padrões (EPM, do inglês *Approximate Pattern Matching*). A diferença básica entre ambos é que o EPM não permite incompatibilidades e o EPM permite. Este problema pode ser definido como segue: seja um padrão $p = p_1, p_2, p_3, \dots, p_k$ e um texto longo $t = t_1, t_2, t_3, \dots, t_m$ o problema da correspondência aproximada de padrões consiste na identificação de todas as ocorrências do padrão p no texto t com no máximo d incompatibilidades. O Algoritmo 4 exibe as suas principais características.

As diferenças mais importantes entre o Algoritmo 2 (EPM) e o Algoritmo 4 (EPM) estão destacadas em vermelho, nas linhas 9-11 e 13-15. Na linha 9-11, o EPM calcula a distância de *Hamming* entre o padrão p e o texto $t[i \dots (i + j - 1)]$. Se a distância exceder o valor do parâmetro d , então a posição referente à essa *substring* não será retornada. A análise de complexidade do caso médio é a seguinte: seja X uma variável aleatória que conta a quantidade de comparações antes de d incompatibilidades. Se $d = 1$, então $X \sim GEOM(p)$, porém se $d > 1$, então $X \sim NB(d, p)$, no qual p e d representam a probabilidade e a quantidade de incompatibilidades

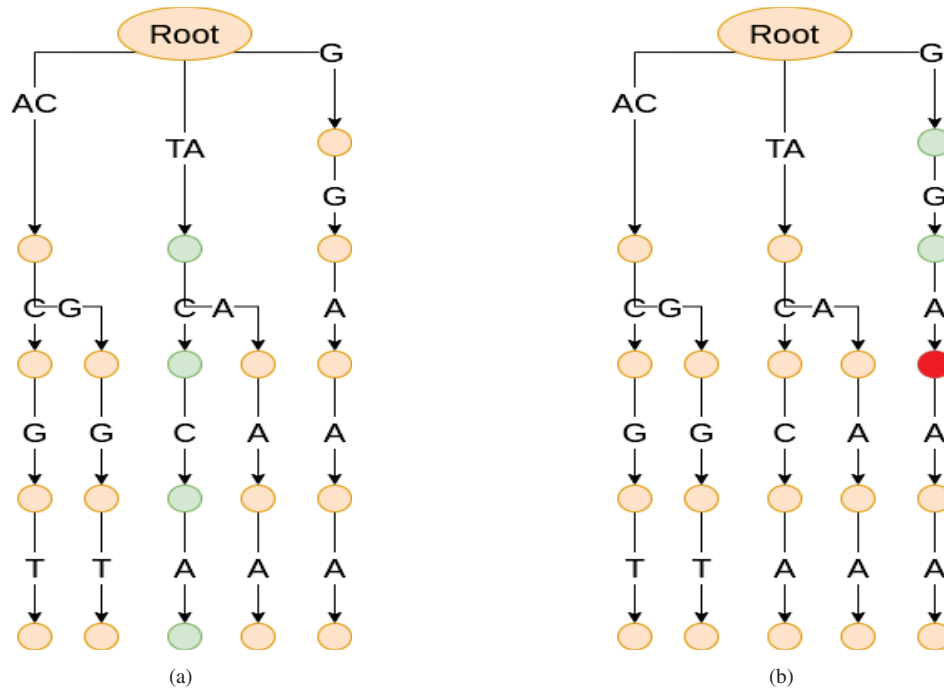


Figura 2.9: (a) Busca bem sucedida do padrão *TACCA*. (b) Busca mal sucedida do padrão *GGATT*. Fonte: o autor.

Algoritmo 4 APPROXIMATE-PATTERN-MATCHING:

```

1:  $w \leftarrow |p|$ 
2:  $m \leftarrow |t| - w + 1$ 
3:  $pos \leftarrow \emptyset$ 
4:  $k \leftarrow$  número máximo de incompatibilidades permitidas
5: for  $i = 1$  to  $m$  do
6:    $t_i = t[i \dots (i + w - 1)]$ 
7:    $dist = 0$ 
8:   for  $j = 1$  to  $w$  do
9:     if  $t[i, j] \neq p[j]$  then
10:       $dist = dist + 1$ 
11:    end if
12:  end for
13:  if  $dist \leq k$  then
14:     $pos.add(i)$ 
15:  end if
16: end for
17: return  $pos$ 

```

respectivamente. O valor esperado $\mathbb{E}(X) = \frac{d}{p}$, para $p = \frac{3}{4}$, temos $\mathbb{E}(X) = \frac{d}{\frac{3}{4}} = \frac{4d}{3} \approx 1,33d$. Para $d = 1$, temos exatamente a mesma complexidade do EPM. Porém, para $d > 1$, a complexidade será $O(dm)$. Isso significa que quanto maior é o valor de d , mais operações serão realizadas. Assim como no EPM, KTREES e STREES podem ser utilizadas para melhorar o tempo de execução do EPM. Estudos clássicos de algoritmos que utilizam métodos enumerativos na descoberta de MOTIFS SÃO: SPEELER (Sagot, 1998), WEEDER (Pavesi et al., 2001) e YMF (Sinha e Tompa, 2003);

2.3.2 Algoritmos probabilísticos

Os algoritmos desta categoria empregam modelos estatísticos para representar as sequências alvo. Geralmente, são utilizados dois modelos distintos: um para descrever as sequências de interesse e outro para representar as sequências de fundo ou controle, comumente referidas na literatura como `BACKGROUND`. Além disso, os algoritmos probabilísticos se dividem em duas sub-classe: i) determinísticos e ii) não-determinísticos. Dentre os algoritmos mais conhecidos na literatura que empregam modelos probabilísticos podemos destacar o `MEME` (Bailey e Elkan, 1995), `BIOPROSPECTOR` (Liu et al., 2004) e o `GIBBS MOTIF SAMPLER` (Hughes et al., 2000). Enquanto o primeiro é determinístico e dois últimos são não-determinísticos.

2.3.2.1 Cadeias de Markov

Cadeia de `MARKOV` é um tipo de processo estocástico discreto muito utilizado na bioinformática e na biologia computacional com o objetivo de modelar sequências biológicas (DURBIN et al., 1998). Ele é capaz de descrever uma sequência de eventos que ocorrem no tempo e possui como principal particularidade a falta de memória, caracterizada pela propriedade de `MARKOV`, cujo enunciado diz que a probabilidade de um evento ocorrer no tempo $i + 1$ depende apenas do evento atual no tempo i e não de todo o histórico de eventos passados $i - 1, i - 2, i - 3, \dots, i_2, i_1$. Esse tipo de modelo é chamado de cadeia de `MARKOV` de ordem 1. Entretanto, outros tipos podem ser utilizados, como cadeias de `MARKOV` de ordem τ para $\tau \geq 0$. Uma cadeia de `MARKOV` de ordem τ é aquela cuja a probabilidade de evento ocorrer no tempo $i + 1$ depende dos τ eventos anteriores. Se $\tau = 0$, então os eventos são independentes. O modelo apresentado nesta seção não inclui a análise de variáveis latentes que de acordo com Rabiner (1989) é denominada de cadeias de `MARKOV` observáveis. Na Seção 2.3.2.4 falaremos sobre outros tipos de modelos que presumem a existência de variáveis latentes.

A forma mais usual encontrada na literatura para descrever cadeias de `MARKOV` é através de grafos, conforme mostra a Figura 2.10. Neste exemplo, os estados são representados pelos nucleotídeos A, G, C, T e existe uma aresta ligando cada um deles a todos os outros (*fully connected*). Em cada aresta existe uma probabilidade condicional associada⁶ $a_{st} = P(x_i = t | x_{i-1} = s)$ conforme mostra a Equação 2.11. Isso pode ser entendido como a probabilidade de visualizar o símbolo t no tempo i se s foi o símbolo imediatamente anterior a t . Para calcular a_{st}^+ , a frequência absoluta c_{st}^+ é computada. Essa frequência pode ser interpretada como o número de vezes que o símbolo s sofreu transição para o símbolo t . Esse valor é normalizado pela frequência absoluta $\sum_{t'} c_{st'}^+$, que é a quantidade de vezes que o símbolo s sofreu transição para todos os outros símbolos $\{A, C, G, T\}$.

$$a_{st}^+ = \frac{c_{st}^+}{\sum_{t' \in \{A, C, G, T\}} c_{st'}^+} \quad (2.11)$$

Embora seja possível representar através de grafos pequenos sistemas com poucos estados, a representação de cadeias maiores com muitos estados se torna problemática. Por essa razão, é bastante comum representar cadeias de `MARKOV` através de matrizes de transição. A grande vantagem disso é que a matriz de transição é um objeto matemático que pode ser facilmente convertida em código programável. A Tabela 2.1 mostra um exemplo com duas matrizes de transição retiradas de (DURBIN et al., 1998).

⁶Omitida na figura para evitar a poluição da imagem.

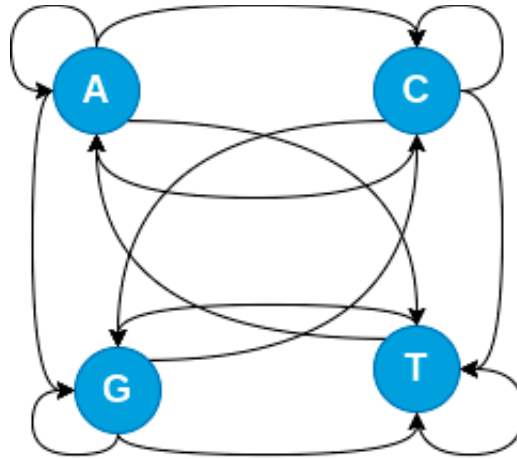


Figura 2.10: Grafo totalmente conectado entre os nucleotídeos A, C, G, T. Fonte: (DURBIN et al., 1998).

Tabela 2.1: (a) Modelo de Markov de ordem 1 construído a partir de dados extraídos de ilhas CpG do organismo *H.sapiens*. (b) Modelo de Markov de ordem 1 construído a partir de dados extraídos de regiões aleatórias do organismo *H.sapiens*.

+	A	C	G	T
A	0,180	0,274	0,426	0,120
C	0,171	0,368	0,274	0,188
G	0,161	0,339	0,375	0,125
T	0,079	0,335	0,384	0,182

a

-	A	C	G	T
A	0,300	0,205	0,285	0,210
C	0,322	0,298	0,078	0,302
G	0,248	0,246	0,298	0,208
T	0,177	0,239	0,292	0,292

b

Fonte: (DURBIN et al., 1998)

A Tabela 2.1a mostra o modelo transição positivo, construído a partir de 48 ilhas CpG⁷ com aproximadamente 60.000 nucleotídeos. A Tabela 2.1b mostra o modelo de transição negativo, construído com nucleotídeos selecionados de regiões aleatórias do genoma. Através destes modelos, podemos usar a propriedade de MARKOV e calcular facilmente a probabilidade de uma sequência x conforme mostra a Equação 2.12.

$$\begin{aligned}
 P(x) &= P(x_k, x_{k-1}, x_{k-2}, \dots, x_2, x_1) \\
 &= P(x_k | x_{k-1}) P(x_{k-1} | x_{k-2}) \dots P(x_1) \\
 &= P(x_1) \prod_{i=2}^k a_{x_{i-1}x_i}
 \end{aligned} \tag{2.12}$$

Conforme exige a Equação 2.13, o termo $P(x_1)$ pode ser suprimido da Equação 2.12, pois espera-se que a probabilidade da sequência x começar com qualquer um dos símbolos existentes no alfabeto seja a mesma. Neste caso, $\pi = \{A = 0,25, C = 0,25, G = 0,25, T = 0,25\}$. Desta forma, esse modelo se mantém homogêneo no sentido de que as probabilidades de transição são as mesmas em qualquer ponto no tempo (inclusive no início da cadeia). Caso as probabilidades se alterem em função do tempo, então o modelo é chamado de cadeia de MARKOV não homogênea.

$$P(x) = \prod_{i=1}^k a_{x_{i-1}x_i} \tag{2.13}$$

⁷Regiões genômicas ricas em CG, cuja principal causa é a falta de metilação.

Ainda na Equação 2.13, é possível observar que as probabilidades são multiplicadas k vezes, o que faz o valor de $P(x)$ decrescer rapidamente próximo a zero, causando instabilidade numérica quando implementado em sistemas computacionais. Uma alternativa é calcular o $\log$ da multiplicação, que algebricamente se transforma na soma dos $\log$ s, conforme mostra a Equação 2.14.

$$\begin{aligned}\log P(x) &= \log \prod_{i=1}^w a_{x_{i-1}x_i} \\ &= \sum_{i=1}^w \log a_{x_{i-1}x_i}\end{aligned}\quad (2.14)$$

Podemos ajustar as equações apresentadas acima para calcular o ESCORE da sequência x utilizando *log-odds ratio*, conforme mostra a Equação 2.15. $S(x) > 1$, indica maior probabilidade de x pertencer ao grupo positivo (CpG). $S(x) \leq 1$ indica maior probabilidade de x pertencer ao grupo negativo (controle).

$$\begin{aligned}S(x) &= \log \frac{P(x|\text{modelo+})}{P(x|\text{modelo-})} = \sum_{i=1}^w \log \frac{a_{x_{i-1}x_i}^+}{a_{x_{i-1}x_i}^-} \\ &= \sum_{i=1}^w \beta_{x_{i-1}x_i}\end{aligned}\quad (2.15)$$

Na Equação 2.15, podemos verificar que $\beta_{i,j}$ representa o logaritmo da divisão entre o modelo positivo e o modelo negativo, que em notação matricial seria $\beta = \log \frac{A^+}{A^-}$. Em outras palavras, esse cálculo representa o logaritmo da divisão entre as probabilidades de transição $a_{i,j}^+$ e $a_{i,j}^-$, conforme podemos ver na Tabela 2.2

Tabela 2.2: Matriz *log-odds ratio* construída a partir do cálculo do logaritmo resultante da divisão entre a matriz positiva e a matriz negativa.

β	A	C	G	T
A	-0,740	0,419	0,580	-0,803
C	-0,913	0,302	1,812	-0,685
G	-0,624	0,461	0,331	-0,730
T	-1,169	0,573	0,393	-0,679

Através da matriz 2.2, podemos discriminar facilmente novas sequências ainda não classificadas. Por exemplo, seja $x_1 = ACGCGTCG$ e $x_2 = ATTATATT$, com $k = 8$. De acordo com a Equação 2.15, $S(x_1) = 0,149 + 1,182 + 0,331 + 1,812 + (-0,730) + 0,393 + 1,182 = 3,689$ e $S(x_2) = (-0,803) + (-0,679) + (-1,169) + (-0,803) + (-1,169) + (-0,803) + (-0,679) = -6,105$. Portanto, de acordo com o modelo proposto, x_1 possui maior probabilidade de pertencer ao grupo positivo e x_2 de pertencer do grupo negativo. Fazendo uma rápida análise, fica evidente que existem muitas transições do tipo CG em x_1 o que eleva o seu valor de ESCORE. Da mesma forma, existem muitas transições do tipo TA em x_2 o que contribui para diminuir seu valor de ESCORE.

2.3.2.2 Position Weight Matrix

Também são conhecidas na literatura pela denominação *Position Specific Scoring Matrix* (PSSM). Formalmente, são modelos aditivos lineares de MARKOV não homogêneos de ordem zero,

muito utilizados na modelagem de MOTIFS biológicos sem lacunas (Bao et al., 2021; Stormo, 2000). As PWMs são importantes, pois possuem estreita relação com parâmetros bio-físicos de energia livre (Berg e von Hippel, 1988; Stormo e Fields, 1998). Existem outros tipos de representações que assumem a não-linearidade da energia livre⁸ que podem ser empregadas para esta finalidade, entretanto, de acordo com (Benos et al., 2002), As PWMs fornecem uma aproximação bastante precisa da realidade, capturando de forma fiel os aspectos fundamentais de um alinhamento de MOTIFS. Este modelo demonstra um ótimo desempenho na prática.

Diferentemente das cadeias de MARKOV vistas na Seção 2.3.2.1, as PWMs são do tipo não homogênea e possuem ordem zero, i.e., as probabilidades variam no tempo e existe a presunção de que os nucleotídeos de uma coluna do alinhamento ocorram de forma independente dos demais. Essa representação é muito utilizada para realizar a modelagem de sequências pequenas e essa é uma das principais razões pela qual elas são indicadas para modelagem de MOTIFS biológicos. A título de comparação, considere um dataset com sequências de tamanho k . A matriz de transição do modelo de MARKOV apresentado na Seção 2.3.2.1 não depende de k . Portanto essa matriz terá dimensões $n \times n$, mais especificamente 4×4 para sequências de DNA.

No modelo PWM, a matriz de transição depende de k e terá dimensões $4 \times k$. Portanto, quanto maior for o valor de k , mais larga será a matriz resultante. Como genes e outras regiões genômicas possuem milhares ou milhões de pares de base, o modelo PWM torna-se pouco eficiente para realizar a modelagem destes sítios. Por sua vez, MOTIFS costumam ser pequenos, algo em torno de $5 \leq k \leq 25$ pares de base, o que contribui para a sua utilização.

Da mesma forma que as matrizes de pontuação PAM e BLOSUM⁹, as PWMs são baseadas em *log-odds ratio*. De posse destas matrizes, podemos formular as seguintes hipóteses: H_0 (hipótese nula): as sequências não possuem relação entre si e H_a (hipótese alternativa): as sequências possuem relação entre si. Desta forma, é possível calcular o ESCORE de acordo com a Equação 2.16.

$$S(x, y) = \log_2 \frac{P(x, y|H_a)}{P(x, y|H_0)} \quad (2.16)$$

As PWMs utilizam *log-odds* de uma maneira similar. O objetivo é modelar a probabilidade de observar um determinado nucleotídeo em uma posição particular do alinhamento local, sendo sua construção realizada a partir de amostras de MOTIFS já anotados. Por exemplo, dado um alinhamento com n MOTIFS de tamanho k , é possível calcular a razão de verossimilhança através da Equação 2.17.

$$P(x, i) = \frac{q(x, i)}{p(x)}, i = 1, 2, 3, \dots, k \quad (2.17)$$

Onde $H_a = q(x, i)$ é a probabilidade de observar o nucleotídeo x na posição i e $H_0 = p(x)$ é a probabilidade de observar o nucleotídeo x em qualquer posição no genoma. Na literatura, $q(x, i)$ é denominada distribuição de interesse e $p(x)$ distribuição de fundo. A partir desta expressão, podemos calcular o *log-odds position specific scoring system* através da Equação 2.18.

$$S(x, i) = \log_2 P(x, i), i = 1, 2, 3, \dots, k \quad (2.18)$$

Por último, a Equação 2.19 apresenta o processo de cálculo da probabilidade de interesse q . No que diz respeito à distribuição de fundo p , há diversas abordagens para calculá-la. No

⁸Ainda não existem evidências suficientes que comprovem a não-linearidade desta associação.

⁹Consulte (Mount, 2008), para obter mais informações sobre essas matrizes.

entanto, a estratégia mais comum consiste em contar a quantidade de símbolos A, C, G, T e dividi-los pelo total de símbolos presentes. Uma outra maneira mais sofisticada é através de modelos de MARKOV de ordem τ , conforme mostrado na Seção 2.3.2.1. É interessante destacar que Thijs et al. (2001a) argumentaram que a representação de modelos de fundo utilizando cadeias de MARKOV de ordem $\tau \geq 1$, no geral, pode contribuir com a melhora na predição MOTIFS biológicos.

A modelagem do modelo de fundo é uma etapa importante para a separação efetiva de regiões conservadas do ruído genômico. Embora a utilização de cadeias de MARKOV de ordem superior possa capturar mais detalhes e complexidade na estrutura de sequências, é necessário considerar a possibilidade de sobre-ajuste (*overfitting*). Este fenômeno ocorre quando o modelo é excessivamente ajustado para captar as características do conjunto de dados de treinamento, comprometendo sua capacidade de generalizar para novos dados.

Além do mais, um outro efeito indesejado pode ocorrer. Uma cadeia de MARKOV de ordem superior consegue capturar muito mais informação contextual (dependências entre símbolos vizinhos) do que uma cadeia de ordem inferior. Se o tamanho dos MOTIFS é similar à ordem da cadeia de MARKOV empregada, a informação contextual adicional pode incluir partes dos próprios MOTIFS, o que pode causar o efeito inverso fazendo com que as regiões de interesse se pareçam mais com o ruído genômico, dificultando assim a sua detecção.

$$q(x, i) = \frac{c(x, i) + \psi}{n + |\Sigma|}, i = 1, 2, 3, \dots, k \quad (2.19)$$

É importante destacar alguns pontos na Equação 2.19. O primeiro, é o termo $c(x, i)$ que representa a frequência absoluta que o nucleotídeo x aparece na i -ésima coluna do alinhamento. O segundo ponto, é a constante ψ chamada de pseudo-contador. Ela é imprescindível para a estabilidade numérica da Equação 2.18. Para ver isso, suponha que $\psi = 0$ e $c(x, i) = 0$. Desta forma, tanto $p(x, i)$ quanto $P(x)$ também serão iguais a zero. Isso levará a Equação 2.18 para $-\infty$. Portanto, é importante que ψ seja um número positivo maior que zero. Em geral, escolhe-se valores pequenos para ψ . Para a maioria dos problemas, $\psi = 1$ será suficiente Lawrence et al. (1993). Por fim, o terceiro ponto a ser destacado diz respeito ao termo $|\Sigma|$, que representa o tamanho do alfabeto, neste caso $\Sigma = 4$.

Podemos empregar a Equação 2.18 para construir o modelo PWM. Em particular, $PWM_{x,i} = S(x, i)$. A Tabela 2.3 mostra um modelo PWM construído com aproximadamente de 5 mil MOTIFS¹⁰ do tipo NKX2-5 com tamanho $k = 11$, pertencentes ao organismo *H. sapiens*. Como $\log \frac{a}{b} = \log a - \log b$, então, cada índice deste modelo pode ser interpretado como a diferença, em bits, entre o logaritmo da distribuição de interesse e o logaritmo distribuição de fundo. Quanto maior esse valor, maior é a divergência entre as distribuições e consequentemente mais conservada será a coluna.

Após a construção do modelo PWM, podemos calcular o SCORE de uma sequência x arbitrária e classificá-la de acordo com as hipóteses iniciais: H_0 ou H_a . A Equação 2.20 mostra como essa computação é realizada. Através dela, podemos concluir que se $S(x) \geq 1$, então rejeitamos H_0 classificamos x como MOTIF, caso contrário, aceitamos H_0 e classificamos x como BACKGROUND.

$$S(x) = \sum_{i=1}^k S(x_i, i) = \sum_{i=1}^k PWM_{x_i, i} \quad (2.20)$$

¹⁰5.444 para ser mais exato.

Tabela 2.3: (a) Matriz de frequência absoluta. (b) Matriz de frequência relativa normalizada por coluna. (c) Modelo PWM.

	1	2	3	4	5	6	7	8	9	10	11
A	1723	1526	96	5246	144	135	222	4819	4931	1485	1827
C	1024	1646	5002	48	5098	82	4319	120	165	1406	1251
G	1687	1318	24	76	49	61	62	217	98	1478	1084
T	1010	954	322	74	153	5166	841	288	250	1075	1282

a

	1	2	3	4	5	6	7	8	9	10	11
A	0,32	0,28	0,02	0,96	0,03	0,02	0,04	0,89	0,91	0,27	0,34
C	0,19	0,30	0,92	0,01	0,94	0,02	0,79	0,02	0,03	0,26	0,23
G	0,31	0,24	0,00	0,01	0,01	0,01	0,01	0,04	0,02	0,27	0,20
T	0,19	0,18	0,06	0,01	0,03	0,95	0,15	0,05	0,05	0,20	0,24

b

	1	2	3	4	5	6	7	8	9	10	11
A	0,34	0,17	-3,83	1,95	-3,24	-3,33	-2,62	1,82	1,86	0,13	0,42
C	-0,41	0,27	1,88	-4,83	1,91	-4,05	1,67	-3,50	-3,04	0,05	-0,12
G	0,31	-0,05	-5,83	-4,16	-4,80	-4,48	-4,46	-2,65	-3,80	0,12	-0,33
T	-0,43	-0,51	-2,08	-4,20	-3,15	1,92	-0,69	-2,24	-2,44	-0,34	-0,09

c

Como exemplo, considere $x = ACCACTCAAAA$ e o modelo PWM exibido na Tabela 2.3c. Aplicando a Equação 2.20, temos que $S(x) = 0,34 + 0,27 + 1,88 + 1,95 + 1,91 + 1,92 + 1,67 + 1,82 + 1,86 + 0,13 + 0,42 = 14,17$ bits. Como $S(x) \geq 1$, então a hipótese nula é rejeitada e x é classificado como MOTIF. De fato, $x = ACCACTCAAAA$ é a sequência CONSENSO, pois $x_i = \max PWM_{x_i,i}$, para $i = 1, 2, 3, \dots, k$. Considere agora $y = CTGCGGGCGTG$. Aplicando a mesma equação, temos que $S(y) = (-0,43) + (-0,51) + (-5,83) + (-4,83) + (-4,80) + (-4,48) + (-4,46) + (-3,50) + (-3,80) + (-0,34) + (-0,33) = -24,35$ bits. Como $S(y) < 1$, então a hipótese nula é aceita e a sequência é classificada como BACKGROUND. As Tabela 2.4a e 2.4b exibem a computação de $S(x)$ e $S(y)$.

Tabela 2.4: (a) Cálculo de $S(x = ACCACTCAAAA)$. (b) Cálculo de $S(y = CTGCGGGCGTG)$. Os caminhos em azul e vermelho mostram as células selecionadas para realizar os dois cálculos respectivamente.

	1	2	3	4	5	6	7	8	9	10	11
A	0,34	0,17	-3,83	1,95	-3,24	-3,33	-2,62	1,82	1,86	0,13	0,42
C	-0,41	0,27	1,88	-4,83	1,91	-4,05	1,67	-3,50	-3,04	0,05	-0,12
G	0,31	-0,05	-5,83	-4,16	-4,80	-4,48	-4,46	-2,65	-3,80	0,12	-0,33
T	-0,43	-0,51	-2,08	-4,20	-3,15	1,92	-0,69	-2,24	-2,44	-0,34	-0,09

a

	1	2	3	4	5	6	7	8	9	10	11
A	0,34	0,17	-3,83	1,95	-3,24	-3,33	-2,62	1,82	1,86	0,13	0,42
C	-0,41	0,27	1,88	-4,83	1,91	-4,05	1,67	-3,50	-3,04	0,05	-0,12
G	0,31	-0,05	-5,83	-4,16	-4,80	-4,48	-4,46	-2,65	-3,80	0,12	-0,33
T	-0,43	-0,51	-2,08	-4,20	-3,15	1,92	-0,69	-2,24	-2,44	-0,34	-0,09

b

Por fim, podemos calcular o valor esperado de $\mathbb{E}(S(x))$, para todo $x \in X$. Esse valor fornece o `SCORE` do alinhamento como um todo e representa a esperança da diferença logarítmica entre as distribuições de interesse e fundo medido em bits, podendo ser empregado para realizar a seleção de modelos. $\mathbb{E}(S(x))$ pode ser calculado considerando todas as sequências $x \in X$ de tamanho k , conforme mostra a Equação 2.21.

$$\mathbb{E}(S(X)) = \sum_{x \in X} Q(x)S(x) \quad (2.21)$$

Na Equação 2.21, $Q(X)$ representa a probabilidade de x pertencer à distribuição de interesse e $S(x)$ representa o *log odds ratio* exibida na Equação 2.20. Um problema desta abordagem é que $|X|$ cresce exponencialmente com k . Sendo assim, gerar todas as combinações possíveis pode se tornar rapidamente inviável. Uma forma alternativa de realizar o mesmo cálculo é processando a soma de todos os elementos do modelo `PWM`, conforme mostra a Equação 2.22.

$$\mathbb{E}(S(X)) = \sum_{i=1}^4 \sum_{j=1}^k Q_{i,j} PWM_{i,j} \quad (2.22)$$

Como exemplo, considere o modelo `PWM` da Tabela 2.3c. Aplicando a Equação 2.22, temos que $\mathbb{E}(S(X)) \approx 10,32$ bits. Isso significa que a diferença entre a distribuição de interesse e a distribuição de fundo é de aproximadamente 10,32 bits. Quanto maior esse valor, mais conservado será o alinhamento como um todo. O valor esperado de $S(X)$, obtido através Equação 2.22, também é conhecido na literatura como distância de Kullback-Leibler ou Conteúdo da Informação, ambos discutidos no início deste capítulo.

Por fim, o modelo `WAM` (*Weight Array Methods*) proposto por Zhang e Marr (1993), é uma generalização do modelo `PWM` que consegue capturar dependências entre posições adjacentes. Entretanto, devido à sua maior complexidade e à necessidade de estimar uma quantidade consideravelmente maior de parâmetros, o modelo `WAM` é menos eficiente em termos computacionais. Como resultado, é menos frequente seu uso na modelagem de `MOTIFS` biológicos em comparação ao modelo `PWM`.

Entretanto, em casos bastantes específicos, quando a dependência entre as bases do alinhamento são mais evidentes, o modelo `WAM` pode obter resultados melhores do que o modelo `PWM`. Contudo, essa não é a regra e, no geral, o modelo `PWM` é mais simples e acaba se sobressaindo em relação aos demais. Além do mais, a Navalha de Ocam (Hamilton, 1861) parece funcionar bem neste caso. Outros modelos probabilísticos com potencial para ser utilizados na modelagem de `MOTIFS` biológicos podem ser encontrados em (Siddharthan, 2010), (Mathelier e Wasserman, 2013), (Lee et al., 2011a), (Lee et al., 2011b) e (Burge e Karlin, 1997).

2.3.2.3 Modelos de mistura

Conforme mostrado até o momento, estamos interessados em modelar duas distribuições de probabilidades: a distribuição de interesse que contém os `MOTIFS` e a distribuição de fundo, que não contém (para facilitar o entendimento, a partir deste ponto, chamarei a distribuição de interesse de α e a distribuição de fundo de β). Entretanto, o que temos inicialmente são datasets com centenas ou milhares de sequências sem nenhuma anotação e não sabemos a priori o que representam as diversas regiões presentes em sua estrutura interna. Desta forma, além da escolha do melhor modelo, temos dois problemas a serem resolvidos: i) clusterizar os dados; e ii) estimar os parâmetros do modelo. Após a escolha do modelo apropriado, se as distribuições são conhecidas a priori, então a estimação dos parâmetros é realizada de maneira direta, através das amostras fornecidas, conforme discutimos nas Seções 2.3.2.1 e 2.3.2.2.

Por outro lado, se as distrições subjacentes e as posições na qual os MOTIFS ocorrem não são conhecidas, então primeiro será preciso estimá-las. Em particular, muitos algoritmos de descoberta de MOTIFS executam essas duas etapas, uma após a outra, até a convergência. As misturas finitas de distribuições (MFM, do inglês *Mixture Finite Models*), como o próprio nome sugere, são empregadas em situações nas quais temos um conjunto de dados cujas amostras advêm de duas ou mais fontes geradoras diferentes e quando o uso de distribuições isoladas acarretam baixo desempenho discriminatório. As MFMS são formadas por K distribuições de probabilidades com κ_i parâmetros a serem estimados. Neste modelo, a correlação entre as variáveis observadas são explicadas através de variáveis ocultas, chamadas de variáveis latentes. Podemos calcular a função densidade de probabilidade (PDF, do inglês *Probability Density Function*) ou a função de massa de probabilidade (PMF, do inglês *Probability Mass Function*) de uma MFM conforme mostra a Equação 2.23.

$$p(x) = \sum_{k=1}^K w_k p_k(x) \quad (2.23)$$

Na Equação 2.23, w é um vetor de probabilidade, no qual w_k indica a plausibilidade de p_k , cujo comportamento é mostrado na Equação 2.24 e $p_k(x)$ é a k -ésima PDF (ou PMF no caso discreto). De acordo com a Equação 2.24, w_k é um número entre 0 e 1 e a soma $w_1 + w_2 + w_3 + \dots + w_k = 1$, portanto w representa a distribuição a priori ou peso inicial dos k -modelos. Como exemplo, considere que temos $k = 2$, $w_1 = 0,5$, $w_2 = 0,5$ e as distribuições $p_1(x) \sim EXP(\lambda)$ e $p_2(x) \sim N(\mu, 1)$. Desta forma, $p(x)$ é dada através da distribuição preditiva $p(x) = w_1 p_1 + w_2 p_2 = w_1 EXP(\lambda) + w_2 N(\mu, 1)$. Para $x = 3,5$, $\mu = 2$ e $\lambda = 1$, temos que $p(x) = w_1 \lambda \exp^{-\lambda x} + w_2 \frac{1}{\sqrt{2\pi}} \exp^{-\frac{1}{2}(x-\mu)^2} = 0,5 \times 0,03 + 0,5,13 = 0,079$.

$$0 \leq w_k \leq 1, \sum_{k=1}^K w_k = 1 \quad (2.24)$$

Considerando que $x_i \perp x_j, \forall i \neq j$ e $i, j \in \{1, 2, 3, \dots, n\}$, podemos utilizar a Equação 2.23 para calcular a estimativa de máxima verossimilhança (MLE, do inglês *Maximum Likelihood Estimation*) através da Equação 2.25.

$$\begin{aligned} P(X) &= \prod_{x \in X} p(x|\theta) \\ P(X) &= \prod_{x \in X} \sum_{k=1}^K w_k p_k(x|\theta_k) \end{aligned} \quad (2.25)$$

A Equação 2.25 multiplica $p(x|\theta)$ n vezes, causando instabilidade numérica. Por essa razão, é melhor trabalhar com o $\log p(x|\theta)$, pois, desta forma, trocamos a multiplicação das probabilidades pela soma. Como a Equação 2.25 é monotonicamente crescente, o valor de $\log p(x|\theta)$ exibe as mesmas relações de ordem que $p(x|\theta)$, tal que $p(x|\theta_1) > p(x|\theta_2) \iff \log p(x|\theta_1) > \log p(x|\theta_2)$. Além do mais, computacionalmente falando, a soma é mais eficiente que a multiplicação. Por essas razões, o $\log$ de $P(X)$ é computado, resultando na Equação 2.26 que é conhecida na literatura como *log likelihood* ou *log* da verossimilhança.

$$\begin{aligned}
\log P(X) &= \log \prod_{x \in X} \sum_{k=1}^K w_k p_k(x|\theta_k) \\
\log P(X) &= \sum_{x \in X} \log \sum_{k=1}^K w_k p_k(x|\theta_k) \\
\mathcal{L}(\theta; X) &= \sum_{x \in X} \log \sum_{k=1}^K w_k p_k(x|\theta_k)
\end{aligned} \tag{2.26}$$

A Equação 2.26 fornece uma maneira eficiente de computar a probabilidade conjunta de todas as amostras do dataset dado θ sem o problema da instabilidade numérica. O problema agora torna-se de otimização, cujo objetivo consiste em encontrar valores para θ que maximize a Equação 2.26, conforme mostra a Equação 2.27.

$$\max_{\theta} \mathcal{L}(\theta; X) \tag{2.27}$$

Infelizmente não é possível obter uma solução analítica para a Equação 2.27. Mas, felizmente, existem outras maneiras de lidar ela. Neste trabalho, veremos como resolvê-la utilizando a maximização de expectativa (EM, do inglês *Expectation Maximization*). É interessante destacar que na análise de MOTIFS, frequentemente o modelo de mistura utilizado é dado pelos seguintes parâmetros: $k = 2$, $\theta = \{\alpha, \beta, w_\alpha, w_\beta\}$ tal que $w_\alpha \sim \text{BETA}$, $w_\beta = 1 - w_\alpha$, $p_\alpha(x) \sim \text{CATEGORICAL}(\alpha)$ e $p_\beta(x) \sim \text{CATEGORICAL}(\beta)$, α e $\beta \sim \text{DIRICHLET}^{\text{u}}$.

Outro modelo de mistura bastante empregado na bioinformática e na descoberta de MOTIFS são as cadeias ocultas de MARKOV (HMM, do inglês *Hidden Markov Chain*). A principal diferença entre HMM e os modelos apresentados anteriormente consiste na dependência estatística entre os estados. No HMM, os estados não são visíveis e são modelados através de uma cadeia de MARKOV de ordem τ (em muitos casos $\tau = 1$). Isso significa que, para $s \in S$, $s_i \perp s_j$ não ocorre para estados adjacentes, sendo que S é o conjunto de todos os estados possíveis. Não podemos confundir a dependência entre os estados ocultos com a dependência entre as variáveis observáveis. No modelo HMM canônico, as variáveis observáveis são independentes.

De acordo com Rabiner (1989), um modelo HMM é constituído pelos seguintes elementos: i) n estados ocultos; ii) m símbolos; iii) matriz de distribuição de probabilidade de transição A ; iv) matriz de distribuição de probabilidade dos símbolos observáveis B ; e v) vetor de distribuição de probabilidade inicial w . O modelo final é $\theta = \{A, B, w\}$. Ainda segundo Rabiner (1989), através deste modelo, é possível responder as seguintes questões.

1. Dado as observações $X = \{x_1, x_2, x_3, \dots, x_t\}$ e o modelo $\theta = \{A, B, w\}$, como computar de maneira eficiente $P(X|\theta)$, a probabilidade das observações dado o modelo?
2. Dado as observações $X = \{x_1, x_2, x_3, \dots, x_t\}$ e o modelo $\theta = \{A, B, w\}$, como escolher uma sequência de estados $Q = \{q_1, q_2, q_3, \dots, q_t\}$ que é ótima?
3. Como reestimar os parâmetros do modelo $\theta = \{A, B, w\}$ para maximizar $P(X|\theta)$?

Para responder a primeira pergunta, o algoritmo denominado FORWARD (FW) é utilizado. Ele possui este nome porque calcula a probabilidade de uma sequência arbitrária da esquerda

^uA Distribuição de Dirichlet é uma distribuição de probabilidade contínua multivariada, usada para modelar proporções com soma total restrita a 1. Serve como a distribuição a priori em modelos bayesianos para proporções, sendo parametrizada por um vetor de parâmetros positivos que influenciam sua forma.

para direita. Para entender como esse algoritmo funciona, considere a sequência de observações $X = \{x_1, x_2, x_3, \dots, x_t\}$. Se a sequência de estados $Q = \{q_1, q_2, q_3, \dots, q_t\}$ for conhecida, então podemos calcular $P(X|\theta)$, conforme mostra a Equação 2.28.

$$\begin{aligned} P(X|Q, \theta) &= \prod_{i=1}^t b_{q_i}(x_i) \\ &= b_{q_1}(x_1)b_{q_2}(x_2)b_{q_3}(x_3) \dots b_{q_t}(x_t) \end{aligned} \quad (2.28)$$

Na Equação 2.28, o termo $b_{q_i}(x_i)$ se refere a probabilidade de emissão do símbolo x_i no estado q_i . Como nesta equação assumimos que os estados são conhecidos, então não precisamos utilizar a matriz de transição. A probabilidade de um sequência arbitrária de estados é definida pela Equação 2.29.

$$\begin{aligned} P(Q|\theta) &= w_{q_1} \prod_{i=2}^t a_{q_{i-1}, q_i} \\ &= w_{q_1} a_{q_1, q_2} a_{q_2, q_3} a_{q_3, q_4} \dots a_{q_{t-1}, q_t} \end{aligned} \quad (2.29)$$

A Equação 2.29 computa a probabilidade de um estado Q arbitrário, sem levar em consideração as probabilidades de emissão. Através de w_{q_1} , a probabilidade da sequência começar no estado q_1 é calculada. Em seguida, a matriz de transição A é utilizada para calcular as transições entre $q_i, q_j, i < j | j = i + 1$. Enfim, unindo as Equações 2.28 e 2.29, chegamos a Equação 2.30, que considera a transição de estados e a emissão dos símbolos.

$$\begin{aligned} P(X|\theta) &= \sum_{Q \in \mathbb{Q}} P(X|Q, \theta) P(Q|\theta) \\ &= \sum_{Q \in \mathbb{Q}} w_{q_1} b_{q_1}(x_1) \prod_{i=2}^t a_{q_{i-1}, q_i} b_{q_i}(x_i) \end{aligned} \quad (2.30)$$

A Equação 2.30 pode ser interpretada como segue. No tempo $t = 1$, estamos no estado q_1 com probabilidade w_{q_1} na qual o símbolo x_1 é emitido com probabilidade $b_{q_1}(x_1)$. No tempo $t = 2$, o algoritmo deixa o estado q_1 e vai em direção ao estado q_2 com probabilidade a_{q_1, q_2} . Neste estado o símbolo x_2 é emitido com probabilidade $b_{q_2}(x_2)$. Isso continua até realizarmos a última transição no tempo $t = n$ do estado q_{n-1}, q_n com probabilidade a_{q_{n-1}, q_n} . Neste estado, o último símbolo é emitido com probabilidade $b_{q_n}(x_n)$. Para calcular $P(X|\theta)$, basta computar o somatório dos cálculos exibidos acima para todo $Q \in \mathbb{Q}$. Porém, é fácil verificar que a quantidade de combinações possíveis para $\mathbb{Q}$ é exponencial em n , exibindo a seguinte complexidade assintótica $O(N^n)$. Por essa razão, a Equação 2.30 torna-se rapidamente impraticável para conforme n cresce.

O algoritmo FORWARD (FW) consegue resolver a Equação 2.30 utilizando programação dinâmica muito mais rapidamente com o auxílio de uma matriz α de computações intermediárias. As Equações 2.31, 2.32 e 2.33 mostram todas as etapas deste algoritmo.

$$\text{Inicialização : } \alpha_{i,1} = w_i O_i(x_1), 1 \leq i \leq N \quad (2.31)$$

$$\text{Indução : } \alpha_{i,t+1} = b_{q_i}(x_{t+1}) \sum_{j=1}^N a_{i,j} \alpha_{j,t}, 1 \leq j \leq N, 1 \leq t \leq n-1 \quad (2.32)$$

$$\text{Terminação : } P(X|\theta) = \sum_{i=1}^N \alpha_{i,n} \quad (2.33)$$

É importante destacar que a matriz α possui dimensões $N \times n$, sendo que cada linha representa um estado e cada coluna a variação temporal. A Equação 2.31 mostra a etapa de inicialização, na qual a probabilidade da sequência começar nos estados $q_1, q_2, q_3, \dots, q_N$ e emitir o símbolo x_1 é computada. A Equação 2.32 mostra a etapa de indução, no qual a seguinte probabilidade condicional é calculada: $P(x_1, x_2, x_3, \dots, x_k, q_t = s_i | \theta)$. Portanto, a posição $\alpha_{i,t}$ representa a observação parcial da sequência $x_1, x_2, x_3, \dots, x_t$ e do estado s_i no tempo t , dado os parâmetros definidos em θ . Por fim, a terminação é exibida na Equação 2.33 no qual é computada a soma entre os elementos de $\alpha_{1,n}, \alpha_{2,n}, \alpha_{3,n}, \dots, \alpha_{N,n}$, concluindo o algoritmo.

Além do FW, existe ainda o procedimento BACKWARD (BW), que atua de forma ligeiramente diferente e faz os mesmos cálculos de maneira invertida. BW é importante apenas se for necessário calcular o melhor conjunto de estados que explicam as observações ou se quisermos reestimar os parâmetros do modelo θ . No primeiro caso, os principais algoritmos são: i) FORWARD-BACKWARD; e ii) VITERBI. Já para o segundo caso, o algoritmo mais utilizado é o BAUM-WELCH. Uma boa revisão sobre estes algoritmos pode ser encontrada em (Drake, 1967), (Rabiner, 1989) e (DURBIN et al., 1998).

2.3.2.4 Maximização da expectativa

O algoritmo de maximização da expectativa (EM, do inglês *Expectation Maximization*) é uma ferramenta muito utilizada para estimar parâmetros de modelos estatísticos por meio da otimização da função de máxima verossimilhança na presença de variáveis latentes. Ele também pode ser considerado um método para estimar a função de densidade de probabilidade (PDF) ou função de massa de probabilidade (PMF) de modelos estatísticos quando as equações que representam o modelo não podem ser resolvidas diretamente. Isso ocorre, por exemplo, em modelos de mistura finita, nos quais variáveis latentes e parâmetros desconhecidos impedem a resolução analítica do problema (Deisenroth et al., 2020).

Conforme ilustrado na Equação 2.23 na Seção 2.3.2.3, os modelos de mistura podem ser representados como uma combinação convexa de κ distribuições de probabilidades. Além disso, introduzimos a Equação 2.26 que descreve a função de máxima verossimilhança para esse tipo de modelo. Infelizmente essa equação não pode ser resolvida diretamente, devido à presença de parâmetros ocultos ou latentes, e por essa razão o algoritmo não pode encontrar o ótimo global. Porém, existe um resultado matemático muito importante que garante a este método encontrar ótimos locais (Bishop e Nasrabadi, 2006). Como resultado, é comum que estratégias de inicialização inteligentes sejam utilizadas em conjunto com EM. As Equações 2.34, 2.35, 2.36 e 2.37 exibem mais uma vez a função de verossimilhança.

$$\log P(X) = \log \prod_{x \in X} \sum_{k=1}^K w_k p_k(x|\theta_k) \quad (2.34)$$

$$\log P(X) = \sum_{x \in X} \log \sum_{k=1}^K w_k p_k(x|\theta_k) \quad (2.35)$$

$$\mathcal{L}(\theta; X) = \sum_{x \in X} \log \sum_{k=1}^K w_k p_k(x|\theta_k) \quad (2.36)$$

$$\mathcal{L}(\theta; X) = \sum_{x \in X} \log P(x|\theta) \quad (2.37)$$

Lembrando que o objetivo é maximizar $\mathcal{L}(\theta; X)$ com relação à $\theta = \{w_k, \theta_k, k = 1, 2, 3, \dots, K\}$. Para isso, precisamos calcular as derivadas parciais, conforme mostram as Equações 2.38 e 2.39.

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{\partial \sum_{x \in X} \log P(x|\theta)}{\partial \theta} \quad (2.38)$$

$$= \sum_{x \in X} \frac{\partial \log P(x|\theta)}{\partial \theta} \quad (2.39)$$

Como a derivada é um operador linear, na Equação 2.38, a soma das derivadas se transforma na derivada das somas, o que resulta na Equação 2.39. Calculando a derivada da Equação 2.39, ficamos com a Equação 2.40.

$$\frac{\partial \log P(x|\theta)}{\partial \theta} = \frac{1}{P(x|\theta)} \frac{\partial P(x|\theta)}{\partial \theta} \quad (2.40)$$

Resolvendo a Equação 2.40 para θ_k e w_k , obtemos a Equação 2.41 e 2.42. Nestas equações, temos dois termos, r_{nk} e N_k que são calculados conforme mostra a Equações 2.43 e 2.44.

$$\theta_k^{new} = \frac{1}{N_k} \sum_{x \in X} x r_{nk} \quad (2.41)$$

$$w_k^{new} = \frac{N_k}{N} \quad (2.42)$$

O termo r_{nk} presente na Equação 2.41, pode ser interpretado como a responsabilidade da k -ésima distribuição sobre a n -ésima amostra. Em outras palavras, r_{nk} nada mais é do que a regra de Bayes que computa a probabilidade a posteriori $P(\theta_k|x_n)$, conforme mostra a Equação 2.43. O termo N_k , presente na Equação 2.42 calcula a soma de r_{nk} . Isso pode ser entendido como a parcela de N que pertence ao k -ésima distribuição, sendo que $N = N_1, N_2, N_3, \dots, N_k$.

$$P(\theta_k|x) = r_{nk} = \frac{w_k P(x|\theta_k)}{\sum_{j=1}^K w_j P(x|\theta_j)} \quad (2.43)$$

$$N_k = \sum_{n=1}^N r_{nk} \quad (2.44)$$

No contexto do algoritmo EM, Equações 2.43 e 2.44 são executadas na etapa de *expectation*, pois estão envolvidas no cálculo da probabilidade das amostras, e as Equações 2.41 e 2.42 são executadas na etapa de *maximization*, pois realizam a reestimação dos parâmetros do modelo. Como exemplo, considere as amostras $X = \{AACT, CGTA, ACGT, TACG\}$ e o modelo inicial $\theta = \{\alpha, \beta, w_\alpha, w_\beta\}$ tal que α e $\beta \sim \text{DIRICHLET}$, $p(x|\alpha) \sim \text{CATEGORICAL}(\alpha)$, $p(x|\beta) \sim \text{CATEGORICAL}(\beta)$, $w_\alpha \sim \text{BETA}$ e $w_\beta = w_\alpha - 1$, conforme mostram as Equações 2.45, 2.46 e 2.47. Os hiperparâmetros de concentração da distribuição de Dirichlet e da distribuição Beta podem ser escolhidos tal que o suporte seja próximo a uma distribuição uniforme. A Figura 2.11 exibe as etapas fundamentais do algoritmo EM.

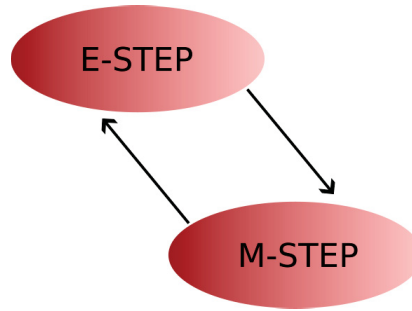


Figura 2.11: Representação gráfica simplificada do algoritmo EM. As etapas *E-STEP* e *M-STEP* são executadas alternadamente até que a convergência seja atingida. Fonte: o autor.

$$\alpha = \begin{bmatrix} A & 0.26 & 0.25 & 0.29 & 0.20 \\ C & 0.25 & 0.14 & 0.27 & 0.34 \\ G & 0.22 & 0.20 & 0.18 & 0.40 \\ T & 0.19 & 0.24 & 0.30 & 0.27 \end{bmatrix} \quad (2.45)$$

$$\beta = \begin{bmatrix} A & C & G & T \\ 0.19 & 0.23 & 0.20 & 0.38 \end{bmatrix} \quad (2.46)$$

$$w_\alpha = 0.30, w_\beta = 1 - w_\alpha = 0.70 \quad (2.47)$$

É importante destacar alguns pontos. Para gerar os modelos das Equações 2.45 e 2.46, foi utilizada uma distribuição de Dirichlet com hiperparâmetros de concentração $\{10, 10, 10, 10\}$. O valor de w_α foi escolhido através de uma distribuição Beta com parâmetros $\{10, 10\}$. A Tabela 2.5 sumariza os valores obtidos após o cálculo de todas as probabilidades.

As colunas mais importante da Tabela 2.5 são as que calculam os valores de $r_{i\alpha}$ e $r_{i\beta}$, pois elas serão utilizadas na reestimação de $\theta = \{\alpha, \beta, w_\alpha, w_\beta\}$, de acordo com as Equações 2.48, 2.49, 2.50 e 2.51.

Tabela 2.5: Cálculo das probabilidades utilizando o modelo representado pelas Equações 2.45, 2.46 e 2.47.

Amostras	$P(x_i \alpha)$	$P(x_i \beta)$	$P(x_i \theta = \{\alpha, \beta, w_\alpha, w_\beta\})$	$r_{i\alpha} = P(\alpha x_i)$	$r_{i\beta} = P(\beta x_i)$
AACT	0,0035	0,0032	0,00329	0,32	0,68
CGTA	0,0051	0,0033	0,00384	0,40	0,60
ACGT	0,0018	0,0033	0,00285	0,19	0,81
TACG	0,0030	0,0033	0,00321	0,28	0,72

$$\alpha_{j,A} = \sum_{j=1}^w \mathbb{1}\{x_{i,j} = A\} r_{i\alpha} \quad (2.48)$$

$$\beta_A = \sum_{j=1}^w \mathbb{1}x_{i,j} = A r_{i\beta} \quad (2.49)$$

$$w_\alpha = \frac{1}{n} \sum_{i=1}^n r_{i\alpha} \quad (2.50)$$

$$w_\beta = \frac{1}{n} \sum_{i=1}^n r_{i\beta} \quad (2.51)$$

A Equação 2.48, atualiza os valores de α da seguinte maneira. Se $x_{i,j} = A$, então $\alpha_{j,A} = r_{i\alpha}$. Por exemplo, considere a primeira amostra $x_1 = \{AACT\}$ com seu respectivo valor $r_{1\alpha} = 0,32$. Como $x_{1,1} = A$ então $\alpha_{1,A} = \alpha_{1,A} + 0,32$. O mesmo raciocínio pode ser empregado para explicar a atualização das outras bases na distribuição α e também para realizar a atualização da distribuição β (Equação 2.49). Os parâmetros w_α e w_β são atualizados conforme as Equações 2.50 e 2.51, através da divisão de $N_\alpha = \sum_{i=1}^n r_{i\alpha}$ e $N_\beta = \sum_{i=1}^n r_{i\beta}$ por n , que representa o número de amostras no dataset. As Equações 2.52, 2.53 e 2.54 mostram o modelo após a atualização.

$$\alpha^T = \begin{bmatrix} A & C & G & T \\ 0.32 + 0.19 & 0.40 & \epsilon & 0.28 \\ 0.32 + 0.28 & 0.19 & 0.40 & \epsilon \\ \epsilon & 0.32 + 0.28 & 0.19 & 0.40 \\ 0.40 & \epsilon & 0.28 & 0.32 + 0.19 \end{bmatrix} = \begin{bmatrix} A & C & G & T \\ 0.51 & 0.40 & 10^{-3} & 0.28 \\ 0.60 & 0.19 & 0.40 & 10^{-3} \\ 10^{-3} & 0.60 & 0.19 & 0.40 \\ 0.40 & 10^{-3} & 0.28 & 0.51 \end{bmatrix}$$

$$\alpha^T = \begin{bmatrix} A & C & G & T \\ 0.4282 & 0.3359 & 0.0008 & 0.2351 \\ 0.5038 & 0.1595 & 0.3359 & 0.0008 \\ 0.0008 & 0.5038 & 0.1595 & 0.3359 \\ 0.3359 & 0.0008 & 0.2353 & 0.4286 \end{bmatrix} \quad (2.52)$$

$$\beta^T = \begin{bmatrix} A & 0.68 + 0.68 + 0.60 + 0.81 + 0.72 \\ C & 0.68 + 0.60 + 0.81 + 0.72 \\ G & 0.60 + 0.81 + 0.72 \\ T & 0.68 + 0.60 + 0.81 + 0.72 \end{bmatrix} = \begin{bmatrix} A & 3.49 \\ C & 2.81 \\ G & 2.13 \\ T & 2.81 \end{bmatrix} = \begin{bmatrix} A & 0.31 \\ C & 0.25 \\ G & 0.19 \\ T & 0.25 \end{bmatrix} \quad (2.53)$$

$$w_\alpha = \frac{1.19}{4} = 0.3125, w_\beta = 1 - w_\alpha = 0.6875 \quad (2.54)$$

Os modelos exibidos nas Equações 2.52 e 2.53 foram construídos através da execução das Equações 2.48 e 2.49 para $\alpha_{i,j}$ e β_j , $i = 1, 2, 3, \dots, w$ e $j \in \{A, C, G, T\}$. É interessante destacar o símbolo ϵ presente em alguns elementos do modelo α . Ele é um pseudocontador definido com valor próximo a zero (neste exemplo, $\epsilon = 10^{-3}$) cujo objetivo é não permitir que probabilidades iguais a zero estejam presentes no modelo, o que consequentemente causaria instabilidade numérica nos cálculo de logaritmo.

Como exemplo, considere a primeira coluna das amostras exibidas na Tabela 2.5. O símbolo G não está presente nela, o que leva a probabilidade de T ser zero neste ponto. Porém, é quase certo que a probabilidade de G nesta coluna não seja zero. Isso só ocorreu porque a amostragem é muito pequena, sendo nestes casos aconselhável o uso de pseudocontadores. É fácil verificar que conforme a quantidade de amostras cresce, o uso de pseudocontadores se torna cada vez menos importante. A Equação 2.54 mostra a reestimação dos pesos w_α e w_β através da aplicação das Equações 2.50 e 2.51.

Uma outra forma de lidar com o problema é através do uso explícito de variáveis latentes. Considere o vetor de variáveis latentes $z = \{z_1, z_2, z_3, \dots, z_n\}$ tal que $z_i \in \{0, 1, 2, 3, \dots, k\}$. Sem perda de generalidade, considere que $k = 2$ e $z_i \in \{\alpha, \beta\}$. Neste modelo, z_i indica qual é a distribuição da i -ésima amostra. Por exemplo, se $z_{10} = \alpha$, isso indica que a décima amostra pertence à distribuição α . É fácil verificar que $P(z_i = \alpha) = P(\alpha|x_i) = r_{i\alpha}$. Desta forma, temos que $z_i \sim B(r_{i\alpha})$ para $k = 2$. Se $k > 2$, então podemos utilizar a distribuição multinoulli também conhecida como distribuição categórica para modelar z_i . Isso é ligeiramente diferente do mostrado por Dempster et al. (1977), na qual $z_i \sim B(w_\alpha)$.

Se z for conhecido, estimar θ é direto. Basta utilizar para z agrupar as sequências que pertencem à α e β e realizar os cálculos através das Equações 2.19 e 2.20. Estimar w_α e w_β também é simples, sendo que $w_\alpha = \frac{1}{n} \sum_{i=1}^n \mathbb{1}_{z_i = \alpha}$ e $w_\beta = 1 - w_\alpha$. O problema é que z é desconhecido, então precisamos estimá-lo. Porém para isso, primeiro é preciso calcular θ . Desta

forma, entramos em um *loop*: para obter z , precisamos de θ e para obter θ precisamos z . Isso mostra a razão pela qual não podemos obter uma solução analítica para modelos de mistura.

Uma forma de resolver esse impasse é assumir um algoritmo no qual z esteja disponível à priori. Deste modo, com z podemos obter θ . Uma vez que θ foi calculado, podemos então reestimar z . Isso é executado sucessivamente até que a convergência seja atingida. Como exemplo, considere $X = \{AACT, CGTA, ACGT, TACG\}$ e $z = \{1, 0, 1, 0\}$. Empregando as Equações 2.19, encontramos o modelo θ descrito nas Equações 2.55, 2.56 e 2.57.

$$M^+ = \begin{bmatrix} A & A & C & T \\ A & C & G & T \end{bmatrix} M^- = \begin{bmatrix} C & G & T & A \\ T & A & C & G \end{bmatrix} \quad (2.55)$$

$$\alpha^T = \begin{bmatrix} A & C & G & T \\ 2 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix} (\alpha^T + 1) = \begin{bmatrix} A & C & G & T \\ 3 & 1 & 1 & 1 \\ 2 & 2 & 1 & 1 \\ 1 & 2 & 2 & 1 \\ 1 & 1 & 1 & 3 \end{bmatrix} = \begin{bmatrix} A & C & G & T \\ 0.50 & 0.16 & 0.16 & 0.16 \\ 0.33 & 0.33 & 0.16 & 0.16 \\ 0.16 & 0.33 & 0.33 & 0.16 \\ 0.16 & 0.16 & 0.16 & 0.50 \end{bmatrix} \quad (2.56)$$

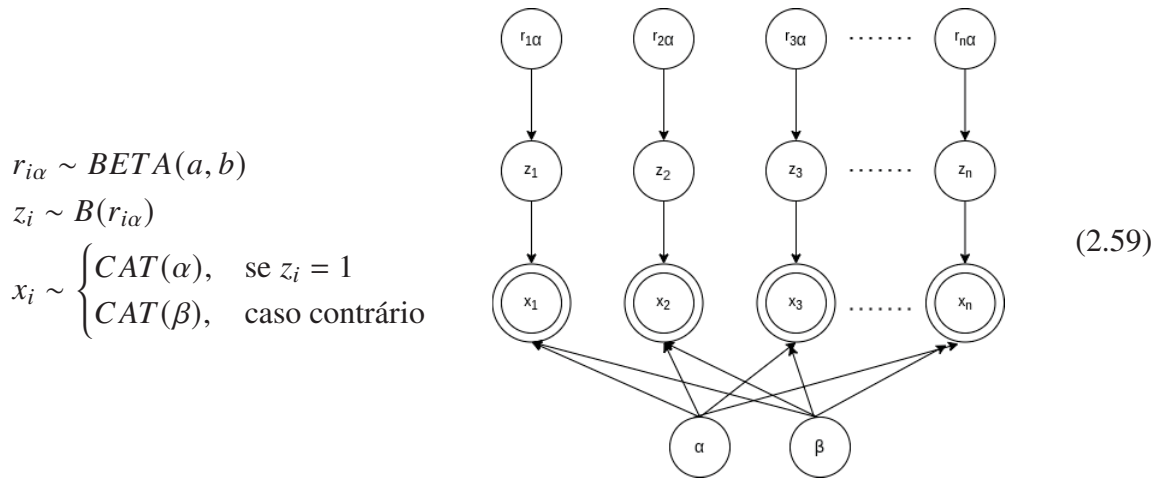
$$\beta^T = \begin{bmatrix} A & 2 \\ C & 2 \\ G & 2 \\ T & 2 \end{bmatrix} = (\beta^T + 1) = \begin{bmatrix} A & 3 \\ C & 3 \\ G & 3 \\ T & 3 \end{bmatrix} = \begin{bmatrix} A & 0.25 \\ C & 0.25 \\ G & 0.25 \\ T & 0.25 \end{bmatrix} \quad (2.57)$$

$$w_\alpha = \frac{2}{4} = \frac{1}{2}, w_\beta = 1 - w_\alpha = \frac{1}{2} \quad (2.58)$$

Podemos agora usar θ para computar as probabilidade da Tabela 2.6 e através dos valores presentes nas colunas $r_{i\alpha}$ e $r_{i\beta}$, reestimar z , tal que $z_i \sim B(r_{i\alpha})$. Desta forma, temos que $p(z_i = 1 | r_{i\alpha}) = r_{i\alpha}$. Por exemplo, de acordo com o modelo, a probabilidade da primeira amostra $x_1 = AACT$ pertencer à α é $\approx 0,87$ e assim $z_1 \sim B(0,87)$. A Equação 2.59 exhibe o modelo completo ao lado da imagem que representa a notação de placas (*plate notation*).

Tabela 2.6: Cálculo das probabilidades utilizando o modelo representado pelas Equações 2.55, 2.56, 2.57 e 2.58.

Amostras	$P(x_i \alpha)$	$P(x_i \beta)$	$P(x_i \theta = \{\alpha, \beta, w_\alpha, w_\beta\})$	$r_{i\alpha} = P(\alpha x_i)$	$r_{i\beta} = P(\beta x_i)$
AACT	0,0272	0,0039	0,0156	0,8726	0,1274
CGTA	0,0007	0,0039	0,0023	0,1710	0,8290
ACGT	0,0272	0,0039	0,0156	0,8726	0,1274
TACG	0,0031	0,0039	0,0035	0,4496	0,5504



2.4 CHIP-SEQ

A tecnologia CHIP-SEQ é uma poderosa abordagem baseada em sequenciamento de próxima geração (NGS) que permite a identificação e análise de interações proteína-DNA em um contexto genômico global (Park, 2009; Furey, 2012). Com o avanço das técnicas de sequenciamento, o CHIP-SEQ se tornou uma ferramenta amplamente utilizada para investigar a regulação da expressão gênica, a organização da cromatina e a função dos fatores de transcrição e outras proteínas que interagem com o DNA (Landt et al., 2012). O protocolo CHIP-SEQ tem uma ampla gama de aplicações na biologia molecular e na pesquisa genômica, incluindo:

- Mapeamento do perfil de fatores de transcrição e modificação de histonas: o CHIP-SEQ é frequentemente utilizado para identificar os sítios de ligação de fatores de transcrição e as modificações de histonas associadas à regulação da expressão gênica (Park, 2009). A análise desses dados pode revelar padrões de atividade regulatória em diferentes condições celulares, tecidos e espécies (Furey, 2012);
- Identificação de regiões regulatórias e elementos funcionais: ao mapear a distribuição de fatores de transcrição e modificações de histonas, o CHIP-SEQ pode ajudar a identificar regiões regulatórias, como promotores, intensificadores e silenciadores, bem como outros elementos funcionais do genoma, como insuladores e locais de *splicing* (Landt et al., 2012);
- Estudo de mecanismos de regulação epigenética: o CHIP-SEQ é uma ferramenta útil para investigar a regulação epigenética, como modificações de histonas e metilação do DNA, que desempenham um papel importante no desenvolvimento, diferenciação celular e doenças humanas (Park, 2009);
- Análise de redes de regulação gênica: ao combinar os dados do CHIP-SEQ com outras técnicas de genômica funcional, como RNA-seq e ATAC-seq, é possível construir redes de regulação gênica que descrevem as interações entre fatores de transcrição, co-fatores e seus alvos gênicos (Furey, 2012).

Basicamente o processo consiste em fixar proteínas de interesse ao DNA e então fragmentá-lo em pequenas regiões circunstantes. Esses fragmentos são então ligados a uma plataforma de sequenciamento e as regiões de ligação das proteínas são identificadas por meio da

análise dos dados de sequenciamento. Para isso, a imunoprecipitação da cromatina é utilizada juntamente com o sequenciamento paralelo massivo para mapear interações entre proteínas de interesse e suas respectivas regiões de ligação nos mais variados genomas (Robertson et al., 2007). Por essa razão, essa técnica é muito empregada para encontrar TFBS experimentalmente. Conforme mostrado na Figura 2.12, este protocolo envolve várias etapas, cujas mais importantes são:

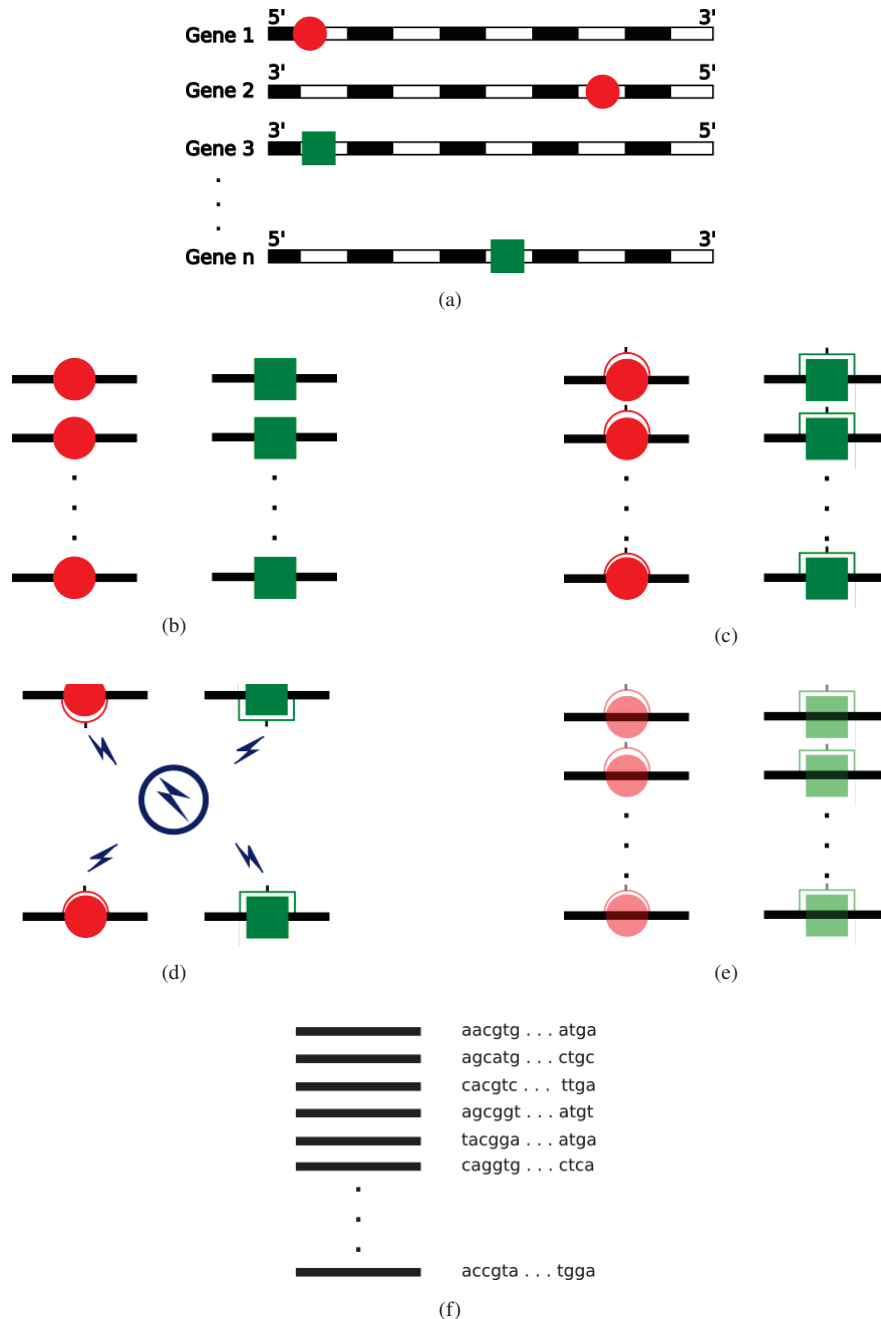


Figura 2.12: Principais etapas do protocolo ChIP-seq. (a) *Crosslinking*. (b) Isolamento. (c) Imunoprecipitação. (d) Purificação. (e) *Crosslinking* reverso. (f) Sequenciamento. Fonte: o autor.

1. *Crosslinking*: a proteína de interesse (poi, do inglês *Protein of Interest*) é “colada”¹² à cromatina por meio de técnicas específicas, como por exemplo através do uso de formaldeído (Park, 2009);

¹²O termo utilizado em inglês é *cross-linked*.

2. Isolamento: pequenos fragmentos de DNA + POI são “cortados”¹³ através de abordagens específicas, como *sonication* ou por meio das enzimas de restrição (Schmidt et al., 2009);
3. Imunoprecipitação: anticorpos específicos (SAB, do inglês *Specific Antibody*) são criados com o objetivo de se ligarem as proteínas de interesse (Park, 2009);
4. Purificação: através de um processo magnético, o complexo SAB + POI + DNA é separado dos demais objetos (Park, 2009);
5. *Crosslinking* reverso: através da incubação em altas temperaturas ou da digestão das proteínas envolvidas (POI+SAB) o fragmento de DNA é completamente isolado (Fareed et al., 2022);
6. Sequenciamento: com os fragmentos de DNA devidamente isolados, é possível sequenciá-los através de modernas técnicas de sequenciamento paralelo de alto desempenho (Shendure e Ji, 2008), como por exemplo *Roche 454* (Luo et al., 2012) ou *Illumina HiSeq* (Caporaso et al., 2012). Os fragmentos após serem sequenciados são chamados de *reads*¹⁴.

Em média, as *reads* possuem cerca de 100 à 300 pares de base. O tamanho e a precisão final dependem de vários fatores, tais como a tecnologia utilizada no sequenciamento e o tipo de organismo a ser estudado. Após a obtenção de todos os fragmentos, várias análises podem ser executadas, como por exemplo a realização do alinhamento das *reads* contra o genoma de referência ou a criação de modelos que representam e capturam as suas principais características. No primeiro caso, o interesse consiste em localizar a posição dos fragmentos no genoma. Por exemplo, se uma alta concentração de uma determinada *read* for localizada na região promotora de diferentes genes, isso pode ser um indício de que este fragmento é um sítio de ligação cuja função pode estar relacionada com a regulação de alguns ou de todos os genes envolvidos.

No segundo caso, objeto de estudo deste trabalho, a finalidade é obter a localização das regiões mais conservadas no interior das *reads* para a construção de modelos. Isso é importante pois, através deste processo, é possível realizar a anotação de outras regiões do genoma que ainda não possuem função conhecida. É interessante notar que na etapa de isolamento, além da região de interesse, as bordas dessa região também sofrem o processo de *shearing*. Desta forma, a resolução final de um fragmento, além de possuir um formato irregular, fica em torno de $100 \leq t \leq 300$ pares de base que é muito maior do que a sua região conservada, que mede cerca de $5 \leq k \leq 25$ pares de base, necessitando assim de análise computacional complementar para separar os fragmentos de interesse das demais áreas.

Em muitos experimentos CHIP-SEQ, uma amostra de controle também é gerada, chamada de amostra negativa ou BACKGROUND. Diferentemente da amostra positiva, a amostra negativa não está enriquecida com os fragmentos de interesse. Desta forma, as amostras positivas e negativas possuem distribuição de probabilidade diferentes. Entretanto, é possível realizar a análise dos dados de CHIP-SEQ apenas com as sequências positivas. A vantagem disto é que o experimento acaba com um custo operacional menor, uma vez que os fragmentos negativos não precisam ser sequenciados. Entretanto, caso seja necessário aumentar a precisão do experimento, o sequenciamento dos fragmentos negativos torna-se importante. É bastante comum que modelos teóricos baseados na distribuição binomial negativa e Poisson sejam empregados para modelar as distribuição de controle (Zhang et al., 2008).

¹³O termo utilizado em inglês é *DNA shearing*.

¹⁴Não existe uma tradução adequada deste termo para o português, o mais próximo seria fragmento.

De acordo com [Park \(2009\)](#), o CHIP-CHIP é uma abordagem que combina imunoprecipitação da cromatina com a técnica de *microarray*. Antes do advento do CHIP-SEQ, essa era a técnica mais utilizada para identificar regiões enriquecidas. Contudo, em comparação ao CHIP-SEQ, o CHIP-CHIP apresenta várias desvantagens, tais como: i) devido ao processo de hibridização em *microarray*, há um maior risco de viés e ruídos causados por hibridizações cruzadas; ii) o enriquecimento é calculado por meio da medição da intensidade de hibridização, o que muitas vezes resulta em saturação em sinais mais elevados; iii) as interações detectadas ficam limitadas às regiões genômicas para as quais as sondas estão disponíveis; iv) a precisão dessa técnica varia entre $300 \leq t \leq 1000$.

Por essas razões, a técnica CHIP-SEQ revolucionou a descoberta de elementos regulatórios *in vivo*. No entanto, é importante salientar que mesmo com todo o avanço biológico e experimental, só é possível realizar a análise e interpretação dos dados gerados por este protocolo por meio de sofisticadas ferramentas computacionais ([Ladunga, 2010](#)).

Embora seja bastante preciso, dados oriundos de experimentos CHIP-SEQ ainda possuem muitos problemas, tais como baixa resolução, elevado ruído de fundo e erros sistêmicos¹⁵. A grande maioria dessas características indesejáveis podem ser mitigadas com a utilização de ferramentas computacionais. Por exemplo, de acordo com [Ladunga \(2010\)](#), a análise computacional de MOTIFS é responsável por reduzir consideravelmente a quantidade de falsos positivos e negativos. Apesar de suas muitas aplicações e avanços, o CHIP-SEQ ainda enfrenta vários desafios técnicos e de análise de dados:

- **Qualidade e especificidade do anticorpo:** a qualidade e especificidade do anticorpo utilizado no experimento CHIP-SEQ são críticas para a confiabilidade dos resultados ([Landt et al., 2012](#)). O uso de anticorpos não específicos ou de baixa qualidade pode levar a resultados falsos positivos ou negativos;
- **Reprodutibilidade e padronização:** a reprodutibilidade dos experimentos CHIP-SEQ entre laboratórios e condições experimentais é uma preocupação, especialmente na comparação de diferentes estudos ([Furey, 2012](#)). A padronização de protocolos e critérios de qualidade é essencial para garantir a confiabilidade e comparabilidade dos dados;
- **Análise de dados e identificação de picos:** a análise de dados CHIP-SEQ é complexa e envolve várias etapas, incluindo mapeamento de *reads*, normalização, identificação de picos e extração de MOTIFS ([Park, 2009](#)). A escolha dos algoritmos e parâmetros corretos é importante para obter resultados biologicamente significativos.

Devido a problemas na especificidade dos anticorpos, estima-se que de 60% a 90% dos fragmentos obtidos pelo protocolo CHIP-SEQ tenham origem em outras interações, que não são as realizadas pelos fatores de transcrição de interesse ([Pepke et al., 2009](#)). Este problema pode se tornar ainda mais evidente em famílias multi-gênicas. Famílias multi-gênicas são grupos de genes dentro de um organismo que são derivados de um ancestral comum e, portanto, compartilham sequências similares e, frequentemente, funções biológicas relacionadas. Esses genes podem codificar para proteínas que desempenham funções essenciais e muitas vezes sobrepostas dentro de processos celulares, incluindo, mas não se limitando a, fatores de transcrição. A presença de múltiplas cópias genéticas com alta similaridade sequencial dentro de famílias multi-gênicas pode complicar a distinção entre os membros específicos da família durante experimentos,

¹⁵Um erro sistêmico bastante comum é a troca de um nucleotídeo por outro no momento do sequenciamento.

especialmente quando se utilizam anticorpos para identificar a ligação de proteínas específicas, como é o caso dos experimentos de CHIP-SEQ.

O desafio associado às famílias multi-gênicas em experimentos de CHIP-SEQ é exacerbado pela dificuldade em desenvolver anticorpos que possam distinguir de forma confiável entre proteínas altamente similares. Devido à correlação moderada entre o epítipo (a parte da antígeno a que o anticorpo se liga) e os resíduos reais de ligação do fator de transcrição de interesse, criar anticorpos que sejam altamente específicos para membros individuais de famílias multi-gênicas de TF's se torna um desafio significativo. Isso ocorre porque os anticorpos podem cruzar a reatividade entre os membros da família multi-gênica, levando a uma maior proporção de fragmentos de CHIP-SEQ que representam interações não específicas ou indesejadas, em vez de apenas aquelas realizadas pelo fator de transcrição de interesse.

Devido a essa correlação, projetar anticorpos altamente específicos para TFS multifamiliares continua sendo um grande desafio (Klinck et al., 2008). Apesar disso, CHIP-SEQ ainda é a principal tecnologia disponível para reconhecimento de TFBS *in vivo* da atualidade e por essa razão, existem diversos repositórios online, públicos e comerciais, que disponibilizam dados para análise. O principais bancos de dados de elementos regulatórios extraídos por meio do protocolo CHIP-SEQ são:

- JASPAR: <https://jaspar.genereg.net>;
- TRANSFAC: <https://genexplain.com/transfac>;
- ENCODE: <https://www.encodeproject.org/>.

JASPAR (Castro-Mondragon et al., 2022) é um banco de dados de público e gratuito que recebe atualizações constantes e armazena diversos perfis TFBS curados manualmente por meio de matrizes de frequência PFMS (Equação 2.1). Conforme visto, as PFMS calculam as ocorrências de cada nucleotídeo em cada posição de um alinhamento local de sequências sem lacunas. Com isso, é possível converter as matrizes de frequência em modelos probabilísticos ou energéticos PWMS ou PSSMS (Equação 2.2), que podem ser usadas posteriormente para escanear qualquer sequência de DNA para classificar TFBSS.

TRANSFAC é um banco de dados de fatores de transcrição eucarióticos disponibilizado em conjunto com seus sítios de ligação genômicos e perfis de ligação ao DNA. Sua proposta é muito similar ao JASPAR, porém a base de dados TRANSFAC é paga e seu uso é estritamente comercial. Segundo Matys et al. (2006), TRANSFAC é uma compilação muito antiga (existe desde 1996), e vem sendo cuidadosamente mantida e curada, o que a ajudou se tornar muito relevante na área de elementos regulatórios. Assim como JASPAR, possui uma biblioteca de matrizes de frequência, regularmente empregadas por pesquisadores em análises de sequências genômicas para detecção de possíveis locais de ligação dos fatores de transcrição (TFBSS).

ENCODE (Consortium et al., 2012) é um projeto internacional que tem como objetivo identificar todos os elementos funcionais no genoma humano. O ENCODE disponibiliza vários dados públicos de CHIP-SEQ, incluindo dados de histonas e fatores de transcrição em diferentes tipos de células e tecidos humanos. Por exemplo, um dos arquivos disponíveis no ENCODE é um conjunto de dados de CHIP-SEQ para o fator de transcrição CTCF em células K562. Este conjunto de dados contém as leituras do sequenciamento obtidas após o enriquecimento de DNA para a proteína CTCF. Cada leitura contém informações como a posição no genoma onde ela foi mapeada, a qualidade da leitura e a orientação da leitura. Esses dados podem ser usados para identificar regiões do genoma onde o fator de transcrição CTCF se liga.

JASPAR, TRANSFAC e ENCODE são os principais repositórios de elementos regulatórios disponíveis para a comunidade científica. Além destes, existem ainda: i) HOCOMOCO: <https://hocomocol1.autosome.org>; ii) REGULATORY ANNOTATION: <https://www.bcgsc.ca/resources/software/oreganno>; iii) PRODORIC: <https://www.prodoric.de>; iv) REGULON: <https://regulondb.ccg.unam.mx>; (v) *Gene Expression Omnibus* (GEO) <https://www.ncbi.nlm.nih.gov/geo/>.

HOCOMOCO (Kulakovskiy et al., 2018) é um banco de dados que possui diversos elementos regulatórios dos organismos *H. sapiens* e *M. musculus* e suas respectivas matrizes de frequência. REGULATORY ANNOTATION (Lesurf et al., 2016) é um banco de dados aberto que está em amplo crescimento para a curadoria de elementos regulatórios conhecidos da literatura científica. A anotação é coletada através de usuários para os mais variados tipos de ensaios biológicos e é automaticamente cruzado com dados presentes em vários bancos de dados, tais como: PUBMED, ENTREZ GENE, ENSEMBL, DBSNP, EVOC e o banco de dados público de taxonomia. PRODORIC (Dudek e Jahn, 2022) é um banco de dados criado recentemente pelo Professor Dr. Dieter Jahn da Universidade Técnica de Braunschweig, cuja principal característica é curadoria de elementos procarióticos.

Segundo Tierrafría et al. (2022), REGULON DB é um banco de dados que fornece várias informações do organismo *E. coli* a respeito de diversos aspectos transcricionais, tais como organização do operon e sua decomposição em unidades de transcrição, promotores e seu tipo sigma, genes e seus sítios de ligação ribossomal, terminadores, sítios de ligação de reguladores transcricionais específicos, bem como sua organização em frases regulatórias, conformações ativas e inativas e regulons simples e complexos.

Por fim, o banco de dados GEO (Sayers et al., 2023) é um repositório público de dados de expressão gênica, que armazena informações sobre como os genes são ativados ou desativados em diferentes condições biológicas. Com mais de 2 milhões de amostras de expressão gênica de diferentes espécies e condições, o GEO é uma importante fonte de dados para pesquisadores de biologia molecular, genética e outras áreas relacionadas. Os dados do GEO são acessíveis a todos os pesquisadores, permitindo a reutilização de dados e a descoberta de novos insights biológicos.

O protocolo CHIP-SEQ revolucionou a pesquisa em biologia molecular e genômica, permitindo a investigação das interações proteína-DNA em uma escala genômica global. Com a crescente disponibilidade de plataformas de sequenciamento de próxima geração e avanços nas técnicas de análise de dados, espera-se que o CHIP-SEQ continue a ser uma ferramenta inestimável para estudar a regulação da expressão gênica, a organização da cromatina e a função de fatores de transcrição e outras proteínas que interagem com o DNA.

À medida que a tecnologia continua a evoluir, é provável que novas abordagens e melhorias no protocolo CHIP-SEQ sejam desenvolvidas para lidar com desafios técnicos e de análise de dados. Além disso, a integração do CHIP-SEQ com outras técnicas de genômica funcional e a aplicação em um contexto clínico e translacional têm o potencial de expandir significativamente nosso entendimento dos mecanismos subjacentes à regulação gênica e ao desenvolvimento de doenças.

2.5 CONCLUSÃO

Neste capítulo, apresentamos os fundamentos necessários para entender o problema de descoberta de MOTIFS biológicos e o contexto no qual nossa pesquisa se insere. Inicialmente, descrevemos o problema de identificação de regiões conservadas, que é determinante para compreensão da regulação gênica e as interações proteína-DNA. Exploramos também uma variedade de métodos para descoberta de MOTIFS. Iniciamos com o método de força bruta,

procedemos para a estratégia de `BRANCH AND BOUND` e terminamos com técnicas modernas de contagem utilizando `KTREES` e `STREES`. Em seguida, discutimos os principais algoritmos e estruturas que serão utilizados em nosso estudo, incluindo *expectation maximization*, *multi-armed bandits*, cadeias de `MARKOV` e modelos `PWM`. Esses algoritmos representam abordagens distintas e complementares para abordar o problema de descoberta de `MOTIFS` e serão explorados em detalhes ao longo de nossa pesquisa. Também apresentamos vários repositórios de dados de imunoprecipitação da cromatina seguida de sequenciamento de alto desempenho (`CHIP-SEQ`). Esses dados são fundamentais para a compreensão das interações proteína-DNA e nos permitirão avaliar a eficácia e o desempenho dos algoritmos selecionados no contexto da descoberta de `MOTIFS` biológicos. Ao lançar os alicerces teóricos relativos, este capítulo oferece uma base robusta para os capítulos subsequentes. Neles, iremos explorar a aplicação dos algoritmos mencionados, avaliar os resultados alcançados, e discutir as implicações e limitações de nossas descobertas. No próximo capítulo, daremos início a uma ampla revisão bibliográfica da literatura. Aprofundaremos nossa compreensão dos conceitos e princípios subjacentes, constituindo uma base confiável para o desenvolvimento e aplicação das metodologias exploradas ao longo desta pesquisa.

3 REVISÃO BIBLIOGRÁFICA

O objetivo deste capítulo é mostrar os principais avanços conquistados pela comunidade científica na predição e descoberta de MOTIFS biológicos através da exposição de diversos artigos publicado ao longo dos últimos anos. Para atingir este objetivo, uma extensa revisão sistemática da literatura foi realizada em diversas bases de dados com a finalidade de construir uma visão ampla do problema. Após a obtenção dos estudos iniciais, foram aplicados vários filtros com o propósito de selecionar somente os trabalhos mais importantes. Para finalizar, levantamos algumas questões que ainda estão em aberto na literatura, com o intuito de auxiliar os pesquisadores da área a evoluir o estado da arte. Também serão abordados os métodos e técnicas empregadas na área de estudo, incluindo uma discussão detalhada sobre as abordagens metodológicas adotadas. Essa análise permitiu identificar as lacunas e oportunidades existentes na literatura, bem como justificar a relevância e a necessidade do estudo proposto. Este capítulo está dividido como segue. A Seção 3.1 descreve como a revisão bibliográfica foi realizada, bem como a *string* de busca empregada e as fontes de dados utilizadas. A Seção 3.2 descreve os trabalhos relacionados mais relevantes que foram selecionados na etapa anterior. Por fim, a Seção 3.4 apresenta algumas questões que ainda não foram respondidas pela comunidade científica e ainda estão em aberto.

3.1 METODOLOGIA UTILIZADA

A revisão bibliográfica deste trabalho foi desenvolvida e alicerçada em sólidos pilares descritos principalmente por Boland et al. (2017), com o objetivo de ser o mais abrangente possível. O objetivo deste capítulo é: (i) aprofundar a investigação nos processos que definem e caracterizam os MOTIFS biológicos e (ii) estudar as principais abordagens que integram o estado da arte na descoberta MOTIFS biológicos. Nesta seção (e nas demais dentro deste capítulo), apresentaremos uma ampla revisão bibliográfica dos principais trabalhos publicados nesta área, a fim de responder as questões (i) e (ii).

3.1.1 Grupo controle e *string* de busca

O grupo controle engloba estudos que foram manualmente identificados como importantes. Esses trabalhos foram selecionados seguindo métricas quantitativas, como por exemplo o número de total citações na literatura e o grau de afinidade que eles apresentam com este estudo. Desta forma, conforme mostra a Tabela 3.1, cinco artigos foram selecionados. Eles foram utilizados no processo de extração de palavras-chave que irão compor a *string* de busca. Portanto, após a aplicação da *string* de busca nas bases de dados selecionadas, todos devem ser retornados pela pesquisa. Em outras palavras, a *string* de busca deve ser confeccionada de tal forma a devolver, dentre outros estudos relevantes, estes cinco escolhidos manualmente.

A Figura 3.1 descreve a *string* de busca empregada neste trabalho. É interessante destacar que foram utilizados na pesquisa somente os metadados “*Título/Title*” e “*Resumo/Abstract*”, pois desta forma, foi possível recuperar apenas os trabalhos mais relevantes, que utilizam as *keywords* diretamente nestes dois ambientes. Artigos que apenas mencionam as *keywords* de forma isolada, no meio do texto, foram descartados, pois isso aumenta as chances do estudo tratar os tópicos de interesse de maneira secundária. É importante notar que a *string* de busca pode mudar em função da fonte de pesquisa utilizada, porém a essência permanece a mesma para todas elas.

Tabela 3.1: Artigos utilizados como *baseline* para composição da *string* de busca.

Título	Publicação	Última versão	#Citações
Prosampler: an ultrafast and accurate motif finder in large ChIP-seq datasets for combinatory motif discovery	2019	2023	18
Homer: Hypergeometric Optimization of Motif EnRichment	2010	2019	9587
Meme: Fitting a mixture model by expectation maximization to discover motifs in biopolymers	1994	2023	5516
Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment	1993	2020	2394
Identifying DNA and protein patterns with statistically significant alignments of multiple sequences	1989	2000	1573

```

("motif finder"[Title/Abstract]) OR
("finding-motifs"[Title/Abstract]) OR
("finding-motif"[Title/Abstract]) OR
("motif-finding"[Title/Abstract]) OR
("motif finding"[Title/Abstract]) OR
("discovery motif"[Title/Abstract]) OR
("discovery motifs"[Title/Abstract]) OR
("sequence motif finder"[Title/Abstract]) OR
("sequence motif discovery"[Title/Abstract]) OR
("finding motifs"[Title/Abstract]) OR
("predict motifs"[Title/Abstract]) OR
("prediction motifs"[Title/Abstract]) OR
("identifying motifs"[Title/Abstract]) OR
("prediction of transcription factor binding sites"[Title/Abstract]) OR
("prediction of transcription factor binding site"[Title/Abstract]) OR
("detect overrepresented motifs"[Title/Abstract]) OR
("detecting overrepresented motifs"[Title/Abstract])

```

Figura 3.1: Exemplo da *String* de busca empregada para recuperar o rol de artigos utilizados neste trabalho. Neste caso particular, a *string* exibida está com a sintaxe formatada para a busca no banco de dados PubMed, porém, pode ser facilmente modificada para outros tipos de bancos de dados.

3.1.2 Fontes de pesquisa

Foram utilizadas quatro fontes de pesquisa para compor o rol inicial de estudos a serem investigados. São elas: (i) **PubMed** – <https://pubmed.ncbi.nlm.nih.gov/>, (ii) **PubMed Central** – <https://www.ncbi.nlm.nih.gov/pmc/>, (iii) **Scopus** – www.scopus.com e (iv) **IEEE Xplore** – explore.ieee.org. PUBMED é um mecanismo de busca gratuito mantido pela Biblioteca Nacional de Medicina dos Estados Unidos (NLM, do inglês *National Library of Medicine*) que indexa principalmente os estudos catalogados no Sistema de Análise e Recuperação de Literatura Médica Online (MEDLINE, do inglês *Medical Literature Analysis and Retrieval System Online*). PUBMED CENTRAL OU PMC, além de um mecanismo de busca, também um repositório gratuito digital desenvolvido pelo Centro Nacional de Informações sobre Biotecnologia dos Estados Unidos (NCBI, do inglês, *National Center for Biotechnology Information*) que também mantido pelo NLM.

Embora existam muitos estudos que são indexados tanto pelo PUBMED quanto PMC, isso não é uma regra. O PUBMED possui mais estudos, pois indexa todos os tipos de artigos, pagos e gratuitos, enquanto que o PMC indexa somente os gratuitos. Por essa razão, as duas fontes de dados foram utilizadas na extração dos estudos iniciais. SCOPUS é um banco de dados mantido pela ELSEVIER que centraliza trabalhos de milhares de editoras das mais variadas áreas do conhecimento. Por fim, IEEE XPLORE é um banco de dados para pesquisa de artigos publicados em conferências e revistas vinculadas ao Instituto de Engenheiros Elétricos e Eletrônicos (IEEE, do inglês *Institute of Electrical and Electronics Engineers*) e possui um viés mais computacional, sendo esta um das principais razões de sua escolha.

3.1.3 Critérios de seleção

Após a aplicação da *string* de busca, 1090 artigos foram retornados, conforme mostra a Tabela 3.2. Ao todo, foram empregados 11 filtros com o objetivo de separar apenas os trabalhos com maior relevância e que possuem conexão com a proposta desta tese. A Tabela 3.3 apresenta todos os filtros utilizados na pesquisa. Os três primeiros são inclusivos, i.e., se o artigo a ser analisado possuir alguma característica em comum com um destes três critérios, então ele será incluso no rol inicial de artigos.

Tabela 3.2: Quantidade de resultados encontrados por cada fonte de pesquisa.

Fonte de pesquisa	#Estudos encontrados
PubMed	347
PubMed Central	32
Scopus	562
IEEE Xplore	149
Total	1090

Os filtros 1 e 2 apresentam os critérios de inclusão que definem a afinidade dos estudos analisados com o tema desta tese. O filtro 3 especifica o intervalo aberto de todos os artigos que foram publicados até dez/2023 dos quais farão parte do rol de estudos avançados (desde que atendam aos critérios 1 e 2). Os demais filtros, são exclusivos e contribuem com a remoção de artigos que apresentam pouca ou nenhuma relação com este trabalho. Por exemplo, os filtros 4 e 5 excluem estudos que se destinam estudo de MOTIFS estruturais em RNA's e proteínas, pois estes tópicos fogem do escopo desta trabalho¹.

Os filtros 6 e 7 removem os *surveys* e os artigos que foram publicados em anais de congresso, pois o objetivo desta etapa é selecionar apenas trabalhos cujo finalidade foi a publicação de novas abordagens para descoberta de MOTIFS. É importante notar que não foram removidos os trabalhos publicados em anais de congresso da fonte de pesquisa IEEE XPLORE, pois, neste canal, a grande maioria são deste tipo. Os filtros 8 e 9 removem respectivamente os estudos que disponibilizam apenas o resumo ou que foram duplicados devido a aparição em dois ou mais mecanismos de busca. Por fim, os filtros 10 e 11 eliminam os trabalhos que tratam de MOTIFS não biológicos, como por exemplo os MOTIFS musicais e estudos escritos em língua diferente da inglesa.

Após a aplicação da *string* de busca, o resumo de todos os 1090 estudos selecionados foram lidos e os filtros aplicados. Caso os trabalhos satisfaçam aos critérios de inclusão, eles são selecionados para a próxima etapa, em um rol chamado de artigos avançados. É importante notar

¹Se o artigo envolve a descoberta de MOTIFS regulatórios em sequências de RNA e proteínas, então ele será aceito.

Tabela 3.3: Critérios de inclusão e exclusão aplicados na pesquisa bibliográfica.

#	Filtro	Tipo
1	O estudo apresentou o desenvolvimento de um novo algoritmo para a descoberta de <i>motifs</i> em sequências de DNA	Inclusão
2	O estudo apresentou uma investigação sobre sítios de ligação dos fatores de transcrição e propôs uma nova ferramenta para descoberta de <i>motifs</i>	Inclusão
3	O estudo foi publicado até dez/2023	Inclusão
4	O estudo foi sobre <i>motifs</i> estruturais em RNA's	Exclusão
5	O estudo foi sobre <i>motifs</i> estruturais em proteínas	Exclusão
6	O estudo foi publicado em anais de conferências ou similares*	Exclusão
7	O estudo foi alvo de revisão sistemática ou <i>survey</i>	Exclusão
8	Estudos sem resumo ou com apenas o resumo disponível	Exclusão
9	Estudos duplicados	Exclusão
10	Estudos com <i>motifs</i> não biológicos	Exclusão
11	Estudos não escritos em inglês	Exclusão

* Foram considerados todos os estudos da fonte de dados IEEE Explore, inclusive os publicados em anais de eventos, pois estes são maioria neste canal de pesquisa.

que se houver dúvidas entre aceitação ou exclusão de um estudo, então ele deverá ser aceito, já que, caso seja necessário, será possível removê-lo nas etapas subsequentes. Ao final desta primeira triagem, dos 1090 trabalhos, 228 foram selecionados. A Figura 3.2 mostra um resumo dos artigos separados por área.

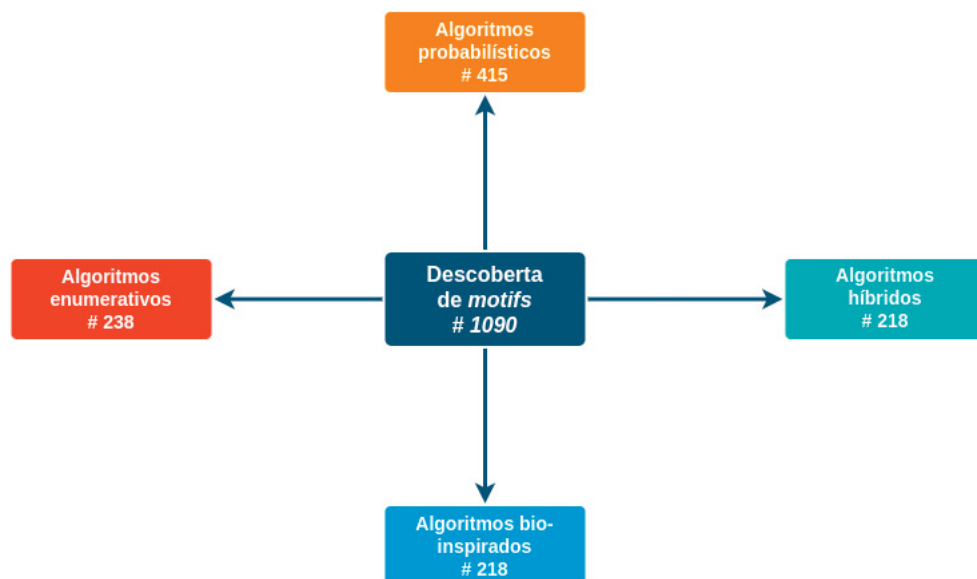


Figura 3.2: Principais abordagens empregadas no problema de descoberta de *motifs* biológicos. Os valores posicionados logo abaixo dos rótulos representam a frequência absoluta de artigos publicados na literatura levando em consideração os estudos recuperados pela *string* de busca. Fonte: o autor.

Em seguida, foi realizada a leitura da introdução e conclusão dos 228 trabalhos do rol avançado, sendo 79 trabalhos selecionados para a fase seguinte. Por fim, o último passo foi a leitura completa dos 79 artigos selecionados na fase anterior, dos quais 45 foram eleitos para o rol final (que serão detalhados na próxima seção), sendo estes os mais relevantes, que apresentaram alguma inovação importante ou que possuem relação com o tema proposto. A Figura 3.3 exibe todas as etapas executadas na confecção desta revisão e a Tabela 3.4 apresenta um resumo dos 45 artigos selecionados para a etapa final.

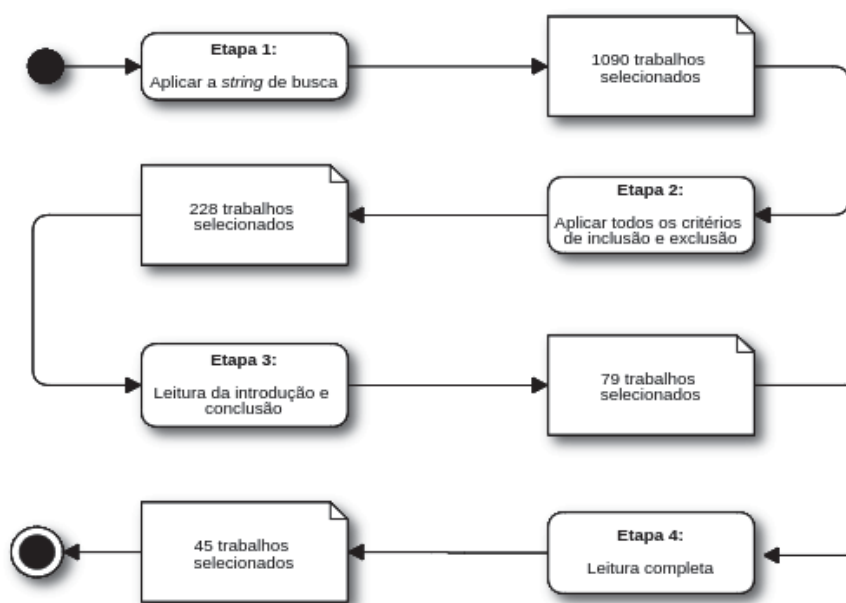


Figura 3.3: Fluxograma das atividades executadas na revisão sistemática. Fonte: o autor.

3.2 TRABALHOS RELACIONADOS

A finalidade desta seção é revisar e classificar os principais estudos e técnicas existentes na literatura cujo foco é a descoberta de MOTIFS biológicos. Com isso, pretendemos identificar o estado da arte, bem como reconhecer as questões que ainda estão em aberto. A Figura 3.4 exibe as áreas de pesquisa mais relevantes dentro do contexto de MOTIFS biológicos, das quais podemos destacar duas: i) descoberta de MOTIFS e ii) análise de enriquecimento. A principal diferença entre ambas reside na forma como os MOTIFS são determinados. Na primeira, os padrões são identificados através de métodos matemáticos em um conjunto de sequências de entrada sem nenhuma informação a priori e sem dados de treinamento. Portanto, os algoritmos que se destinam a resolver este problema são frequentemente do tipo não supervisionados, que possuem como principal característica o agrupamento de sequências similares de acordo com algum critério pré-estabelecido.

Por outro lado, a análise de enriquecimento tem como objetivo comparar a frequência de um determinado MOTIF de DNA em um conjunto primário de sequências com a frequência esperada em um conjunto de sequências de controle. Uma maior frequência do MOTIF no conjunto primário pode ser uma evidência de enriquecimento e sugerindo uma possível função biológica para o MOTIF. Isso pode ser feito através de informações fornecidas por um biólogo, tais como conjuntos de genes co-regulados e bibliotecas de MOTIFS conhecidas, dos quais fatores controlam a transcrição de determinados conjuntos de genes.

Tabela 3.4: Artigos utilizados na revisão sistemática ordenados do mais antigo para o mais recente.

#	Algoritmo	Princípio de operação	Referência
1	–	Enumerativo	Tomba (1999)
2	Consensus	Probabilístico	Stormo e Hartzell (1989)
3	–	Probabilístico	Lawrence e Reilly (1990)
4	–	Probabilístico	Lawrence et al. (1993)
5	MEME	Probabilístico	Bailey et al. (1994)
6	–	Probabilístico	Liu et al. (1995)
7	wConsensus	Probabilístico	Hertz e Stormo (1999)
8	AlignACE	Probabilístico	Hughes et al. (2000)
9	Motif Sampler	Probabilístico	Thijs et al. (2001b)
10	Biopropector	Probabilístico	Liu et al. (2001)
12	Multiprofiler	Probabilístico	Keich e Pevzner (2002b)
13	–	Enumerativo	Buhler e Tomba (2002)
14	MDSscan	Híbrido	Liu et al. (2002)
15	Motif regressor	Híbrido	Conlon et al. (2003)
16	ST-GA	Inspirado na natureza	Stine et al. (2003)
17	EMnEM	Híbrido	Moses et al. (2003)
18	YMF	Enumerativo	Sinha e Tomba (2003)
19	–	Probabilístico	McLaughlin et al. (2005)
20	GAME	Inspirado na natureza	Wei e Jensen (2006)
21	–	Híbrido	Reddy et al. (2006)
22	MOPAT	Híbrido	Hu et al. (2008)
23	MOGAMOD	Inspirado na natureza	Kaya (2009)
24	–	Híbrido	Shashidhara et al. (2010)
25	HOMER	Probabilístico	Heinz et al. (2010)
26	MotEvo	Probabilístico	Arnold et al. (2012)
27	MAMotif	Inspirado na natureza	Bi (2012)
28	TFBSGroup	Enumerativo	Jia et al. (2013)
29	GAMI	Inspirado na natureza	Thompson e Congdon (2014)
30	STEME	Probabilístico	Reid e Wernisch (2014)
31	GAEM	Híbrido	Wang e Miao (2014)
32	MotifHound	Enumerativo	Kelil et al. (2014)
33	FMotif	Enumerativo	Jia et al. (2014)
34	SIOMICS	Enumerativo	Ding et al. (2015)
35	MCES	Enumerativo	Yu et al. (2015)
36	FastMotif	Híbrido	Colombo e Vlassis (2015)
37	MFMD	Inspirado na natureza	Garbelini et al. (2018)
38	MCAT	Híbrido	Yang et al. (2019)
39	MOICA	Inspirado na natureza	Gohardani et al. (2019)
40	nEpigram	Híbrido	Ngo et al. (2019)
41	–	Enumerativo	Riza et al. (2019)
42	ProSampler	Híbrido	Li et al. (2019)
43	MFEA	Inspirado na natureza	Ashraf e Shafi (2020)
44	DeepGRN	Inspirado na natureza	Chen et al. (2021a)
45	STREME	Probabilístico	Bailey (2021)

Através do modelo estimado, o algoritmo procura por regiões que exibam alta probabilidade ou alguma outra medida estatística ou de distância que possa servir como discriminador. As regiões encontradas são chamadas de regiões enriquecidas ou picos. Embora o foco deste trabalho esteja concentrado na área (i) – descoberta de MOTIFS – mais conhecida pelo seu termo em inglês *de novo motif analysis* ou *de novo motif discovery*, nosso algoritmo também é capaz de

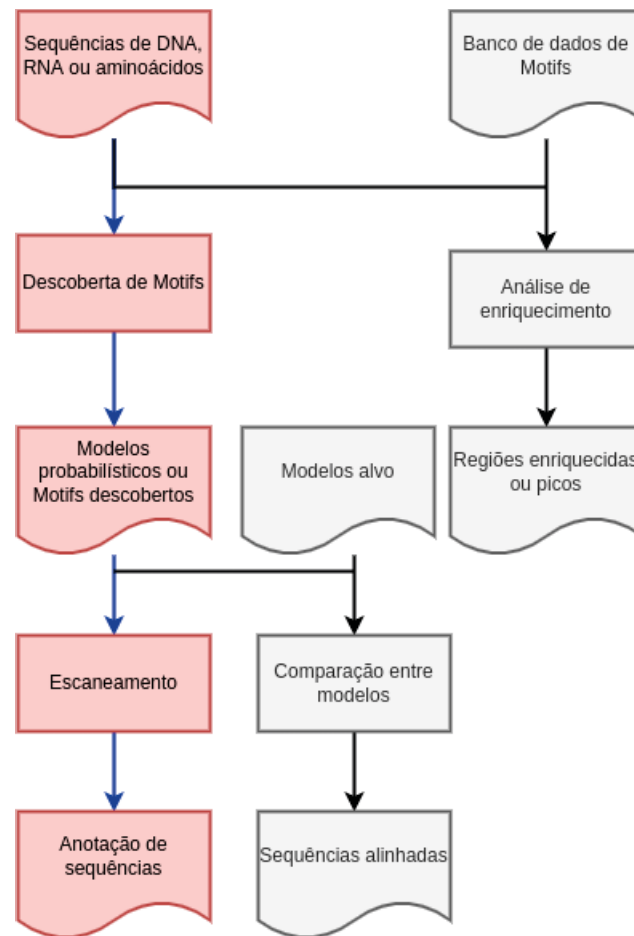


Figura 3.4: Taxonomia das áreas de pesquisa em *motifs* biológicos. O escopo deste trabalho reside nas áreas destacadas em vermelho ligadas pelas setas azuis. Fonte: o autor.

fazer análise de enriquecimento e também pode encontrar fragmentos conservados dentro dessas regiões.

Desta forma é importante ressaltar que: i) para funcionar corretamente, a análise de enriquecimento precisa de sequências co-reguladas e uma biblioteca de MOTIFS conhecidos enquanto que a análise de MOTIFS atua sobre quaisquer conjunto de sequências e obviamente não precisa de uma biblioteca de MOTIFS conhecidos, pois a descoberta destes padrões é justamente o seu principal objetivo e ii) a análise de enriquecimento frequentemente possui centenas de bases de resolução enquanto que a análise de MOTIFS possui apenas algumas dezenas ou menos.

Existem muitos trabalhos na literatura que se destinam a estudar a descoberta de MOTIFS. Eles utilizam as mais variadas técnicas, das quais podemos destacar: i) algoritmos enumerativos, ii) algoritmos probabilísticos, iii) algoritmos inspirados na natureza e iv) algoritmos híbridos. A Figura 2.1 mostra a frequência absoluta dos 1090 estudos iniciais exibidos na Tabela 3.2 divididos por área. É possível observar que as abordagens probabilísticas possuem um maior número de publicações, seguido pelas técnicas enumerativas, métodos híbridos e algoritmos bio-inspirados respectivamente.

Dentro dos algoritmos enumerativos, temos duas sub-classes principais: i) *pattern driven* e *sample driven*. Enquanto as abordagens baseadas em *pattern driven* enumeram todos os *k-mers* (até mesmo os que não fazem parte do dataset de entrada), os métodos baseado em *sample driven*, enumeram apenas os padrões de comprimento *k* que estão presentes no dataset de entrada. Desta forma, o primeiro é mais preciso e lento e o segundo é mais rápido, porém menos preciso.

Alguns representantes dos algoritmos *pattern driven* são: SPELLER (Sagot, 1998), WEEDER (Pavesi et al., 2001) e YMF (Sinha e Tompa, 2003). Por outro lado, alguns trabalhos relevantes que utilizaram *sample driven* são: FAST-MOTIF (Colombo e Vlassis, 2015), TFBS-GROUP (Jia et al., 2014) e SIOMICS (Ding et al., 2015).

As abordagens probabilísticas podem ser divididas em basicamente em duas grandes sub-categorias: i) baseados no algoritmo *Expectation Maximization* e ii) baseados no algoritmo de *Gibbs Sampling*. Existem ainda os algoritmos gulosos, que embora sejam em menor número, são bastantes importantes e podem ser utilizados como heurísticas de inicialização. A principal característica apresentada por estas metodologias é a representação dos MOTIFS através de modelos estatísticos. A grande maioria utiliza matrizes PFM (Matriz de Posição de Frequência, do inglês *Position Frequency Matrix*), que nada mais são do que distribuições multinomiais independentes estimadas para cada coluna do alinhamento dos padrões biológicos.

Modelos probabilísticos mais avançados, como por exemplo, *Markov Chain* ou *Hidden Markov Chain*, também podem ser empregados. Os trabalhos mais relevantes que utilizam *Expectation Maximization* são: MEME (Bailey et al., 1994), STEME (Reid e Wernisch, 2014) e STREME (Bailey, 2021). Já os mais importante que empregam *Gibbs Sampling* são: MOTIF SAMPLER (Thijs et al., 2001b), ALIGNACE (Hughes et al., 2000), BIOPROSPECTOR (Liu et al., 2001) e PROSAMPLER (Li et al., 2019). Podemos citar ainda o trabalho pioneiro desenvolvido por Stormo e Hartzell (Stormo e Hartzell, 1989) que envolveu a mistura de algoritmos gulosos e probabilísticos e o trabalho desenvolvido por Lawrence et al. (Lawrence et al., 1993) que utilizou inferência Bayesiana para estimar os parâmetros do modelo. É importante destacar também que Thijs et al. (2001a) utilizou uma cadeia de MARKOV de ordem 2 para modelar as probabilidades de fundo dos genomas utilizados no seu trabalho.

Os métodos inspirados na natureza possuem como principal característica a mimetização, i.e., buscam imitar certos comportamentos de organismos, processos e estruturas naturais ou sociais através de algoritmos de otimização. Eles podem ser classificados em: i) baseados na teoria da evolução DARWINIANA OU LAMARCKISTA, tendo como principal representante os algoritmos genéticos; ii) baseados na teoria memética², tendo como principal abordagem os algoritmos meméticos; iii) baseados em enxames, sendo o exame de partículas seu algoritmo mais fundamental; e iv) baseados no funcionamento de estruturas orgânicas e inorgânicas, como por exemplo as redes neurais artificiais, algoritmos baseados no sistema imunológico e algoritmos baseados no resfriamento de metais e vidros. Alguns trabalhos importantes nesta área de pesquisa que valem ser ressaltados são: ST-GA (Stine et al., 2003), GAME (Wei e Jensen, 2006) e MFMD (Garbelini et al., 2018). Por fim, as abordagens híbridas utilizam em seu pipeline duas ou mais técnicas citadas acima. Alguns trabalhos importantes desta categoria são: MD-SCAN (Liu et al., 2002), EM-N-EM (Moses et al., 2003) e MCAT (Yang et al., 2019).

Embora seja válido argumentar que as redes neurais se originaram de um conceito baseado em biologia, especificamente na forma como os neurônios no cérebro humano interagem, é importante fazer uma ressalva neste contexto. Na literatura da Ciência da Computação e Inteligência Artificial, as redes neurais são frequentemente enquadradas de forma mais ampla, com foco em sua implementação matemática e computacional. Embora o modelo de aprendizado de máquina das redes neurais seja inspirado em um processo biológico, a implementação prática dessa técnica se desviou consideravelmente de sua analogia biológica. Portanto, apesar de suas raízes conceituais, as redes neurais não são classificadas estritamente como uma técnica bio-inspirada na literatura predominante. Entretanto, neste trabalho, com o objetivo de organizar os diferentes tipos de algoritmos, as redes neurais são apresentadas juntamente com os métodos bio-inspirados.

²Termo cunhado por Richard Dawkins no seu livro *The selfish gene*.

3.2.1 Algoritmos enumerativos

O objetivo deste tipo de abordagem é enumerar todas as palavras de um determinado tamanho k , sobre o alfabeto $\Sigma = \{a, c, g, t\}$ e computar a similaridade entre elas através de alguma função $g(\cdot)$. Algumas abordagens mais sofisticadas permitem que as palavras contenham certo grau de degeneração. Isso é necessário, pois, embora MOTIFS sejam padrões conservados, dificilmente eles serão idênticos. O problema de MOTIFS que permite degeneração é chamado de (k,d) -MOTIF, onde k é a largura dos MOTIFS e d é a quantidade de bases degeneradas.

A principal característica dos algoritmos enumerativos é a sua capacidade de encontrar soluções exatas, pois o algoritmo faz uma busca exaustiva no espaço de soluções. Em contrapartida, o tempo de execução deste tipo de abordagem é bastante alto e depende da largura dos MOTIFS k , da largura das sequências L e da quantidade de sequências disponíveis no dataset N . Árvores de sufixo e programação paralela são frequentemente utilizadas para acelerar a execução desta classe de algoritmo, contudo, não são suficientes quando os valores de k , L e N são grandes.

3.2.1.1 Principais algoritmos

[Tompas \(1999\)](#) apresentou um algoritmo para descoberta de MOTIFS curtos, cujo objetivo principal foi a detecção dos sítios de ligação de ribossomos. Ele utilizou uma abordagem similar ao trabalho proposto por [Staden \(1989\)](#), porém, ao invés de utilizar somente o valor da frequência de cada padrão, neste estudo, [Tompas \(1999\)](#) utilizou como critério discriminador o z -score. Para isso, ele criou um algoritmo que constrói uma tabela com todos os k -mers pertencentes ao dataset de entrada e calcula a significância estatística para cada um deles levando em consideração a quantidade de ocorrências do padrão no dataset versus a quantidade de ocorrências do padrão caso eles fossem extraídos de uma distribuição de background discreta não-uniforme. Essa abordagem obteve resultados bastante promissores e conseguiu, na maioria dos casos, detectar as regiões conservadas corretamente.

[Pavesi et al. \(2001\)](#) desenvolveram o WEEDER, o primeiro algoritmo a empregar árvores de sufixo para armazenar sub-strings oriundas de sequências de DNA. Além disso, o WEEDER foi também uma das primeiras abordagens que obtiveram um relativo êxito na tentativa de descobrir MOTIFS de tamanhos arbitrários, sem que fosse necessário que o usuário informasse este parâmetro a priori. De acordo com os autores, o algoritmo é capaz de descobrir MOTIFS com uma taxa de erros $\epsilon < 1$, encontrando todos os padrões que aparecem em pelos menos q sequências com até $\epsilon \times m$ mutações, sendo que m é a largura do MOTIF.

[Buhler e Tompas \(2002\)](#) perceberam que algumas instâncias do problema de descoberta de MOTIFS, denominadas (k,d) -MOTIF, no qual os padrões possuem largura k com até d mutações distribuídas uniformemente entre suas posições, são bastante difíceis de serem resolvidas pelos algoritmos clássicos, ainda mais quando estes são inicializados randomicamente. Em particular, os autores identificaram que o problema se torna cada vez mais difícil quando o número de mutações aumenta. Por exemplo, se considerarmos $k = 6$ e $d = 2$, então a probabilidade que um dado L-MER ocorra com até d substituições em uma dada posição de uma sequência aleatória é $\approx 0,038$. Isso significa que de cada 100 sequências aleatórias com tamanho k , aproximadamente 4 delas serão similares ao MOTIF procurado. Se mantivermos a largura em $k = 6$, porém, aumentarmos o número de mutações para $d = 4$, então a probabilidade de ocorrência deste padrão será de aproximadamente $\approx 0,466^3$. Portanto, neste último exemplo, de cada 100 sequências aleatórias de largura k , quase metade será similar à distribuição dos MOTIF, o que dificulta muito

³Se as probabilidades de fundo forem uniformes, i.e. $a = c = g = t = \frac{1}{4}$, então podemos utilizar a distribuição binomial para computar a probabilidade do MOTIF de tamanho k possuir até d mutações, tal que $d \sim \text{Binom}(k, p)$, no qual d é a quantidade de substituições, k é a largura do MOTIF e $p = 1 - \frac{1}{4} = \frac{3}{4}$ é a probabilidade de sucesso.

o processo de busca e separação entre o verdadeiro e o falso⁴ (codificado pela distribuição de fundo) padrão. Por essa razão, [Buhler e Tompa \(2002\)](#) desenvolveram um novo algoritmo baseado na técnica de Projeções Randômicas, do inglês *Random Projections*, conseguindo excelentes resultados em diversos problemas considerados difíceis, como o (14,4), (16,5) e (18,6)-MOTIF. [Riza et al. \(2019\)](#) estenderam este algoritmo desenvolvendo uma nova abordagem empregando a biblioteca de alta performance PBD-MPI ([Chen et al., 2021b](#)) da linguagem de programação R que possui suporte a multi-core.

[Sinha e Tompa \(2003\)](#) apresentaram o YMF, um algoritmo enumerativo baseado em uma versão mais simplificada publicada por [Tompa \(1999\)](#). O YMF foi empregado inicialmente para a descoberta de MOTIFS no organismo *S. cerevisiae*, porém, os autores mostraram que o algoritmo poderia ser utilizado para descobrir MOTIFS em quaisquer organismos. Entretanto, o YMF é uma abordagem enumerativa exata, portanto, o tempo de execução cresce muito rápido em função da largura dos MOTIFS o que implica que este método só pode ser empregado em MOTIFS e conjunto de dados pequenos. Essa abordagem foi uma das primeiras a tratar os MOTIFS como dímeros e utilizar essa característica no cálculo dos p-valores dos sinais encontrados.

[Jia et al. \(2013\)](#) construíram uma abordagem enumerativa inspirada na teoria dos grafos chamada de TFBS-GROUP. Diferente das maioria das abordagens anteriores baseadas em *pattern-driven*, o TFBS-GROUP utilizou a técnica *sample-driven*. A diferença entre ambas é que a primeira enumera todas as 4^k sequências de tamanho k não levando em consideração o dataset de entrada, o que resulta em altos tempos de execução, tornando impraticável sua aplicação em problemas com k grande (≥ 10). Porém, embora possua um tempo de execução elevado, essa abordagem consegue, na maioria das vezes, encontrar o ótimo global. Por outro lado, a técnica *sample-driven* enumera apenas os k -mers pertencentes ao dataset de entrada. Com isso, o algoritmo ganha performance, porém, não existe a garantia de que o ótimo global seja encontrado. O desafio do TFBS-GROUP foi justamente conseguir unir performance na execução do algoritmo com boa acurácia na busca. Para isso, os autores utilizaram um grafo N-PARTITE ligados através de vértices, formando CLIQUES. O TFBS-GROUP foi comparado com outros algoritmos similares e conseguiu bons resultados, principalmente no quesito tempo de execução. Baseado nesta abordagem, [Jia et al. \(2014\)](#) apresentaram uma extensão para este algoritmo com objetivo de detectar padrões em sequências de CHIP-SEQ e CHIP-EXO chamado de FMOTIF. Para isso, os autores utilizaram o método MACS ([Zhang et al., 2008](#)) para detectar a localização de picos nas regiões enriquecidas no sequenciamento bruto e uma estrutura de dados baseada em árvore de sufixos para armazenar os fragmentos resultantes da excisão do cromatina, que os autores chamaram de *mismatched suffix tree*. Baseado nos trabalhos de [Sagot \(1998\)](#) e [Pavesi et al. \(2001\)](#), o FMOTIF conseguiu bons resultados em comparação com os algoritmos do estado da arte, principalmente no quesito tempo de execução.

[Ding et al. \(2014\)](#) desenvolveram o algoritmo SIOMICS, uma abordagem enumerativa cuja finalidade se concentra na descoberta de vários padrões conservados ao mesmo tempo utilizando uma estrutura de dados do tipo árvore, no qual os MOTIFS mais conservados ficam próximos ao nó raiz e os menos conservados ficam nas folhas. Diferente da maioria das abordagens, o SIOMICS tenta identificar múltiplos padrões e seus respectivos co-fatores. Ao invés de comparar os resultados do SIOMICS com outros algoritmos do estado-da-arte, os autores optaram por utilizar o algoritmo STAMP ([Mahony e Benos, 2007](#)) e comparar os MOTIFS descobertos com MOTIFS reais extraídos dos sítios JASPAR ([Castro-Mondragon et al., 2022](#)) e TRANSFAC ([Wingender et al., 2018](#)). Um ano mais tarde, os mesmos autores apresentaram uma extensão deste algoritmo chamado SIOMICS_EXTENSION ([Ding et al., 2015](#)) que podia encontrar MOTIFS de tamanhos variados, não

⁴A palavra falso foi utilizada com a intenção de caracterizar a distribuição de fundo e não com a conotação original da palavra, uma vez que a distribuição de fundo existe e é verdadeira.

sendo mais necessário fornecer a largura dos padrões a priori. Embora a segunda versão seja mais flexível, ela não foi capaz de superar as métricas de desempenho alcançadas pela primeira versão.

Kelil et al. (2014) apresentaram o MOTIF-HOUND, um novo algoritmo para busca de MOTIFS lineares em proteínas. MOTIFS lineares são sequências curtas de aminoácidos responsáveis pela interação proteína-proteína (PPI, do inglês *Protein-Protein Interaction*), que estão envolvidas em funções secundárias, como por exemplo a ligação com outras proteínas para formar um complexo maior ou em outros casos agindo como regulador de funcionalidades, podendo ligar ou desligar determinadas características. O MOTIF-HOUND foi desenvolvido para detectar MOTIFS com certo grau de degeneração e os autores utilizaram um método baseado em *pattern driven* para calcular a significância estatísticas dos padrões encontrados. O algoritmo apresentou um desempenho satisfatório, exibindo na maioria dos casos um bom compromisso entre tempo de execução e acurácia.

Yu et al. (2015) apresentaram o MCES, um algoritmo enumerativo cujo propósito é a identificação dos sítios de ligação dos fatores de transcrição em sequências de CHIP-SEQ utilizando mineração e combinação de sub-strings no modelo ZOOPS (do inglês *Zero or One Occurrence per Sequence*)⁵. Para isso, o MCES utilizou uma estratégia baseada na técnica MAP-REDUCE que, segundo os autores, ajudou a melhorar a performance do algoritmo, permitindo uma execução muito mais rápida. Os resultados obtidos pelo MCES foram comparados com o algoritmo FMOTIF, sendo que o primeiro conseguiu melhores resultados em vários datasets. O MCES foi testado com dados sintéticos e reais e se mostrou bastante rápido na detecção dos padrões conservados.

3.2.2 Algoritmos probabilísticos

Por sua vez, algoritmos probabilísticos lidam bem com MOTIFS amplos, porém, devido a vários fatores, abordagens desenvolvidas sem o devido cuidado tendem a ser pouco precisas. A utilização de algoritmos enumerativos ou probabilísticos depende do contexto. Se uma solução exata é necessária, então temos que dispor de alguma técnica enumerativa. Por outro lado, se uma solução aproximada já é suficiente, então algoritmos probabilísticos se tornam mais atrativos. Na maioria das vezes, conhecer a posição aproximada de um MOTIF já é o bastante para descrevê-lo. Além do mais, com o refinamento das técnicas estatísticas e a quantidade cada vez maior de dados disponíveis, os algoritmos probabilísticos estão se tornando cada vez mais precisos e conseguindo encontrar padrões com resoluções cada vez mais altas. Outra característica importante desta classe de algoritmos é a sua flexibilidade.

Abordagens probabilísticas tendem a ser mais genéricas do que as enumerativas, conseguindo capturar de forma abrangente MOTIFS com alto grau de substituições. Existe ainda um outro problema bastante comum que afeta o desempenho dos algoritmos enumerativos, chamado de degeneração, no qual as bases nucleicas sofrem mutação devido a pressão seletiva, causando a alteração do nível de conservação das sequências biológicas. Com isso, essas abordagens frequentemente sofrem para capturar tais mudanças e tendem ser menos precisas do que técnicas probabilísticas quando esse tipo de evento ocorre.

3.2.2.1 Principais algoritmos

Um dos primeiros trabalhos de descoberta de MOTIFS publicados na literatura foi escrito por Stormo e Hartzell (1989), um dos mais influentes pesquisadores na área de Bioinformática, sendo responsável por inúmeras contribuições e pela publicação de diversos artigos. Neste estudo

⁵Neste modelo, zero ou uma ocorrência do MOTIF por sequência podem ocorrer.

pioneiro, os autores desenvolveram um algoritmo guloso que detecta padrões em sequências de DNA através da construção e comparação de modelos probabilísticos. Para isso, os autores utilizaram o Conteúdo da Informação (IC, do inglês *Information Content*) computado de cada matriz, cujos parâmetros são estimados através de diversos alinhamentos de sequências. Este trabalho é de suma importância para a Bioinformática, principalmente para a área de descoberta de MOTIFS, pois foi a primeira vez que o termo Matriz de Escore de Posição Específica (PSSM, do inglês *Position Specific score Matrix*) foi mencionado. As matrizes PSSM são tão eficientes na representação de MOTIFS biológicos, que atualmente elas ainda são a forma mais popular para representar este tipo de estrutura. Neste trabalho, os autores analisaram o dataset CRP (*cAMP Receptor Protein*), que possui 18 sequências com 105 pares de base cada, conseguindo excelentes resultados na confirmação de 23 sítios de ligação.

Um ano mais tarde, Lawrence e Reilly (1990) publicaram um estudo utilizando a Maximização da Expectativa (EM, do inglês *Expectation Maximization* (Dempster et al., 1977)), sendo esta a primeira vez que o EM foi empregado na descoberta de MOTIFS. Neste artigo, os autores propuseram uma função de avaliação baseado na log-verissimilhança que, ao invés de considerar apenas os dados dos sítios de ligação, também incorporou as sequências que não faziam parte desta classe. Além disso, este trabalho foi um dos primeiros a apresentar uma modelagem que empregou a correlação entre bases para melhorar a resposta final das predições.

Embora o EM seja bastante eficiente na busca por padrões biológicos, ele é bastante suscetível aos parâmetros escolhidos na inicialização, o que pode levar o algoritmo a ficar preso em ótimos locais. Em outras palavras, não existe garantias de que o EM encontre o ótimo global, sendo necessário recorrer as algumas técnicas, como por exemplo, a reinicialização aleatória (*Random Restart*), cujo objetivo é reconfigurar os parâmetros iniciais, fazendo a abordagem recomençar a modelagem a partir de outro ponto. Pensando nessas limitações, Lawrence et al. (1993) desenvolveram uma técnica baseada na amostragem de Gibbs (*Gibbs Samplig*) visando tratar este problema e melhorar a performance da busca. Assim como no estudo publicado por Stormo em 1989, neste trabalho, os autores propuseram uma função baseada no IC, também chamada de divergência de Kullback-Leibler, sendo o objetivo principal do algoritmo realizar a estimação dos parâmetros na tentativa de maximizar esta função. Além do mais, este artigo foi pioneiro na utilização da amostragem de Gibbs aplicado ao problema de descoberta de MOTIFS obtendo ótimos resultados práticos.

Os estudos publicados por Lawrence et al. em 1990 e 1993, possuíam uma grande deficiência: eles eram capazes de modelar apenas um MOTIF por sequência, o que ficou conhecido na literatura como OOPS (*One Occurrence Per Sequence*)⁶. Essa limitação é bastante significativa e, em geral, diminui a capacidade de predição dos algoritmos, pois, em muitos casos, devido a alta complexidade que determinados genes são regulados, podem existir várias cópias (degeneradas) de um mesmo MOTIF por sequência. Por essa mesma razão, em algumas situações, as sequências de entrada podem exibir zero MOTIFS. Falando de outra forma, a presunção de que cada sequência possua exatamente uma cópia de cada tipo de MOTIF é muito forte. Com isso em mente, Bailey et al. (1994) publicaram o MEME (*Multiple EM for Motif Elicitation*), uma abordagem que modificou o algoritmo originalmente criado por Lawrence e Reilly (1990), incorporando uma modelagem mais robusta, capaz de identificar uma quantidade arbitrária de MOTIFS por sequência com relativa precisão. O MEME também inovou implementando um algoritmo secundário, que, ao invés de simplesmente escolher o modelo inicial ao acaso, utilizava uma estratégia gulosa para selecionar os parâmetros iniciais. Até a data de escrita desta tese, o MEME ainda é o algoritmo mais popular e o mais utilizado pela comunidade científica para realizar a análise de MOTIFS.

⁶O termo foi preservado em inglês.

[Liu et al. \(1995\)](#) publicaram uma extensão dos algoritmos de [Lawrence et al. \(1993\)](#), porém, neste estudo eles empregaram uma abordagem Bayesiana juntamente com o algoritmo de *Gibbs Sampling*. Para isso, os autores utilizaram a distribuição de Dirichlet para modelar a distribuição de probabilidade a priori e a distribuição Multinomial para modelar a distribuição dos dados, resultando na conhecida distribuição conjugada Dirichlet-Multinomial. A grande vantagem de se empregar uma distribuição conjugada é que a distribuição a posteriori pode ser calculada de forma fechada, e isso se traduz em algoritmos mais rápidos. A desvantagem é que nem sempre a distribuição conjugada é a mais adequada para realizar a modelagem estatística e, em muitos casos, outras distribuições mais complexas, que não possuem forma fechada, são mais pertinentes. Foi a primeira vez que este tipo de abordagem foi utilizada na busca por MOTIFS, alcançando excelentes resultados em diversos datasets reais.

[Hertz e Stormo \(1999\)](#) expandiram as capacidades do algoritmo guloso publicado em 1989 por Stormo e Hartzell. Eles utilizaram a mesma base da antiga abordagem, mas desta vez, implementaram algumas modificações que permitiu ao usuário não especificar a largura dos MOTIFS diretamente. Ao invés disso, o programa aceita como entrada um valor de desvio padrão e a partir desta informação, calcula a melhor largura possível. Além disso, os autores também demonstraram duas maneiras de se calcular o p-valor de um alinhamento de MOTIFS. Isso foi útil, pois, a partir da publicação destes algoritmos, tornou-se possível comparar modelos com larguras e quantidade de sequências distintas, o que na prática, estabeleceu uma forma bastante confiável de realizar a seleção de modelos com diferentes graus de liberdade.

O ALIGNACE é um algoritmo baseado na amostragem de *Gibbs* que foi concebido por [Hughes et al. \(2000\)](#), tomando como base a abordagem desenvolvida por [Lawrence et al. \(1993\)](#). As principais mudanças deste estudo em relação ao original, guardam relação com aspectos pertinentes aos tipos de organismos avaliados. Por exemplo, para o organismo *S. cerevisiae*, os autores utilizaram as probabilidades de fundo estimada através da análise do genoma deste espécime. Além disso, as duas fitas (direta e inversa) do gene foram utilizadas para construir o alinhamento de MOTIFS e consequentemente realizar a estimação dos parâmetros. É interessante notar que, ao contrário da grande maioria dos artigos publicados nesta área, este estudo não utilizou sequências em sobreposição. Infelizmente não ficou totalmente claro a razão pela qual esse sistema foi adotado. Por fim, os autores utilizaram um procedimento de mascaramento de nucleotídeos probabilístico, elevando a capacidade do ALIGNACE na detecção de MOTIFS de diferentes famílias.

Ainda nesta linha, [Thijs et al. \(2001b\)](#) desenvolveram o MOTIF-SAMPLER, um algoritmo baseado na amostragem de *Gibbs* que também estendeu a abordagem original, publicada por [Lawrence et al. \(1993\)](#), introduzindo duas modificações: a inclusão de uma distribuição de probabilidade para estimar a quantidade de cópias de um mesmo MOTIF em cada sequência e a utilização de um modelo de fundo mais sofisticado, empregando cadeias de MARKOV de ordem superior, cujos resultados práticos foram publicados pelos mesmos autores de forma mais detalhada em outro trabalho ([Thijs et al., 2001a](#)). A importância deste estudo se deve principalmente pela sua capacidade de mostrar que modelar as probabilidades de fundo levando em consideração a dependência estatísticas entre nucleotídeos vizinhos pode contribuir positivamente na melhoria da acurácia geral dos algoritmos. O MOTIF SAMPLER alcançou bons resultados na localização de MOTIFS nos organismos *Arabidopsis thaliana* e *Saccharomyces cerevisiae*.

[Liu et al. \(2001\)](#) publicaram o BIOPROPECTOR, uma abordagem baseada no algoritmo de *Gibbs* e que também foi fortemente influenciado pelo clássico artigo publicado por [Lawrence et al. \(1993\)](#). Neste estudo, os autores desenvolveram um abordagem chamada de THRESHOLD-SAMPLER, capaz de reconhecer vários MOTIFS por sequência através da definição de valores de corte iterativamente estimados ao longo da execução do algoritmo. Influenciados por [Thijs et al.](#)

(2001b), os autores do BIOPROPECTOR modelaram as probabilidade de fundo utilizando cadeias de MARKOV de ordem superior. Por fim, também foi desenvolvida uma forma rudimentar de lidar com MOTIFS separados através de *gaps*. Embora seja raro que MOTIFS apresentem *gaps*, esta foi a primeira vez que um algoritmo tentou superar este problema. Com isso, o BIOPROPECTOR conseguiu bons resultados em diversos datasets reais e sintéticos.

Keich e Pevzner (2002a) publicaram um estudo no qual foi identificado que as abordagens tradicionais, que empregam por exemplo algoritmos gulosos, maximização da expectativa ou o algoritmo de Gibbs falham. Eles perceberam que a precisão destas técnicas decresce em função de três fatores principais: i) incremento na largura dos MOTIFS; ii) incremento na largura das sequências; e iii) incremento no nível de degeneração dos MOTIFS. Em outras palavras, eles chegaram na conclusão de que existe um limite teórico no qual as abordagens tradicionais não conseguiam ultrapassar. Levando isso em consideração, Keich e Pevzner (2002b) desenvolveram o MULTIPROFILER uma abordagem que combina a busca de *strings* em diversas vizinhanças com a modelagem de vários perfis probabilísticos utilizando dinucleotídeos parcialmente dependentes. Este algoritmo conseguiu excelentes resultados no dataset CRP (o mesmo utilizado por Stormo e Hartzell (1989)) e também em diversos datasets sintéticos, em particular na resolução do problema (15, 4)-MOTIF.

McLaughlin et al. (2005) inovaram ao apresentar um algoritmo que não realiza a predição de MOTIFS diretamente, mas sim indiretamente, através da análise da estrutura tri-dimensional das proteínas associadas aos fatores de transcrição. De acordo com os autores, o propósito da abordagem não é substituir os algoritmos clássicos e sim criar uma nova técnica que seja complementar, capaz de atuar em situações nas quais elas não podem ser utilizados. Os autores utilizaram dados de diversos bancos de proteínas, tais como UNIPROT, SWISSPROT, TREMBLE, TIGRFAM, PROSITE, PRINTS e PFAM para construir modelos capazes de identificar potenciais proteínas que interagem com DNA. Primeiramente foi desenvolvido um modelo base inspirado no trabalho publicado por McLaughlin e Berman (2003), porém, desta vez os autores utilizaram *alpha helix* extraídos do dataset RCBS Protein data Bank. Ainda que na média as métricas de desempenho desta abordagem sejam inferiores às alcançadas pelos algoritmos clássicos, resultados bem sucedidos foram obtidos na detecção de diversos MOTIFS em diferentes organismos, mostrando que a análise tri-dimensional das proteínas pode ser um caminho promissor na busca e detecção de MOTIFS.

Heinz et al. (2010) investigaram os mecanismos regulatórios dos macrófagos e das células B em relação a sua linhagem e descobriram que fator comum PU.1 possui interações colaborativas com outros pequenos conjuntos de fatores de transcrição e estabelecem ligação com a maioria dos promotores distais de regiões genômicas marcadas por modificações epigenéticas H3K4me1. Para isso eles desenvolveram uma abordagem chamada HOMER (*Hypergeometric Optimization of Motif Enrichment*) que utiliza a distribuição hipergeométrica para encontrar MOTIFS utilizando dois conjuntos de sequências: o primeiro é um dataset de entrada conhecidamente enriquecido; o segundo, chamado de dataset de controle, possui sequências não enriquecidas. Os autores obtiveram resultados bastante promissores na análise de dados de CHIP-SEQ sugerindo que combinações simples de fatores de transcrição podem ser essenciais para especificar os locais genômicos responsáveis pela identidade celular e respostas específicas do tipo de célula a diversas entradas de sinalização.

Arnold et al. (2012) desenvolveram o MOTEVO, um arcabouço Bayesiano que realiza a inferência de regiões cis-regulatórias através do alinhamento de múltiplas sequências e informações filogenéticas de organismos ortólogos. O algoritmo recebe como entrada um conjunto de sequências, uma coleção de uma ou mais matrizes PWM e a árvore filogenética das espécies relacionadas. O MOTEVO devolve como saída a distribuição a posteriori e a versão

atualizada do conjunto de matrizes PWM e as probabilidades associadas a cada posição válida nas sequências de entrada de conter ou não um elemento regulatório. O MOTEVO foi comparado com vários algoritmos do estado da arte obtendo importantes resultados na detecção de diversos elementos regulatórios em organismos eucarióticos superiores.

Reid e Wernisch (2014) apresentaram uma nova abordagem denominada STEME, um algoritmo que estende as funcionalidades do MEME empregando um modelo de mistura finita juntamente com o que os autores chamaram de *Posição Específica a Priori*, método que inclui nos dados de entrada, informações adicionais que não participam das sequências iniciais, ajudando no processo de busca. Alguns exemplos são: i) informações filogenéticas; ii) ocupação dos nucleossomos; e iii) sequências negativas de controle. Além disso, devido ao EM ser sensível ao local de inicialização, os autores também apresentaram um modelo discriminativo/enumerativo utilizado no processo de inicialização, com o objetivo de encontrar sinais e servir de molde para a criação do modelo inicial. O STEME foi comparado com os algoritmos DREME e TRAWLER obtendo bons resultados em dados de CHIP-SEQ, principalmente em MOTIFS grandes de com alto grau de degeneração.

Por fim, Bailey (2021) publicou o STREME, um dos algoritmos mais recentes desta categoria, que utiliza árvores de sufixo genéricas para armazenar as sequências de entrada de um dado conjunto de dados. Segundo o autor, diferentemente de outras abordagens que frequentemente utilizam árvores de sufixo para acelerar a contagem de *matches* e assim calcular alguma medida de escore, o STREME utiliza essa mesma estrutura para computar os *matches* das todas as sequências extraídas do conjunto de dados com matrizes PWM, que representam os MOTIFS candidatos. Além disso, o STREME utiliza o teste exato de FISHER (Fisher, 1922) ou o teste BINOMIAL para avaliar a significância estatística dos MOTIFS encontrados. Esta abordagem foi comparada a diversos métodos do estado da arte (incluindo o MEME, inventado pelo mesmo autor) conseguindo resultados promissores na maioria dos experimentos.

3.2.3 Algoritmos inspirados na natureza

Abordagens inspiradas na natureza englobam algoritmos que utilizam técnicas que foram tipicamente influenciadas por eventos naturais (Dhiman e Kumar, 2018). Exemplos clássicos são: algoritmo genéticos e meméticos, enxame de partículas, arrefecimento simulado e redes neurais artificiais. Essa classe de algoritmos possui como principal característica a flexibilidade, pois podem resolver uma ampla gama de problemas e se adaptar as mais variadas circunstâncias apenas com pequenas modificações. Por essa razão, são abordagens largamente utilizadas em métodos de otimização complexa (no qual a derivada de uma função não existe ou não pode ser calculada analiticamente), em problemas combinatórios e em problemas cuja solução exata e correta demandaria recursos computacionais extremos ou mesmo inexistentes, como ocorre em problemas NP-completos.

Infelizmente, essa generalização tem um preço. Frequentemente, essa classe de algoritmos tende a ser menos precisa do que outros tipos de abordagens que foram desenvolvidas de forma específica para um dado problema. A principal desvantagem dos algoritmos bio-inspirados é sua incapacidade de obter respostas exatas. Isso ocorre principalmente nos algoritmo genéticos e meméticos e nas abordagens baseadas em enxame de partículas. Entretanto, embora essa característica seja verdadeira, algoritmos bio-inspirados conseguem encontrar na maioria das vezes boas soluções, bastante próximas da solução ótima. Isso os torna muito atrativos, principalmente em problemas da classe NP-completo, no qual soluções exatas só podem ser obtidas para instâncias pequenas de problemas de decisão.

3.2.3.1 Principais algoritmos

Stine et al. (2003) desenvolveram uma abordagem utilizando algoritmos genéticos para descobrir padrões conservados em sequências genômicas co-expressadas. Segundo os autores, foi utilizado um Algoritmo Genético Estruturado (ST-GA, do inglês *Structured Genetic Algorithm*), cuja função de avaliação foi baseada no algoritmo de alinhamento local BLAST (Altschul et al., 1990). Além disso, os autores inseriram um operador de aglomeração determinístico (*Deterministic Crowding*) para lidar com a multimodalidade da função objetivo. O ST-GA foi capaz de superar diversos algoritmos do estado da arte no problema (κ, D) -MOTIF, porém conquistou resultados ruins quando $d \geq 4$ mutações.

Congdon et al. (2005) publicaram o GAMI (*Genetic Algorithm for Motif Inference*), uma abordagem evolutiva que utiliza o algoritmo genético clássico para descobrir novos MOTIFS em sequências de DNA co-reguladas. Este método utilizou uma função objetivo inspirada no programa BLAST (Altschul et al., 1990), no qual é computado um escore entre o todos os k -mers do dataset e uma população de soluções gerada de forma aleatória. A diferença entre eles é a forma como a similaridade é computada. Enquanto o BLAST utiliza um modelo baseado no maior escore possível (que segue uma distribuição de Gumbel (1935)), o GAMI calcula o maior escore consecutivo possível. O GAMI também modificou o operador de elitismo, salvando não apenas o melhor indivíduo de cada geração, mas 50% deles. Anos mais tarde, Thompson e Congdon (2014) apresentaram uma versão melhorada chamada de GAMI-CRM, com objetivo de encontrar módulos cis-regulatórios (CRM, do inglês, *Cis-Regulatory Modules*) em sequência co-reguladas. Os CRM's são elementos maiores, que apresentam elevado grau de conservação e que possuem em seu interior um ou mais TFBS. O GAMI, tanto na versão original quanto na melhorada, apresentou bons resultados na descoberta de MOTIFS e de elementos cis-regulatórios, conseguindo superar algumas abordagens clássicas em diversos datasets sintéticos e reais.

Wei e Jensen (2006) apresentaram o GAME (*Genetic Algorithm for Motif Elicitation*), uma das primeiras abordagens a empregar algoritmos genéticos com seus respectivos operadores clássicos de crossover e mutação. A grande inovação do GAME foi a apresentação de dois novos operadores genéticos, específicos para o problema de MOTIFS: SHIFT e ADJUST. O operador SHIFT é bastante similar ao empregado no trabalho publicado por Lawrence et al. (1993), movendo o alinhamento de MOTIFS para a direita ou esquerda buscando uma melhora na função de *fitness*. O operador ADJUST é mais interessante. Ele adiciona (ou remove) iterativamente outras sub-sequências de mesmo tamanho ao alinhamento de MOTIFS tentando melhorar o escore da função *fitness*. Por fim, o GAME também utilizou uma função objetivo baseada na estatística Bayesiana desenvolvida por Jensen et al. (2004) que leva em consideração a distância de *Kullback-Leibler* e um parâmetro p que mede a abundância de MOTIFS no dataset. De acordo com os autores, vários experimentos foram realizados tanto em datasets sintéticos quanto reais mostrando que o GAME foi capaz de superar algoritmos do estado da arte, como o MEME e o GIBBS MOTIF SAMPLER em diversos datasets.

Kaya (2009) desenvolveram o MOGAMOD (*Multi-objective Genetic Algorithm*), a primeira abordagem genética multi-objetivo a lidar com o problema de descoberta de MOTIFS. O MOGAMOD utilizou o algoritmo NSGA-II com três objetivos a serem maximizados: similaridade, tamanho e suporte⁷. Novamente um operador de SHIFT bastante similar ao desenvolvido por Lawrence et al. (1993) foi utilizado neste estudo para melhorar as soluções finais encontradas pelo algoritmo. Segundo os autores, o MOGAMOD conseguiu resultados promissores quando comparado a

⁷No contexto deste artigo, o termo suporte está sendo empregado como sinônimo de quantidade. Portanto, suporte seria a quantidade que se deseja estimar de padrões conservados existentes no conjunto de dados

algoritmos populares, como MEME, ALIGNACE e WEEDER, superando esses algoritmos clássicos principal no tempo de execução.

Bi (2012) desenvolveu o MA-MOTIF, um algoritmo memético capaz de encontrar sinais regulatórios em sequências biológicas. Segundo os autores, a principal vantagem do algoritmo é a seu tempo de execução, conseguindo ser até 16 vezes mais rápido do que métodos que utilizam a maximização da expectativa. O MA-MOTIF utiliza todas as etapas clássicas dos algoritmos genéticos como otimizador global, tais como geração da população inicial, mutação, crossover, elitismo e seleção, bem como uma etapa de adicional de otimização local que emprega o algoritmo de maximização da expectativa. Em datasets sintéticos, o MA-MOTIF foi comparado apenas com outros algoritmos meméticos e conseguiu resultados melhores do que seus pares. Em datasets reais, ele foi comparado com algoritmo clássicos, conseguindo também resultados expressivos.

Garbelini et al. (2018) publicaram o MFMD (*Memetic Framework for Motif Discovery*), um algoritmo memético para descoberta de MOTIFS em sequências biológicas não codificantes de genes supostamente co-regulados. O MFMD é composto por três etapas: i) pré-processamento; ii) *pattern discovery* e; iii) *pattern matching*. A principal etapa é a *pattern discovery*, no qual o algoritmo memético é empregado. A etapa de pré-processamento utiliza entropia de Shannon para remover sequências espúrias de baixa complexidade e a etapa de *pattern matching* transforma os scores (obtidos através do cômputo do conteúdo da informação, proposto inicialmente por Stormo e Hartzell (1989)) em z-scores. Após isso, o algoritmo calcula os p-valores de cada um dos *k-mers* pertencentes ao dataset de entrada para obter a significância estatística de cada um deles. O MFMD inovou ao empregar uma função bi-objetivo, utilizando a soma de pesos ponderado para estabelecer o valor final do fitness de cada indivíduo, representados por matrizes PWM. Além disso, o algoritmo empregou o GRASP (*Greedy Randomized Search Procedure*) como heurística construtiva. Vários testes foram conduzidos, tanto com datasets sintéticos quanto reais, sendo demonstrado que o MFMD conseguiu superar os algoritmos clássicos, tais como o MEME e GIBBS MOTIF SAMPLER em diversas situações, conquistando métricas de desempenho melhor do que seus pares.

Gohardani et al. (2019) desenvolveram o MOICA (*Muti-objective Imperialist Competitive Algorithm*), um algoritmo multi-objetivo baseado nas abordagens publicadas por Kaya (2009) e González-Álvarez et al. (2011), que pode ser considerado uma extensão de ambos. Tal qual o MOGAMOD, o MOICA também utilizou 3 funções objetivos: similaridade, suporte e tamanho. A inovação ficou por conta do uso de uma função de complexidade, estudada primeiramente por (Fogel et al., 2004). Essa função é semelhante a função de entropia de Shannon, no qual valores menores implicam que a sequência é mais homogênea e valores maiores indica que existe maior heterogeneidade. Isso é importante, pois sequências com baixa complexidade podem indicar espúrios. O MOICA foi comparado com diversos algoritmos multi-objetivo utilizando dados do sítio TRANSFAC (Wingender et al., 1996) e conseguiu resultados importantes na maioria dos experimentos condizidos.

Ashraf e Shafi (2020) apresentaram o MFEA, um algoritmo evolutivo mono-objetivo simples que utilizou sequências consenso para a codificação da população inicial. Além disso, os autores empregaram um sistema de mutação baseado na distribuição Gaussiana, permitindo que, ao sofrer mutação, o novo valor sorteado para o gene fique próxima ao valor antigo, garantindo pequenos passos em direção ao ótimo global. O MFEA foi testado no conjunto de dados proposto por Tompa et al. (2005) contra diversos algoritmos do estado da arte e conseguiu resultados bastante animadores, sendo capaz superar tais abordagens em alguns deles.

Finalmente temos o DEEP-GRN (Chen et al., 2021a) um algoritmo que utiliza aprendizado profundo, modelos convolucionais (cnn's) e redes recorrentes (rnn's) para representar MOTIFS biológicos. Os experimentos foram realizados utilizando dados de CHIP-SEQ extraídos do sítio

ENCODE-DREAM (Sloan et al., 2016) e os resultados foram comparados com outros algoritmos similares, que também empregaram aprendizado profundo. O DEEP-GRN foi capaz de modelar e prever com sucesso diversos tipos de MOTIFS biológicos e conseguiu resultados promissores quando comparado a outros métodos análogos.

3.2.4 Algoritmos híbridos

Essa classe de algoritmos é caracterizada por abordagens que empregam duas ou mais técnicas computacionais. Isso ocorre principalmente quando o problema a ser tratado é muito complexo. Desta forma, em muitos casos, é mais eficiente dividir o problema em partes menores e utilizar diferentes algoritmos para resolver cada parte. Ao dividir um problema grande em fragmentos menores é possível também realizar uma análise mais minuciosa e identificar as características fundamentais de cada parte, empregando diferentes algoritmos em cada uma delas. Isso é essencial para a correta compreensão do problema a fim de se obter melhores resultados quando comparados a execução de abordagens executadas isoladamente.

3.2.4.1 Principais algoritmos

Com o advento de técnicas mais avançadas para encontrar MOTIFS experimentalmente, como CHIP-ON-CHIP (Blat e Kleckner, 1999), ficou cada vez mais simples e eficiente descobrir estes padrões no nível genômico através de ensaios científicos. Entretanto, a maioria das abordagens experimentais consegue mapear apenas os locais de interação entre proteína-DNA no interior de uma determinada área de resolução, que pode variar entre 1 a vários quilobases. Desta forma, Liu et al. (2002) desenvolveram o MDSCAN, uma abordagem que une benefícios de duas estratégias conhecidas para encontrar MOTIFS – enumeração de palavras e matrizes PSSM – em um único algoritmo. Além disso, eles utilizaram o ranqueamento do CHIP-ARRAY para agilizar a busca e melhorar a taxa de acertos. Os autores obtiveram resultados promissores em experimentos relacionados à levedura e conseguiram prever dois MOTIFS que não haviam sido identificados nos experimentos laboratoriais. Aproveitando o bom desempenho deste algoritmo, Conlon et al. (2003) utilizaram o MDSCAN em conjunto com um modelo de regressão linear e criaram o MOTIF REGRESSOR, obtendo resultados significativamente melhores do que o MDSCAN nos mesmo conjunto de dados.

Moses et al. (2003) apresentaram uma abordagem denominada EM-N-EM, que estende o algoritmo MEME, utilizando um modelo de mistura no qual a maximização da expectativa foi empregada juntamente com informações filogenéticas obtidas a priori, com o objetivo de minimizar a quantidade de falsos positivos. Foram utilizados vários modelos filogenéticos, como por exemplo o modelo de substituição de Jukes-Cantor, Felsenstein'81 e Halpern-Bruno. Os autores conseguiram bons resultados nos organismos *S. bayanus*, *S. cerevisiae*, *S. kudriavzevii*, *S. mikatae* e *S. paradoxus*, obtendo respostas superiores quando comparado com o algoritmo MEME executado sem as informações filogenéticas.

Reddy et al. (2006) desenvolveram uma abordagem híbrida que utiliza o algoritmo RANDOM PROJECTIONS para encontrar soluções promissoras e em seguida executa o EXPECTATION MAXIMIZATION nas soluções iniciais para encontrar ótimos locais. Após isso, um algoritmo de busca sistemática em vizinhança local é aplicado nas soluções devolvidas pelo EM com objetivo de realizar pequenas perturbações e chegar eventualmente no ótimo global. Por fim, o algoritmo reporta a sequência consenso mais promissora encontrada nas fases anteriores. Segundo os autores, essa técnica conseguiu bons resultados tanto em datasets sintéticos quanto em reais e em alguns casos conseguiu superar as métricas conquistadas por vários algoritmos clássicos.

Hu et al. (2008) publicaram o MOPAT (*Motif Pair Tree*), uma abordagem híbrida que utiliza uma estrutura de dados do tipo árvore juntamente com tabelas hash para encontrar módulos cis-regulatórios (CRM's, do inglês *Cis Regulatory Modules*). Diferente de outras técnicas, o MOPAT é um algoritmo supervisionado, que necessita de um conjunto de matrizes PWM como entrada para construir um modelo probabilístico e posteriormente varrer sequências genômicas em buscas de regiões que contenham CRM's. O MOPAT utiliza como métrica a razão *log-likelihood* para medir o desempenho de alinhamentos e calcular a distribuição dos MOTIFS através de técnicas de *monte-carlo*. Os autores testaram o MOPAT em diversos datasets e compararam seu desempenho com as abordagens COMPEL, CBUST e DIRE conseguindo bons resultados, principalmente em relação ao COMPEL e ao CBUST.

Shashidhara et al. (2010) desenvolveram um algoritmo híbrido utilizando o RANDOM PROJECTIONS (RP) e o EXPECTATION MAXIMIZATION. Neste trabalho, os autores utilizaram o RP para obter soluções promissoras e após isso aplicaram o EM nos melhores *buckets* com o objetivo de refinar a busca. Após isso, o algoritmo realiza mais uma etapa de refinamento através de uma técnica chamada pelos autores de Pesquisa de Perfil de Vizinhança (do inglês, *Neighborhood Profile Search*). Nesta etapa, os *eigenvectors* são calculados a partir da matriz Hessiana (gerada com base na matriz PSSM inicial). Com isso, os pontos de sela são calculados e as regiões estabilizadoras (convergência) são encontradas. Essas regiões são utilizadas como entrada do algoritmo EM, que segue executando até que um número máximo de iterações seja atingido. Conforme exposto pelos autores, este algoritmo foi comparado com o ALIGNACE, WINNOVER e SP-STAR, conseguindo resultados melhores na maioria dos experimentos.

Wang e Miao (2014) apresentaram o GAEM (*Genetic Algorithm with Expectation Maximization*), uma abordagem para descoberta de MOTIFS baseada no HIGEDA (Le et al., 2010). Diferente de outras abordagens que buscam apenas MOTIFS contíguos sem *gaps*, o principal objetivo de GAEM é encontrar padrões que possuam deleções, inserções e substituições, i.e., MOTIFS com *gaps*. A principal razão para a maioria dos algoritmos não considerar *gaps*, é porque não é comum MOTIFS reais apresentarem essa característica. Porém, em alguns casos, modelar *gaps* pode ser útil, principalmente quando o objetivo é encontrar MOTIFS em determinados organismos superiores. O GAEM faz isso através de um algoritmo de alinhamento dinâmico entre a sequência encontrada e a sequência consenso. Segundo os autores, vários experimentos foram conduzidos comparando o GAEM com o HIGEDA e com outros similares. Os resultados mostraram que o GAEM conseguiu superar seus concorrentes em diversos datasets, em especial nos casos no qual os MOTIFS exibiram maior degeneração, apresentando um número maior de deleções, substituições e inserções.

Colombo e Vlassis (2015) publicaram o FAST-MOTIF uma abordagem que inovou ao utilizar a decomposição espectral ao invés do algoritmo EM para estimar os parâmetros do modelo estatístico subjacente. Além disso, em conjunto com a decomposição espectral, foi empregada uma modelagem estatística levemente diferente da usual, denominada mistura do produto de distribuições (do inglês *Mixture of Product Distributions*). O FAST-MOTIF foi testado utilizando dados de HT-SELEX extraídos do estudo publicado por Jolma et al. (2013) e colocados para consulta no ENA (*European Nucleotide Archive*). Devido aos métodos utilizados na sua confecção, o FAST-MOTIF foi desenvolvido para ser utilizado com grandes volumes de dados. Por essa razão, seu melhor desempenho ocorreu justamente neste tipo de dataset quando comparado com os algoritmos DREME (Bailey, 2011) e Multinomial (Jolma et al., 2013).

Yang et al. (2019) desenvolveram o MCAT, do inglês (*Motif Combining and Association Tool*), uma ferramenta de *ensemble* que combina seis diferentes programas de descoberta de MOTIFS: MEME, BIOPROSECTOR, DECOD, XXMOTIF, WEEDER e CMF. O coração desta abordagem é o algoritmo de voto, baseado em ranqueamento. O MCAT calcula o escore dos MOTIFS através de

três formas distintas, sendo que duas delas são bem conhecidas: *z-score* e *p-value*. A última foi chamada pelos autores de *comparison score* e leva em consideração a posição em relação às suas sequências originais. O MCAT foi comparado com outras duas abordagens baseadas em *ensemble*, DYNAMIT (Dassi e Quattrone, 2016) e EMD (Hu et al., 2006), e conseguiu resultados melhores, sobretudo no organismo *S. cerevisiae*, em datasets extraídos do sítio www.yeastract.com (Teixeira et al., 2006).

Ngo et al. (2019) apresentaram o M-EPIGRAM, uma atualização baseada no seu antecessor EPIGRAM (Whitaker et al., 2015), cujo objetivo é encontrar MOTIFS em sequências de DNA que sofreram metilação. A metilação em DNA é uma das principais marcas epigenéticas que pode levar tanto ao aumento quanto à diminuição da expressão de um determinado gene. Por essa razão, descobrir essas regiões é de suma importância para a detecção de certas doenças ou mesmo para evidenciar a razão pela qual alguns organismos são imunes a determinados distúrbios. De acordo com os autores, uma das principais vantagens do M-EPIGRAM em relação aos seus concorrentes é sua capacidade de trabalhar com arquivos volumosos. O algoritmo primeiro computa o *score* enriquecido para cada *k-mer* baseado na quantidade de vezes que ele aparece nas sequências de entrada. Após isso, as sequências de entrada são embaralhadas e o mesmo cálculo é realizado. Em seguida, os *k-mers* são ranqueados de forma decrescente, formando uma lista de MOTIFS, que são escolhidos através de seu *p-value*. O M-EPIGRAM foi testado em datasets reais e sintéticos extraídos do projeto ENCODE, da plataforma GEO e do sítio HOCOMOCO, conseguindo resultados melhores que algoritmos do estado da arte, especialmente nos casos em que os MOTIFS sofreram metilação.

Por fim, Li et al. (2019) publicaram uma nova abordagem para descoberta de MOTIFS chamada PROSAMPLER que combina a técnica enumerativa *sample-driven* com o algoritmo de amostragem de Gibbs desenvolvido por Lawrence et al. (1993). O PROSAMPLER possui 5 etapas: i) geração das sequências de controle através de cadeias de Markov de ordem 3; ii) contagem da frequência de todos os *k-mers* pertencentes ao dataset de entrada; iii) identificação dos *k-mers* mais significativos e construção das matrizes PWM; iv) mesclagem das matrizes PWM similares utilizando a distância Sandelin-Wasserman (Wasserman e Sandelin, 2004) e; v) utilização do algoritmo de amostragem de Gibbs para refinar a busca. O PROSAMPLER foi comparado com diversos algoritmos do estado da arte e conseguiu bons resultados em dados de CHIP-SEQ sintéticos e reais extraídos dos sítios JASPAR e *Gene Expression Omnibus*, principalmente no quesito tempo de execução.

3.3 ANÁLISE CRÍTICA DOS TRABALHOS RELACIONADOS

Nesta seção, uma análise comparativa dos algoritmos-chave discutidos nas seções anteriores foi realizada, evidenciando as particularidades, vantagens, limitações e respectivas áreas de aplicação de cada um. A literatura revisada para esta tese apresenta uma diversidade de algoritmos para a descoberta de MOTIFS de sequência, cada um com seus próprios pontos fortes e limitações. O intuito desta análise é identificar oportunidades de melhoria ou lacunas não atendidas pelos métodos existentes, bem como destacar as características-chave de cada método para fornecer uma comparação informada e justificar a direção da pesquisa deste trabalho. Para aprofundar a análise, selecionamos os seis artigos detalhados na Tabela 3.1, por representarem de forma abrangente o estado da arte e porque apresentam resultados relevantes para nossa área de estudo.

Começando com o trabalho de (Stormo e Hartzell, 1989; Stormo, 2000), *Identifying protein-binding sites from unaligned DNA fragments*, no qual os autores apresentam uma técnica inovadora para a identificação de sítios de ligação de proteínas. Este trabalho foi um dos

primeiros métodos robustos e eficazes para encontrar sequências conservadas em conjuntos de fragmentos de DNA. Embora forneça informações importantes sobre a ligação de proteínas, sua aplicação é limitada pelo pressuposto de que os sítios de ligação são independentes uns dos outros. Além disso, o método depende fortemente da qualidade e quantidade dos dados de entrada. Se apenas alguns fragmentos de DNA que contêm o MOTIF estão disponíveis, ou se esses fragmentos são muito curtos, a eficácia do método pode ser comprometida. Da mesma forma, a presença de ruído nos dados de entrada (por exemplo, sequências que não são realmente sites de ligação para a proteína de interesse) pode prejudicar a precisão dos resultados. Estas limitações do modelo de Stormo e Hartzell pode levar a resultados imprecisos e pobres dependência do contexto.

Lawrence et al. (1993), em *Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment*, oferecem uma solução alternativa, propondo uma estratégia baseada em amostragem de Gibbs para o alinhamento de múltiplas sequências. Este método é eficaz para detectar sinais sutis de sequência, permitindo que o algoritmo explore eficientemente o espaço de todos os possíveis alinhamentos, em busca daqueles que maximizam uma medida de similaridade. Este aspecto da abordagem de Lawrence et al. diferencia o seu trabalho de outros métodos de alinhamento de sequências da época, que geralmente confiavam em heurísticas menos rigorosas. Entretanto, um dos desafios associados à amostragem de Gibbs é a escolha dos parâmetros iniciais, que pode afetar consideravelmente os resultados finais. Além disso, o método pode ficar preso em ótimos locais, particularmente quando se lida com um grande número de sequências ou MOTIFS. Também é importante mencionar que, embora o método seja capaz de identificar e alinhar múltiplos MOTIFS, ele faz a suposição simplificadora de que cada sequência contém exatamente um exemplar de cada motif. Esta suposição pode não ser válida em muitos casos biológicos.

Em contraste, o artigo de Heinz et al. (2010), *Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and B cell identities*, oferece uma abordagem mais biologicamente orientada, examinando os fatores de transcrição determinantes de linhagem. Um dos pontos fortes desse estudo é a exploração detalhada da regulação gênica em macrófagos e células B, trazendo insights importantes para o entendimento da diferenciação celular e a expressão gênica nessas células. O estudo usa uma variedade de técnicas experimentais, incluindo a cromatina imunoprecipitação seguida por sequenciamento (CHIP-SEQ) para identificar regiões do DNA que interagem com fatores de transcrição específicos, fornecendo uma visão abrangente da regulamentação gênica nessas células. Embora a análise seja extensa, a interpretação dos dados pode ser complexa, dada a natureza multifatorial e interativa da regulação gênica. Além disso, enquanto o estudo traz insights importantes sobre a regulação gênica em macrófagos e células B, a generalização desses resultados para outros tipos de células pode não ser direta. Outro ponto negativo é que os autores não discutem em detalhes as técnicas utilizadas na extração dos MOTIFS, se concentrando mais nos aspectos biológicos do experimento.

No trabalho intitulado *Fitting a mixture model by expectation maximization to discover motifs in bipolymers*, Bailey et al. (1994) propõem uma técnica de otimização baseada na maximização da expectativa para a identificação de MOTIFS em biopolímeros. Este método, apesar de sua eficiência, carrega demandas computacionais expressivas. Destaca-se como uma inovação metodológica a utilização do algoritmo de EM para ajustar um modelo de mistura, habilidade que permite a detecção efetiva de fragmentos conservados em sequências de biopolímeros. A aplicabilidade desta técnica abrange diversos problemas de bioinformática, incluindo o reconhecimento de sítios de ligação de fatores de transcrição, sítios de splicing e MOTIFS proteicos. A clareza e o rigor na apresentação dos resultados, aliados à precisão metodológica, consolidam a robustez deste trabalho, favorecendo a replicação do método por outros pesquisadores. No

entanto, o método tem limitações importantes. A primeira delas é o comportamento do algoritmo em frente de conjuntos de dados de alta dimensionalidade. Essa ferramenta estatística pode encontrar obstáculos, devido à tendência de convergência para ótimos locais. Além disso, a sensibilidade do método à inicialização dos parâmetros do modelo de mistura pode resultar em variabilidade nos resultados, dependendo da escolha inicial.

Li et al. (2019), por meio do *ProSampler*, propõem metodologias de alto desempenho e grande versatilidade. Este algoritmo se destaca pela alta taxa de processamento e pela habilidade de trabalhar com grandes volumes de dados de CHIP-SEQ. Contudo, a evidência empírica que comprova essa superioridade de desempenho carece de detalhamento adequado. A eficiência do PROSAMPLER é, em grande parte, atribuída ao seu algoritmo de amostragem *Gibbs*, que é projetado para ser altamente otimizado. A amostragem *Gibbs* é uma técnica de Monte Carlo usada para obter uma sequência de amostras a partir de uma distribuição multivariada, particularmente quando a distribuição direta não é conhecida. Os autores afirmam que o PROSAMPLER é capaz de descobrir rapidamente os principais MOTIFS em um conjunto de dados de CHIP-SEQ com milhares de sequências. Entretanto, é importante ressaltar que as comparações de desempenho dependem fortemente das configurações do sistema e dos parâmetros escolhidos.

Os estudos apresentados revelam uma série de limitações que poderiam ser abordadas ao desenvolver um novo algoritmo para identificação de sítios de ligação de proteínas. De maneira geral, as limitações incluem a dependência da qualidade e quantidade dos dados de entrada, a pressuposição de independência entre os sítios de ligação, a complexidade e instabilidade do algoritmo de amostragem de *Gibbs*, a simplificação de que cada sequência contém apenas um exemplar de cada MOTIF, e as dificuldades inerentes à regulação gênica multifatorial e interativa. Adicionalmente, uma limitação importante é a tendência do algoritmo de Maximização da Expectativa em convergir para ótimos locais, que pode resultar em soluções subótimas, especialmente em situações com conjuntos de dados de alta dimensionalidade.

A dificuldade em lidar com sequências pouco conservadas e mais sutis é uma limitação adicional observada nos métodos discutidos. Isso ocorre porque os métodos atuais podem não ter sensibilidade suficiente para detectar MOTIFS ou sítios de ligação que não são altamente conservados ou que apresentam variações sutis. Esses MOTIFS pouco conservados podem ter um papel importante na regulação gênica e, portanto, é importante que um novo algoritmo seja capaz de identificar esses sinais com eficácia.

3.4 DESAFIOS

A pesquisa bibliográfica permitiu identificar diversos aspectos e desafios que ainda precisam ser superados. Como o problema é NP-completo, ainda está em aberto e é altamente improvável que um único algoritmo seja capaz de abordar toda a variedade de eventos possíveis. Essa complexidade se reflete na ampla quantidade de artigos publicados na literatura, bem como na diversidade de técnicas que podem ser empregadas.

Não há uma abordagem universal e, provavelmente, nunca haverá uma que trate todas as nuances do problema. Por exemplo, as abordagens probabilísticas são mais adequadas para modelar fragmentos degenerados e extensos, enquanto os algoritmos enumerativos são mais apropriados para identificar fragmentos menores e altamente conservados. Portanto, a seleção do método ideal depende das características dos fragmentos analisados e dos objetivos específicos da pesquisa em questão, o que torna o desenvolvimento de soluções eficazes um desafio contínuo neste campo.

A partir da análise da variabilidade, surge um desafio complexo denominado problema (κ, D) -MOTIF, no qual lidamos com fragmentos de tamanho k , que apresentam até d mutações

distribuídas uniformemente ao longo de sua extensão. Resolver esse problema requer abordagens sofisticadas que consideram a diversidade e a complexidade das sequências em estudo. O problema se torna cada vez mais difícil conforme a largura k diminui e o grau de degeneração d aumenta. Em muitos casos, os fatores de transcrição aparecem como dímeros, i.e., uma molécula de proteína composta por sub-unidades similares chamadas de monômeros. Neste caso, as mutações ocorrem de forma comportada no centro da proteína. Isso se traduz em sítios de ligação que possuem sequências conservadas nas bordas com mutações ocorrendo no centro. Fragmentos com essas características, como o homodímero *operon lac* do organismo *E. coli* são mais simples de serem modelados e consequentemente encontrados.

Por outro lado, muitos fatores de transcrição apresentam uma distribuição de mutações mais uniforme ao longo de suas sequências proteicas. Este último caso é um exemplo típico de (K,D) -MOTIF, onde d mutações estão distribuídas ao longo de uma sequência de tamanho k . Isso representa um desafio na análise de sequências por duas razões principais. Primeiro, a Divergência de Kullback-Leibler (DKL) dos fragmentos diméricos será maior do que a dos fragmentos não-diméricos, o que dificulta a sua identificação por abordagens que se baseiam exclusivamente na DKL como função objetivo.

Um outro problema mais sutil surge da distância média de hamming. Em alinhamentos diméricos, ela tende a ser menor do que em alinhamentos não-diméricos. Na prática, a distância média de hamming é utilizada para estimar a probabilidade de um fragmento arbitrário ser encontrado ao acaso. Por exemplo, considere dois alinhamentos distintos A_1 e A_2 , com sequências de largura $k = 6$ nucleotídeos. Considere ainda que cada sequência possui $d = 2$ mutações e que a distância média de hamming seja respectivamente $DHA_1 = 1,8$ e $DHA_2 = 3,8$. Podemos concluir que, mesmo possuindo o mesmo número de mutações por sequência, o alinhamento A_1 é mais conservado, pois possui uma distância média de hamming menor (e consequentemente um IC maior).

Podemos empregar a distribuição binomial para calcular a probabilidade de cada alinhamento. Realizando os cálculos com $k = 6$, $p = \frac{3}{4}$ e arredondando $DHA = 2$, $DHB = 4$, temos que $p(A_1) = 0,038$ e $p(A_2) = 0,466$. Em termos práticos, para cada 100 fragmentos arbitrários de tamanho $k = 6$, temos a esperança de que 3,8 deles pertençam ao alinhamento A_1 e 46,6 pertençam ao alinhamento A_2 . Isso torna o A_2 muito semelhante à distribuição de fundo tornando-o mais difícil de ser detectado. [Buhler e Tompa \(2002\)](#) postularam que a maior parte das falhas ocorre quando a razão entre d e k ultrapassa 25%, o que pode ser visto em muitos fragmentos de organismos eucariotos.

Além da variabilidade, as chamadas técnicas de sequenciamento de nova geração foram desenvolvidas, levando a um aumento exponencial na quantidade de dados disponíveis. Anteriormente, a análise de sequências era realizada com apenas algumas dezenas de amostras, resultando em uma demanda computacional relativamente baixa. No entanto, com o surgimento dessas técnicas, a complexidade e o custo do sequenciamento de genomas inteiros diminuíram significativamente. Isso levou a um crescimento considerável na quantidade de dados disponíveis, exigindo que as abordagens computacionais sejam constantemente aperfeiçoadas para acompanhar esse avanço. Dois problemas surgem na análise de grande volumes de dados. O primeiro tem relação com o tempo e o segundo com o espaço. É bastante óbvio que se a quantidade de dados aumenta, o tempo e o espaço necessário para lidar com eles também deve aumentar.

Frequentemente, algoritmos de otimização utilizam funções objetivo ou algum tipo de teste de hipótese para validar uma descoberta. Em muitos problemas sintéticos ou do mundo real, a solução com o maior score tende a ser a mais adequada. No entanto, isso nem sempre é verdade, em especial na análise e descoberta de MOTIFS. Devido à presença de fragmentos espúrios, sequências de baixa complexidade sem função celular definida podem apresentar um

escore mais alto do que as sequências de interesse. Esses fragmentos, portanto, representam um desafio na busca bem-sucedida de MOTIFS verdadeiramente relevantes.

A busca pelo ótimo global em um espaço de alta dimensionalidade é um desafio recorrente em muitos algoritmos de otimização, em particular no algoritmo EM, frequentemente empregado na descoberta de MOTIFS. A função de log-verossimilhança que o EM tenta maximizar infelizmente é não-convexa, complexa e repleta de ótimos locais. Portanto, é bastante comum que este método, ao operar a partir de um ponto inicial aleatório, se encontre preso em um desses ótimos locais, limitando a eficácia geral do algoritmo. Este obstáculo é particularmente expressivo em contextos como a descoberta de MOTIFS, onde a dimensão do espaço de busca é ampla.

Uma das estratégias mais empregadas para mitigar este problema é a inicialização múltipla a partir de diferentes pontos, na esperança de que pelo menos um deles esteja em uma região próxima ao ótimo global. A estratégia de múltiplas inicializações, apesar de simples e intuitiva, pode ser ineficiente em diversas circunstâncias. As principais razões são⁸:

- Alto custo computacional: cada inicialização demanda a execução completa do algoritmo EM, que pode ser computacionalmente caro, especialmente quando o espaço de busca é grande. Esta estratégia pode, portanto, resultar em um uso desproporcional de recursos computacionais.
- Ineficiência em espaços de busca de alta dimensão: em um espaço de busca de alta dimensionalidade, a probabilidade de uma inicialização aleatória estar próxima do ótimo global é extremamente pequena. Portanto, são necessárias muitas inicializações para garantir uma cobertura adequada do espaço de busca, o que pode ser inviável em termos de tempo e recursos.
- Falta de garantia de convergência: mesmo com um número significativo de inicializações, não há garantias de que o algoritmo EM convergirá para o ótimo global. Ainda é possível que todas as inicializações conduzam a ótimos locais.

Essas limitações apontam para a necessidade de métodos mais avançados de inicialização, como por exemplo o fornecimento de uma ou mais “sementes” iniciais localizadas em regiões promissoras do espaço de busca. Isso poderia afetar positivamente a performance dos algoritmos de otimização, aumentando as chances de fuga de ótimos locais. Entretanto, a busca por sementes adequadas é um desafio por si só, pois exige uma compreensão aprofundada da estrutura do problema de otimização. A superação deste obstáculo, contudo, tem o potencial de transformar profundamente a eficiência do algoritmo EM permitindo que melhores soluções sejam encontradas.

3.5 CONCLUSÃO

Neste capítulo, exploramos a vasta gama de abordagens e métodos existentes no campo da descoberta de MOTIFS biológicos. Ao analisar a literatura, procuramos compreender as diferentes estratégias e técnicas utilizadas pelos pesquisadores, bem como identificar as principais tendências e desenvolvimentos ao longo do tempo. Examinamos em detalhes várias abordagens, destacando suas características distintas, vantagens e limitações. Essa análise nos permitiu adquirir uma compreensão abrangente das metodologias existentes e forneceu insights importantes que podem orientar nossas próprias investigações. No próximo capítulo, intitulado “Material

⁸É interessante destacar que essas limitações não se aplicam somente ao algoritmo EM, mas sim a qualquer algoritmo de otimização exata.

e Métodos”, delinearemos as metodologias empregadas, bem como os materiais e recursos utilizados em nossas análises, com o objetivo de abordar o problema da descoberta de MOTIFS biológicos de maneira sistemática e rigorosa.

4 MATERIAL E MÉTODOS

Neste capítulo, apresentamos o BIOMAPP::CHIP (BIOLOGICAL MOTIF APPLICATION FOR CHIP-SEQ DATA), um novo arcabouço desenvolvido para identificar MOTIFS em conjuntos de dados de DNA obtidos por meio do protocolo CHIP-SEQ. O BIOMAPP::CHIP é uma estrutura híbrida composta por três fases principais: pré-processamento, inicialização e otimização. Cada uma dessas fases engloba várias sub-fases, trabalhando em conjunto para aprimorar as métricas de desempenho. Na fase de pré-processamento, as regiões de pico obtidas a partir dos dados brutos, são normalizadas e preparadas para a análise subsequente. A fase de inicialização emprega técnicas de agrupamento e contagem para separar as sequências em grupos, com base em suas características e similaridades. Finalmente, a etapa de otimização emprega algoritmos sofisticados com o objetivo de aprimorar e aumentar a precisão das estimativas relacionadas à correta identificação dos MOTIFS.

4.1 ESTRATÉGIAS PARA SUPERAR OS DESAFIOS IDENTIFICADOS NO CAPÍTULO ANTERIOR

Nesta seção, faremos referência aos desafios que foram delineados durante a revisão bibliográfica, apresentando como o BIOMAPP::CHIP se propõe a solucionar essas questões. Por meio de uma abordagem metódica, cada desafio será enfrentado de forma sistemática, com foco na aplicação das estratégias e técnicas específicas. Inspirados nos avanços documentados na literatura, propomos soluções robustas, que visam contribuir positivamente aos paradigmas existentes.

4.1.1 MOTIFS sutis e o problema (k, d) -MOTIF

O problema (k, d) -MOTIF é uma questão bem estabelecida na bioinformática, particularmente na descoberta de MOTIFS em sequências de DNA. Ele consiste em identificar padrões conservados de tamanho k que aparecem frequentemente em sequências de DNA, com a particularidade de que estes podem apresentar no máximo d discordâncias ou mutações. Em outras palavras, o desafio é encontrar sequências de k nucleotídeos que se repetem ao longo de múltiplas sequências de DNA, permitindo até d diferenças entre elas. A identificação de (k, d) -MOTIFS é uma tarefa complexa, porém essencial para a compreensão dos mecanismos de regulação genética, uma vez que estes fragmentos podem representar, por exemplo, sítios de ligação para proteínas que influenciam a expressão gênica.

Este problema é mais do que um desafio teórico, pois encontra ressonância em diversas situações biológicas naturais. As sequências de DNA, RNA e proteínas frequentemente apresentam MOTIFS recorrentes com variações sutis - caracterizados precisamente como (k, d) -MOTIFS. Esses fragmentos, apesar das mutações, desempenham funções biológicas importantes, como a ligação de proteínas ao DNA ou RNA, ou a formação de estruturas tridimensionais em proteínas. Sendo assim, a identificação eficiente e precisa dessas regiões é um passo essencial na compreensão dos mecanismos subjacentes à função biológica e na previsão de interações moleculares. Logo, a resolução do problema (k, d) -MOTIF pode oferecer contribuições importantes para o campo da genômica, proteômica e, mais amplamente, para a biologia molecular.

O problema (k, d) -MOTIF representa um desafio computacionalmente complexo, particularmente quando o número de mutações permissíveis d é maior que 25% do comprimento

k ou em outras palavras, o nível de conservação é inferior a 75%. Essa complexidade surge devido ao vasto espaço de pesquisa que precisa ser explorado para identificar todos os possíveis fragmentos de comprimento k que diferem em até d posições de um MOTIF de referência. Além disso, a presença de um alto número de mutações pode obscurecer os verdadeiros fragmentos biológicos, tornando a tarefa de identificação ainda mais desafiadora. A necessidade de navegar por esse enorme espaço de soluções e a dificuldade em distinguir ruído de sinais biologicamente relevantes fazem do problema (k,d) -MOTIF um desafio complexo, mas de vital importância na análise de sequências genômicas.

Com o objetivo de abordar este desafio, a estratégia escolhida envolve o uso da SMT (SPARSE MOTIF TREE) (Garbelini et al., 2024) (sub-seção 4.2.1.1), uma estrutura de dados que combina a eficiência das árvores de sufixo com a compactação das matrizes esparsas para contar KMERS de maneira otimizada. A SMT, foi desenvolvida especificamente para a contabilização eficiente de KMERS e apresenta uma vantagem distinta na manipulação de grandes volumes de dados genômicos.

Para complementar a estratégia, o algoritmo KDIVE¹ (sub-seção 4.2.1.1) é empregado. Ele consiste em um método de busca em profundidade que permite encontrar fragmentos com até d mutações dentro da SMT. A importância deste algoritmo reside na sua eficiência computacional, uma vez que realiza a busca em tempo linear, tornando a tarefa de identificar (k,d) -MOTIFS mais factível. Este enfoque integrado une as capacidades de contagem eficiente de KMERS da SMT com a rapidez e precisão do algoritmo KDIVE, proporcionando uma solução robusta para o problema (k,d) -MOTIF.

4.1.2 Análise de dados derivados do protocolo CHIP-SEQ

Este desafio também é abordado por nosso algoritmo através da SMT. A principal vantagem de nossa abordagem se deve ao fato de que o tempo de construção da SMT é linear em relação ao tamanho de k . Além disso, a complexidade espacial da SMT é otimizada por meio do uso de uma matriz esparsa de alto rendimento. Este design inovador permite o gerenciamento efetivo de uma grande quantidade de dados gerados pelo CHIP-SEQ, e pode superar as limitações das técnicas convencionais de bioinformática ao lidar com conjuntos de dados de grande escala.

4.1.3 Soluções sub-ótimas

Embora amplamente utilizado, o algoritmo EM possui uma característica limitante que deteriora a sua eficácia: ele garante apenas a localização de um ótimo local, ou seja, uma solução que é ideal apenas dentro de uma determinada região do espaço, mas não necessariamente a melhor solução global. Isso significa que a solução encontrada pode ser sub-ótima quando consideramos o espaço de solução total. Portanto, é essencial o desenvolvimento de estratégias que previnam que o algoritmo fique aprisionado em tais regiões, facilitando a exploração do espaço de busca de maneira mais ampla e, idealmente, a identificação da solução global ótima.

Nós abordamos este desafio através do uso conjunto da SMT, do algoritmo KDIVE e do FAST-EM (sub-seção 2.3.2.4), uma versão melhorada do EM canônico capaz de lidar melhor com grandes volumes de dados. A SMT é empregada para a extração dos KMERS enriquecidos de todo o conjunto de dados. Uma vez obtidos esses KMERS, o KDIVE é empregado para buscar todos os KMERS “irmãos”, que estão a uma distância de d mutações a partir dos fragmentos enriquecidos. Esta combinação proporciona uma seleção criteriosa de modelos estatísticos pré-otimizados, que são posteriormente usados para alimentar o FAST-EM. Dessa forma, nosso método proporciona

¹O nome KDIVE foi criado para dar a impressão de mergulho em busca de fragmentos de tamanho k .

um ponto de partida melhorado, permitindo a obtenção de soluções mais robustas, minimizando o risco do algoritmo ficar preso em ótimos locais.

4.2 BIOMAPP-CHIP

Além dos algoritmos já mencionados nas seções introdutórias, nosso arcabouço incorpora um conjunto adicional de algoritmos essenciais que operam em um papel de suporte, tais como K-SEARCH, IUPAC-SEARCH, KHMAP e FAST-EM. Nas seções seguintes, todos os algoritmos serão objeto de uma análise mais aprofundada, com explicações detalhadas sobre suas funcionalidades e respectivas contribuições para a estrutura geral do BIOMAPP::CHIP.

O BIOMAPP::CHIP é um arcabouço híbrido, implementado em C++ e R, que reúne diferentes técnicas analíticas e numéricas, otimizadas para o tratamento aprofundado dos dados, em especial dados provenientes de experimentos CHIP-SEQ. O objetivo desta abordagem é construir respostas efetivas que auxiliem pesquisadores e especialistas da área de biologia molecular, mais especificamente no estudos de sequências conservadas, facilitando investigações complexas que envolvem a análise de MOTIFS de sequência. Por meio da combinação adequada de métodos, alguns já estabelecidos² e outros criados especificamente para nosso arcabouço, o BIOMAPP::CHIP oferece uma ferramenta robusta e adaptável para o cenário da genômica funcional. A Figura 4.1 ilustra todos os módulos do arcabouço e suas respectivas interações.

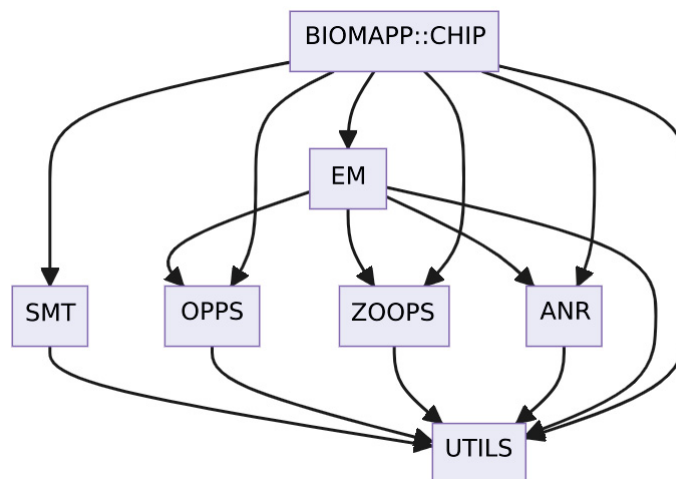


Figura 4.1: O BIOMAPP::CHIP é estruturado em diversos módulos que trabalham de forma interconectada. Cada módulo é responsável por uma parte específica da funcionalidade geral do pacote. O módulo UTILS contém uma coleção de funções utilitárias que auxiliam diversos outros módulos. O módulo SMT se destina ao manejo da SPARSE MOTIF TREE, incluindo sua criação e manipulação. Os módulos OPPS, ZOOPS e ANR abrigam a implementação do algoritmo EM adaptado para cada uma das respectivas distribuições de MOTIFS. Finalmente, o módulo BIOMAPP::CHIP atua como o núcleo do arcabouço, integrando todos os módulos anteriores e implementando a estrutura final do pacote. Fonte: o autor.

Podemos observar nesta figura que o arcabouço é composto por sete módulos principais: UTILS, SMT, OPPS, ZOOPS, ANR, EM e BIOMAPP::CHIP. O módulo UTILS é a base do algoritmo, fornecendo funcionalidades que são utilizadas por todos os outros módulos. O módulo SMT é construído sobre o módulo UTILS, assim como os módulos OPPS, ZOOPS, ANR e EM. O módulo EM, além de depender do módulo UTILS, também depende dos módulos OPPS, ZOOPS e ANR. Por fim, o módulo BIOMAPP::CHIP é o mais complexo, pois depende de

²Os algoritmos empregados em nosso arcabouço que não foram totalmente implementados são o REPEAT-MASKER e DUST, ambos operam na fase de pré-processamento.

todos os outros módulos. Ele utiliza as funcionalidades fornecidas pelos módulos UTILS, SMT, OOPS, ZOOPS, ANR e EM para realizar suas operações. Todos os módulos possuem suporte a paralelização através das bibliotecas OPENMP: <https://www.openmp.org/> e THREAD BUILDING BLOCKS: <https://www.intel.com/content/www/us/en/developer/tools/oneapi/onetbb.html>. O BIOMAPP::CHIP pode ser encontrado em: <https://github.com/jadermcg/biomapp-chip>.

Observando a Figura 4.1, é possível notar que neste trabalho optamos por não empregar técnicas de aprendizado profundo. Embora o aprendizado profundo tenha demonstrado bom desempenho em muitos domínios da aprendizagem de máquina, esta abordagem também apresenta várias limitações que a torna menos atrativa para o nosso contexto específico. Por exemplo, os modelos de aprendizado profundo são frequentemente criticados como CAIXAS-PRETAGS devido à sua falta de interpretabilidade. Apesar desses modelos produzirem previsões relativamente precisas, é geralmente difícil entender exatamente como ou porquê chegaram a essas conclusões. Esta falta de transparência pode ser problemática, especialmente em contextos científicos, onde é importante fazer previsões corretas e também entender os mecanismos subjacentes que levam a essas previsões.

Além disso, ainda que as técnicas de aprendizado profundo possam, em teoria, aprender representações de alto nível dos dados, na prática, o desempenho desses modelos depende fortemente do ajuste cuidadoso dos hiperparâmetros. A necessidade de uma extensa pesquisa de hiperparâmetros aumenta a complexidade computacional do aprendizado profundo e também pode levar a *overfitting* os dados de treinamento.

Por fim, e talvez mais importante, nosso trabalho se concentra no aprendizado não supervisionado, uma área em que as técnicas de aprendizado profundo ainda não mostraram o mesmo nível de sucesso que nas tarefas de aprendizado supervisionado. No aprendizado não supervisionado, o objetivo é aprender a estrutura subjacente dos dados sem a ajuda de rótulos de classe ou outros tipos de FEEDBACK supervisionado. Embora existam técnicas de aprendizado profundo projetadas para este propósito, como as AUTOENCODERS e as redes ADVERSARIAIS GENERATIVAS, elas ainda estão longe de ser uma solução universal para todos os problemas de aprendizado não supervisionado.

Dado o exposto, acreditamos que uma abordagem específica, projetada para aproveitar ao máximo o conhecimento de domínio sobre o problema, pode fornecer resultados superiores em comparação com uma abordagem genérica baseada em aprendizado profundo. Isso é especialmente verdadeiro em nosso caso, onde a natureza não supervisionada do problema impõe desafios adicionais que as técnicas de aprendizado profundo ainda estão aprendendo a superar.

Antes de nos aprofundarmos na descrição do método, é importante compreender como os dados são adquiridos. Eles podem ser obtidos a partir de várias fontes públicas e privadas. Para obter mais informações sobre bancos de dados de CHIP-SEQ, consulte o Capítulo 2, Seção 2.4. O processo de obtenção de dados pode ser resumido nas seguintes etapas:

1. Realização do experimento: as proteínas de interesse são imunoprecipitadas juntamente com os fragmentos de DNA associados.
2. Processamento dos dados brutos: os dados brutos do sequenciador são processados para obter sequências curtas de DNA, chamadas de READS.
3. Identificação de picos: programas específicos, como MACS (Zhang et al., 2008) ou SICER (Zang et al., 2009), são usados para identificar regiões de enriquecimento de READS, chamadas de picos.

4. Pré-processamento: remoção de picos em regiões não específicas, filtragem de picos com base em critérios de qualidade, normalização dos dados e remoção de espúrios.
5. Análise: realização da análise de interesse, que podem ser: extração de MOTIFS, anotação funcional, análise de redes regulatórias, análise de conservação, dentre outros.

As etapas 1 e 2 são parte do experimento biológico e geralmente são conduzidas por um biólogo. A etapa 3, denominada análise de enriquecimento, tem como objetivo identificar as regiões de pico, que vão de 100 a 300 pb³, no qual existe uma alta probabilidade de que os fragmentos de interesse estejam presentes. A etapa 4 é crítica, pois envolve a remoção de sequências repetitivas e de baixa complexidade. A última etapa consiste na análise dos dados pré-processados.

O BIOMAPP::CHIP atua a partir do final da etapa 3, no qual recebe como entrada as regiões de pico devidamente extraídas do genoma. Na etapa 4, algoritmos especializados são utilizados para a realização do pré-processamento e a etapa 5 é implementada integralmente. O BIOMAPP::CHIP é composto por três fases principais: pré-processamento, inicialização e otimização.

A fase de pré-processamento envolve a adaptação do conjunto de dados para a execução dos experimentos, como a padronização do tamanho das sequências e a conversão de todos os caracteres para maiúsculas. Entretanto, o principal objetivo desta etapa é identificar sequências de baixa complexidade, elementos transponíveis, como ALUS (*Arthrobacter luteus*) e SINES (*Short Interspersed Nuclear Elements*), inserções de *E. coli*, ou quaisquer outros componentes que possam levar a conclusões imprecisas nas etapas subsequentes. Isso é realizado com o auxílio de algoritmos especializados, como DUST (Tatusov e Lipman, 1996) e REPEAT MASKER (Smit et al., 1996). Essa fase envolve a eliminação de sequências redundantes e a correção de erros, assegurando a qualidade e a integridade dos dados.

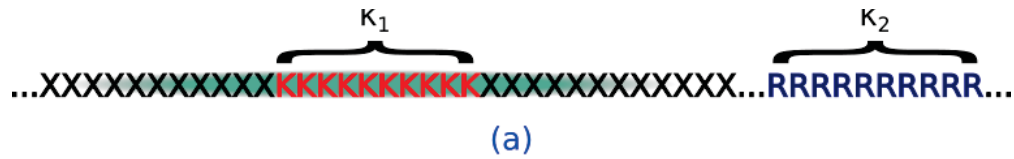
A fase de inicialização recebe como entrada regiões de pico previamente processadas, nos quais se presume que haja fragmentos da distribuição de interesse envoltos por elementos da distribuição de fundo. O propósito do algoritmo de inicialização consiste na identificação, contagem e agrupamento eficiente dos KMERS enriquecidos bem como na realização de testes de hipótese para determinar sua significância estatística.

O cerne desta etapa consiste em 2 componentes especialmente desenvolvidos para este fim: i) SMT: estrutura de dados escalável, baseada em árvores de sufixo e responsável por armazenar todos os KMERS do conjunto de dados principal; ii) KDIVE: algoritmo de busca eficiente inspirado na busca em profundidade limitada e no algoritmo BRANCH AND BOUND apresentado no Capítulo 2.

Um problema frequentemente que acompanha a análise de KMERS é o “problema do entorno”. Para entender melhor, suponha que o KMER κ seja o mais abundante em um conjunto de sequências arbitrário, aparecendo ν vezes. Ocorre que o entorno de κ também será enriquecido, levando a erros de análise. Por exemplo, considere $\kappa = CAGCTG$. Se κ aparece ν , então, considerando que a distribuição de controle é uniforme, o entorno de κ com deslocamento de h nucleotídeos aparecerá $\frac{\nu}{4^h}$. De forma geral, temos que se um KMER κ aparecer ν vezes, então seu entorno $k + h$ ou $k - h$ com deslocamento de h nucleotídeos aparecerá $\nu \frac{1}{4^h}$. Como exemplo, considere $\bar{\kappa} = CCAGCT$ ($h = 1$). Este fragmento aparecerá $\frac{\nu}{4^1}$. Se $\nu = 1000$, então $|\bar{\kappa}| = 250$. O problema se torna ainda mais sutil e difícil de eliminar se a distribuição de fundo não for uniformemente distribuída. A Figura 4.2 mostra o problema do entorno graficamente.

Na Figura 4.2(a), são destacados dois fragmentos, κ_1 e κ_2 . Observa-se que a área de influência de κ_1 , representada em verde, diminui gradativamente à medida que a distância em

³A resolução das READS depende fortemente da tecnologia de sequenciamento empregada.



Kmer	Freq

KKKKKKKKKK	= 1000
XKKKKKKKKK	= 250
KKKKKKKKKX	= 250
XXKKKKKKKK	= 125
KKKKKKKKXX	= 125
RRRRRRRRRR	= 100

(b)

Figura 4.2: Problema de entorno. (a) Representação de uma sequência com dois kmers, κ_1 e κ_2 . A área em verde representa a área de influência de κ_1 . (b) Tabela de frequência de κ_1 , κ_2 e dos fragmentos do entorno imediato de κ_1 , com $h = 2$. É possível verificar que os fragmentos que representam o entorno de κ_1 aparecem mais vezes do que o fragmento κ_2 . Fonte: o autor.

relação ao fragmento aumenta. No entanto, esta influência continua intensa na proximidade imediata. Esta observação é corroborada pela Figura 4.2(b), onde uma tabela de frequências foi elaborada. Nesta tabela, é possível constatar que a frequência do fragmento κ_1 é superior à frequência do fragmento κ_2 , mesmo considerando até dois deslocamentos a partir de κ_1 . Este problema pode ser mitigado, mas não totalmente resolvido através da geração de vários modelos iniciais.

Na fase de otimização, sementes pré-otimizadas obtidas a partir da SMT são utilizadas como ponto de partida para a execução do algoritmo EM. O algoritmo EM é responsável por refinar essas sementes, ajustando seus parâmetros de modo a maximizar a probabilidade das observações dadas. Como resultado da execução dessa fase, cada semente é transformada em um modelo PWM, que representa uma descrição mais precisa e ajustada do MOTIF em questão. Este modelo PWM pode então ser empregado em novos dados para encontrar padrões que ainda não foram rotulados.

Para determinar qual variante do algoritmo EM deve ser empregada, é essencial entender as especificidades do processo de geração de sequências mediado pelo protocolo CHIP-SEQ. Compreender essas nuances permite uma seleção mais apropriada, garantindo assim uma análise mais precisa e eficaz. O protocolo CHIP-SEQ é baseado na imunoprecipitação de proteínas de ligação ao DNA, e, portanto, é esperado que muitas das sequências geradas no experimento contenham uma cópia do MOTIF de interesse. No entanto, é importante notar que os dados de CHIP-SEQ também podem conter ruído e sinais não específicos, e nem todas as sequências conseguem capturar a região conservada.

Além disso, os dados de CHIP-SEQ são geralmente pré-processados antes da análise de MOTIFS, e apenas os picos de enriquecimento de maior qualidade são selecionados para a busca. Mesmo assim, é possível que nem todos os picos contenham o fragmento de interesse, devido a fatores como ruído, efeitos indiretos de ligação à proteína ou interações cooperativas entre proteínas de ligação ao DNA.

Com base nesse entendimento, o modelo ZOOPS se torna uma escolha natural para a análise de MOTIFS em dados de CHIP-SEQ, pois permite a presença do fragmento em algumas, mas

não todas, as sequências. Por essa razão, este modelo é geralmente a opção mais apropriada, levando em consideração que nem todas as sequências podem conter o MOTIF de interesse. No entanto, dependendo do contexto biológico e das expectativas em relação à presença e frequência dos MOTIFS nas sequências, os modelos ANR ou mesmo OOPS também devem ser considerados. Em particular, o modelo OOPS pode ser útil em experimentos realizados com maior rigor, onde a quantidade de ruído é menor ou em situações mais simples, nas quais o MOTIF de interesse se liga a apenas um local de ligação.

Por fim, uma versão otimizada⁴ do algoritmo EM, chamada de FAST-EM é executada, obtendo como retorno um conjunto de modelos PWM otimizados. A partir destes modelos, podemos extrair a sequência CONSENSO otimizada com a qual podemos calcular o escore de todos os KMERS do dataset e consequentemente computar os P-VALORES destes fragmentos. Os P-VALORES são empregados para classificar se um determinado fragmento pertence ou não à distribuição de interesse.

É importante lembrar que a escolha do modelo é apenas uma parte da análise de MOTIFS. Outros fatores, como o pré-processamento adequado dos dados, a seleção de picos de alta qualidade e a validação experimental dos resultados, são essenciais para obter uma compreensão abrangente e precisa das interações proteína-DNA no experimento. Segue abaixo um resumo de todas as fases e subfases do BIOMAPP::CHIP.

1. Pré-processamento:

- (a) Obter as regiões de pico.
- (b) Limpar e normalizar os dados.

2. Contagem:

- (a) Carregar o dataset principal pré-processado.
- (b) Carregar as sequências de controle (se disponíveis) ou gerá-las a partir do embaralhamento do dataset principal.
- (c) Gerar o modelo probabilístico de fundo a partir dos dados de controle.
- (d) Estimar o valor de k ou usar um valor pré-definido.
- (e) Construir a SMT a partir das regiões de pico.
- (f) Extrair os KMERS enriquecidos a partir da SMT.
- (g) Gerar os modelos iniciais a partir dos dados enriquecidos.

3. Otimização:

- (a) Otimizar os sementes obtidas a partir da SMT com o EM e construir os modelos finais.

O BIOMAPP::CHIP começa com o pré-processamento, onde as regiões de pico são identificadas e os dados são limpos e normalizados. Em seguida, na fase de inicialização, o dataset principal pré-processado é carregado, juntamente com as sequências de controle, caso estejam disponíveis. Caso contrário, as sequências de controle são geradas a partir do embaralhamento do dataset principal.

O modelo probabilístico de fundo é gerado a partir dos dados de controle, e o valor de k é estimado ou um valor pré-definido é usado. A SMT é construída a partir das regiões de pico, e

⁴Que permitiu uma maior agilidade no processamento.

os **KMERS** enriquecidos são extraídos da **SMT** para gerar os modelos iniciais. Finalmente, na fase de otimização, as sementes são otimizadas por meio do algoritmo **EM**. A Figura 4.3 apresenta de maneira gráfica o pipeline do **BIOMAPP::CHIP**, ilustrando as etapas de pré-processamento, inicialização e otimização ao longo do tempo, bem como as sub-etapas associadas a cada uma dessas fases principais.

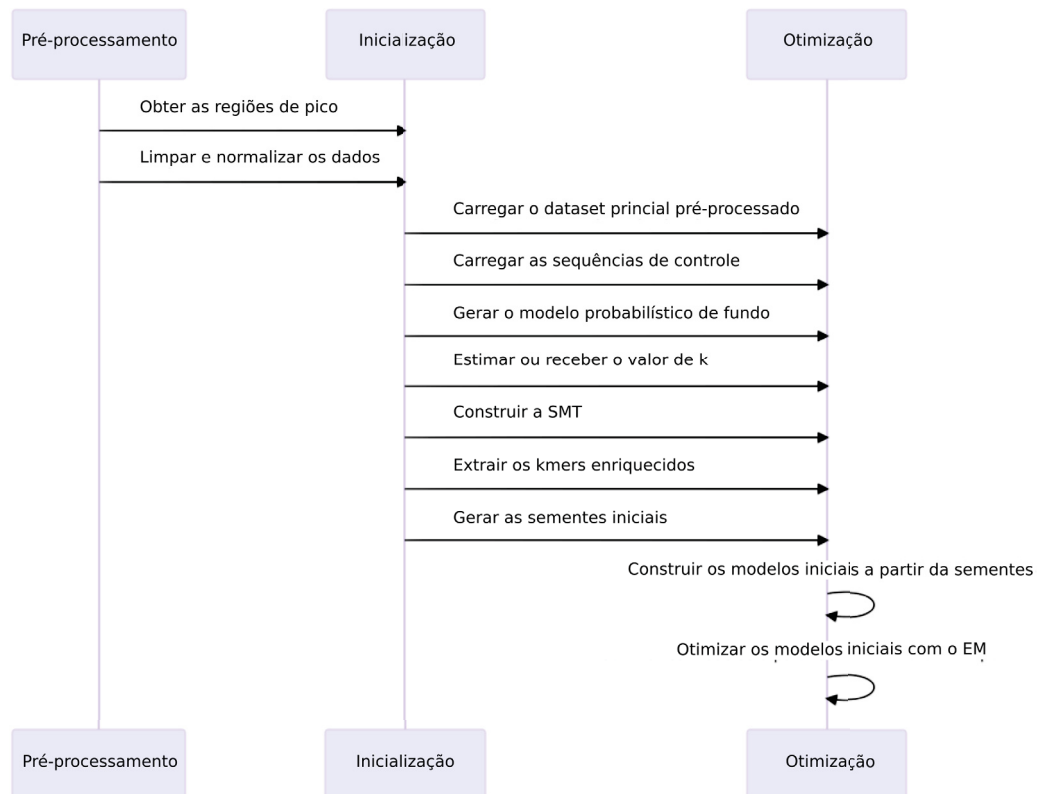


Figura 4.3: Pipeline geral do **BIOMAPP::CHIP**. O processo começa com o pré-processamento, onde as regiões de pico são obtidas e os dados são limpos e normalizados. Em seguida, a fase de inicialização carrega o dataset principal pré-processado e as sequências de controle. Um modelo probabilístico de fundo é gerado a partir dos dados de controle, e o valor de k é estimado. A **SMT** é construída a partir das regiões de pico, e os **K-MERS** enriquecidos são extraídos da **SMT**. Modelos iniciais são então gerados a partir dos dados enriquecidos. Finalmente, na fase de otimização, o **EM** é empregado para otimizar os modelos iniciais. Fonte: o autor.

4.2.1 Inicialização

O algoritmo de inicialização segue um conjunto de passos para identificar **KMERS** enriquecidos no dataset principal e gerar modelos iniciais a partir dessas informações. Primeiramente, o algoritmo carrega os dados pré-processados, que contêm as sequências nucleotídicas de interesse. Em seguida, caso estejam disponíveis, as sequências de controle são carregadas. Caso contrário, são geradas a partir do embaralhamento do conjunto de dados principal utilizando o método de **MARKOV** (Fitch, 1983) ou o método de **EULER** (Altschul e Erickson, 1985).

O método de **MARKOV** preserva a composição de mononucleotídeos e dinucleotídeos (ou outras propriedades de ordem superior, dependendo da ordem do modelo de **MARKOV** utilizado) das sequências originais. Isso significa que as sequências embaralhadas terão uma distribuição similar de pares de nucleotídeos adjacentes, mas a ordem global das sequências será alterada.

O método de **EULER** utiliza o circuito euleriano para embaralhar as sequências. Um circuito euleriano é um caminho em um grafo que visita cada aresta exatamente uma vez e retorna

ao ponto de partida. O método de EULER transforma a sequência original em um grafo de DE BRUIJN, no qual os vértices representam subsequências de tamanho k e as arestas representam a transição entre KMERS consecutivos. Um circuito euleriano é então construído a partir desse grafo, e as sequências embaralhadas são geradas a partir desse circuito. O método de EULER preserva a composição de KMERS das sequências originais, mas a ordem global das sequências é alterada. Este método é particularmente útil quando se deseja preservar as propriedades de KMERS nas sequências embaralhadas.

As sequências de controle são essenciais para estabelecer uma base comparativa e avaliar o enriquecimento dos KMERS. As vantagens em se trabalhar com sequências de controle específicas são: (i) contexto biológico: sequências de controle específicas são derivadas do genoma de interesse, preservando a composição de nucleotídeos e a estrutura genômica, o que permite uma comparação mais realista entre sequências de controle e dados experimentais; (ii) redução de falsos positivos: ao utilizar sequências de controle específicas, é possível identificar e descartar MOTIFS que ocorrem devido a características genômicas comuns, em vez de interações proteína-DNA específicas.

Entretanto, trabalhar com sequência de controle extraídas diretamente do genoma alvo pode apresentar alguns desafios. São eles: (i) complexidade adicional: a obtenção de sequências de controle específicas pode exigir experimentos adicionais, como CHIP-SEQ com anticorpos de controle ou a seleção de sequências genômicas que não sejam alvos do fator de transcrição de interesse; (ii) viés de seleção: a escolha das sequências de controle específicas pode introduzir viés na análise, dependendo do método utilizado para selecioná-las.

Sempre que possível, é mais recomendado trabalhar com sequências de controle extraídas diretamente do genoma do organismo alvo. Porém, em muitos casos isso não é possível, devido a vários fatores, tais como custo, dificuldade para extrair sequências adequadas ou viés de seleção. Neste caso, pode-se usar uma versão embaralhada do conjunto de dados principal. As vantagens disso são: (i) simplicidade: embaralhar o conjunto de dados principal é um processo computacionalmente simples e não requer experimentos adicionais para gerar sequências de controle; (ii) preservação da composição de nucleotídeos: embaralhar as sequências preserva a composição geral de nucleotídeos, permitindo que as sequências de controle tenham uma distribuição de nucleotídeos semelhante à das sequências experimentais.

Porém, este método incorpora alguns problemas adicionais que podem levar a resultados impressos, tais como: (i) perda de contexto genômico: embaralhar as sequências pode eliminar características genômicas importantes, como estrutura de cromatina, ilhas CPG e outros elementos, que podem ser essenciais para a interpretação dos resultados e a compreensão das interações proteína-DNA; (ii) falsos positivos: como as sequências embaralhadas podem não refletir adequadamente o contexto genômico, é possível que os MOTIFS encontrados sejam artefatos ou apresentem significância estatística reduzida em comparação com as sequências de controle específicas.

Ao escolher entre sequências de controle específicas ou embaralhar o conjunto de dados principal, é importante considerar o contexto biológico e os recursos disponíveis. Sequências de controle específicas geralmente fornecem uma análise mais robusta e precisa das interações proteína-DNA, mas podem exigir esforço e recursos adicionais para obtê-las. Embaralhar o conjunto de dados principal é uma abordagem mais simples, mas pode apresentar limitações na interpretação dos resultados e na detecção de falsos positivos.

O próximo passo é gerar um modelo probabilístico de fundo a partir dos dados de controle, que servirá como referência para identificar os KMERS enriquecidos. O valor de k é então estimado ou um valor pré-definido é utilizado, dependendo da abordagem adotada. Com

base nas regiões de pico, a estrutura de dados SMT é construída, permitindo a extração eficiente dos KMERS enriquecidos.

Por fim, os KMERS enriquecidos são extraídos da SMT e utilizados como sementes para gerar os modelos iniciais a partir dos dados enriquecidos. Esses modelos iniciais representam uma aproximação das sequências de interesse que serão usadas nas etapas subsequentes do algoritmo. Ao seguir esse processo, o algoritmo de inicialização consegue identificar, de forma eficiente, os KMERS mais relevantes e estabelecer uma base consistente para a análise subsequente das sequências nucleotídicas.

4.2.1.1 SMT

Diante dos desafios enfrentados na análise e interpretação de complexos conjuntos de dados biológicos, torna-se imperativo o desenvolvimento de ferramentas computacionais mais eficazes e precisas. Neste contexto, surgiu a necessidade de criar uma solução que abordasse as limitações existentes e também introduzisse uma nova perspectiva na maneira de processar e analisar esses dados. Foi com esse objetivo em mente que a SMT foi concebida e implementada.

A SMT é representada através de uma estrutura de dados bidimensional $M^{\nu \times 6}$, sendo ν o número de nós, implementada a partir de fragmentos de texto de mesma largura. Diferentemente das KTREES ou STREES, a SMT foi desenhada especificamente para lidar com KMERS. Ela foi levemente inspirada na teoria dos quadrados de ROOM (Archbold e Johnson, 1958), no qual cada elemento $M_{i,j}$ ou está vazio ou possui um valor configurado entre 1 e ν . Seu principal objetivo é armazenar eficientemente todos os KMERS pertencentes ao conjunto de dados principal, bem sua representação e contagem, permitindo que buscas rápidas sejam efetuadas. Desta forma, cada linha de M representa um nó $\{1, 2, 3, \dots, \nu\}$, as colunas de 1 a 4 representam os nucleotídeos e as duas últimas colunas, 5 e 6, representam a quantidade de vezes que um fragmento apareceu no conjunto de dados e o seu respectivo endereço.

Por exemplo, se $M_{1,T} = 10$, então símbolo T está presente entre os nós 1 e 10, conforme ilustra a Figura 4.4. Em comparação com estruturas similares, a SMT é mais eficiente para o armazenamento e busca de fragmentos de mesmo tamanho e sua construção possui complexidade de tempo linear $O(k)$, sendo k o tamanho da sequência de interesse. Além disso, é possível executar diversos algoritmos sobre a SMT em tempo linear, tais como busca exata e busca aproximada, que permitem por exemplo recuperar rapidamente os KMERS mais abundantes no dataset.

A Figura 4.4(a) mostra as sequências empregadas para construir a árvore. É interessante notar que como todos os fragmentos possuem o mesmo tamanho (neste exemplo, $k = 4$), então a profundidade da árvore será sempre igual a k , conforme mostra a Figura 4.4(b). Ainda nesta figura, podemos observar que cada vértice da árvore representa um nó e cada aresta um símbolo. Por fim, a Figura 4.4(c) representa a matriz de busca $M^{\nu \times 6} = M^{13 \times 6}$, com $\nu = 13$.

É importante destacar que não é necessário criar a árvore exibida na Figura 4.4(b). É possível construir diretamente a matriz M (Figura 4.4(c)) a partir do alinhamento (Figura 4.4(a)). No entanto, em alguns casos, a estrutura em árvore pode ser útil, pois facilita a análise da complexidade e a visualização. Ainda nesta figura, as linhas 5, 8, 9 e 13 representam os nós folha e armazenam os metadados endereço e contagem. Por exemplo, considere a linha 5. Observa-se que não existem valores inseridos nas colunas de 1 a 4, mas apenas nas colunas 5 e 6. Isso ocorre porque essa linha armazena o endereço do primeiro KMER (AACT) e sua respectiva contagem. Desta forma, é possível recuperar em tempo linear todos os fragmentos enriquecidos, bem como sua contagem e endereço absoluto. Algoritmo 5 mostra como a SMT é construída.

Neste algoritmo, na linha 3, a raiz (`root`) é definida com o valor 1 e o novo nó (`new_node`) como 2. Na linha 5, todas as sequências $\{s_1, s_2, s_3, \dots, s_n\}$ são processadas a

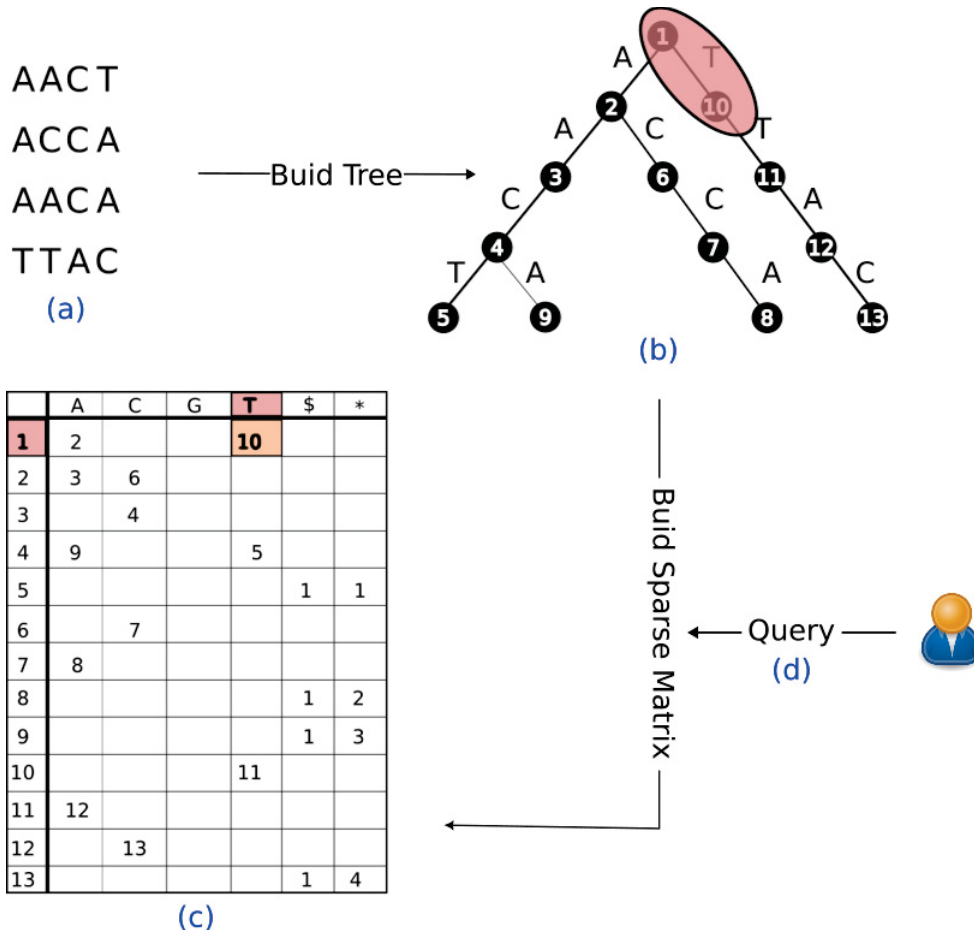


Figura 4.4: Estrutura de dados SMT. (a) 4-MERS empregados para construir a árvore. (b) SMT com profundidade 4. (c) SMT^{13×6}. (d) Usuário realizando pesquisas na SMT. Fonte: o autor.

partir da raiz, na qual cada uma delas possui largura k . Na linha 9 o algoritmo verifica se $M[\text{node}, \text{symbol}]$ é igual a 0. Se isso ocorrer, significa que não existe nenhuma aresta com o símbolo definido pela variável `symbol` saindo da variável `node` e indo em direção a outro nó. Em outras palavras, é necessário acrescentar um novo nó nesta posição e isso é realizado nas linhas 10 e 11. Por fim, a variável `node` recebe o valor do novo nó (`new_node`) e na linha 12 o novo nó é incrementado. Na linha 13, se $M[\text{node}, \text{symbol}]$ for diferente de 0, então o nó já existe e na linha 14 a variável `node` é apenas atualizada com o valor representado por $M[\text{node}, \text{symbol}]$.

A complexidade de espaço da SMT no pior caso é $O(\sigma v)$, sendo que σ é o número de símbolos do alfabeto, neste caso 4, e v é a quantidade de nós da árvore. O valor de v depende diretamente do tamanho e da variância dos fragmentos. Desta forma, quanto maior a variância, maior a quantidade de nós. É possível observar na Figura 4.4 (c) que a maior parte dos elementos de M está vazio. Esse comportamento é recorrente e pode ser calculado. Em particular, podemos calcular o valor esperado do nível de ocupação de M . Para isso, considere que o número máximo e mínimo de filhos que um nó pode ter é 4 e 0 respectivamente. A média seria então $\frac{4+0}{2} = 2$ filhos por nó.

Uma árvore com profundidade k e com cada nó apresentando em média 2 filhos, possuirá em média 2^k folhas. A quantidade média de nós da árvore será $v = \sum_{i=0}^k 2^i = 2^{k+1} - 1$. Neste caso, precisaremos de uma matriz M com $2^{k+1} - 1$ linhas e 4 colunas. Entretanto, cada linha possui 4 colunas sendo que na média apenas 2 delas estarão ocupadas. Com isso, podemos então

Algoritmo 5 CREATE-SMT:

```

1:  $n \leftarrow$  number of nodes
2:  $M \leftarrow \text{zeros}(n, 4)$  // Cria a matriz vazia com  $n$  linhas e 4 colunas.
3:  $root \leftarrow 1$ 
4:  $new\_node \leftarrow 2$ 
5: for  $s \in S$  do
6:    $node = root$ 
7:   for  $i = 1$  to  $w$  do
8:      $symbol = s[i]$ 
9:     if  $M[node, symbol] = 0$  then
10:       $M[node, symbol] = new\_node$ 
11:       $new\_node = new\_node + 1$ 
12:       $node = new\_node$ 
13:   else
14:      $node = M[node, symbol]$ 
15:   end if
16: end for
17: end for
18: return  $M$ 

```

calcular o nível de ocupação através de $\omega = \frac{2\nu}{4\nu} = 0,5$. Na prática, este número será ainda menor, pois a média do número de filhos diminui conforme a profundidade da árvore aumenta. Desta forma, a complexidade de espaço médio de M é $O(\omega\nu)$, como $\omega < \sigma$, então $O(\omega\nu) < O(\sigma\nu)$.

Sendo assim, podemos esperar que apenas metade da capacidade de M seja utilizada, o que caracteriza um grande desperdício de espaço. Por essa razão, o algoritmo CREATE-SMT foi implementado por meio de uma matriz esparsa de alto rendimento através da biblioteca de álgebra linear ARMADILLO (Sanderson e Curtin, 2016) empregando a linguagem de programação C++. A complexidade de tempo da SMT é mais simples de ser calculada e em todos os casos é $O(mk)$, no qual m representa a quantidade e k a largura dos fragmentos. Para ver isso, basta considerar que cada fragmento é composto por k símbolos. Como existem m fragmentos, então será necessário processar mk nucleotídeos no total. Porém, se considerarmos que $k \leq 20$ e que m pode crescer indefinidamente, então temos que $k \ll m$. Como k possui um limite superior constante, então podemos aproximar a complexidade para $O(mc) = O(m)$.

Uma das vantagens da SMT reside na sua eficiência computacional durante a etapa de construção, superando métodos tradicionais como KTREES e STREES, que demandam tempo quadrático para a mesma operação. Além disso, a SMT armazena apenas os fragmentos de interesse, tornando-se mais eficiente para lidar com KMERS. Comparada às tabelas HASH, a SMT também consome uma quantidade menor de espaço. Suponha que tenhamos m KMERS de tamanho k e que utilizemos uma tabela HASH com endereçamento direto, cujas chaves são os KMERS. Caso a distribuição de fundo seja uniforme, a quantidade de espaço utilizado pela tabela HASH será da ordem de $m \times k$ bytes. Em contraste, a SMT armazena apenas as informações necessárias e consome um espaço menor.

Assumindo que a distribuição de fundo seja uniforme, espera-se encontrar $0,25^k \times m$ kmers repetidos, o que significa que a quantidade total de KMERS sem repetição será $m - 0,25^k \times m$. Para armazenar esses KMERS em uma tabela HASH com duas colunas (contagem e endereço), serão necessárias $m - 0,25^k \times m$ linhas.

Por exemplo, considerando $m = 1000$ e $k = 10$, o número de linhas da tabela HASH será $m - m \times 0,25^{10} \approx 1000$. Se cada símbolo ocupa 1 byte e cada KMER contém 10 símbolos, então cada KMER ocupará 10 bytes. Portanto, a quantidade total de bytes necessários para armazenar todos os KMERS sem repetição será $10 \times 1000 = 10000$ bytes, ou aproximadamente 10KB.

Adicionando as colunas responsáveis por armazenar a contagem e o endereço dos respectivos KMERS, teremos um total de $m \times 8$ bytes, que no exemplo dado é igual a $1000 \times 8 = 8000$ bytes ou 8 KB. Somando com o espaço necessário para armazenar os KMERS, o tamanho total da tabela HASH será de aproximadamente 18 KB (10 KB + 8 KB).

Na SMT, mesmo que todos os $m = 1000$ KMERS sejam diferentes, isso resultaria em uma matriz esparsa de 10.000 linhas e 4 colunas. Como cada célula da matriz armazena um número inteiro, a quantidade bruta de dados armazenados seria de $10.000 \times 4 = 40.000$ bytes, ou seja, 40 KB.

No entanto, a SMT utiliza uma matriz esparsa de alto rendimento, o que faz com que a quantidade de dados armazenados seja menor na prática. Esse número dependerá do nível de ocupação ω , que na média é 0,5, mas na prática fica abaixo de 0,3. Para ilustrar, considerando $\omega = 0,2$, teríamos uma quantidade de dados armazenados de $\omega \times 40.000 = 8.000$ bytes, ou seja, 8 KB. É interessante destacar que a performance da SMT piora conforme o índice de ocupação aumenta porém, em nossos testes esse índice sempre figurou abaixo de 0,3. Em parte, isso se deve a uma moderada redundância de KMERS nos genomas analisados.

É importante ressaltar que na análise de MOTIFS, uma parcela considerável dos KMERS são repetidos, o que resulta em uma matriz ainda mais esparsa e, conseqüentemente, uma quantidade de dados armazenados ainda menor. Além disso, a SMT permite o uso de compactação de dados para reduzir ainda mais o espaço necessário de armazenamento.

KDIVE Dentre os algoritmos que executam sobre a SMT, o KDIVE talvez seja o mais importante do ponto de vista da análise de KMERS. Ele foi criado com o objetivo de realizar buscas eficientes de fragmentos de texto na SMT, mesmo que estes apresentem até d degenerações. Em outras palavras, o KDIVE deve retornar verdadeiro para a busca mesmo que a STRING de consulta contenha até d MISMATCHES. É importante destacar que o algoritmo assume como premissa que a probabilidade de um MISMATCH ocorrer é a mesma para qualquer posição da sequência.

Por exemplo, se a probabilidade de um MATCH é igual a $\frac{1}{4}$, então a de um MISMATCH é $1 - \frac{1}{4} = \frac{3}{4}$. Portanto, a ocorrência de um MISMATCH é três vezes mais provável do que a de um MATCH. Essa característica é importante, porque altera a esperança do número de computações executadas e conseqüentemente modifica a complexidade do algoritmo. A construção do KDIVE foi realizada tomando como base a busca em profundidade limitada (Norvig e Russell, 2013) e o Algoritmo 1 (BRANCH AND BOUND) estudado no Capítulo 2. O Algoritmo 6 mostra o funcionamento básico do KDIVE.

O Algoritmo 6 foi implementado de forma recursiva e recebe como principais parâmetros a matriz SMT (construída a partir do Algoritmo 5), o fragmento s e o número total de degenerações permitidas d . Se $s \in M$ com no máximo d incompatibilidades, o algoritmo retorna o valor booleano TRUE, indicando a presença do fragmento com até d mutações.

O algoritmo realiza sua tarefa percorrendo a SMT para identificar correspondências entre a STRING de pesquisa e os KMERS armazenados, levando em conta a quantidade de MISMATCHES permitidos. É relevante destacar que este algoritmo pode ser facilmente adaptado para retornar o conjunto completo de elementos degenerados, em vez de apenas um valor booleano (TRUE ou FALSE). Esta modificação pode ser útil em determinadas aplicações, pois permite obter informações mais detalhadas sobre as correspondências encontradas. Por exemplo, é possível

Algoritmo 6 KDIVE:

```

1: if  $k > d_{max}$  : then
2:   return // A string de pesquisa ultrapassou  $d$  mismatches e a árvore será podada.
3: end if
4: if  $i \geq k$  : then
5:    $resp = 1$  // Encontrou a string de pesquisa com até  $d$  mismatches.
6: end if
7:  $symbol = s[i]$ 
8: for  $j = 1$  to 4 do
9:    $next\_node = M[node, j]$ 
10:  if  $next\_node \neq 0$  and  $symbol = j$  then
11:     $kdiv(M, s, d_{max}, d, i + 1, k, next\_node, resp)$  // Match!
12:  else
13:     $kdiv(M, s, d_{max}, d + 1, i + 1, k, next\_node, resp)$  // Mismatch!
14:  end if
15:  if  $resp = 1$  then
16:    break
17:  end if
18: end for

```

identificar quais *KMERS* da SMT correspondem ao fragmento de pesquisa, considerando as mutações permitidas.

Os parâmetros d_{max} , d , i e k permitem controlar a busca por correspondências exatas ou aproximadas. A recursão permite percorrer a árvore em profundidade e verificar todos os caminhos possíveis até encontrar uma correspondência ou atingir a profundidade máxima definida pelo parâmetro d_{max} . Uma possível chamada ao Algoritmo 6 seria $kdiv(M, s, d_{max} = 2, d = 0, i = 1, k = 10, node = root, resp = FALSE)$. Neste caso, o algoritmo espera encontrar um *match* mesmo que até $d = 2$ mutações sejam detectadas em sequências de tamanho $k = 10$.

O Algoritmo 6 possui dois casos base, mostrados nas linhas 1 e 4. O primeiro verifica se a quantidade de mutações ultrapassou o limite d_{max} , ou seja, se $d \geq d_{max}$. Caso essa condição seja satisfeita, ocorrerá uma poda na busca. O segundo caso base verifica se o algoritmo analisou a *STRING* de pesquisa por completo, o que ocorrerá se $i \geq k$. Se essa condição for verdadeira, então o padrão foi encontrado e a variável $resp$ é atualizada para o valor *true*.

Vamos considerar a execução do Algoritmo 6 com $d_{max} = 1$, $s = CACT$ e a SMT mostrada na Figura 4.4 (c). A Figura 4.5 ilustra o comportamento do algoritmo para este exemplo. No primeiro nível da árvore, ao expandir o nó raiz, os símbolos A e T são avaliados. No entanto, nenhum deles corresponde a C, que é o primeiro símbolo de s . Por essa razão, o valor de d é incrementado de 0 para 1. O próximo nó a ser expandido é o 2, cujos filhos são os nós 3 e 6. A aresta entre 2 e 3 possui o símbolo A, que corresponde ao segundo símbolo de s , e portanto o valor de d não é incrementado. A expansão do nó 6 resulta no nó 7, cuja aresta possui o símbolo C. Como o segundo símbolo de s é A, então d é incrementado de 1 para 2.

Neste momento, ocorre a poda da árvore, pois na próxima chamada recursiva, o caso base localizado na linha 1 do Algoritmo 6 será executado, o que encerrará a recursão. Entretanto, outras chamadas recursivas ainda estarão em execução e ao final, o algoritmo encontrará a sequência sombreada em azul e retornará a variável $resp = true$. Por outro lado, se $d_{max} = 1$

e $s = CACG$, o Algoritmo 6 retornará `resp = false`, indicando que o fragmento de pesquisa com até 1 mutação não foi encontrado na SMT.

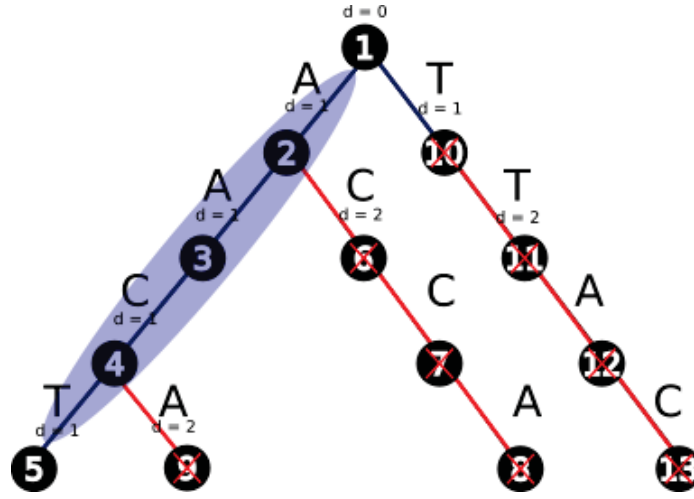


Figura 4.5: Exemplo de execução do D-DFS para $d = 1$ e $s = CACT$. O algoritmo encontrou a sequência sombreada em azul e alterou o estado da variável `RESP` para `TRUE`. Fonte: o autor.

A complexidade de tempo do `KDIVE` pode ser medida através da distribuição BINOMIAL NEGATIVA. Considere uma SMT com v nós, d MISMATCHES e fragmentos de largura k . Considere ainda a v.a. $X \sim NB(d, p)$, que conta a quantidade de comparações que o algoritmo `KDIVE` realiza. É fácil verificar que $p = \frac{3}{4}$, pois se a probabilidade de MATCH é $\frac{1}{4}$ a probabilidade de um MISMATCH será $1 - \frac{1}{4} = \frac{3}{4}$, portanto $X \sim NB(d, \frac{3}{4})$. A PMF $P_x(d, p)$ é dada pela Equação 4.1 e a esperança pela Equação 4.2.

$$P_x(d, p) = \binom{n-1}{d-1} p^d (1-p)^{n-d} \quad (4.1)$$

$$E(X) = \sum_{x \in X} x P_x(d, p) \quad (4.2)$$

A distribuição BINOMIAL negativa é uma generalização da distribuição geométrica que é definida por $X \sim GEOM(p) = NB(1, p)$. Portanto, a distribuição geométrica é a distribuição BINOMIAL negativa com $d = 1$ sendo possível empregá-la para calcular o valor esperado. Como exemplo, considere $X \sim NB(1, p)$, $Y \sim NB(1, p)$ e $Z = X + Y$. Se X e Y forem independentes, então $Z = X + Y \sim NB(2, p)$. Portanto, podemos calcular $E(Z) = E(X) + E(Y) = \frac{1}{p} + \frac{1}{p} = \frac{2}{p}$. Desta forma, $X \sim NB(d, p)$ pode ser escrito como $Z = X_1 + X_2 + X_3 + \dots + X_d$, sendo que $X_i \sim NB(1, p)$. Calculando $E(Z) = E(X_1) + E(X_2) + E(X_3) + \dots + E(X_d) = \frac{1}{p} + \dots + \frac{1}{p} = \frac{d}{p}$.

Com este resultado, podemos escrever que a complexidade de tempo do `KDIVE` é $O(\frac{d \times 4}{3}) = O(d)$. É fácil verificar que quanto maior o tamanho de d , mais operações serão necessárias. Em particular, a complexidade cresce linearmente com d . É importante destacar que MOTIFS dificilmente ultrapassam a largura de 20 nucleotídeos e a quantidade de mutações não costuma ser maior do que 20% de seu tamanho. Se considerarmos isso, esperamos encontrar em média $d = 5$ MISMATCHES, o que resulta em $\frac{5 \times 4}{3} \approx 6,666$ comparações por fragmento. Considere como exemplo que temos $n = 1000$ STRINGS de pesquisa com $k = 20$ para submeter a uma SMT. A Figura 4.6 mostra o tempo teórico de execução para vários valores de d e n .

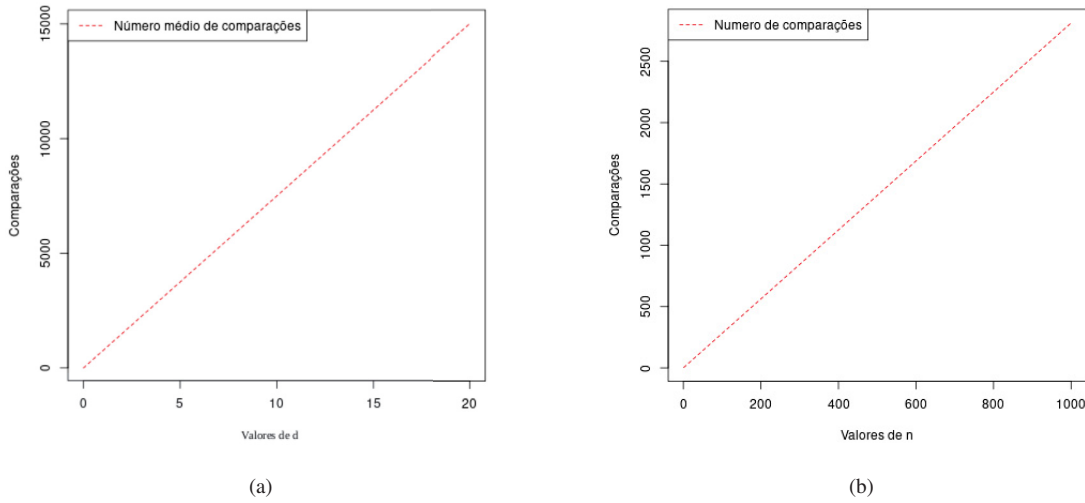


Figura 4.6: (a) Tempo teórico de execução do `KDIVE` para $n = 1000$ strings e $d = 0, 1, 2, 3, \dots, 20$ mismatches. (b) Tempo teórico de execução do `D-DFS` para $d = 5$ mismatches e $n = 0, 1, 2, 3, \dots, 1000$ strings. Fonte: o autor.

A Figura 4.6(a) mostra o número de comparações quando variamos o valor de d de 0 até 20. É possível observar que para $d = 20$ e $n = 1000$, em média 15000 operações são realizadas. A Figura 4.6(b) exibe um cenário mais realista, que mostra o número de comparações quando deixamos o valor de $d = 5$ e alteramos o valor de n de 0 até 1000. Podemos observar nesta figura que para $n = 1000$ e $k = 20$, são necessárias aproximadamente 2500 comparações para processar todas as STRINGS de pesquisa.

Podemos calcular a complexidade do algoritmo `KDIVE` em relação ao tamanho das sequências de entrada. Considere um conjunto de dados com n sequências de largura t . Temos um total de $m = t - k + 1$ fragmentos de tamanho k em cada sequência de tamanho t . Seja X o número de comparações realizadas por cada fragmento. Sabemos que X segue uma distribuição BINOMIAL negativa com parâmetros $p = \frac{3}{4}$ e d e que $\mathbb{E}(X) = \frac{d}{p} = \frac{4d}{3}$. Então, a quantidade de comparações por sequência será $m \frac{4d}{3}$. Se considerarmos por exemplo um fragmento de tamanho $k = 20$ com $d = 5$ mutações, então $\mathbb{E}(X) = \frac{5 \times 4}{3} \approx 6,666$, o que resultará em média em $6,666 \dots \times m = O(m)$. É interessante destacar que este valor será ainda menor na prática, porque o `KDIVE` utiliza o algoritmo `BRANCH AND BOUND` para realizar podas na árvore de pesquisa.

Mesmo no pior caso, o algoritmo ainda exibirá um comportamento assintótico linear. Em uma análise menos aprofundada, chegamos à seguinte complexidade: $k(m - k + 1) = km - k^2 + k$, no qual o termo dominante é k^2 e, portanto, a complexidade desse algoritmo será da ordem de $O(k^2)$. Porém, k frequentemente é bem menor que m , e desta forma podemos fazer uma análise mais refinada. Dado que $k \ll m$, a expressão $k(m - k + 1)$ pode ser aproximada como: $k(m - k + 1) \approx km$.

A complexidade do algoritmo é, em geral, dominada pelo termo km e pode ser expressa como $O(km)$. No entanto, considerando que k não é maior que 20 e m pode crescer indefinidamente, podemos ajustar a análise de complexidade. Como k possui um limite superior constante ($k \leq 20$), a complexidade do algoritmo é, principalmente, influenciada pelo termo m . Portanto, podemos expressar a complexidade do algoritmo como $O(m)$.

KSEARCH Este algoritmo tem como principal objetivo realizar buscas exatas na estrutura de dados SMT. Ele destaca-se por sua eficiência, já que opera com uma complexidade temporal de $O(k)$, no qual k é o tamanho do `KMER` a ser encontrado. Dessa forma, o algoritmo

percorre o fragmento uma única vez, fazendo consultas diretas à estrutura SMT para encontrar correspondências exatas. Sua eficiência o torna particularmente útil em cenários onde é imperativo minimizar o tempo de busca, o que é frequentemente o caso em aplicações de bioinformática e biologia computacional. O Algoritmo 7 ilustra como essa funcionalidade foi implementada.

Algoritmo 7 KSEARCH

```

1:  $k \leftarrow \text{kmer.size}()$ 
2:  $\text{next} \leftarrow 0$ 
3: for  $i = 1$  to  $k$  do
4:    $c \leftarrow \text{kmer}[i]$ 
5:    $\text{next} \leftarrow \text{SMT}(\text{next}, c)$ 
6:   if  $\text{next} == 0$  then
7:     return False
8:   end if
9: end for
10: return True

```

O algoritmo KSEARCH é bastante simples e começa por determinar o tamanho do KMER e armazena essa informação na variável k . Em seguida, inicializa a variável next com o valor 0. Esta variável será usada para manter o estado atual da busca dentro da estrutura SMT. Após isso, o algoritmo entra no LOOP principal que percorre cada caractere c do KMER. Dentro desse LOOP, a SMT é chamada com o estado atual next e o caractere c para obter o próximo estado. Se em algum ponto o estado next se torna 0, o algoritmo conclui que o KMER não foi encontrado e retorna FALSE. A eficiência deste algoritmo é destacada pela sua complexidade de tempo $O(k)$, que é linear em relação ao tamanho do KMER. Isso torna o KSEARCH uma ferramenta extremamente eficaz para buscas exatas dentro da SMT.

IUPAC-SEARCH O algoritmo IUPAC-SEARCH é uma função de busca especializada projetada para identificar e contar padrões especificados no formato IUPAC dentro de uma SMT. O formato IUPAC é uma maneira padrão de representar sequências de ácidos nucleicos, tornando este algoritmo particularmente útil para bioinformática e o estudo de sequências genéticas. Quando um padrão IUPAC é fornecido para a função, ela percorre a SMT para identificar todas as instâncias desse padrão.

A função então retorna todos os KMERS que correspondem ao padrão IUPAC junto com suas contagens, armazenados em um mapa onde cada KMER é mapeado para a respectiva contagem. Isso fornece um resumo conciso e de alto nível de todos os locais na SMT onde ocorre o padrão de interesse, bem como a frequência com que cada KMER associado é encontrado. a notação IUPAC para ácidos nucleicos, usada para representar sequências genômicas, inclui letras para os quatro nucleotídeos padrão e também letras para representar possíveis combinações destes. Abaixo estão as definições para as letras de IUPAC que representam padrões degenerados.

- R: representa uma purina, ou seja, é uma abreviação para uma posição que pode ser uma Adenina (A) ou Guanina (G).
- Y: representa uma pirimidina, ou seja, é uma abreviação para uma posição que pode ser uma Citosina (C) ou Timina (T).
- S: STRONG ou *strong interaction*, é uma ligação de 3 hidrogênios. Representa uma posição que pode ser uma Guanina (G) ou Citosina (C).

- W: *WEAK* ou *weak interaction*, é uma ligação de 2 hidrogênios. Representa uma posição que pode ser uma Adenina (A) ou Timina (T).
- K: *KETO*, ou seja, os nucleotídeos com grupos cetônicos. Representa uma posição que pode ser uma Guanina (G) ou Timina (T).
- M: *AMINO*, ou seja, os nucleotídeos com grupos amino. Representa uma posição que pode ser uma Adenina (A) ou Citosina (C).
- B: representa uma posição que pode ser qualquer base, exceto Adenina (A) (ou seja, pode ser C, G ou T).
- D: representa uma posição que pode ser qualquer base, exceto Citosina (C) (ou seja, pode ser A, G ou T).
- H: representa uma posição que pode ser qualquer base, exceto Guanina (G) (ou seja, pode ser A, C ou T).
- V: representa uma posição que pode ser qualquer base, exceto Timina (T) (ou seja, pode ser A, C ou G).
- N ou X: representa qualquer base, ou seja, é uma abreviação para uma posição que pode ser uma Adenina (A), Guanina (G), Citosina (C) ou Timina (T).

O Algoritmo 8 exibe o funcionamento do IUPAC-SEARCH. O algoritmo começa com uma condição que verifica se a variável i é igual à k na linha 1. Se as duas são iguais, isso indica que o algoritmo atingiu o final do padrão IUPAC que está procurando na árvore. Neste caso, entre as linhas 2 a 5, o algoritmo recupera a contagem e o endereço do nó atual na árvore, registra a contagem no mapa de padrões que corresponde ao endereço e, em seguida, encerra a execução do algoritmo.

Se i não é igual a k , o algoritmo procede para obter o caractere atual do padrão IUPAC e inicializa as variáveis *symbol* e *next* com zero nas linhas 8 a 9. Uma nova condição é verificada na linha 10, onde se o caractere atual for um dos caracteres de bases nucleotídicas (A, C, G, T), o algoritmo executa as instruções nas linhas 11 a 15. Neste bloco, o algoritmo transforma o caractere atual em seu respectivo valor inteiro (utilizando a função *char2int*), obtém o nó filho correspondente na árvore e, se tal nó existir, chama recursivamente a função IUPAC-SEARCH para continuar a busca a partir desse nó.

Na linha 16, se o caractere atual for *R* (indicando purina), as linhas 17 a 22 são executadas. Aqui, o algoritmo verifica cada um dos símbolos possíveis para purina (A e G). Para cada símbolo, obtém o nó filho correspondente na árvore e, se esse nó existir, chama recursivamente a função IUPAC-SEARCH.

Na linha 23, se o caractere atual for *Y* (indicando pirimidina), o algoritmo executa as linhas 24 a 28. Este bloco é semelhante ao anterior, porém, verifica-se cada um dos símbolos possíveis para pirimidina (C e T). Finalmente, a linha 31 nota que outras letras IUPAC (como S, W, K, M, B, D, H, V, N, X) devem seguir um formato semelhante ao explicado acima. No fim, o algoritmo armazena todos os *KMERS* e suas respectivas contagens no mapa *patterns*.

A principal vantagem do IUPAC-SEARCH sobre o algoritmo de força bruta reside na sua superior eficiência computacional. Enquanto o método de força bruta procede à análise exaustiva de todos os *KMERS* do conjunto de dados, tentando encontrar correspondências com o padrão IUPAC, o IUPAC-SEARCH, por outro lado, opera de maneira mais inteligente e seletiva sobre a SMT. Este último algoritmo se destaca em termos de velocidade, uma vez que consegue fazer podas

Algoritmo 8 IUPAC-SEARCH

```

1: if  $i == k$  then
2:    $count \leftarrow SMT(node, 4)$ 
3:    $address \leftarrow SMT(node, 4 + 1)$ 
4:    $patterns[address] \leftarrow count$ 
5:   return
6: end if
7:  $c \leftarrow iupac[i]$ 
8:  $symbol \leftarrow 0$ 
9:  $next \leftarrow 0$ 
10: if  $c == A$  OR  $c == C$  OR  $c == G$  OR  $c == T$  then
11:    $symbol \leftarrow char2int(c)$ 
12:    $next \leftarrow SMT(node, symbol)$ 
13:   if  $next \neq 0$  then
14:     IUPACSearch( $SMT, patterns, iupac, next, i + 1, k$ )
15:   end if
16: else if  $c == R$ (puRine) then
17:   for  $symbol \in \{A, G\}$  do
18:      $next \leftarrow SMT(node, symbol)$ 
19:     if  $next \neq 0$  then
20:       IUPACSearch( $SMT, patterns, iupac, next, i + 1, k$ )
21:     end if
22:   end for
23: else if  $c == Y$ (pYrimidine) then
24:   for  $symbol \in \{C, T\}$  do
25:      $next \leftarrow SMT(node, symbol)$ 
26:     if  $next \neq 0$  then
27:       IUPACSearch( $SMT, patterns, iupac, next, i + 1, k$ )
28:     end if
29:   end for
30: end if
31: Outros casos (como S, W, K, M, B, D, H, V, N, X) seguem um formato semelhante.

```

na árvore de pesquisa. Isso significa que nós subordinados a padrões que não correspondem ao critério de busca não são sequer considerados, otimizando assim o processo de busca. A Figura 4.7 ilustra de forma clara o funcionamento do IUPAC-SEARCH, oferecendo uma visão mais abrangente de seu funcionamento.

KHMAP O algoritmo KHMAP utiliza uma abordagem recursiva para gerar tabelas HASH de contagem a partir de uma SMT previamente construída. Se uma SMT foi inicialmente criada com um valor de k igual a κ , o KHMAP é capaz de extrair tabelas de contagem para qualquer valor de k tal que $1 \leq k \leq \kappa$. A execução deste processo é facilitada pelos Algoritmos 9 e 10.

O algoritmo COUNT K-MERS começa uma função recursiva que aceita cinco argumentos: a estrutura de dados SMT, o nó atual da árvore ($node$), o tamanho máximo do KMER (k_{max}), um mapa hash para armazenar as contagens ($hmap$) e um contador j para manter o controle durante a recursividade. A primeira coisa que o algoritmo faz é verificar se o contador j é igual a k_{max} . Se for o caso, ele entra em um bloco condicional para executar a contagem. Neste

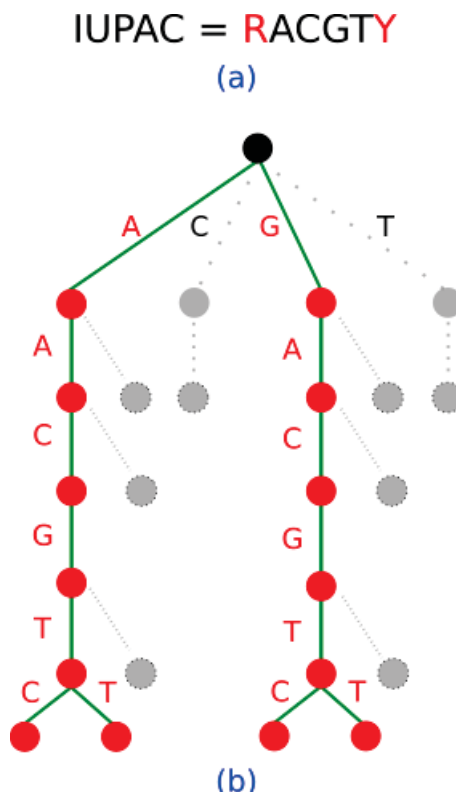


Figura 4.7: Funcionamento do algoritmo IUPAC SEARCH. A Figura (a) exibe o padrão no formato IUPAC, no qual R e Y significam purina e pirimidina respectivamente. (b) Em verde, temos o caminho realizado pelo algoritmo na SMT para recuperar todos os padrões que correspondentes. Fonte: o autor.

bloco condicional, o algoritmo obtém o valor da contagem armazenado no nó atual através de `smt(node, 4)`. Este valor de contagem é então adicionado ao mapa hash `hmap` associado ao `KMER` em análise. Logo após, a função retorna, finalizando essa instância da recursão.

Se j não for igual a $k_{\max}$, o algoritmo prossegue para um `LOOP` variando de 1 a 4, que basicamente são as representações numéricas para os quatro nucleotídeos em uma sequência de DNA. Para cada iteração, o algoritmo verifica se há um próximo nó válido na árvore SMT. Se houver, a função chama a si mesma recursivamente, avançando para o próximo nó e incrementando o contador j . Essa estratégia permite que o algoritmo explore eficientemente a árvore SMT e extraia as contagens de todos os `KMERS` possíveis de tamanho até $k_{\max}$.

O algoritmo `CREATING HASH TABLE` é uma extensão natural do algoritmo `COUNTING K-MERS` e serve para criar uma tabela hash de contagens de `KMERS`. Ele também é uma função recursiva e começa com a função `HASH`, que aceita sete argumentos: a estrutura de dados SMT, o nó atual na árvore (`node`), o tamanho do `KMER` desejado (k), o tamanho máximo do `KMER` ($k_{\max}$), a sequência atual do `KMER`, o mapa hash (`hmap`) e um contador j para controlar a recursão. Semelhante ao algoritmo anterior, o primeiro passo é verificar se o contador j alcançou o tamanho máximo k . Se for verdade, o algoritmo chama a função `COUNT_KMERS` para realizar a contagem do `KMER` e armazená-la no mapa hash. Neste ponto, a função retorna, encerrando a instância atual da recursão.

Caso contrário, o algoritmo entra em um `LOOP` que itera de 1 a 4. Cada valor nesta iteração representa um dos quatro nucleotídeos em uma sequência de DNA. Dentro do `LOOP`, o algoritmo verifica se há um próximo nó válido na estrutura SMT. Se existir, a função `HASH` chama a si mesma recursivamente, avançando para o próximo nó. Além disso, ela também atualiza o valor de `KMER` acrescentando o caractere correspondente ao próximo nó e incrementa

Algoritmo 9 COUNTING K-MERS

```

1: function COUNT_KMERS(SMT, node, kmax, hmap, j)
2: if j == kmax then
3:   count = SMT(node, 4)
4:   hmap(kmer) += count
5:   return
6: end if
7: for i = 1 to 4 do
8:   next = SMT(node, i)
9:   if next > 0 then
10:    COUNT_KMERS(SMT, node, kmax, hmap, j + 1)
11:   end if
12: end for

```

Algoritmo 10 CREATING HASH TABLE

```

1: function HASH(SMT, node, k, kmax, kmer, hmap, j)
2: if j == kmax then
3:   COUNT_KMERS(SMT, node, kmax, hmap, j)
4:   return
5: end if
6: for i = 1 to 4 do
7:   next = SMT(node, i)
8:   if next > 0 then
9:    HASH(SMT, next, k, kmax, kmer + char(i), hmap, j + 1)
10:  end if
11: end for

```

o contador j . Assim, o algoritmo mantém um registro de todos os **KMERS** em um mapa hash, proporcionando um acesso rápido e eficiente aos dados. Isso permite que a tabela hash seja criada de forma eficiente, navegando pela árvore SMT e coletando as informações necessárias.

4.2.2 Otimização

O objetivo desta etapa é otimizar as sementes iniciais obtidas na fase anterior através do algoritmo **EM**. A utilização deste algoritmo permitiu refinar os parâmetros associados às sementes, tornando-as representações mais acuradas. Este processo de otimização é importante para a qualidade dos modelos resultantes e para o sucesso das análises subsequentes. Serão apresentados os principais modelos probabilísticos que são integralmente suportados pelo **BIOMAPP::CHIP**. Faremos uma descrição aprofundada sobre a implementação desses modelos em nosso arcabouço, detalhando como eles operam e interagem com os outros componentes do sistema. Estes modelos serão elucidados em termos de suas especificidades técnicas e conceituais. Também abordaremos como cada modelo contribui para a robustez do arcabouço, enfatizando a sua relevância para a capacidade global do sistema de identificar e caracterizar **MOTIFS**. Os modelos utilizados em nosso estudo são: **OOPS**, **ZOOPS** e **ANR**, que podem ser visualizados graficamente na Figura 4.8.

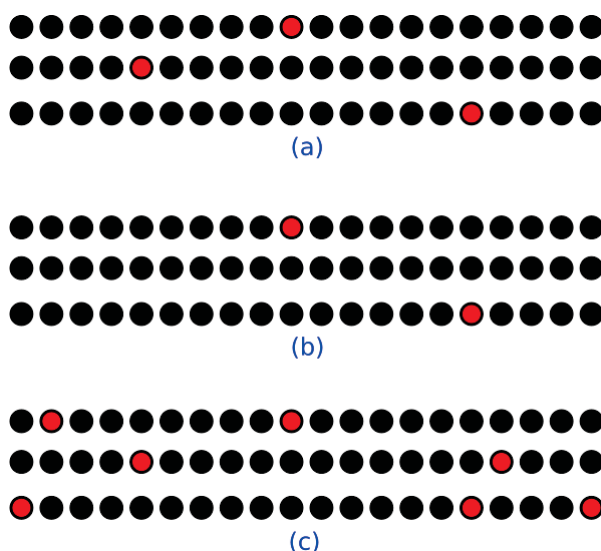


Figura 4.8: Representação abstrata dos modelos **OOPS**, **ZOOPS** e **ANR**, no qual os círculos vermelhos representam o **MOTIF** conservado e os círculos pretos representam a distribuição de fundo. (a) No modelo **OOPS**, é possível observar que em cada sequência, existe exatamente uma cópia do círculo vermelho. (b) No modelo **ZOOPS**, é possível observar que em cada sequência, existe zero ou uma cópia do círculo vermelho. (c) No modelo **ANR**, é possível observar que em cada sequência, existe um número arbitrário de cópias do círculo vermelho. Fonte: o autor.

4.2.2.1 *OOPS*

Representado pela Figura 4.8(a), este modelo possui como premissa a ocorrência de exatamente uma cópia do **MOTIF** de interesse por sequência. Isso torna o **OOPS** útil em cenários onde se espera uma regularidade estrita na ocorrência dos fragmentos conservados, como em determinados processos de ligação de proteínas ao DNA. Por exemplo, em estudos genéticos onde um fator de transcrição específico se liga a um único local no promotor de cada gene, o modelo **OOPS** é comumente empregado. Além disso, em certas análises de **RNA-SEQ** onde se procura por um único **MOTIF** que influencia a estabilidade do RNA ou a eficiência de tradução, o modelo **OOPS** também pode ser apropriado. O Algoritmo 11 mostra como o **OOPS** foi implementado.

Algoritmo 11 OOPS

```

1:  $\alpha_i \sim \text{DIRICHLET}(a_1, a_2, a_3, a_4)$ 
2: while True do
3:    $\gamma_{ij} = \frac{1}{m}$  // Inicializa o vetor  $\gamma_i$  o valor  $\frac{1}{m}$ , para  $j = 1, 2, 3, \dots, m$ .
4:    $\hat{\alpha} = \text{zeros}(4, k)$ 
5:   for  $i = 1$  to  $n$  do
6:      $\text{seq}_i = \text{seqs}[i]$ 
7:     for  $j = 1$  to  $m$  do
8:        $\gamma_{ij}^{(t)} = \frac{\gamma_{ij} \text{Pr}(\text{seq}_i | \rho=j)}{\sum_{l=1}^m \gamma_{il} \text{Pr}(\text{seq}_i | \rho=l)}$  // E-step.
9:       for  $c = j$  to  $j + k - 1$  do
10:         $l = \text{seq}_i[c]$ 
11:         $\hat{\alpha}[l, c] = \hat{\alpha}[l, c] + \gamma_{ij}^{(t)}$  // M-step.
12:      end for
13:    end for
14:  end for
15:   $\alpha = \frac{1}{n} \hat{\alpha}$  // Normalização.
16:  if  $|Q^{(t)} - Q^{(t-1)}| \leq \epsilon$  then
17:    BREAK // Verifica convergência.
18:  end if
19: end while
20: return ( $\alpha$ )

```

Este algoritmo começa com a construção de um modelo inicial α a partir de uma distribuição de `DIRICHLET` (linha 1). Uma vez que o modelo inicial está definido, o algoritmo entra em um `LOOP` que continuará até que a convergência seja alcançada, indicada pelo critério de que a diferença absoluta entre as estimativas de log-verossimilhança entre iterações consecutivas seja menor ou igual a um pequeno valor ϵ (linha 16).

Dentro do `LOOP`, o vetor γ_i é inicializado com o valor $1/m$ (linha 3) e uma matriz auxiliar $\hat{\alpha}$ é definida com zeros (linha 4). Em seguida, o algoritmo percorre todas as sequências (linha 5). Para cada sequência, o algoritmo passa por todas as posições possíveis dentro da sequência (linha 7). Na linha 8, é realizado o passo E (`EXPECTATION`), onde o valor γ_{ij} é atualizado com base na probabilidade da sequência estar em uma determinada posição dada a distribuição de probabilidade corrente.

Em seguida, para cada posição no intervalo de $1, 2, 3, \dots, m$ (linhas 9-12), o algoritmo atualiza a matriz $\hat{\alpha}$ somando o valor atualizado de γ_{ij} ao valor corrente de $\hat{\alpha}[l, c]$, onde l é a base do `MOTIF` na posição atual e c indica a coluna do fragmento, que vai de j até $j + k - 1$. Este é o passo M (`MAXIMIZATION`) onde o modelo é atualizado para refletir as novas estimativas de posição do fragmento conservado. Na linha 15, a matriz α é atualizada para ser a média dos valores em $\hat{\alpha}$ divididos pelo número total de sequências n .

O algoritmo continua iterando até que a diferença entre os valores de log-verossimilhança entre duas iterações consecutivas seja menor ou igual a um pequeno valor ϵ (linhas 16-18). Quando essa condição é satisfeita, o `LOOP` termina, indicando que o modelo convergiu para um valor estável. Finalmente, o modelo α aprendido é retornado (linha 20). Este modelo contém as probabilidades atualizadas para cada base em cada posição do `MOTIF`, refletindo os locais mais prováveis do fragmento de interesse no conjunto de dados. A Figura 4.9 exibe graficamente o funcionamento do algoritmo OOPS.

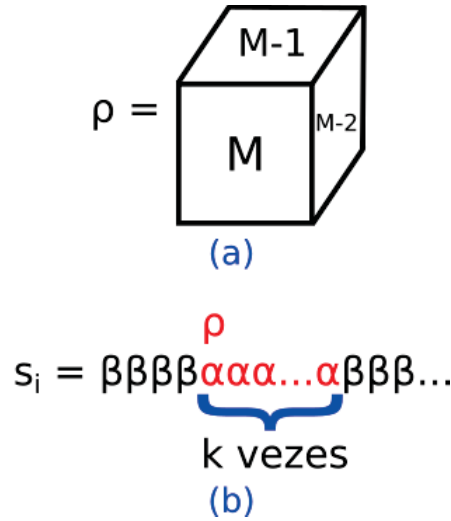


Figura 4.9: Representação gráfica do modelo oops. No painel (a), postulamos que, a priori, cada posição ρ tem a capacidade de contribuir de maneira equitativa para a presença ou ausência do fragmento conservado. No painel (b), apresentamos uma modelagem gráfica elucidativa, demonstrando a abordagem do modelo em lidar com um fragmento situado em uma posição específica, denotada como ρ . Fonte: o autor.

As linhas (16-18) do algoritmo oops têm particular relevância, pois é onde ocorre a verificação da convergência. Frequentemente, a função Q , que denota o cálculo da log-verossimilhança completa dos dados, é usada para este propósito, conforme expresso na Equação 4.3. No entanto, a convergência também pode ser monitorada pela log-verossimilhança (LL), conforme descrito na Equação 4.4, ou pelo conteúdo de informação (IC), tal como elucidado no Capítulo 2 e representado na Equação 2.22. A escolha da métrica para monitorar a convergência pode depender do contexto específico e das características do conjunto de dados em questão.

$$Q = \sum_{i=1}^n \sum_{j=1}^m \gamma_{ij} [\log Pr(s_i|\rho = j) + \log Pr(\rho = j)] \quad (4.3)$$

$$LL = \sum_{i=1}^n \log \sum_{j=1}^m Pr(\rho = j) Pr(s_i|\rho = j) \quad (4.4)$$

Tanto a função Q quanto a função LL desempenham papéis importantes na avaliação da convergência do algoritmo EM. A função Q é otimizada a cada iteração do EM. Se ela não apresentar uma mudança significativa entre duas iterações consecutivas, isso pode ser um sinal de que o algoritmo EM está convergindo para um máximo local ou global. A quantidade de mudança que é considerada “significativa” pode variar e é geralmente definida por uma tolerância predeterminada.

A função Q é definida como a expectativa do logaritmo da verossimilhança completa com respeito à distribuição a posteriori das variáveis latentes. Em outras palavras, é a média ponderada dos logaritmos das probabilidades condicionais dos dados observados e das variáveis latentes, onde os pesos são as probabilidades a posteriori das variáveis latentes, conforme mostram as Equações 4.5 ... 4.10. Em particular, as Equações 4.10 e 4.3 representam a mesma quantidade.

$$Q(\theta|\theta^t) = \prod_{i=1}^n \prod_{j=1}^k (w_j f(x_i|\theta_j))^{\mathbb{1}_{\{Z_i=j\}}} \quad (4.5)$$

$$\log Q(\theta|\theta^t) = \log \prod_{i=1}^n \prod_{j=1}^k (w_j f(x_i|\theta_j))^{\mathbb{1}_{\{Z_i=j\}}} \quad (4.6)$$

$$\log Q(\theta|\theta^t) = \sum_{i=1}^n \sum_{j=1}^k \log(w_j f(x_i|\theta_j))^{\mathbb{1}_{\{Z_i=j\}}} \quad (4.7)$$

$$\log Q(\theta|\theta^t) = \sum_{i=1}^n \sum_{j=1}^k \mathbb{1}_{\{Z_i = j\}} (\log w_j + \log f(x_i|\theta_j)) \quad (4.8)$$

$$\log Q(\theta|\theta^t) = \sum_{i=1}^n \sum_{j=1}^k E(\mathbb{1}_{\{Z_i = j\}}) (\log w_j + \log f(x_i|\theta_j)) \quad (4.9)$$

$$Q = \sum_{i=1}^n \sum_{j=1}^k \gamma_{ij} (\log w_j + \log f(x_i|\theta_j)) \quad (4.10)$$

Nesta expressão, γ_{ij} simboliza a probabilidade a posteriori de que o ponto de dados x_i pertença à componente j da mistura. Essa probabilidade é calculada na etapa E do algoritmo EM e representa a “responsabilidade” da componente j para o ponto de dados x_i . w_j é a probabilidade a priori de que um ponto de dados arbitrário pertença à componente j da mistura. $f(x_i|\theta_j)$ é a densidade de probabilidade do ponto de dados x_i dada a componente j e os parâmetros do modelo θ_j .

Ao calcular a função Q, estamos efetivamente estimando o valor esperado do logaritmo da verossimilhança completa sob a distribuição a posteriori das variáveis latentes. A maximização dessa função Q na etapa M do algoritmo EM nos ajuda a atualizar os parâmetros do modelo θ para uma estimativa melhorada, movendo-se em direção à maximização da verossimilhança.

Ao iterar entre as etapas E e M, buscamos encontrar os parâmetros θ que maximizam a função Q, resultando em estimativas máximas da verossimilhança. Esse processo iterativo continua até que ocorra convergência, ou seja, quando as estimativas param de mudar consideravelmente entre as iterações.

A função LL, que é a soma dos logaritmos das verossimilhanças de cada ponto de dados sob o modelo atual, também é uma ferramenta importante para monitorar a convergência do EM. Assim como a função Q, se a LL estabilizar entre as iterações, isso pode indicar a convergência do algoritmo. Vale lembrar que nem a função LL nem a função Q podem diminuir ao longo das iterações, fornecendo assim uma forma de verificar a correta implementação do algoritmo. No entanto, em alguns casos, essas funções podem apresentar um comportamento diferente. Isso pode ocorrer devido a erros numéricos, erros de arredondamento ou quando a semente inicial que é submetida ao EM já está muito próxima de um ótimo local, sendo o último caso de ocorrência bastante comum.

Embora seja uma medida que reflita a qualidade do MOTIF em si, e não o ajuste do modelo aos dados, conforme vimos no Capítulo 2, o IC também pode ser empregado no acompanhamento

da convergência do EM. Observar o IC ao longo das iterações pode dar uma visão do quanto o MOTIF está se tornando mais definido e, possivelmente, mais biologicamente relevante. Porém, é importante destacar que o IC não é uma medida de convergência do EM, sendo possível ser utilizado somente em casos específicos, como na medição da entropia de alinhamentos múltiplos locais. Cada uma dessas medidas tem um papel importante no processo de descoberta de MOTIFS e fornece uma perspectiva diferente sobre o problema.

Um último ponto a se destacar é a forma como o modelo α é inicializado. Inicializar o modelo α através de uma distribuição de DIRICHLET tem suas vantagens e desvantagens. Entre as vantagens, está o fato de que a distribuição de DIRICHLET é a distribuição a priori conjugada para a distribuição multinomial, o que significa que a posteriori também será uma DIRICHLET. Isso simplifica muito os cálculos e a atualização dos parâmetros em modelos BAYESIANOS, por exemplo. Além disso, a distribuição de DIRICHLET pode modelar diversas situações, devido à sua flexibilidade em termos de formato, variando de uniforme a concentrada em pontos específicos. Mas, talvez a maior vantagem seja a simplicidade. Usar uma distribuição de DIRICHLET para inicializar α é uma ação direta.

No entanto, a principal desvantagem de usar a DIRICHLET para inicializar α é que esta é uma abordagem completamente não informativa, o que pode não ser ideal se tivermos alguma informação prévia sobre a localização dos MOTIFS. Se tivéssemos alguma indicação ou conhecimento prévio sobre regiões promissoras do espaço de busca, seria possível melhorar potencialmente a eficiência do algoritmo EM, fazendo com que ele convergisse mais rapidamente para uma solução ideal. A escolha de uma inicialização não informativa pode levar a uma convergência mais lenta e, em muitos casos, o algoritmo pode ficar preso em ótimos locais. Portanto, a inclusão de informações prévias, quando disponíveis, pode ser uma estratégia importante para melhorar o desempenho do algoritmo EM.

4.2.2.2 ZOOPS

Representado pela Figura 4.8(b), este modelo é uma extensão do OOPS e assume que cada sequência pode conter zero ou uma ocorrência do MOTIF de interesse. Isso proporciona maior flexibilidade, pois permite que algumas sequências não apresentem o padrão conservado. O ZOOPS é adequado para situações onde há variabilidade na presença do MOTIF entre as sequências. Além disso, ZOOPS é frequentemente útil em estudos de CHIP-SEQ, onde um fator de transcrição pode se ligar a zero ou uma vez em cada região do genoma capturada. A implementação do modelo ZOOPS é delineada no Algoritmo 12.

Na linha 1, assim como no OOPS, o algoritmo ZOOPS começa inicializando a matriz α amostrando-a de uma distribuição de DIRICHLET. Esta matriz é usada para representar o modelo α através de uma matriz PWM, com a distribuição de DIRICHLET sendo uma distribuição de probabilidade multivariada utilizada para representar incertezas em proporções. Seguindo para a Linha 2, a variável λ é inicializada ao ser amostrada de uma distribuição BETA.

A partir da linha 3, o algoritmo entra no LOOP principal, que continuará até que um critério de convergência seja atingido. Dentro desse LOOP, para cada sequência i no conjunto de dados (linhas 4-5), na linha 7, a probabilidade a posteriori é computada. Essa probabilidade é calculada levando em consideração todas as m posições, assumindo que elas podem ou não possuir o MOTIF conservados. Essa linha caracteriza a principal diferença entre o modelo ZOOPS e o modelo OOPS apresentado anteriormente.

Na linha 9, o algoritmo atualiza a estimativa para λ somando os valores das probabilidades posteriores de todas as subsequências. Na linha 10, a matriz α é atualizada para refletir as novas estimativas dadas as sequências e suas probabilidades posteriores. As linhas 12 e 13 marcam

Algoritmo 12 ZOOPS

```

1:  $\alpha_i \sim \text{DIRICHLET}(a_1, a_2, a_3, a_4)$ 
2:  $\lambda \sim \text{BETA}(a, b)$ 
3: while true do
4:   for  $i = 1$  to  $n$  do
5:      $seq_i = fasta_i$ 
6:     for  $j = 1$  to  $m$  do
7:        $z_j = P(z_j = 1 | seq_i) = \frac{p(seq_i | z_j=1)\lambda}{\lambda \sum_{l=1}^m p(seq_i | z_l=1) + (1-\lambda) \sum_{l=1}^m p(seq_i | z_l=0)}$  // E-step.
8:     end for
9:      $\hat{\lambda} = \sum_{j=1}^m z_j$ 
10:     $\hat{\alpha} = \text{update}(\hat{\alpha}, seq_i, z)$  // M-step.
11:  end for
12:   $\alpha = \frac{\hat{\alpha}}{\sum_{l=1}^4 \hat{\alpha}_l}$  // Normalização.
13:   $\lambda = \frac{\hat{\lambda}}{n}$  // Normalização.
14:  if  $|Q^{(t)} - Q^{(t-1)}| \leq \epsilon$  then
15:    BREAK // Verifica convergência.
16:  end if
17: end while
18: return  $\alpha, \lambda$ 

```

o início do passo de normalização. Aqui, a matriz α e a variável λ são normalizadas para que sejam representações válidas de uma distribuição de probabilidade.

Finalmente, na linha 13, o algoritmo verifica se o critério de convergência foi atingido. Se isso for verdadeiro (o valor absoluto da diferença entre as log-verossimilhanças de duas iterações consecutivas é menor ou igual a um pequeno valor ϵ), o algoritmo interrompe o LOOP. O algoritmo retorna a matriz α e a variável λ como resultado na Linha 18, ambas devem fornecer um modelo ajustado aos dados. É interessante destacar a linha 10, na qual a função UPDATE é definida. Ela é responsável por atualizar o modelo α após o cálculo das probabilidades a posteriori de cada sequência. O Algoritmo 13 mostra como essa função é implementada.

Algoritmo 13 UPDATE

```

1: for  $j = 1$  to  $m$  do
2:    $kmer = \text{substr}(seq_i, j, j + k - 1)$ 
3:   for  $l = 1$  to  $k$  do
4:      $c = kmer[l]$ 
5:      $\hat{\alpha}[l, c] = \hat{\alpha}[l, c] + z_j^{(t)}$ 
6:   end for
7: end for
8: return  $(\hat{\alpha})$ 

```

O algoritmo começa com um LOOP na linha 1 que percorre todas as subsequências possíveis (KMERS) na sequência de entrada. A seguir, o algoritmo executa um segundo LOOP (linha 3), que percorre cada posição na sub-sequência atual. Para cada posição, ele obtém o caractere atual c (linha 4) e atualiza o elemento correspondente na matriz $\hat{\alpha}$ (linha 5), adicionando a probabilidade posterior da sub-sequência previamente computada.

Este procedimento é realizado para todas as posições dentro do k MER, e para todos os k MERS dentro da sequência. Assim, a matriz $\hat{\alpha}$ é incrementada pelas probabilidades posteriores de todos os k MERS, ponderadas pela presença dos respectivos caracteres na posição correspondente. Finalmente, a matriz $\hat{\alpha}$ é retornada (linha 8). Este processo é fundamental para o algoritmo ZOOPS, pois permite reestimar os parâmetros do modelo com base nas probabilidades posteriores das subsequências. A Figura 4.10 mostra o resumo gráfico deste algoritmo.

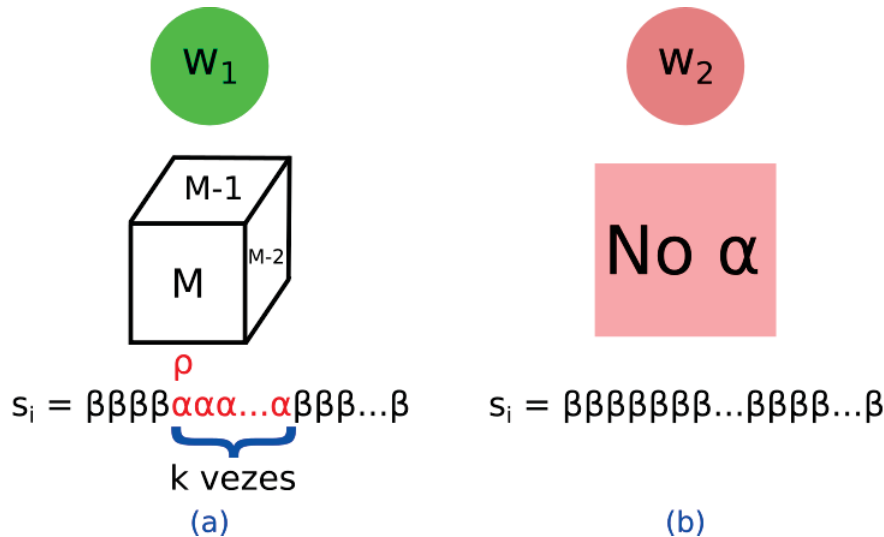


Figura 4.10: Representação gráfica do modelo ZOOPS. Na parte superior temos duas “moedas” simbolizando as probabilidades w_1 e $w_2 = 1 - w_1$. Essas probabilidades representam a plausibilidade de cada modelo (oops e zops) e são parâmetros que, além do modelo α , também devem ser estimados pelo algoritmo. No painel (a) temos o conhecido modelo oops e no painel (b) o modelo zops: *Zero Occurrences per Sequence*. Fonte: o autor.

4.2.2.3 ANR

Semelhante ao OOPS, o modelo ANR também apresenta duas variações distintas: as versões *soft* e *hard*. Apesar desses algoritmos compartilharem uma estrutura comum em sua formulação, eles se distinguem principalmente na maneira como atualizam suas respectivas matrizes durante o processo de otimização. A principal diferença reside na abordagem que cada um adota ao lidar com a probabilidade posterior de um k MER pertencer a uma sequência. Na versão *SOFT*, todos os k MERS influenciam na atualização do modelo, mas a extensão dessa influência é determinada pela sua probabilidade posterior.

Representado pela Figura 4.8(c), este é o modelo mais flexível dos três, pois permite que um MOTIF ocorra qualquer número de vezes dentro de uma sequência. O ANR é aplicável em situações onde se espera que o fragmento conservado possa ocorrer várias vezes dentro de uma mesma sequência, como em determinados processos de regulação genética. Este modelo é especialmente útil em estudos de CHIP-SEQ onde um fator de transcrição pode se ligar a múltiplos locais em uma única região genômica. Além disso, em análises de RNA-SEQ ou 3'-seq, onde podem existir múltiplos sítios que influenciam a estabilidade do RNA, a tradução, ou o local de corte e poliadenilação, o ANR pode ser o modelo mais adequado. O processo de implementação do algoritmo ANR é descrito em detalhes no Algoritmo 14.

O algoritmo ANR começa pela inicialização de α_i de acordo com uma distribuição de DIRICHLET (linha 1) e também inicializa o peso w_α através de uma distribuição BETA (linha 2), que representa o peso inicial do modelo positivo. Em seguida, entra-se em um LOOP principal (linha 3) que prossegue enquanto a diferença absoluta entre $Q^{(t)}$ e $Q^{(t-1)}$ for maior do que um

Algoritmo 14 ANR

```

1:  $\alpha_i \sim \text{DIRICHLET}(a_1, a_2, \dots, a_4)$ 
2:  $w_\alpha \sim \text{BETA}(a, b)$ 
3: while True do
4:    $n_\alpha = 0$ 
5:    $\hat{\alpha} = \text{zeros}(4, k)$ 
6:    $\gamma_i = \text{zeros}(m)$ 
7:   for  $i = 1$  to  $n$  do
8:      $\text{seq}_i = \text{seqs}[i]$ 
9:     for  $j = 1$  to  $m$  do
10:       $\text{kmer}_j = \text{seq}_i[j \dots j + k - 1]$ 
11:       $\gamma_{ij}^{(t)} = \frac{w_\alpha \text{Pr}(\text{kmer}_j | \alpha)}{w_\alpha \text{Pr}(\text{kmer}_j | \alpha) + (1 - w_\alpha) \text{Pr}(\text{kmer}_j | \beta)}$ 
12:       $n_\alpha = n_\alpha + \gamma_{ij}^{(t)}$ 
13:      for  $c = 1$  to  $k$  do
14:         $l = \text{kmer}_j[c]$ 
15:         $\hat{\alpha}[l, c] = \hat{\alpha}[l, c] + \gamma_{ij}^{(t)}$ 
16:      end for
17:    end for
18:  end for
19:   $w_\alpha = \frac{n_\alpha}{nm}$ 
20:   $\alpha = \frac{\hat{\alpha}}{n_\alpha}$ 
21:  if  $|\mathcal{Q}^{(t)} - \mathcal{Q}^{(t-1)}| \leq \epsilon$  then
22:    BREAK
23:  end if
24: end while
25: return ( $\alpha$ )

```

pequeno valor ϵ (linhas 21-23). Dentro deste LOOP, $\hat{\alpha}$ é inicializado como uma matriz de zeros de tamanho 4 por k (linha 5), e γ_i é inicializado como um vetor de zeros de tamanho m (linha 6). Para cada sequência (linhas 6-16), o algoritmo percorre cada k -mer na sequência (linhas 7-18). O vetor γ_{ij} é atualizado (linha 11) e para cada posição no k -mer (linhas 13-16), a entrada correspondente em $\hat{\alpha}$ é atualizada (linha 15).

Após o LOOP para todas as sequências, o algoritmo atualiza o peso w_1 como a proporção de n_1 pelo total de sequências (linha 19), e re-estima α dividindo $\hat{\alpha}$ por n_1 (linha 20). Finalmente, se a condição de convergência for atendida (linhas 21-23), o LOOP é interrompido (linha 22), e o algoritmo retorna a matriz α (linha 25). A Figura 4.11 ilustra visualmente o comportamento do algoritmo ANR.

É relevante enfatizar que, independentemente da sua variante, o modelo ANR distingue-se fundamentalmente dos modelos OOPS e ZOOPS pela metodologia adotada para a reestimação de parâmetros. Enquanto os modelos OOPS e ZOOPS incorporam a totalidade da sequência em seus processos de reestimação, o modelo ANR segue uma estratégia alternativa, direcionando sua atenção exclusivamente aos KMERS e construindo para cada um deles uma distribuição a posteriori específica. Adicionalmente, o ANR não se beneficia de uma suposição a priori acerca da distribuição dos MOTIFS. Desta maneira, ele não faz pressuposições prévias a respeito da frequência de ocorrência de um fragmento em cada sequência. Isso também é conhecido na literatura por TCM, do inglês *Two Component of Mixtures*. Embora essa característica torne o

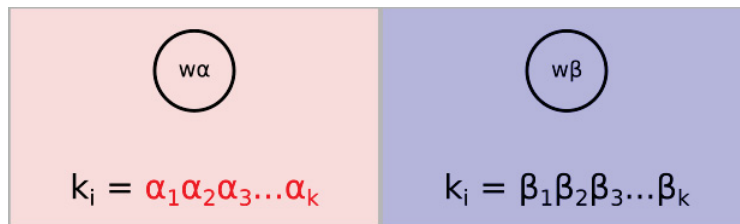


Figura 4.11: Representação gráfica do modelo ANR. Na parte superior da figura temos w_α e w_β que simbolizam as probabilidades a priori das componentes da mistura, sendo que ρ_α (retângulo vermelho) representa a distribuição de interesse e ρ_β (retângulo azul) a distribuição de controle. Ambas devem ser estimadas através do algoritmo. (a) Modelo positivo que representa a distribuição de interesse α que também deve ser estimada. (b) Modelo negativo que representa a distribuição β , construída a partir da extração de sequências genômicas sem o fragmento de interesse ou através embaralhamento do conjunto de dados principal. Fonte: o autor.

algoritmo mais versátil, vale observar que, quando uma distribuição de MOTIFS é conhecida a priori, os algoritmos oops e zoops são, em geral, capazes de alcançar escores superiores.

Além disso, os modelos oops e zoops empregam o Algoritmo 13, que realiza uma atualização suavizada dos parâmetros, chamada de SOFT-EM. Isso torna o modelo mais complexo e mais difícil de interpretar, pois estamos permitindo que vários locais contribuam para um único MOTIF. Desta forma, as abordagens oops e zoops deste trabalho são capazes de representar múltiplos MOTIFS por sequência (MOPS, do inglês *Multiple Motifs per Sequence*) incluindo o zero no caso do zoops.

4.2.2.4 Otimização das sementes iniciais

Nesta etapa, as matrizes PWM pré-otimizadas são aprimoradas através do algoritmo FAST-EM, um desenvolvimento que emergiu como uma extensão do nosso esforço de pesquisa, conforme documentado em Garbelini et al. (2022a,b). Este algoritmo foi especialmente projetado para lidar com o desafio de processar grandes volumes de dados, uma necessidade premente na identificação de MOTIFS biologicamente relevantes com o BIOMAPP::CHIP. A eficiência e a precisão introduzidas pelo FAST-EM refletem o compromisso contínuo com a melhoria das técnicas de bioinformática, conforme ilustrado na Figura 4.12, que detalha os passos elaborados nesta fase.

FAST-EM O FAST-EM representa uma otimização significativa sobre os modelos tradicionais oops e zoops. Esta modificação acelera o tempo de execução e também aprimora a eficiência na manipulação de conjuntos de dados mais amplos. Em outras palavras, FAST-EM potencializa as capacidades dos modelos originais, tornando-os mais adaptáveis e robustos quando confrontados com grandes volumes de informações.

O cálculo das probabilidades marginais representa uma das fases mais onerosas em termos computacionais no algoritmo EM. Este estágio exige a computação da probabilidade de um MOTIF ser encontrado em cada uma das $n \times (t - k + 1)$ posições válidas, dadas as n sequências de entrada. Esse processo pode tornar-se um ponto de estrangulamento intenso, especialmente em experimentos CHIP-SEQ, onde frequentemente se lida com um volume elevado de sequências. Isso, por sua vez, faz com que o valor de n seja consideravelmente alto. Embora seja possível mitigar esse desafio computacional utilizando apenas um subconjunto das sequências como amostra, essa abordagem compromete o poder preditivo do modelo.

Para entender seu funcionamento, precisamos recapitular alguns conceitos. A regra de BAYES para o algoritmo EM diz o seguinte: $P(p_j | s_i) = \frac{P(s_i | p_j) \times P(p_j)}{P(s_i)}$, que pode ser interpretada

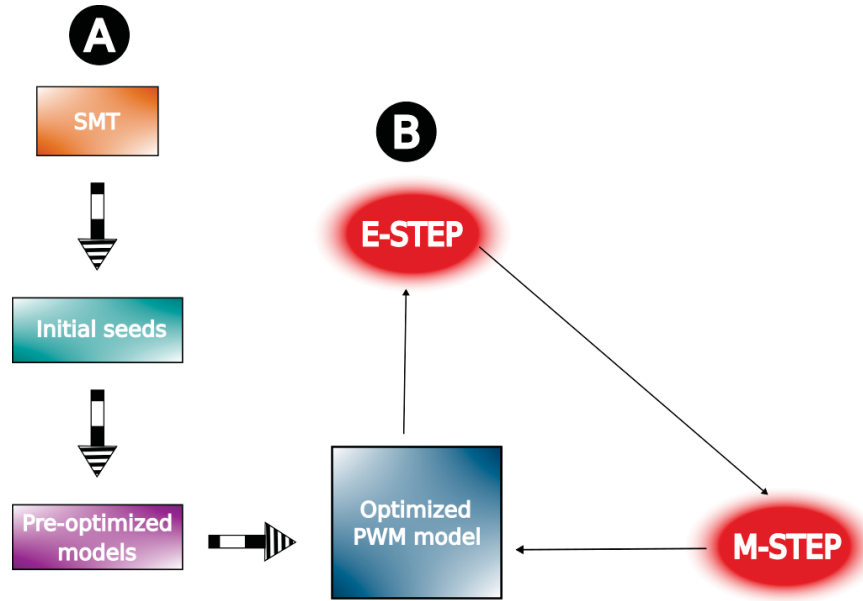


Figura 4.12: Pipeline do algoritmo de otimização do BIOMAPP-CHIP. Na etapa (A), as sementes iniciais fornecidas pela SMT são transformadas em modelos iniciais pré-otimizados que por sua vez são submetidos na etapa (B) ao EM para serem aperfeiçoados. Fonte: o autor.

como: “probabilidade de existir um MOTIF na posição p_j dado que estamos observando a sequência i , dividido pela probabilidade marginal de s_i ”. A probabilidade marginal pode ser escrita como a soma ponderada: $p(s_i) = P(s_i|p_1)P(p_1) + \dots + p(s_i|p_m)P(p_m)$. Se considerarmos que um MOTIF pode estar em qualquer posição com a mesma probabilidade, então essa soma pode ser simplificada para: $p(s_i) = P(s_i|p_1) + \dots + p(s_i|p_m)$.

Para calcular cada termo da probabilidade marginal, é necessário empregar tanto o modelo positivo (α) quanto o modelo negativo (β) através da seguinte equação: $P(s_i|p_u) = \prod_{j=1}^u P(s_{ij}|\beta) \prod_{j=u+1}^{u+k} P(s_{ij}|\alpha) \prod_{j=u+k+1}^m P(s_{ij}|\beta)$, para $1 \leq u \leq m$. Em outras palavras, esta equação precisa ser executada para todas as m posições válidas em cada sequência. O algoritmo FAST-EM faz isso computando uma única vez a $P(s_i|\beta) = \prod_{j=1}^m P(s_{ij}|\beta)$. Depois, para cada $1 \leq u \leq m$, este valor é dividido por $P(p_u|\beta) = \prod_{j=u}^{u+k} P(s_{ij}|\beta)$ e multiplicado por $P(p_u|\alpha) = \prod_{j=u}^{u+k} P(s_{ij}|\alpha)$. Esta simples otimização, quando combinada com técnicas de MULTITHREADING, torna o algoritmo FAST-EM substancialmente mais rápido e, portanto, mais adequado para manipular grandes volumes de dados.

4.3 CONCLUSÃO

Este capítulo apresentou uma descrição abrangente dos elementos constituintes do arcabouço BIOMAPP::CHIP, fornecendo detalhes sobre cada componente e suas respectivas funcionalidades. Abordamos a estrutura da SMT e sua aplicação em algoritmos distintos como KDIVE, K-SEARCH, IUPAC-SEARCH, KHMAP e FAST-EM. Além disso, este capítulo elucidou a implementação do algoritmo EM e suas variantes – OOPS, ZOOPS e ANR –, demonstrando a flexibilidade e adaptabilidade da estrutura em diferentes cenários de pesquisa. Também foi destacada a importância da realização de testes de hipótese para avaliar a significância estatística dos KMERS. Em conjunto, os métodos e mecanismos aqui descritos formam uma estrutura coerente e robusta para a análise de MOTIFS biológicos, oferecendo um rol de ferramentas que auxiliam na solução de problemas complexos na intersecção entre biologia e ciência da computação.

5 EXPERIMENTOS E RESULTADOS

Neste capítulo, apresentaremos os experimentos realizados e os resultados obtidos, organizados em três seções principais, cada uma abordando aspectos importantes da nossa pesquisa. A primeira seção, RESPONDENDO A HIPÓTESE INICIAL, foi dedicada a explorar os resultados que apoiam a hipótese inicial levantada no início deste trabalho. Esta análise crítica visa estabelecer uma estrutura estável para a validação de nossa proposta teórica. A segunda seção, intitulada SMT, foca nos resultados obtidos pelo algoritmo SMT. Aqui, comparamos o desempenho desta abordagem com outros métodos de referência do estado da arte para a contagem de k-mers, proporcionando uma avaliação detalhada de sua performance. Por fim, na terceira seção, BIOMAPP::CHIP, apresentamos uma análise comparativa do arcabouço BIOMAPP::CHIP em relação a outros algoritmos do estado da arte na descoberta de MOTIFS. Esta seção visa demonstrar a aplicabilidade e a superioridade do BIOMAPP::CHIP em contextos práticos de bioinformática. Com este capítulo, buscamos validar nossa hipótese inicial e também posicionar nossas contribuições em relação às soluções existentes, destacando nossos avanços na área.

5.1 RESPONDENDO A HIPÓTESE INICIAL

Nesta seção, abordaremos a validação de nossa hipótese teórica com base em experimentos detalhados. Inicialmente, foram gerados 1995 datasets sintéticos e seus respectivos modelos de referência variando os seguintes parâmetros: número de bases (n), tamanho do k-mer (k) e nível de mutação (c). Estes datasets, criados com distribuição de fundo uniforme, variaram de 1 a 10 milhões de bases e incorporaram MOTIFS de referência via matrizes PWM.

Aplicamos quatro modelos - RANDOM, RANDOM+EM, SMT, SMT+EM - e os comparamos com o modelo de referência através da distância euclidiana. O foco desses experimentos foi demonstrar que a contagem de k-mers fornece um ponto de partida no qual modelos sub-ótimos podem ser criados. Além disso, evidenciamos que tais modelos se aproximam de ótimos locais, potencialmente ótimos globais, quando submetidos a algoritmos de otimização exata, como a maximização da expectativa, alcançando uma aproximação significativa dos modelos de referência empregados.

O modelo RANDOM cria matrizes PWM utilizando uma distribuição de DIRICHLET com parâmetros de concentração ($\alpha = 1, 1, 1, 1$). O modelo RANDOM+EM otimiza a matriz PWM gerada pelo RANDOM através do algoritmo EM. O modelo SMT realiza a contagem eficiente de k-mers para identificar os mais frequentes, empregando posteriormente o KDIVE para encontrar k-mers com até 3 mutações a partir do mais frequente, formando assim a base do modelo pré-otimizado. Finalmente, o modelo SMT+EM utiliza o FAST-EM, para refinar ainda mais o modelo SMT anterior, buscando se aproximar ainda mais dos modelos de referência.

A utilização de 3 mutações é uma decisão que se equilibra entre a sensibilidade e especificidade do modelo e a complexidade computacional. Esta escolha permite detectar variações importantes nos k-mers mais frequentes, mantendo uma boa especificidade, essencial para evitar a identificação de MOTIFS que não sejam biologicamente relevantes. Do ponto de vista biológico, pequenas variações como 3 mutações costumam ser suficientes para capturar a diversidade funcional dos fragmentos, sendo que variações maiores podem não ser tão relevantes ou representar eventos evolutivos diferentes.

A Figura 5.1 exibe um gráfico do tipo BOXPLOT que ilustra a distância euclidiana desses modelos em relação ao modelo de referência. No gráfico, a medida de tendência central para

cada algoritmo é indicada pela linha horizontal dentro das caixas, que representa a mediana das distâncias. O comprimento das caixas, que se estende do primeiro quartil (Q1) ao terceiro quartil (Q3), reflete o intervalo interquartil (IIQ), dando uma noção da dispersão dos dados em torno da mediana. As linhas verticais que se estendem das caixas, conhecidas como “bigodes”, delimitam o alcance típico dos dados, e os pontos pretos representam valores discrepantes (*outliers*), que estão fora do alcance dos bigodes.

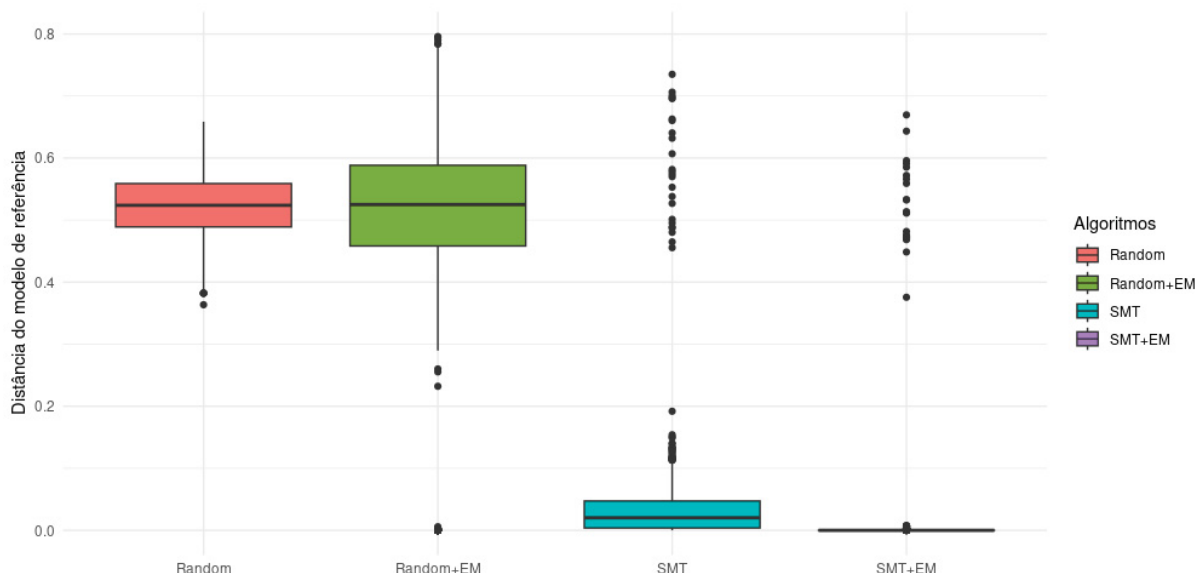


Figura 5.1: Comparação do desempenho dos modelos RANDOM, RANDOM+EM, SMT e SMT+EM em relação ao modelo de referência. Cada BOXPLOT representa a distribuição das distâncias obtidas por cada algoritmo, com a linha central da caixa indicando a mediana. Os bigodes estendem-se até o ponto mais distante dentro do intervalo interquartil, e os pontos representam outliers. Os resultados evidenciam a superioridade dos modelos SMT e SMT+EM, com medianas significativamente mais baixas e menor dispersão em comparação aos modelos RANDOM e RANDOM+EM, indicando uma maior precisão e consistência na aproximação do modelo de referência. Fonte: o autor.

O algoritmo RANDOM atua como um controle, com a maior parte de suas medidas centradas em torno de 0,5. Isso está em consonância com o comportamento antecipado para execuções aleatórias, onde não se espera uma aproximação sistemática do modelo de referência. O algoritmo RANDOM+EM demonstra uma distribuição de resultados com uma quantidade considerável de *outliers*, especialmente abaixo da mediana, o que pode ser interpretado como um sinal de que o método EM é capaz de otimizar a distância em determinados casos, mesmo partindo de condições iniciais aleatórias. Esses pontos extremos, mais próximos do modelo de referência, sugerem que o EM tem o potencial de refinar significativamente os resultados em situações específicas.

Os modelos SMT e SMT+EM representam um avanço qualitativo evidente quando comparados ao modelo RANDOM e RANDOM+EM. O modelo SMT por si só já demonstra um perfil de desempenho superior, com uma mediana substancialmente mais baixa, indicando uma eficácia geral na minimização da distância até o modelo de referência. No entanto, é com a incorporação do EM que observamos a menor mediana entre todos os métodos analisados, sugerindo que este algoritmo mantém a consistência do modelo SMT e também se beneficia da otimização proporcionada pelo EM.

Essa constatação é reforçada pelo número reduzido de *outliers* e pela concentração de resultados em torno de valores baixos, ilustrando que o modelo SMT+EM consegue aprimorar a

tendência central em direção ao modelo de referência e também aumenta a confiabilidade do algoritmo.

A Figura 5.2 apresenta uma série de gráficos de dispersão que analisam a precisão dos quatro modelos em relação ao modelo de referência, considerando diferentes taxas de mutação implantadas nas sequências. Estas taxas de mutação, representadas pelos níveis 0, 1, 0, 15, 0, 2, 0, 5, 0, 3, refletem a frequência com que os MOTIFS são alterados. O tamanho do MOTIF é denotado pela variável k , enquanto o eixo x exibe a quantidade de bases dos datasets, refletindo o volume de dados processados.



Figura 5.2: Distribuição da distância ao modelo de referência para os modelos RANDOM, RANDOM+EM, SMT e SMT+EM em relação à quantidade de bases, segmentada por níveis de mutação (0.1, 0.15, 0.2, 0.25, 0.3) e tamanho de k (indicado pela coloração dos pontos, variando de azul para vermelho conforme o incremento do tamanho de k). Cada subfigura corresponde a um nível de mutação específico, demonstrando como cada algoritmo se comporta em diferentes contextos de complexidade e perturbação. Os modelos SMT e SMT+EM mostram uma tendência de menores distâncias ao modelo de referência em todos os níveis de mutação. A partir do nível de mutação de 0.2, observa-se uma elevação na dispersão para conjuntos de dados menores e valores de k maiores, refletindo o aumento na dificuldade de otimização sob essas condições. Fonte: o autor.

O modelo RANDOM, mostra resultados condizentes com as expectativas teóricas, congregando a maior parte das suas medições em torno da mediana 0,5. Este comportamento reforça sua função de controle, demonstrando uma distribuição aleatória de distâncias em relação ao modelo de referência.

O modelo RANDOM+EM manteve uma grande concentração de pontos em torno do valor mediano de 0,5, mas também apresentou uma série de resultados que se distanciam para a extremidade inferior do gráfico. Estes pontos sugerem que o método EM foi capaz de otimizar a solução a partir de condições aleatórias, alcançando uma proximidade maior com o modelo de referência em determinadas instâncias.

Em contraste, os modelos SMT e SMT+EM exibiram uma performance substancialmente mais próxima do modelo de referência, com a maioria dos pontos aglomerados próximos ao valor zero no eixo y . Este resultado evidencia a eficiência destes métodos em identificar e preservar os MOTIFS corretos, mesmo diante de perturbações introduzidas pelas taxas de mutação.

No entanto, a partir do nível de mutação 0,2, observa-se uma dispersão de pontos para os níveis superiores do gráfico nos casos de datasets menores e valores maiores de k , particularmente nos pontos coloridos de vermelho. Esta tendência indica que, sob condições de menor volume

de dados, maior complexidade de MOTIFS (k elevado) e maiores taxas de mutações, ambos os algoritmos SMT e SMT+EM podem enfrentar dificuldades, o que já era um resultado esperado. Em conjuntos de dados reduzidos, a resolução acurada de MOTIFS complexos torna-se um desafio crescente, levando a uma maior distância do modelo de referência.

5.1.1 Análise estatística

A avaliação comparativa dos modelos RANDOM, RANDOM+EM, SMT e SMT+EM foi realizada utilizando uma abordagem estatística para identificar diferenças significativas entre as suas performances. Para isso, empregou-se o teste de FRIEDMAN, uma ferramenta não paramétrica robusta para análise de variâncias por ranks em amostras relacionadas. Este teste é particularmente adequado para os casos em que as medidas repetidas estão presentes, permitindo-nos comparar mais de dois tratamentos ou condições sem a necessidade de pressupostos de distribuição normal.

No contexto deste estudo, o ID (que representa o nome do conjunto de dados) foi utilizado para representar blocos de medidas relacionadas, isto é, cada conjunto de dados onde os diferentes algoritmos foram aplicados e avaliados. Assim, o teste de FRIEDMAN pôde comparar as medianas dos algoritmos dentro de cada bloco, proporcionando uma avaliação da consistência dos resultados obtidos por cada método.

Caso diferenças estatisticamente significativas sejam detectadas, procederemos com a análise post-hoc de NEMENYI. Este método é empregado para realizar comparações múltiplas e identificar quais grupos diferem entre si. Para ajustar a taxa de erro do tipo I decorrente das múltiplas comparações, a correção de BONFERRONI será aplicada. Este procedimento é essencial para assegurar a integridade dos resultados estatísticos e a validade das conclusões tiradas a partir dos mesmos.

Os resultados do teste de FRIEDMAN revelaram diferenças estatisticamente significativas entre os algoritmos ($\chi^2 = 4672,8$, $df = 3$, $p\text{-valor} < 2,2e-16$), indicando variações marcantes no desempenho entre eles. Conforme esperado, os modelos SMT e SMT+EM demonstraram ser significativamente superiores aos modelos RANDOM e RANDOM+EM. Este resultado é importante, pois reafirma a eficácia das abordagens SMT e SMT+EM e também destaca a limitação dos modelos aleatórios, que, conforme esperado, não conseguiram se aproximar do desempenho dos modelos mais avançados.

O que é particularmente importante, no entanto, é a diferença significativa observada entre os modelos SMT e SMT+EM, conforme indicado pelos resultados do post-hoc de NEMENYI ($p\text{-valor} < 2e-16$ para todas as comparações). Isso sugere que a técnica de otimização exata aplicada aos modelos pré-otimizados obtidos pela execução da SMT foi eficaz. Em outras palavras, os modelos sub-ótimos produzidos pela SMT estavam posicionados em locais promissores do espaço de busca, permitindo que o algoritmo de otimização melhorasse ainda mais esses modelos, levando-os muito próximos do ótimo global.

Esses resultados são particularmente relevantes no contexto da busca por soluções ótimas ou quase ótimas em espaços de busca complexos. Eles ressaltam a importância de um bom ponto de partida na otimização, onde modelos sub-ótimos, mas bem posicionados, podem ser aprimorados através de técnicas de otimização fina. Portanto, os resultados obtidos validam a superioridade dos modelos SMT e SMT+EM sobre abordagens aleatórias e também enfatizam a eficácia da combinação de uma boa heurística inicial com refinamentos subsequentes por meio de métodos de otimização exatos. Por fim, a Tabela 5.1 apresenta uma comparação detalhada do desempenho dos quatro modelos em relação ao modelo de referência, considerando distintos níveis de mutação. O teste de FRIEDMAN e o subsequente teste post-hoc de NEMENYI foram aplicados para discernir diferenças estatisticamente significativas entre os algoritmos em cada nível de mutação.

Tabela 5.1: Resumo dos resultados do teste de FRIEDMAN e do teste post-hoc de NEMENYI para diferentes níveis de mutação.

NÍVEL DE MUTAÇÃO	χ^2	P-VALOR (FRIEDMAN)	COMPARAÇÃO	P-VALOR (NEMENYI)
0,10	954,31	$< 2, 2e - 16$	RANDOM+EM VS RANDOM SMT VS RANDOM SMT VS RANDOM+EM SMT+EM VS RANDOM SMT+EM VS RANDOM+EM SMT+EM VS SMT	1 $< 2e - 16$ $< 2e - 16$ $< 2e - 16$ $< 2e - 16$ $4, 4e - 14$
0,15	973,58	$< 2, 2e - 16$	RANDOM+EM VS RANDOM SMT VS RANDOM SMT VS RANDOM+EM SMT+EM VS RANDOM SMT+EM VS RANDOM+EM SMT+EM VS SMT	0,61 $< 2e - 16$ $< 2e - 16$ $< 2e - 16$ $< 2e - 16$ $1, 1e - 14$
0,20	960,51	$< 2, 2e - 16$	RANDOM+EM VS RANDOM SMT VS RANDOM SMT VS RANDOM+EM SMT+EM VS RANDOM SMT+EM VS RANDOM+EM SMT+EM VS SMT	0,34 $< 2e - 16$ $< 2e - 16$ $< 2e - 16$ $< 2e - 16$ $< 2e - 16$
0,25	923,48	$< 2, 2e - 16$	RANDOM+EM VS RANDOM SMT VS RANDOM SMT VS RANDOM+EM SMT+EM VS RANDOM SMT+EM VS RANDOM+EM SMT+EM VS SMT	0,57 $< 2e - 16$ $< 2e - 16$ $< 2e - 16$ $< 2e - 16$ $< 2e - 16$
0,30	870,21	$< 2, 2e - 16$	RANDOM+EM VS RANDOM SMT VS RANDOM SMT VS RANDOM+EM SMT+EM VS RANDOM SMT+EM VS RANDOM+EM SMT+EM VS SMT	0,097 $< 2e - 16$ $< 2e - 16$ $< 2e - 16$ $< 2e - 16$ $< 2e - 16$

Os resultados do teste de FRIEDMAN são expressivamente significativos em todos os níveis de mutação, com um p-valor consistentemente inferior a $2, 2e - 16$, o que denota que há uma diferença significativa no desempenho entre os algoritmos. As análises post-hoc de NEMENYI revelam que, enquanto não há diferença significativa entre os modelos RANDOM e RANDOM+EM em nenhuma das taxas de mutação, indicando que o EM tem dificuldade em explorar o espaço de busca a partir da inicialização aleatória, os modelos SMT e SMT+EM mostram-se significativamente superiores aos demais em todos os cenários. Além disso, a abordagem que emprega SMT+EM distingue-se da que emprega somente a SMT, com diferenças significativas em todas as taxas de mutação, indicando que a estratégia de otimização utilizada melhora consistentemente o desempenho.

Baseando-se na hipótese inicial, os dados obtidos a partir dos testes estatísticos de FRIEDMAN e NEMENYI oferecem um suporte robusto à teoria de que a contagem de k-mers é fundamental na formulação de modelos probabilísticos iniciais e na identificação de regiões genéticas com baixo grau de mutações. Os resultados revelam que os modelos encontrados através dos algoritmos SMT e SMT+EM, que são baseados na contagem de k-mers, superam significativamente os modelos RANDOM e RANDOM+EM, estando mais próximos do modelo de referência, conforme evidenciado pelos p-valores do teste post-hoc de NEMENYI.

Além disso, a diferença significativa entre os desempenhos dos modelos SMT e SMT+EM sugere que a etapa de otimização exata, inerente ao SMT+EM, aprimora ainda mais os modelos

iniciais propostos pela SMT. Esta observação corrobora a parte da hipótese que prevê que modelos iniciais bem formulados estarão próximos a ótimos locais e também têm o potencial de aproximar-se do ótimo global quando refinados por um algoritmo de otimização exata. Os modelos SMT e SMT+EM mostraram-se eficientes em identificar regiões promissoras no espaço de busca, o que é um indicativo de que o método de contagem de k-mers fornece uma base concreta para a identificação de sequências genéticas relevantes.

Portanto, os dados analisados corroboram a hipótese teórica apresentada, demonstrando que a contagem de k-mers desempenha um papel importante na formação de modelos probabilísticos iniciais e também que a otimização subsequente desses modelos pode levar a resultados que estão muito próximos do ótimo global. Isso sugere que a abordagem proposta é altamente efetiva para o problema de pesquisa em questão, validando a premissa teórica e contribuindo significativamente para o avanço do conhecimento na área de identificação de regiões genéticas com baixo grau de mutações.

5.2 SMT

Nesta seção, apresentaremos os resultados obtidos a partir da avaliação e análise da SMT, em comparação com os algoritmos do estado da arte JELLYFISH (Marçais e Kingsford, 2011) e KANALYZE (Audano e Vannberg, 2014). Estas comparações foram importantes para validar a performance da SMT no contexto do processamento e análise de dados genômicos. A análise comparativa permitiu corroborar o desempenho superior da SMT em diversos cenários, bem como destacar suas características distintas que contribuem para seu alto desempenho e flexibilidade no manuseio de kmers. Ao longo desta seção, discutiremos os aspectos metodológicos dos testes realizados, assim como as métricas empregadas para a avaliação das estruturas de dados e algoritmos em estudo.

Para avaliar os algoritmos, todos os arquivos .bed do repositório JASPAR com mais de 10.000 sequências foram selecionados, totalizando 131 conjuntos de dados distintos. Os algoritmos foram executados com valores de k variando entre 5 e 30 bases. O monitoramento dos tempos de execução e consumo de RAM foi realizado através da instrução `/usr/bin/time -v`, disponível na maioria dos sistemas operacionais LINUX/UNIX. Todos os testes foram realizados no sistema operacional LINUX UBUNTU 22.04.3 LTS, equipados com processadores AMD EPYC 7B12, 8GB de RAM com clock interno de aproximadamente 2250 GHz. Para fins de reprodutibilidade, as seguintes instruções de linha de comando foram utilizadas.

- SMT: `smt -i <fasta file> -k <size of kmer> -s 500`
- JELLYFISH: `jellyfish count -m <size of kmer> -s 500M`
- KANALYZE: `kanalyze count -k <size of kmer> -f fasta <fasta file>`

A Figura 5.3 exibe o desempenho médio dos algoritmos considerando todos os conjuntos de dados e todos os valores de k. Na parte superior esquerda desta figura, ilustrando o consumo de RAM, observa-se que o algoritmo JELLYFISH requer consideravelmente mais memória em comparação aos outros dois. Especificamente, consome cerca de 1000 Mb, enquanto o KANALYZE consome aproximadamente 250 Mb e a SMT usa cerca de 50 Mb. No gráfico superior direito, que retrata o tempo (em segundos), o KANALYZE apresenta o maior tempo de execução, alcançando cerca de 8 segundos. O JELLYFISH, por sua vez, leva aproximadamente 2 segundos, enquanto que a SMT mostra o menor tempo, próximo a 0,5 segundos.

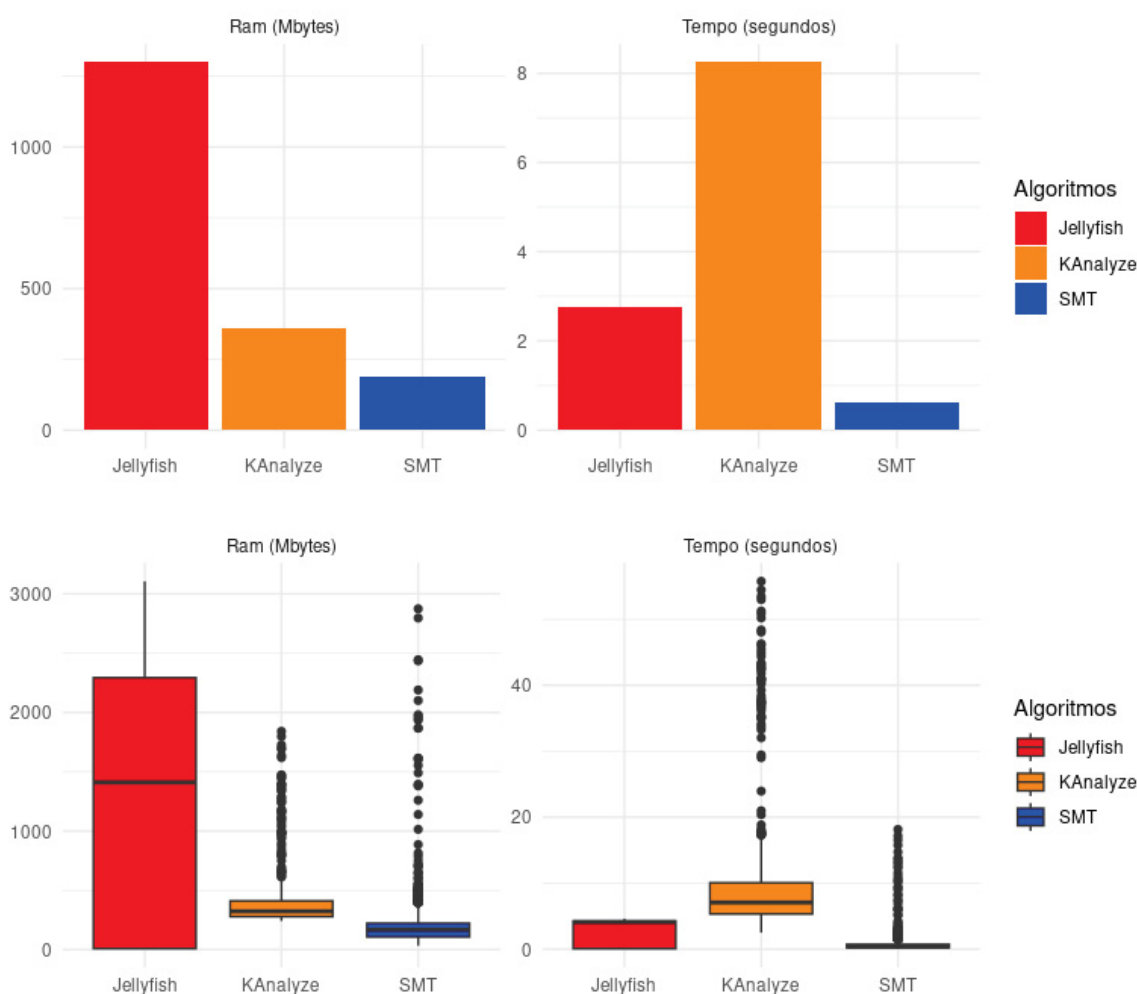


Figura 5.3: Análise comparativa entre os algoritmos SMT, JELLYFISH e KANALYZE em relação ao consumo de RAM (em megabytes) e tempo de execução (em segundos). (a) Consumo médio de RAM. (b) Tempo médio de execução. (c) Distribuição do consumo de RAM, com mediana e outliers. (d) Distribuição do tempo de execução, com mediana e outliers. SMT prova ser mais eficiente em ambos os aspectos em comparação com os outros dois algoritmos. Fonte: o autor.

Prosseguindo com a Figura 5.3, observando o gráfico inferior esquerdo, nota-se que o JELLYFISH tem uma mediana próxima de 1400 Mb com *outliers* excedendo 3000 Mb. O KANALYZE tem sua mediana em torno de 300 Mb, com picos atingindo quase 2000 Mb. Por outro lado, a SMT mantém um consumo menor, com uma mediana próxima de 150 Mb e *outliers* perto de 300 Mb. No gráfico inferior direito, o JELLYFISH mostra baixa dispersão com sua mediana superando 4 segundos. O KANALYZE exibe alta dispersão, com uma mediana próxima de 7 segundos e *outliers* ultrapassando 55 segundos. A SMT continua com o melhor desempenho, com uma mediana próxima de 0,5 segundos e *outliers* aproximando-se de 18 segundos.

A análise do intervalo interquartil (IIQ) para o consumo de RAM dos três algoritmos revela informações importantes sobre sua dispersão. O algoritmo JELLYFISH tem o maior IIQ, com 2284 Mb, indicando uma variação significativa em seu consumo de RAM na metade central dos dados. Em contraste, o KANALYZE e a SMT têm IIQs muito menores, com 133 Mb e 115 Mb, respectivamente. Isso sugere que, enquanto o JELLYFISH apresenta uma dispersão significativa no consumo de RAM, KANALYZE e SMT mostram eficiência neste aspecto, pelo menos na metade

central de suas distribuições. Estas observações complementam a análise do tempo de execução, onde a SMT se destaca com um tempo médio consideravelmente menor em comparação aos outros.

Ao analisar o intervalo interquartil (IIQ) para o tempo de execução dos três algoritmos, podemos extrair informações significativas sobre a variabilidade de cada método. O JELLYFISH apresentou um IIQ de 4,19 segundos, indicando que a metade central de seus tempos de execução varia em torno dessa faixa. O KANALYZE tem um IIQ ligeiramente maior, com 4,73 segundos, sugerindo uma variação um pouco maior em seus tempos centrais em comparação com JELLYFISH. A SMT, por outro lado, exibiu um IIQ de apenas 0,5 segundos, refletindo boa consistência no tempo de execução.

A Figura 5.4 ilustra a relação entre o tamanho do k-mer e duas métricas de desempenho: consumo de RAM e tempo de execução para os algoritmos JELLYFISH, KANALYZE e SMT. Quanto ao consumo de RAM, observamos que o JELLYFISH mostra um aumento do consumo em relação ao tamanho do k-mer, com um aumento acentuado após 20 mers. Surpreendentemente, o KANALYZE exibe uma ligeira diminuição no consumo de RAM em relação ao tamanho do k-mer. Isso pode ter ocorrido devido a alguma técnica de DUMP não documentada que o algoritmo realiza. A SMT, por sua vez, mantém um perfil de consumo quase constante e baixo, independentemente do tamanho do k-mer.

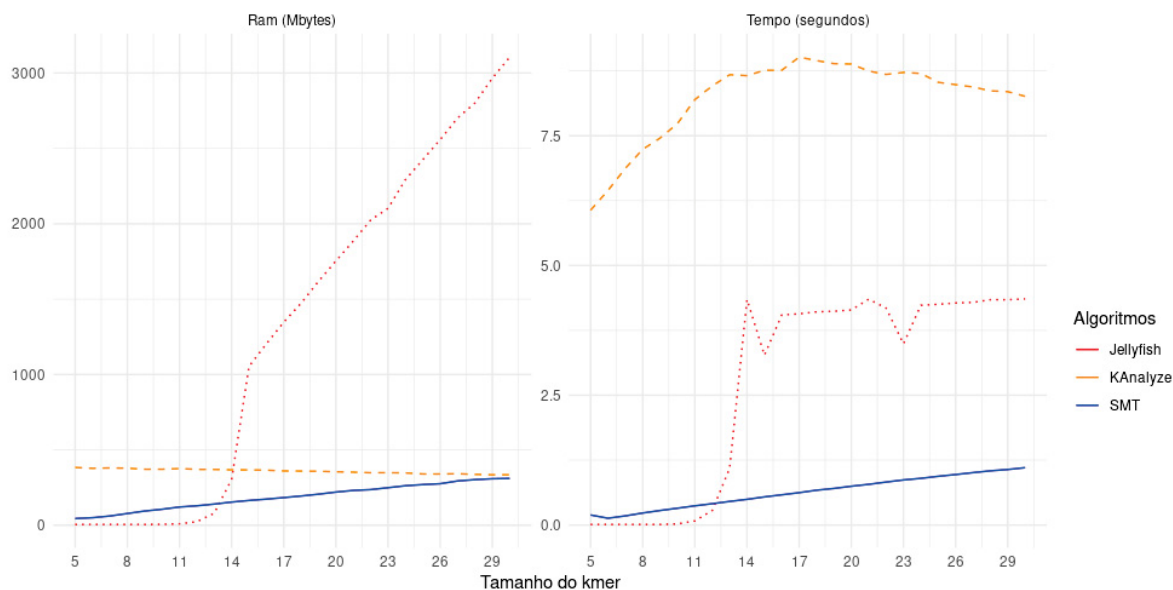


Figura 5.4: Análise comparativa entre os algoritmos SMT, JELLYFISH e KANALYZE em relação ao consumo de RAM (em megabytes) e tempo de execução (em segundos) para vários tamanhos de k. Fonte: o autor.

A Figura 5.5 exibe o comportamento dos algoritmos com relação ao consumo de memória RAM, agrupados por valores de k. De modo geral, observa-se que a SMT tende a ter uma distribuição mais compacta do uso de RAM, enquanto que o JELLYFISH e o KANALYZE exibem variações mais pronunciadas. Em particular, o JELLYFISH mostrou alguns picos no uso de RAM, especialmente nas configurações 8, 12, 15 e 26. Em muitos cenários, ele tende a usar mais RAM do que os outros dois algoritmos.

O KANALYZE parece ter uma variação mais uniforme no uso de RAM em comparação com JELLYFISH, embora ainda mostre algumas oscilações. Em alguns casos (como nas configurações 11, 12 e 15), o uso de RAM do KANALYZE supera o do JELLYFISH e SMT. A SMT frequentemente exibe o menor uso de RAM em quase todas as configurações. No entanto, em algumas situações (como 18 e 29), seu uso de RAM se aproxima ou até supera os outros dois algoritmos.

Algumas configurações, como 7, 13, 21 e 25, mostram uma convergência no uso de RAM entre os três algoritmos, enquanto outras, como 8, 12, 15 e 26, exibem divergências significativas entre eles. A SMT, geralmente apresentou o menor uso de RAM e exibiu uma distribuição mais consistente e compacta, enquanto que o JELLYFISH e o KANALYZE tendem a ser mais voláteis nesta métrica.

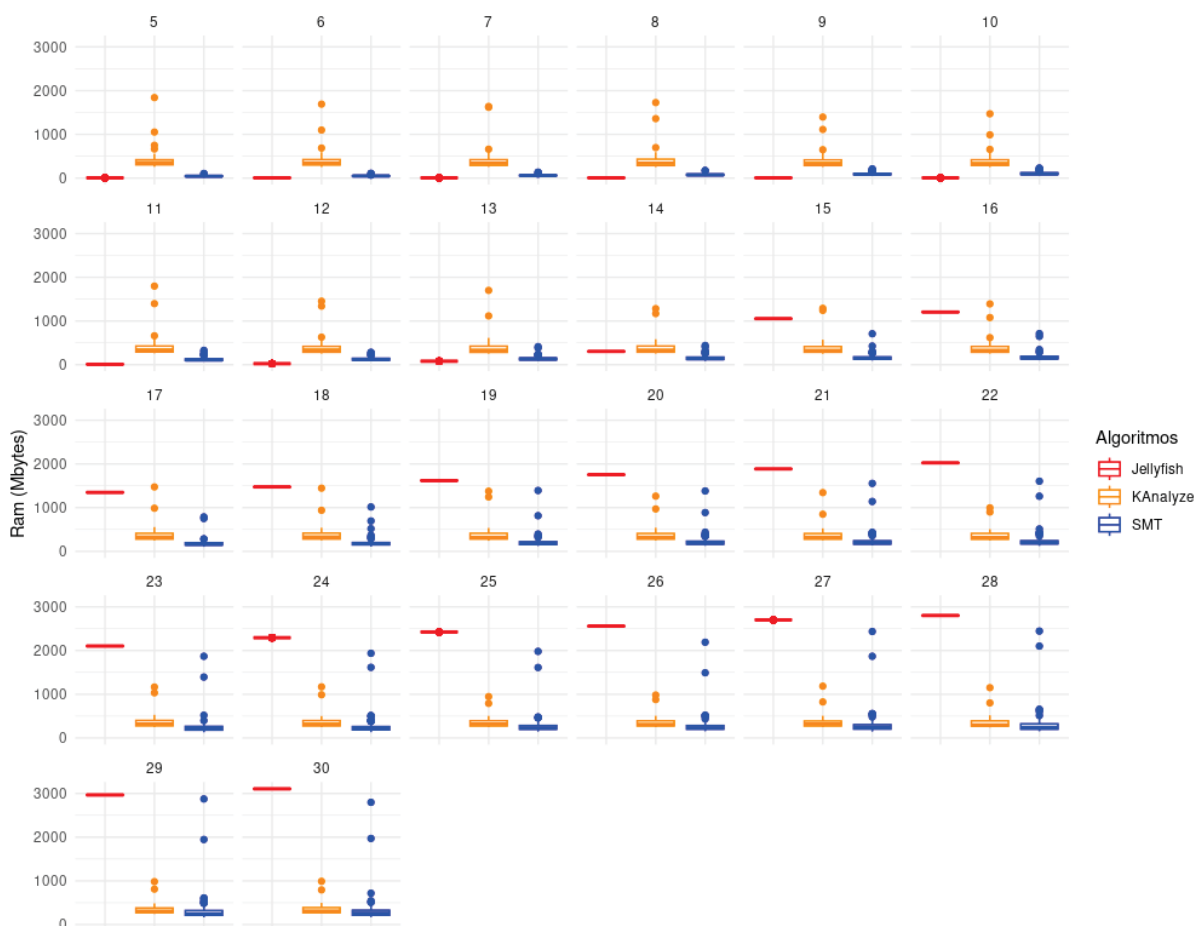


Figura 5.5: Análise comparativa entre os gráficos de boxplot dos algoritmos SMT, JELLYFISH e KANALYZE em relação ao consumo de RAM (em megabytes), agrupados por valores de k . Fonte: o autor.

Por fim, a Figura 5.6 ilustra o desempenho temporal dos algoritmos em diferentes tamanhos de k , variando de 5 a 30. De acordo com esta figura, os algoritmos JELLYFISH e SMT exibem um desempenho temporal mais consistente, com menor variação. Na maioria dos casos, ambos apresentaram tempos de execução abaixo de 10 segundos, com uma ligeira vantagem para o JELLYFISH, já que para alguns valores de k , a SMT exibiu tempos que ultrapassaram 15 segundos. Por outro lado, o KANALYZE apresenta tempos de execução mais elevados, que frequentemente se aproximam ou atingem o limite superior de 40 segundos. Em quase todos os conjuntos de teste, o KANALYZE tem o tempo de execução mais longo entre os três algoritmos.

A Tabela 5.2 fornece uma análise comparativa da média do tempo e consumo de memória RAM entre as três ferramentas: JELLYFISH, KANALYZE e SMT, variando o valor de k de 5 a 30. É evidente que a SMT apresenta um desempenho significativamente melhor em termos de consumo de memória RAM em comparação com os outros algoritmos, mantendo um tempo de execução muito menor em comparação com o JELLYFISH à medida que o tamanho de k aumenta.

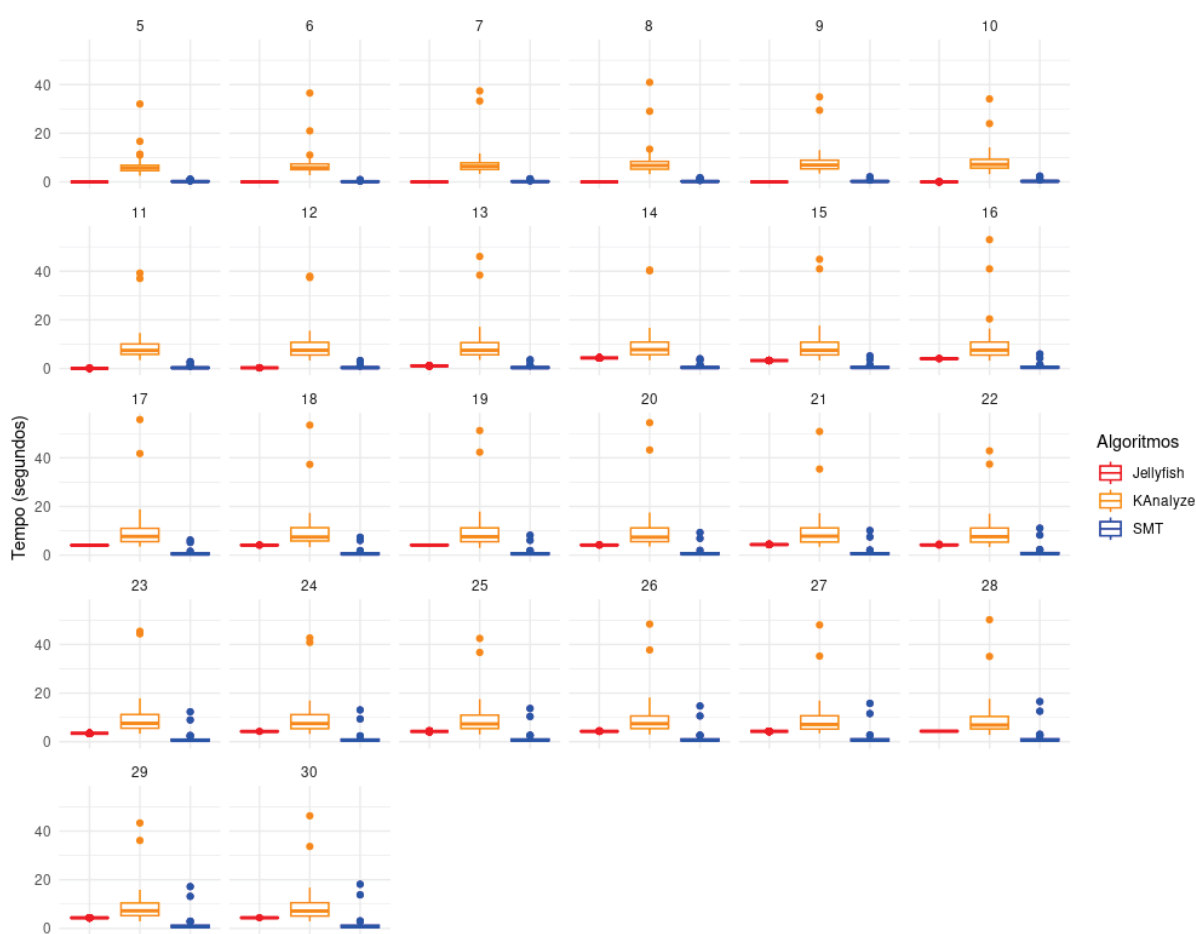


Figura 5.6: Análise comparativa entre os gráficos de boxplot dos algoritmos SMT, JELLYFISH e KANALYZE quanto ao consumo de tempo (em segundos), agrupados por valores de k . Fonte: o autor.

O incremento no tempo de execução e consumo de memória RAM com o aumento de k é mais gradual na SMT, destacando sua eficiência, especialmente em valores mais altos de k .

Além disso, a Tabela 5.2 mostra que, a partir de $k = 13$, o tempo de execução da SMT permanece consistentemente mais baixo do que o do JELLYFISH. Este é um ponto interessante, pois demonstra a eficiência da SMT em manter um tempo de execução baixo mesmo com o aumento de k . Adicionalmente, KANALYZE exibe um tempo de execução significativamente maior em comparação com as outras duas ferramentas em todos os valores de k , destacando a eficiência da SMT e do JELLYFISH em termos de tempo de execução.

Em resumo, enquanto o JELLYFISH e a SMT exibem maior consistência em seu consumo de tempo, o KANALYZE apresenta maior variabilidade. Dependendo do contexto e das necessidades específicas de um projeto, a seleção do algoritmo ideal pode variar. Por exemplo, para projetos com restrições rigorosas de memória, a SMT pode ser a escolha ideal devido à sua eficiência. Em contextos onde a variabilidade no consumo de RAM não é um problema e o desempenho é uma prioridade, tanto a SMT quanto o JELLYFISH são opções viáveis. Por outro lado, apesar de seus tempos de execução mais altos, o KANALYZE tem a vantagem de ser uma ferramenta baseada em JAVA, tornando-a uma opção versátil para sistemas que não dependem de hardware e software específicos.

Tabela 5.2: Análise comparativa do tempo médio e consumo de memória RAM agrupados por valores de k .

k	Jellyfish (tempo)	KAnalyze(tempo)	SMT (tempo)	Jellyfish (Ram)	KAnalyze (Ram)	SMT (Ram)
5	0,01	6,06	0,19	5,12	383,39	44,51
6	0,01	6,45	0,13	5,15	377,63	50,06
7	0,01	6,87	0,18	5,13	380,32	61,39
8	0,01	7,24	0,23	5,14	378,42	77,83
9	0,01	7,46	0,28	5,18	371,73	94,32
10	0,02	7,74	0,32	6,02	371,51	106,11
11	0,08	8,20	0,37	9,53	376,00	121,10
12	0,28	8,46	0,41	23,58	370,78	129,10
13	1,09	8,68	0,45	79,76	370,00	140,45
14	4,35	8,66	0,49	304,45	367,68	153,74
15	3,27	8,76	0,54	1054,17	367,40	164,43
16	4,05	8,76	0,58	1203,98	365,46	173,34
17	4,07	9,01	0,62	1347,63	360,25	183,47
18	4,11	8,95	0,67	1473,61	359,49	194,03
19	4,12	8,89	0,70	1618,81	358,23	206,14
20	4,14	8,88	0,75	1753,24	354,65	220,10
21	4,35	8,75	0,78	1885,82	352,14	231,02
22	4,18	8,68	0,83	2025,08	348,31	236,46
23	3,49	8,72	0,87	2102,76	348,49	249,19
24	4,23	8,70	0,90	2293,25	347,14	262,14
25	4,25	8,53	0,94	2425,23	340,15	269,82
26	4,28	8,48	0,97	2558,67	339,98	275,51
27	4,29	8,44	1,01	2701,95	341,64	293,82
28	4,34	8,36	1,04	2801,81	337,39	302,62
29	4,34	8,35	1,07	2966,11	334,86	308,54
30	4,36	8,26	1,10	3105,75	335,34	310,63

5.2.1 Análise estatística

Nesta seção, empregaremos uma abordagem estatística para discernir diferenças significativas de desempenho entre os algoritmos em termos de tempo de execução e consumo de memória RAM para cada valor de k . Inicialmente, conduziremos o teste de FRIEDMAN para verificar a existência de diferenças entre os algoritmos. Se uma diferença significativa for identificada, prosseguiremos com a análise post-hoc de NEMENYI, com correção de BONFERRONI, para determinar qual algoritmo se destaca como o mais eficiente para cada valor de k . Esta análise nos permitirá tirar conclusões robustas sobre o desempenho relativo dos algoritmos em estudo.

A Tabela 3.2 contém resultados comparativos para os três algoritmos testados: KANALIZE, JELLYFISH e SMT. A comparação foi feita usando o teste de FRIEDMAN com post-hoc de NEMENYI para a variável tempo, e os resultados foram categorizados de acordo com diferentes valores da variável k .

Podemos observar que os valores de χ^2 são substancialmente altos para todos os valores de k , oscilando principalmente em torno de 262 e diminuindo gradualmente para 229,481. Os p-valores associados são extremamente baixos, frequentemente na ordem de 10^{-50} ou menores. Isso indica que as diferenças entre os grupos comparados são estatisticamente significativas. As colunas para comparações pareadas (KAN_VS_JELLY, SMT_VS_JELLY, SMT_VS_KAN) mostram que, para muitos valores de k , os p-valores são tão baixos que quase atingem zero, indicando que as diferenças entre os métodos são estatisticamente significativas. No entanto, um p-valor de 0,967 para $k = 12$ na comparação entre SMT e JELLYFISH destaca-se como uma exceção, indicando que não há diferença significativa entre estes dois métodos neste caso específico. A

Tabela 5.3: Resultados do teste de FRIEDMAN com post-hoc de NEMENYI para a variável tempo agrupada por k , Legenda: (+) SMT ganhou, (-) SMT perdeu, (=) não houve diferença significativa entre o primeiro colocado e os demais.

k	χ^2	P-VALOR	KAN_VS_JELLY	SMT_VS_JELLY	SMT_VS_KAN	RESULTADO
5	262	1,28e-57	$< 2e - 16$	3,94e-14	3,94e-14	-
6	262	1,28e-57	$< 2e - 16$	3,94e-14	3,94e-14	-
7	262	1,28e-57	$< 2e - 16$	3,94e-14	3,94e-14	-
8	262	1,28e-57	$< 2e - 16$	3,94e-14	3,94e-14	-
9	262	1,28e-57	$< 2e - 16$	3,94e-14	3,94e-14	-
10	262	1,28e-57	$< 2e - 16$	3,94e-14	3,94e-14	-
11	259,53	4,4e-57	$< 2e - 16$	5,04e-14	3,59e-14	-
12	196,937	1,72e-43	$< 2e - 16$	0,967	$< 2e - 16$	=
13	254,244	6,19e-56	3,22e-14	1,08e-13	$< 2e - 16$	+
14	243,527	1,31e-53	2,1e-11	2,42e-14	$< 2e - 16$	+
15	258,061	9,18e-57	3,52e-14	5,73e-14	$< 2e - 16$	+
16	246,427	3,08e-54	2,5e-13	3,52e-14	$< 2e - 16$	+
17	248,29	1,21e-54	1,08e-13	3,72e-14	$< 2e - 16$	+
18	250,183	4,71e-55	5,21e-14	3,94e-14	$< 2e - 16$	+
19	242,794	1,9e-53	1,49e-12	3,22e-14	$< 2e - 16$	+
20	248,29	1,21e-54	1,08e-13	3,72e-14	$< 2e - 16$	+
21	239,282	1,1e-52	8,81e-12	2,94e-14	$< 2e - 16$	+
22	244,595	7,71e-54	6,07e-13	3,32e-14	$< 2e - 16$	+
23	256,046	2,51e-56	3,94e-14	5,04e-14	$< 2e - 16$	+
24	244,157	9,59e-54	9,5e-13	3,25e-14	$< 2e - 16$	+
25	241,023	4,6e-53	3,64e-12	3,04e-14	$< 2e - 16$	+
26	232,626	3,06e-51	2,63e-10	2,42e-14	$< 2e - 16$	+
27	235,893	5,98e-52	4,95e-11	2,7e-14	$< 2e - 16$	+
28	234,244	1,36e-51	1,15e-10	2,58e-14	$< 2e - 16$	+
29	235,142	8,7e-52	1,15e-10	2,58e-14	$< 2e - 16$	+
30	229,481	1,48e-50	1,31e-09	2,25e-14	$< 2e - 16$	+

coluna RESULT resume a eficácia do método SMT em relação aos outros. Para valores de k de 5 a 11, a SMT tem desempenho inferior, como indicado pelo sinal – na coluna Resultado. Para valores de k de 13 a 30, a SMT supera os outros métodos, como indicado pelo sinal +.

A análise sugere que o desempenho do método SMT é fortemente dependente do valor da variável k . O algoritmo parece ser menos eficaz para valores menores de k e mais eficaz para valores maiores. Essa característica pode ser atribuída à estratégia da SMT em estimar *a priori* a quantidade de memória necessária para o processamento, que é calculada com base no tamanho do conjunto de dados e no valor de k . O algoritmo procura otimizar o uso de memória e reduzir a necessidade de realocações, um fator crítico para maximizar a eficiência do tempo de processamento.

Finalmente, a Tabela 5.4 contém resultados comparativos para os três algoritmos testados: KANALIZE, JELLYFISH e SMT em relação à variável RAM. Semelhante ao teste anterior, esta comparação foi conduzida usando o teste de FRIEDMAN com post-hoc de NEMENYI, e os resultados foram categorizados de acordo com diferentes valores da variável k .

Analisando os valores obtidos, observamos que eles são consistentemente altos, frequentemente próximos a 262 e gradualmente reduzindo para 226,733 no final da tabela. Os p-valores correspondentes são extremamente baixos, frequentemente na ordem de 10^{-50} ou menores,

Tabela 5.4: Resultados do teste de FRIEDMAN com post-hoc de NEMENYI para a variável RAM agrupada por valores k , Legenda: (+) SMT ganhou, (-) SMT perdeu, (=) não houve diferença significativa entre o primeiro colocado e os demais.

k	χ^2	P-VALOR	KAN_VS_JELLY	SMT_VS_JELLY	SMT_VS_KAN	RESULTADO
5	262	$1,28e - 57$	$< 2e - 16$	$3,94e - 14$	$3,94e - 14$	-
6	262	$1,28e - 57$	$< 2e - 16$	$3,94e - 14$	$3,94e - 14$	-
7	262	$1,28e - 57$	$< 2e - 16$	$3,94e - 14$	$3,94e - 14$	-
8	262	$1,28e - 57$	$< 2e - 16$	$3,94e - 14$	$3,94e - 14$	-
9	262	$1,28e - 57$	$< 2e - 16$	$3,94e - 14$	$3,94e - 14$	-
10	262	$1,28e - 57$	$< 2e - 16$	$3,94e - 14$	$3,94e - 14$	-
11	262	$1,28e - 57$	$< 2e - 16$	$3,94e - 14$	$3,94e - 14$	-
12	262	$1,28e - 57$	$< 2e - 16$	$3,94e - 14$	$3,94e - 14$	-
13	262	$1,28e - 57$	$< 2e - 16$	$3,94e - 14$	$3,94e - 14$	+
14	193,115	$1,16e - 42$	0,0495	$2,89e - 14$	$< 2e - 16$	+
15	258,061	$9,18e - 57$	$5,73e - 14$	$< 2e - 16$	$3,52e - 14$	+
16	260,015	$3,45e - 57$	$4,59e - 14$	$< 2e - 16$	$3,72e - 14$	+
17	260,015	$3,45e - 57$	$4,59e - 14$	$< 2e - 16$	$3,72e - 14$	+
18	262	$1,28e - 57$	$3,94e - 14$	$< 2e - 16$	$3,94e - 14$	+
19	260,015	$3,45e - 57$	$3,72e - 14$	$< 2e - 16$	$4,59e - 14$	+
20	260,015	$3,45e - 57$	$3,72e - 14$	$< 2e - 16$	$4,59e - 14$	+
21	256,137	$2,4e - 56$	$3,32e - 14$	$< 2e - 16$	$5,21e - 14$	+
22	256,137	$2,4e - 56$	$3,32e - 14$	$< 2e - 16$	$5,21e - 14$	+
23	258,061	$9,18e - 57$	$3,52e - 14$	$< 2e - 16$	$5,73e - 14$	+
24	250,55	$3,93e - 55$	$2,94e - 14$	$< 2e - 16$	$6,07e - 13$	+
25	245,237	$5,59e - 54$	$2,58e - 14$	$< 2e - 16$	$8,81e - 12$	+
26	245,237	$5,59e - 54$	$2,58e - 14$	$< 2e - 16$	$8,81e - 12$	+
27	238,58	$1,56e - 52$	$2,14e - 14$	$< 2e - 16$	$2,63e - 10$	+
28	229,511	$1,45e - 50$	$3,2e - 14$	$< 2e - 16$	$2,75e - 08$	+
29	229,511	$1,45e - 50$	$3,2e - 14$	$< 2e - 16$	$2,75e - 08$	+
30	226,733	$5,83e - 50$	$2,83e - 14$	$< 2e - 16$	$1,15e - 07$	+

indicando que as diferenças entre os grupos comparados são estatisticamente significativas. Os p-valores para as comparações pareadas (KAN_VS_JELLY, SMT_VS_JELLY, SMT_VS_KAN) são, na maioria dos casos, extremamente baixos, sugerindo uma diferença significativa entre os métodos comparados. Observa-se que para todas as comparações os p-valores são praticamente zero, reforçando a ideia de diferenças significativas.

Na coluna RESULT, a eficácia do método SMT é resumida em relação a outros algoritmos. Para os valores de k de 5 a 12, o algoritmo SMT mostra desempenho inferior, conforme indicado pelo sinal - na coluna RESULT. Para os valores de k de 13 a 30, o algoritmo SMT demonstra superioridade, representada pelo sinal +. Os dados sugerem que o desempenho do método SMT é fortemente dependente do valor da variável k , similar à observação feita na análise de tempo. O algoritmo é menos eficaz para valores mais baixos de k e mais eficaz para valores mais altos de k . Mais uma vez, esta informação pode ser importante ao escolher o método mais apropriado, dependendo do valor considerado de k e do recurso do sistema em questão.

5.3 BIOMAPP::CHIP

Nesta seção, mostraremos e analisaremos os resultados obtidos pelo arcabouço BIOMAPP-CHIP em comparação com algoritmos do estado da arte. O BIOMAPP-CHIP foi comparado com algoritmos de última geração, incluindo MEME (Bailey et al., 1994), PROSAMPLER (Li et al., 2019) e HOMER (Heinz et al., 2010). Nos experimentos realizados, dois tipos distintos de dados foram empregados. Dados sintéticos foram utilizados para testes de carga, permitindo uma análise comparativa do consumo de tempo e uso de memória RAM entre as diferentes abordagens. Dados reais foram empregados para avaliar as métricas de distância e similaridade, garantindo assim uma avaliação mais realista da sua aplicabilidade em cenários práticos.

É importante ressaltar que todas as métricas de desempenho apresentadas neste estudo foram avaliadas com rigor. Os testes de carga, que incluem medições de tempo de execução e consumo de memória, foram padronizados e quantificados com precisão. O tempo de execução foi medido em segundos, enquanto o consumo de memória foi registrado em megabytes, oferecendo assim uma avaliação transparente e comparável do desempenho do algoritmo. Além das métricas de desempenho, a precisão foi avaliada utilizando diversas medidas de distância e correlação, tais como: distância EUCLIDIANA, distância MANHATTAN, distância HELLINGER, correlação PEARSON, coeficiente BHATTACHARYYA e similaridade SANDELIN-WASSERMAN. Essas medidas foram empregadas para avaliar os modelos encontrados pelos algoritmos em relação aos padrões de referência.

5.3.1 Dados sintéticos

A geração de dados sintéticos foi planejada para permitir uma comparação abrangente entre o arcabouço BIOMAPP-CHIP com outros algoritmos de descoberta de MOTIFS. Foram gerados 50 conjuntos de dados, com tamanhos variando de bases 1×10^5 a 2×10^7 . Cada um deles foi submetido a todos os algoritmos com tamanhos K-MER (k) variando de 5 a 30 bases. Este desenho experimental permitiu uma avaliação do desempenho e eficiência na contagem de K-MER sob diferentes condições de carga e complexidade.

5.3.2 Dados reais

Mais uma vez, os dados reais foram extraídos do banco de dados JASPAR juntamente com os arquivos de posição genômica (.BED). Os experimentos com dados reais constituem uma etapa fundamental na validação de nossa abordagem, pois desta forma podemos avaliar o desempenho dos algoritmos em aplicações práticas, aumentando a confiabilidade dos resultados e a robustez das nossas conclusões. A tabela 5.5 apresenta uma análise detalhada do volume de dados (medido pelo número de regiões de pico) gerado por diferentes tipos de experimentos disponíveis no banco de dados JASPAR.

O protocolo CHIP-SEQ destaca-se como principal fonte de dados, gerando o maior volume de informações, com número médio de regiões de pico de 17.538,78. O volume de dados gerado por este protocolo varia consideravelmente, variando de um mínimo de 22 a um máximo de 322.803 regiões de pico. Outros protocolos significativos em termos de volume de dados incluem CAP-SELEX e HT-SELEX, que geraram uma média de 9.878,26 e 9.856,47 regiões de pico, respectivamente. Embora esses protocolos gerem volumes consideráveis de dados, eles ainda estão significativamente abaixo do CHIP-SEQ.

Nos experimentos envolvendo dados reais, foram selecionados conjuntos de dados contendo mais de 10.000 sequências. Este critério resultou em um conjunto inicial de 148 conjuntos de dados distintos. No entanto, 17 deles foram posteriormente excluídos da análise, pois

Tabela 5.5: Quantidade de regiões de pico extraídas de diversos experimentos disponibilizados no JASPAR 2022. Nota-se de maneira clara que o protocolo CHIP-SEQ é responsável pela produção do volume mais expressivo de dados.

	TIPO	MIN	MAX	MÉDIA
1	CHIP-SEQ	22	322803	17538,78
2	CAP-SELEX	206	22850	9878,26
3	HT-SELEX	55	91919	9856,47
4	NA	84	105198	8801,62
5	NCAP-SELEX	2249	13498	6792,00
6	PBM	98	100001	4424,22
7	COMPILED	6	28379	3661,81
8	DAP-SEQ	22	19758	1222,74
9	SMILE-SEQ	998	1001	999,97
10	CHIP-EXO	48	1000	568,80
11	SELEX	9	1001	561,11
12	CHIP-CHIP	71	4737	475,80
13	SELEX-SEQ	384	384	384,00
14	UNIVERSAL PROTEIN BINDING MICROARRAY	99	101	100,00
15	CSA and/or DIP-CHIP	8	102	99,12
16	EMSA	29	100	64,50
17	BACTERIAL I-HYBRID	13	60	23,13
18	DNASE-I FOOTPRINTING	10	41	16,33

ainda estavam em processo de validação. Portanto, o conjunto final para avaliação consistiu em 131 conjuntos de dados validados. A escolha deste critério de seleção visa garantir uma amostra suficientemente grande e variada para avaliar de forma rigorosa e abrangente o desempenho dos algoritmos. Esta abordagem permitiu testar a eficácia dos algoritmos em dados reais e também forneceu uma análise aprofundada da sua aplicabilidade em cenários mais próximos das condições experimentais frequentemente encontradas em pesquisas na área de descoberta de MOTIFS biológicos.

5.3.3 Parâmetros

Para garantir a reprodutibilidade dos experimentos, estabelecemos os parâmetros para cada algoritmo de acordo com diretrizes específicas. A padronização dessas configurações é importante para garantir a integridade e comparabilidade dos resultados obtidos. Além disso, todos os experimentos foram executados em servidores INTEL XEON CPU E5-2673 v4 2,30 GHz com 8 Gb de memória RAM com sistema operacional LINUX/UBUNTU 22.04. Os parâmetros utilizados nos experimentos seguiram aqueles exibidos na Tabela 5.6.

Tabela 5.6: Parâmetros configurados para os algoritmos utilizados na pesquisa. Cada algoritmo é listado junto aos seus respectivos parâmetros para garantir a reprodutibilidade dos experimentos.

ALGORITMO	PARÂMETROS
BIOMAPP::CHIP	-k <tamanho do kmer> -n 1 -d 2 -c zoops -r 1000 -f 0.001
MEME	-w <tamanho do kmer> -dna -mod zoops -nmotifs 1
PROSAMPLER	-k <tamanho do kmer> -m 1 -l 0
HOMER	-len <tamanho do kmer> -S 1

É fundamental esclarecer alguns aspectos relativos à escolha dos parâmetros. Todos os métodos foram ajustados para identificar apenas o modelo com maior score, conforme eviden-

ciado pelas opções `-n 1`, `-n motifs 1`, `-m 1` e `-S 1` para os algoritmos BIOMAPP::CHIP, MEME, PROSAMPLER e HOMER, respectivamente. O modelo ZOOPS foi empregado nos algoritmos BIOMAPP::CHIP e MEME, enquanto HOMER utiliza o modelo ZOOPS intrinsecamente. O algoritmo PROSAMPLER, por outro lado, não oferece a opção de seleção deste modelo. Além disso, os parâmetros `-r 1000` e `-f 0.001` são usados para governar a convergência de BIOMAPP::CHIP. O processo iterativo será interrompido se o incremento na diferença de pontuação entre duas iterações sucessivas for inferior a `-f 0.001`. Caso contrário, o algoritmo continuará até atingir um limite de `-r 1000` iterações.

5.3.4 Biomapp::chip em dados sintéticos

O objetivo desses experimentos foi medir o desempenho do arcabouço BIOMAPP::CHIP em um conjunto diversificado de conjuntos de dados sintéticos. Esses dados foram gerados com parâmetros variáveis para simular diferentes cenários e complexidades associadas ao mundo real. A utilização de dados sintéticos oferece uma plataforma controlada que permite isolar e avaliar o desempenho e a eficácia dos algoritmos sob condições bem definidas. Este procedimento de análise é essencial para a validação inicial da abordagem metodológica adotada, servindo como etapa preliminar antes do tratamento de dados reais.

Esses testes foram projetados para coletar dados básicos sobre o consumo de tempo e uso de memória RAM pelo algoritmo BIOMAPP::CHIP. Essas informações foram comparadas com métricas de tempo e espaço obtidas de outros algoritmos do estado da arte, como MEME, PROSAMPLER e HOMER. O objetivo desta comparação foi entender como o BIOMAPP::CHIP se posiciona em relação a outras soluções e identificar os pontos fortes e fracos de cada abordagem.

Para a realização dos experimentos, o parâmetro k , representando o tamanho dos *kmers*, foi sistematicamente variado dentro de uma faixa que abrange valores de 5 a 30. Desta forma, foram realizados 5.200 experimentos, 1.300 para cada algoritmo. A captura de métricas relevantes, que incluem tanto o consumo de tempo quanto a alocação de memória RAM, foi realizada usando o comando `UNIX/LINUX /USR/BIN/TIME -v`, garantindo que a coleta de dados fosse consistente e comparáveis entre todos os algoritmos em estudo.

A figura 5.7 ilustra as médias globais dos testes para cada algoritmo, considerando o consumo de tempo e espaço. O BIOMAPP::CHIP provou ser superior em termos de eficiência de tempo, seguido de perto pelo MEME. Por outro lado, PROSAMPLER e HOMER apresentaram menor eficácia neste aspecto, necessitando de períodos maiores para execução. Em relação ao uso de memória, o MEME liderou em desempenho, mas foi seguido de perto por BIOMAPP::CHIP. Este último utiliza a estrutura de dados SMT para inicializar seus modelos, o que demonstra a eficiência desta estrutura em termos de uso de memória. Conforme observado na dimensão tempo, HOMER e PROSAMPLER ocuparam posições inferiores no consumo de memória.

A Figura 5.8 analisa o tempo de execução dos quatro algoritmos – BIOMAPP::CHIP, MEME, HOMER e PROSAMPLER – como uma função do tamanho k . É perceptível que o BIOMAPP::CHIP foi o algoritmo mais eficiente em termos de tempo, seguido de perto pelo MEME, ambos mantendo um tempo médio de execução abaixo de 100 segundos para todos os casos avaliados. Em contraste, os tempos médios de execução para o PROSAMPLER e HOMER foram substancialmente maiores, com o PROSAMPLER excedendo 200 segundos e o HOMER ultrapassando 400 segundos. Além disso, a análise também revela que o tempo de execução para todos os algoritmos avaliados aumenta à medida que o tamanho de k aumenta.

A Figura 5.9 apresenta um gráfico de linhas ilustrando o consumo de memória RAM de cada algoritmo. É um gráfico unificado, permitindo a comparação direta entre todos eles. Semelhante ao que foi observado no tempo de execução, BIOMAPP::CHIP e MEME estão próximos em eficiência de memória, embora o MEME tivesse uma ligeira vantagem, consumindo pouco

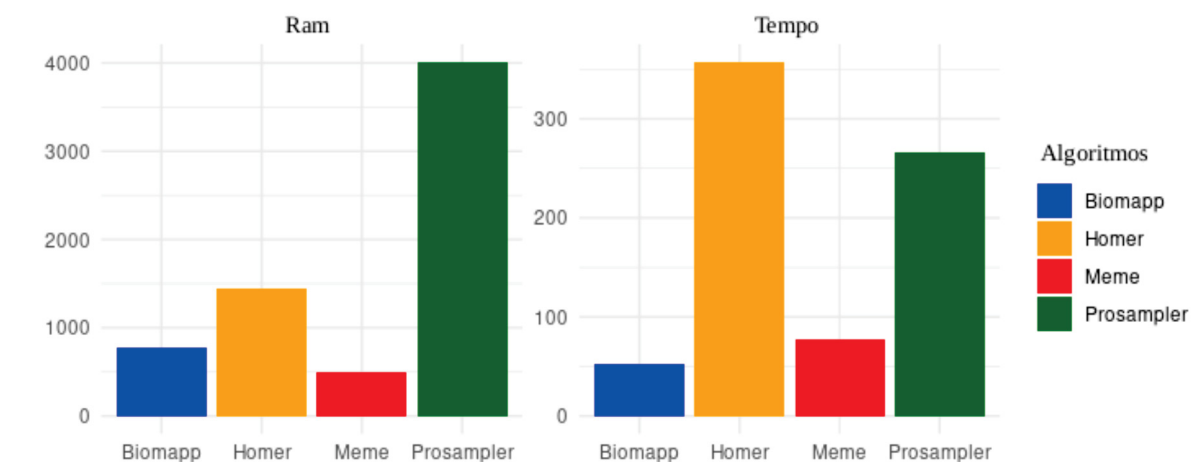


Figura 5.7: Comparação de desempenho entre os algoritmos BIOMAPP-CHIP, MEME, HOMER e PROSAMPLER, com foco em tempo (medido em segundos) e memória (medida em Mbytes) consumo. MEME se destacou pelo uso eficiente de memória, enquanto BIOMAPP-CHIP foi o mais rápido em termos de tempo de execução. Fonte: o autor.

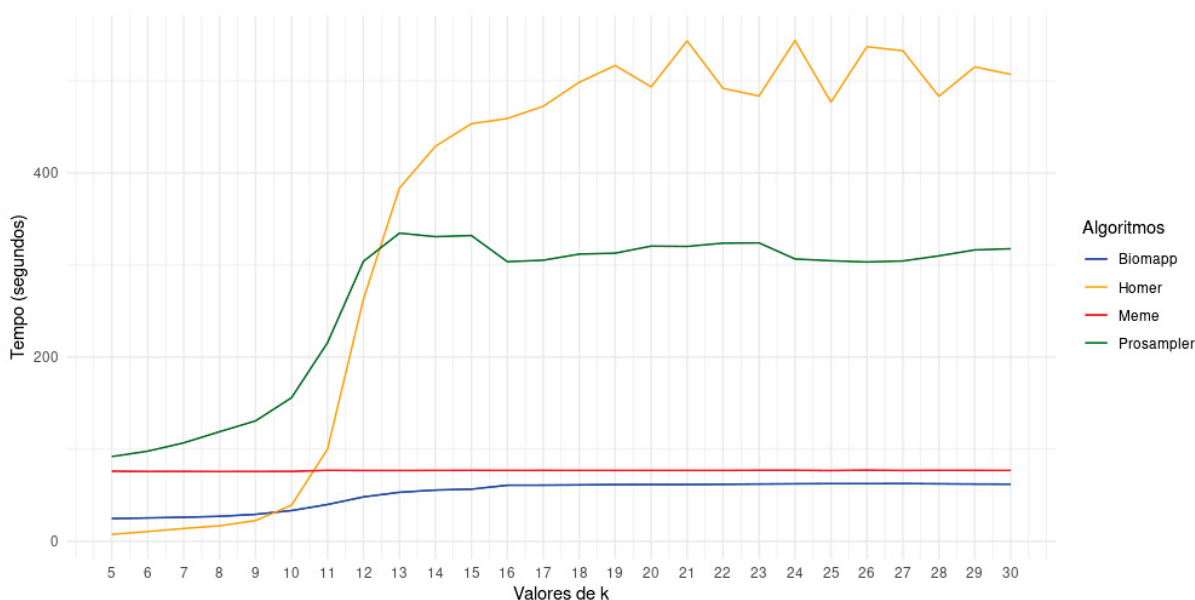


Figura 5.8: Variação do tempo de execução em relação ao k para os algoritmos BIOMAPP::CHIP, MEME, HOMER e PROSAMPLER. Observa-se que o algoritmo BIOMAPP::CHIP apresenta vantagem em termos de eficiência de tempo, seguido pelo algoritmo MEME. Fonte: o autor.

menos de 500 Mbytes no pior caso. O BIOMAPP::CHIP mostrou um consumo médio de memória próximo de 1000 Mbytes. Em contraste, os algoritmos HOMER e PROSAMPLER exibiram um uso muito maior de memória, alcançando pouco mais de 2000 Mbytes e excedendo 4000 Mbytes no pior cenário, respectivamente.

Na análise de desempenho dos algoritmos, é essencial considerar a relação conjunta entre os tempos de execução e o consumo de memória RAM de cada abordagem, pois estes são indicadores importantes da escalabilidade em cenários de processamento de dados de grande volume. A avaliação dos resultados, conforme apresentados na Figura 5.10, sugere que o BIOMAPP e o MEME, identificados pelas cores azul e vermelho, apresentam um perfil de desempenho distinto em comparação com os demais algoritmos.

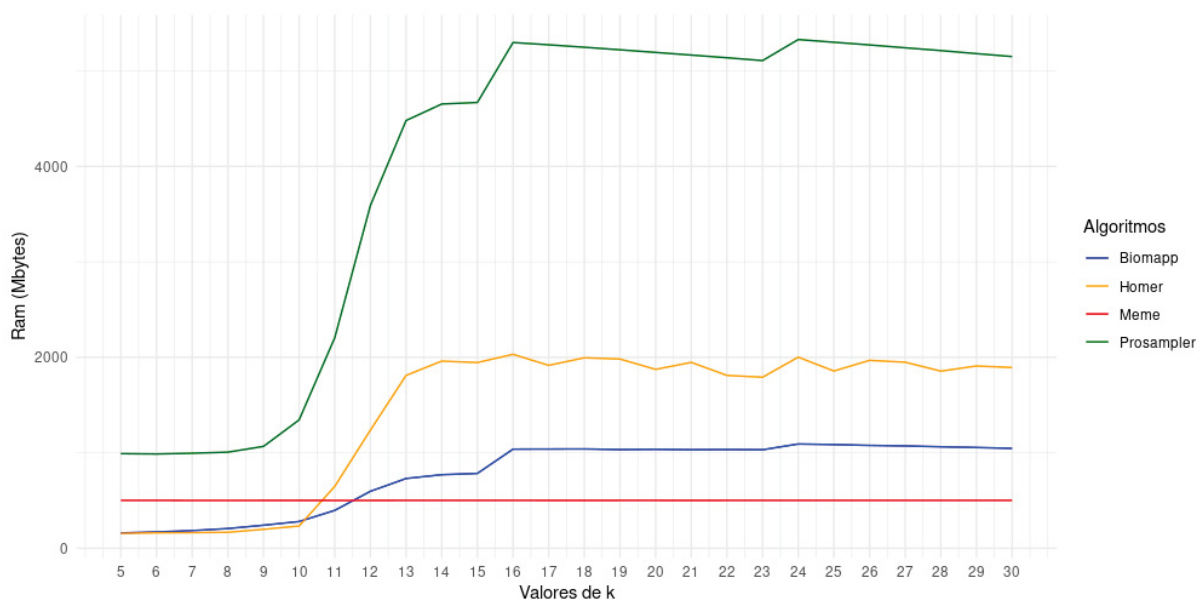


Figura 5.9: Variação do consumo de memória RAM em relação ao k para os algoritmos BIOMAPP::CHIP, MEME, HOMER e PROSAMPLER. Observa-se que o algoritmo MEME possui vantagem em termos de eficiência espacial, seguido pelo algoritmo BIOMAPP::CHIP. Fonte: o autor.

Observa-se que tanto o BIOMAPP quanto o MEME mantém um equilíbrio consistente entre o uso de RAM e o tempo de execução, indicando uma gestão de memória otimizada e um baixo tempo de processamento, mesmo com o aumento da carga de dados. Em particular, o BIOMAPP exibe um coeficiente angular levemente positivo, indicando que o tempo aumenta com o crescimento dos dados. Isso ocorre porque o BIOMAPP realiza a contagem de kmers antes da fase de otimização. Entretanto, esse pequeno aumento no tempo é compensado pela sua superioridade nas métricas de distância e similaridade, como será visto mais adiante.

Em contraste, os algoritmos HOMER e PROSAMPLER, representados respectivamente pelas cores amarelo e verde, tendem a demonstrar um aumento mais acentuado no tempo de execução conforme o consumo de RAM cresce. Esta tendência pode indicar uma menor eficiência no uso de memória ou uma complexidade computacional mais elevada, que resulta em tempos de processamento aumentados. Em particular, o HOMER, apresenta uma variabilidade significativa nos resultados, sugerindo uma sensibilidade maior às condições específicas dos dados ou aos parâmetros de execução.

5.3.5 Biomapp::chip em dados reais

Embora experimentos em ambientes simulados sejam úteis para compreensão inicial e ajuste fino de algoritmos, são os ensaios com dados reais que fornecem o verdadeiro teste de validade para qualquer método computacional em bioinformática. Eles oferecem um cenário muito mais complexo e variável, que representa com mais precisão as condições que os algoritmos enfrentarão em aplicações do mundo real. Portanto, esta seção é a mais crítica e fornece informações mais precisas sobre a viabilidade e robustez dos algoritmos avaliados.

O foco principal deste teste foi a comparação da acurácia entre as abordagens. Por isso, diferentemente dos testes realizados em dados sintéticos, onde o valor de k foi variado sistematicamente, neste cenário, para os testes de acurácia, o valor de k adotado foi o sugerido pela literatura científica. Assim, cada algoritmo foi testado com um valor único e específico de

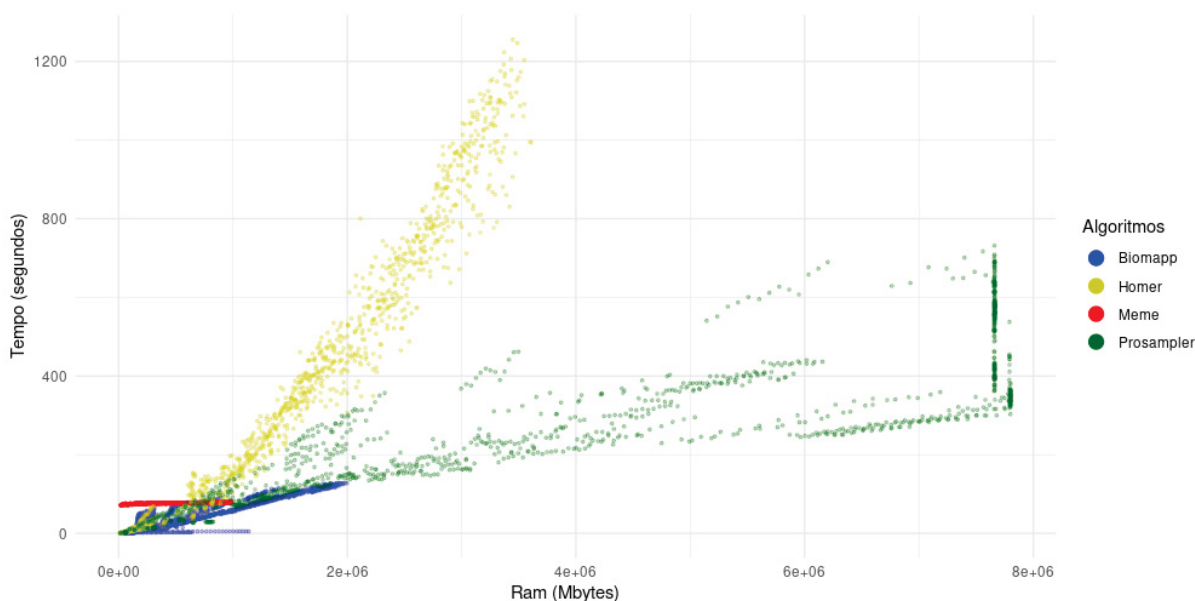


Figura 5.10: Comparação do desempenho entre algoritmos BIOMAPP, MEME, HOMER e PROSAMPLER em termos de tempo de execução e consumo de memória RAM. Dados sintéticos foram utilizados para avaliar a eficiência e a escalabilidade dos algoritmos. Os pontos representam a inter-relação entre a quantidade de RAM utilizada (eixo x) e o tempo necessário para completar a execução (eixo y), com cada cor correspondendo a um algoritmo específico. Fonte: o autor.

k, de acordo com as diretrizes acadêmicas mais recentes. Para os testes de carga, o valor de k variou de maneira similar aos testes com dados sintéticos.

5.3.5.1 Desempenho em relação ao consumo de tempo e espaço

A figura 5.11 revela nuances importantes no desempenho dos quatro algoritmos examinados em termos de tempo de execução e consumo de memória RAM. O BIOMAPP:CHIP se destaca por apresentar o menor tempo médio de execução, apenas 15,9 segundos, sendo a opção mais ágil. Por outro lado, HOMER mostrou-se o mais lento, com tempo médio de 624 segundos e consumo médio de RAM de 1436,16 MB, o que pode ser uma limitação em cenários que exigem respostas rápidas.

Em termos de consumo de RAM, o MEME foi o mais eficiente com uma média de 217 MBYTES, enquanto o HOMER exibiu o maior consumo, com 1436 MBYTES. O PROSAMPLER, que teve desempenho moderado no tempo, registrou o maior pico de utilização de RAM, atingindo 3501 MBYTES. Além disso, PROSAMPLER apresentou a maior variação tanto no tempo quanto no consumo de RAM, sugerindo que seu desempenho pode ser altamente sensível às características do conjunto de dados. No geral, os testes indicaram que BIOMAPP:CHIP e MEME se mostraram mais robustos, tornando-os mais previsíveis em seu desempenho, conforme mostrado na Tabela 5.7.

Tabela 5.7: Comparação do consumo de memória (medido em Mbytes) e tempo (medido em segundos) entre os algoritmos BIOMAPP:CHIP, HOMER, MEME e PROSAMPLER.

ALGORITMO	TEMPO MÉDIO	RAM MÉDIA	TEMPO MAX	RAM MAX	TEMPO MIN	RAM MIN
Biomapp	15,86	335,57	192,79	868,92	1,41	190,08
Homer	624,45	1436,16	1952,69	2270,09	52,47	524,70
Meme	138,36	216,94	159,70	1618,04	118,93	69,32
Prosampler	61,83	548,17	655,92	3500,79	14,26	199,07

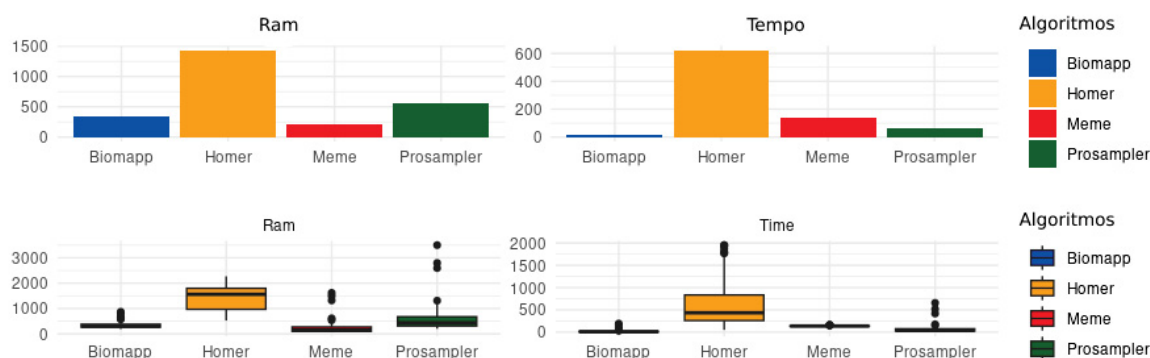


Figura 5.11: Comparação da média geral de memória RAM (medida em Mbytes) e consumo de tempo de execução (medido em segundos) entre diferentes algoritmos aplicados a conjuntos de dados reais. Observa-se que BIOMAPP:CHIP lidera em consumo de tempo e ocupa a segunda posição em consumo de RAM. MEME apresenta o menor consumo de RAM, enquanto PROSAMPLER, apesar de seu desempenho inferior em testes sintéticos, apresenta uma melhoria significativa, ficando em segundo lugar no consumo de memória. O algoritmo HOMER não se destaca em nenhum dos critérios. Fonte: o autor.

A análise dos dados apresentados nas Figuras 5.12 e 5.13 fornece insights importantes sobre o desempenho e a eficiência dos quatro algoritmos avaliados em vários cenários, parametrizados pelo valor de k . Primeiramente, é possível observar que o algoritmo BIOMAPP:CHIP apresenta, em média, o menor consumo de tempo e memória RAM em quase todos os cenários. Seu tempo médio de execução varia de 9,79 a 23 segundos, enquanto o uso médio de RAM fica entre 251,50 e 838,70 MBYTES. O algoritmo também é consistente em manter um tempo de execução máximo e mínimo relativamente baixo, conforme mostrado pelos dados na Tabela 5.8.

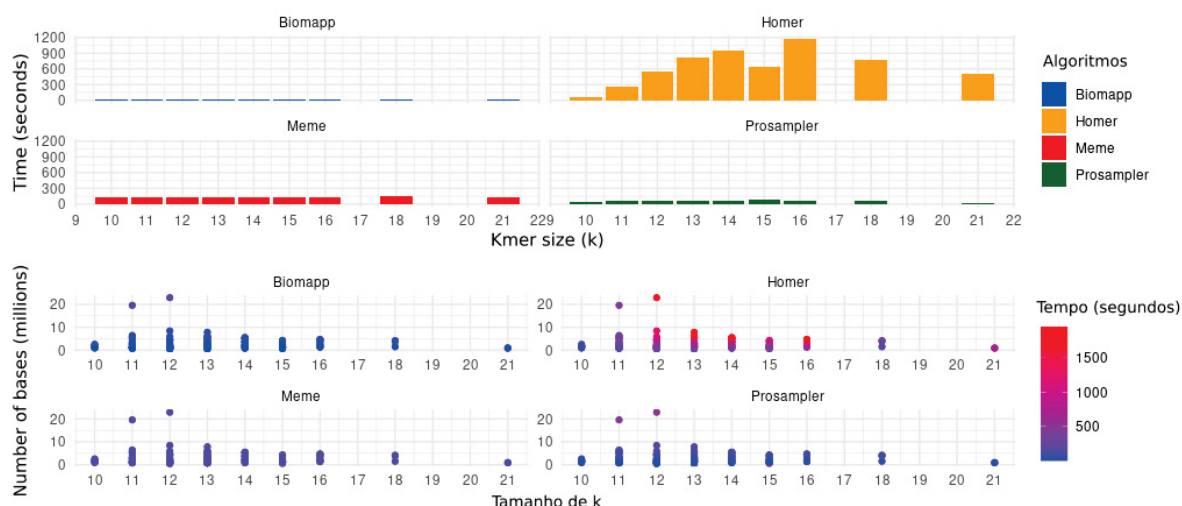


Figura 5.12: Comparação entre o desempenho médio dos algoritmos em relação ao consumo de tempo agrupado por valor de k . É possível verificar que BIOMAPP:CHIP, PROSAMPLER e MEME apresentam os tempos mais curtos enquanto HOMER apresentou tempos de execução significativamente maiores. Fonte: o autor.

Na análise conduzida em conjuntos de dados reais para estudar o relação entre o tempo gasto e a quantidade de memória RAM consumida, observou-se um comportamento similar ao reportado na Figura 5.10, embora com uma inversão entre os algoritmos BIOMAPP e MEME. Na Figura 5.14, é possível identificar que o BIOMAPP exibiu uma relação entre uso de memória e tempo de execução superior ao MEME, traduzindo-se em eficiência aprimorada. Enquanto

Tabela 5.8: Comparação do consumo de memória (medido em Mbytes) e tempo (medido em segundos) entre os algoritmos BIOMAPP::CHIP, HOMER, MEME e PROSAMPLER agrupados por k.

ALGORITMO	K	TEMPO MÉDIO	RAM MÉDIO	TEMPO MAX	RAM MAX	TEMPO MIN	RAM MIN
Biomapp	10	19,34	251,50	95,17	422,70	4,69	208,02
	11	15,83	307,61	150,88	489,64	2,62	221,99
	12	21,53	326,33	192,79	664,13	2,36	235,98
	13	13,25	310,64	40,45	514,13	1,41	190,08
	14	15,99	370,37	34,08	576,42	3,68	262,31
	15	10,43	343,74	28,53	683,56	4,05	271,53
	16	15,84	381,98	24,34	527,36	7,01	281,87
	18	23,00	420,45	34,17	478,62	11,25	374,58
Homer	21	9,79	838,70	15,10	868,92	4,47	808,49
	10	65,81	542,56	85,76	575,83	52,47	524,70
	11	258,96	867,76	421,93	1054,18	133,48	684,94
	12	549,22	1368,86	1802,54	1944,38	152,97	1156,77
	13	806,39	1613,70	1951,66	1796,67	171,83	1467,87
	14	952,28	1825,39	1875,14	2103,16	213,74	1565,54
	15	645,16	1826,03	1584,17	2062,00	326,13	1717,78
	16	1174,47	1950,41	1952,69	2195,72	184,05	1592,81
Meme	18	763,16	1887,06	1931,68	2270,09	142,80	1595,04
	21	497,58	1765,16	709,77	1824,43	285,39	1705,90
	10	139,84	117,86	152,63	166,72	133,59	73,30
	11	138,23	225,86	159,70	1319,03	122,78	72,28
	12	137,20	242,56	155,45	1618,04	126,23	69,32
	13	136,74	202,81	153,11	538,06	118,93	72,25
	14	140,46	223,61	156,88	409,22	129,15	73,24
	15	139,28	240,67	152,55	1491,57	130,57	75,79
Prosampler	16	139,92	218,37	152,61	338,57	129,52	102,36
	18	143,33	248,87	152,39	328,58	136,32	135,78
	21	134,64	86,45	143,97	93,61	125,32	79,28
	10	31,15	350,59	47,54	482,50	16,71	232,54
	11	62,76	563,89	418,14	2796,34	15,60	199,07
	12	69,60	603,25	518,37	3500,79	14,26	213,74
	13	55,79	526,55	155,40	1166,94	16,35	235,40
	14	61,46	571,55	112,23	909,96	17,21	226,32
	15	81,32	566,79	655,92	2596,81	16,70	236,42
	16	58,93	558,32	97,11	817,34	25,89	335,26
	18	69,08	626,60	91,61	794,47	34,94	400,59
	21	20,13	259,94	21,58	265,49	18,68	254,39

isso, o PROSAMPLER demonstrou um consumo de recursos equilibrado, enquanto que o HOMER mostrou-se o mais exigente em termos de recursos, quando comparado com os demais algoritmos.

Em cenários com dados reais, a eficiência aprimorada do BIOMAPP pode ser atribuída à presença de alta redundância, tanto nas bases quanto entre elementos de maior complexidade. Esta redundância favorece o BIOMAPP na etapa de contagem de k-mers e na modelagem de modelos sub-ótimos, o que, por sua vez, impulsiona a eficiência global do algoritmo. A natureza dos dados reais, que frequentemente inclui sequências recorrentes e padrões biomoleculares, parece ser bem capturada e processada pelo BIOMAPP, resultando em uma performance destacada quando comparada aos demais algoritmos que não se beneficiam na mesma medida dessa característica intrínseca dos dados.

O algoritmo MEME apresentou relativa estabilidade no tempo médio de execução, variando de 134,64 a 143,33 segundos, mas com picos de uso de RAM atingindo 1618,04 MBYTES para $k = 12$. O PROSAMPLER apresentou desempenho intermediário em termos de eficiência de

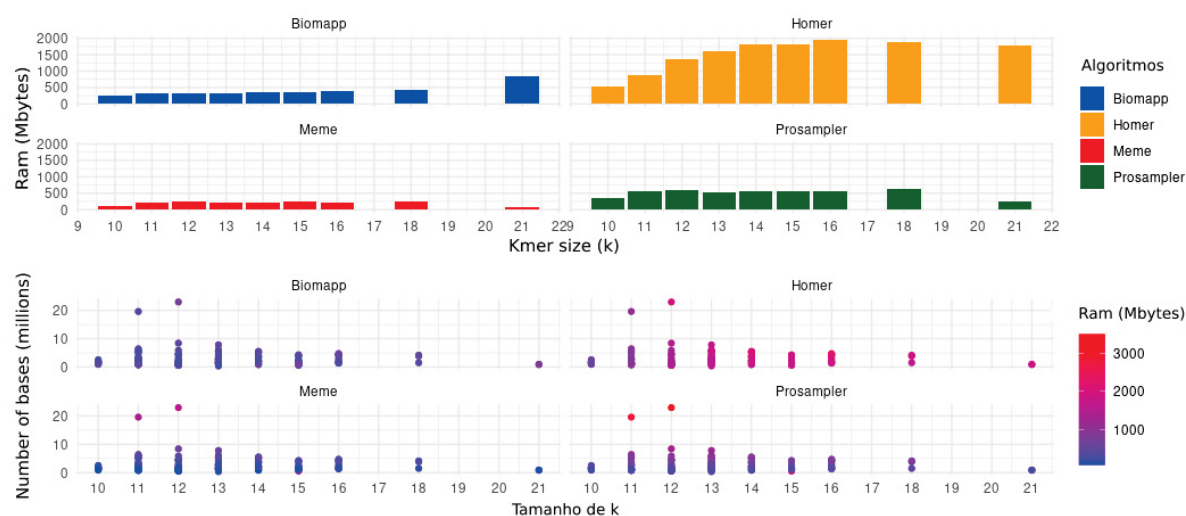


Figura 5.13: Comparação entre o desempenho médio dos algoritmos em relação ao consumo de memória RAM agrupado pelo valor k . É possível verificar que BIOMAPP::CHIP, PROSAMPLER e MEME apresentam os tempos mais curtos enquanto HOMER apresentou tempos de execução significativamente maiores. Fonte: o autor.

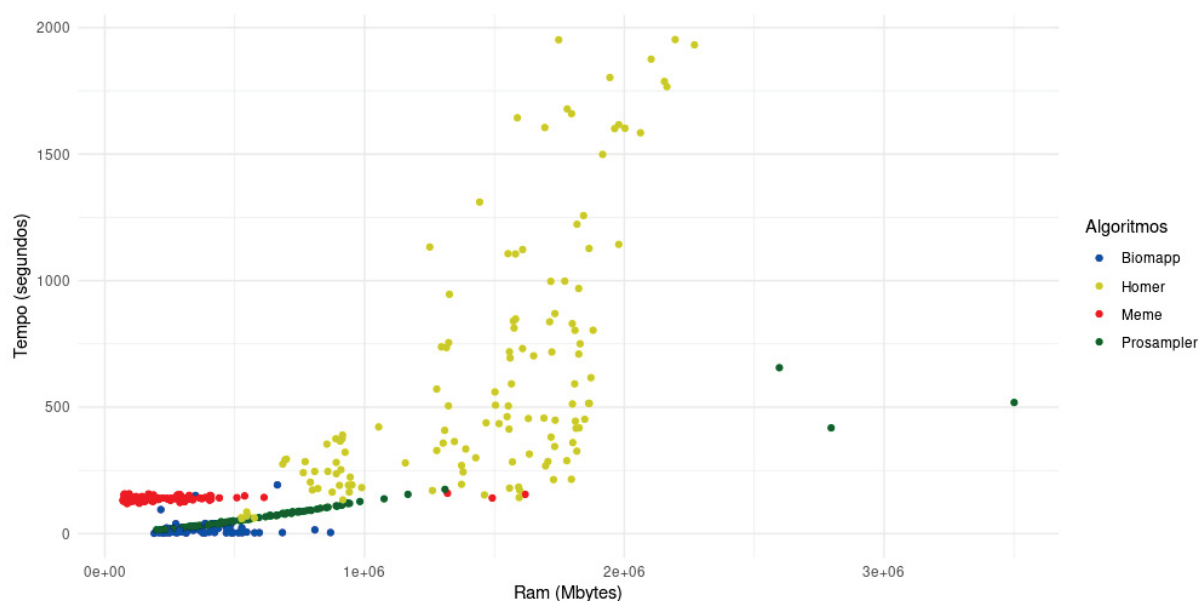


Figura 5.14: Comparação do desempenho entre algoritmos BIOMAPP, MEME, HOMER e PROSAMPLER em termos de tempo de execução e consumo de memória RAM. Dados reais extraídos do repositório JASPAR foram empregados para avaliar a eficiência e a escalabilidade dos algoritmos. Os pontos representam a inter-relação entre a quantidade de RAM utilizada (eixo x) e o tempo necessário para completar a execução (eixo y), com cada cor correspondendo a um algoritmo específico. Fonte: o autor.

tempo e uso de RAM, com grande variação nas métricas máximas e mínimas, principalmente para uso de RAM, que chega a 3500,79 MBYTES para $k = 12$.

5.3.5.2 Comparação com modelos de referência

Nesta subseção abordaremos a avaliação dos modelos gerados por cada algoritmo, contrastando-os com modelos de referência extraídos do banco de dados JASPAR. O objetivo desta comparação é quantificar quão bem os algoritmos foram capazes de aproximar as matrizes PWM encontradas por cada abordagem para suas respectivas matrizes de referência. Para isso,

foram empregadas métricas específicas para medir a distância entre as matrizes, proporcionando assim uma análise abrangente da precisão dos métodos em encontrar representações confiáveis e precisas dos MOTIFS biológicos investigados.

Conforme Figura 5.15 e Tabela 5.9, o algoritmo BIOMAPP::CHIP apresentou superioridade em quase todas as métricas analisadas. Na métrica BHA, o algoritmo BIOMAPP::CHIP registrou média de 0,993 e desvio padrão muito baixo de 0,0308, sinalizando consistência nos resultados. Contudo, é interessante notar que o PROSAMPLER teve o mínimo mais baixo nesta métrica (0,685), o que pode ser um ponto de preocupação em termos de consistência. A eficácia demonstrada pelo BIOMAPP::CHIP pode ser explicada pela sua estratégia de utilizar modelos subótimos como sementes de entrada para o algoritmo FAST-EM. Essa abordagem possibilitou ao BIOMAPP::CHIP explorar o espaço de soluções de maneira mais assertiva, melhorando a probabilidade de encontrar soluções ótimas globais ao refinar progressivamente as sementes busca.

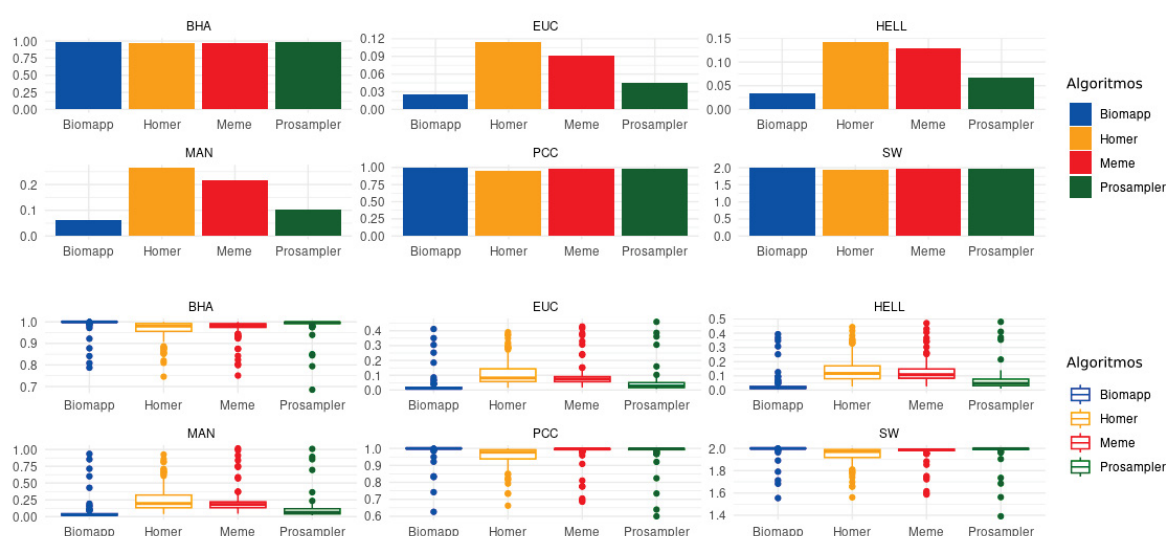


Figura 5.15: Análise comparativa dos algoritmos BIOMAPP::CHIP, MEME, PROSAMPLER e HOMER usando diversas métricas de distância e coeficientes de correlação. As métricas empregadas para comparação incluem distância EUCLIDIANA, distância MANHATTAN, distância HELLINGER, coeficiente de correlação PEARSON, coeficiente SANDELIN-WASSERMAN e BHATTACHARYYA coeficiente. Nota-se que, ao considerar a média geral, o algoritmo BIOMAPP apresentou desempenho superior em relação aos demais algoritmos avaliados. Fonte: o autor.

Quando observamos métricas de distância como EUC, HELL e MAN, BIOMAPP mais uma vez se destaca por ter as médias mais baixas. Aqui, o desempenho de HOMER é notavelmente inferior; por exemplo, na métrica MAN a média é 0,264 com desvio padrão de 0,194, sendo ambos os valores os mais elevados entre os algoritmos analisados. Isto poderia apontar para um desempenho geralmente inferior de HOMER em comparações multidimensionais. Em relação à métrica PCC, todos os algoritmos apresentaram valores médios elevados, indicando um bom grau de correlação. Porém, é interessante notar que PROSAMPLER teve o valor mínimo mais baixo (0,600) nesta métrica. Tal valor mínimo é significativamente inferior aos demais, o que pode indicar instabilidade em casos específicos. A figura 5.16 mostra a evolução de cada algoritmo em função da variável k .

É possível observar nesta figura que o BIOMAPP::CHIP nas distâncias EUCLIDIANA, HELLINGER e MANHATTAN sempre ficou abaixo dos demais algoritmos, para todos os valores de k . No coeficiente BHATTACHARYYA, PROSAMPLER obteve o melhor resultado (0,988) para $k = 10$, seguido por MEME (0,975) e BIOMAPP::CHIP (0,972). Para outros valores de k , o BIOMAPP::CHIP obteve melhores valores nesta métrica. Algo semelhante ocorreu com a correlação PEARSON e

Tabela 5.9: Análise comparativa entre os algoritmos em diversas métricas de distância e correlação. A linha amarela destaca o vencedor de cada métrica em termos de média.

MÉTRICA	ALGORITMO	MÉDIA	DESVIO PADRÃO	MIN	MAX
BHA	Biomapp	0,993	0,031	0,787	1,000
	Homer	0,963	0,045	0,746	0,999
	Meme	0,972	0,040	0,751	0,999
	Prosampler	0,988	0,037	0,685	1,000
EUC	Biomapp	0,026	0,058	0,002	0,412
	Homer	0,115	0,085	0,016	0,391
	Meme	0,091	0,071	0,019	0,429
	Prosampler	0,045	0,065	0,008	0,460
HELL	Biomapp	0,033	0,063	0,003	0,393
	Homer	0,143	0,091	0,026	0,443
	Meme	0,129	0,077	0,027	0,470
	Prosampler	0,067	0,069	0,011	0,479
MAN	Biomapp	0,061	0,137	0,005	0,935
	Homer	0,264	0,194	0,039	0,925
	Meme	0,218	0,167	0,044	1,016
	Prosampler	0,104	0,147	0,019	1,008
PCC	Biomapp	0,991	0,045	0,626	1,000
	Homer	0,955	0,059	0,663	0,999
	Meme	0,983	0,055	0,686	1,000
	Prosampler	0,985	0,054	0,600	1,000
SW	Biomapp	1,988	0,057	1,554	2,000
	Homer	1,940	0,079	1,561	1,999
	Meme	1,969	0,070	1,586	1,999
	Prosampler	1,982	0,074	1,391	2,000

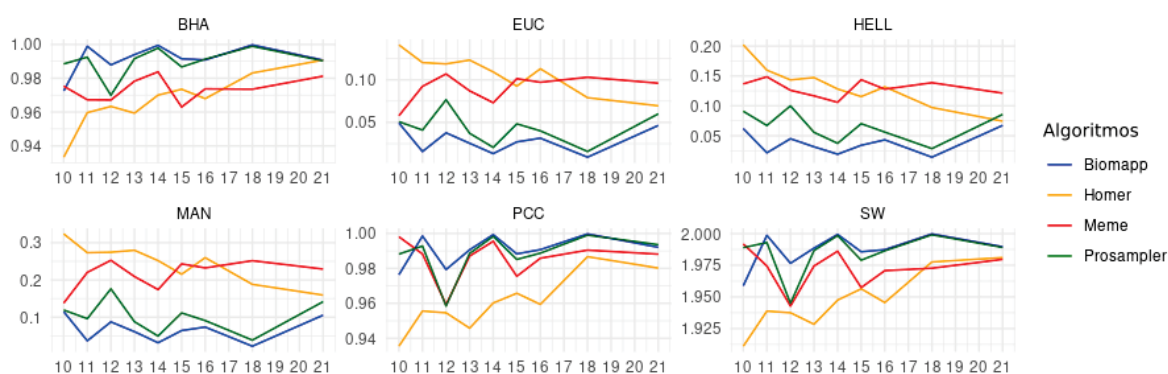


Figura 5.16: Análise comparativa de algoritmos baseada na evolução das métricas em função da variável k . Fonte: o autor.

similaridade SANDELIN-WASSERMAN, em que BIOMAPP::CHIP obteve a pontuação mais baixa para $k = 10$.

A métrica sw mostrou uma competição mais acirrada entre os algoritmos, com todas as médias próximas de 2 e desvios padrão relativamente baixos. Porém, PROSAMPLER apresentou aqui o valor mínimo mais baixo de 1,39, o que mais uma vez levanta questões sobre sua consistência em comparação com os outros algoritmos. Embora o BIOMAPP::CHIP tenha apresentado um desempenho geralmente superior em todas as métricas avaliadas, os demais algoritmos possuem

pontos fracos que merecem atenção. O desempenho inferior do HOMER em métricas de distância e a variabilidade do PROSAMPLER em métricas como BHA e PCC são pontos que requerem investigações adicionais.

A Figura 5.17 exibe o comportamento dos algoritmos com relação ao consumo de tempo vs distância do modelo de referência. Podemos verificar que o algoritmo BIOMAPP (azul) possui pontos em maior concentração próximos ao zero na escala de distância do modelo de referência, sugerindo uma maior precisão em menos tempo, comparado aos outros algoritmos. O PROSAMPLER (verde) mostra uma distribuição densa de pontos em uma faixa mais estreita de tempo e precisão, indicando um desempenho consistente e equilibrado. O MEME (vermelho) exibe uma variabilidade significativa tanto no tempo quanto na precisão, o que pode indicar que seu desempenho é sensível às condições específicas dos dados ou parâmetros de execução. Por fim, o HOMER (amarelo) parece ter um desempenho menos eficiente, com pontos dispersos em uma ampla faixa de tempos de execução e, em sua maioria, maiores distâncias do modelo de referência.

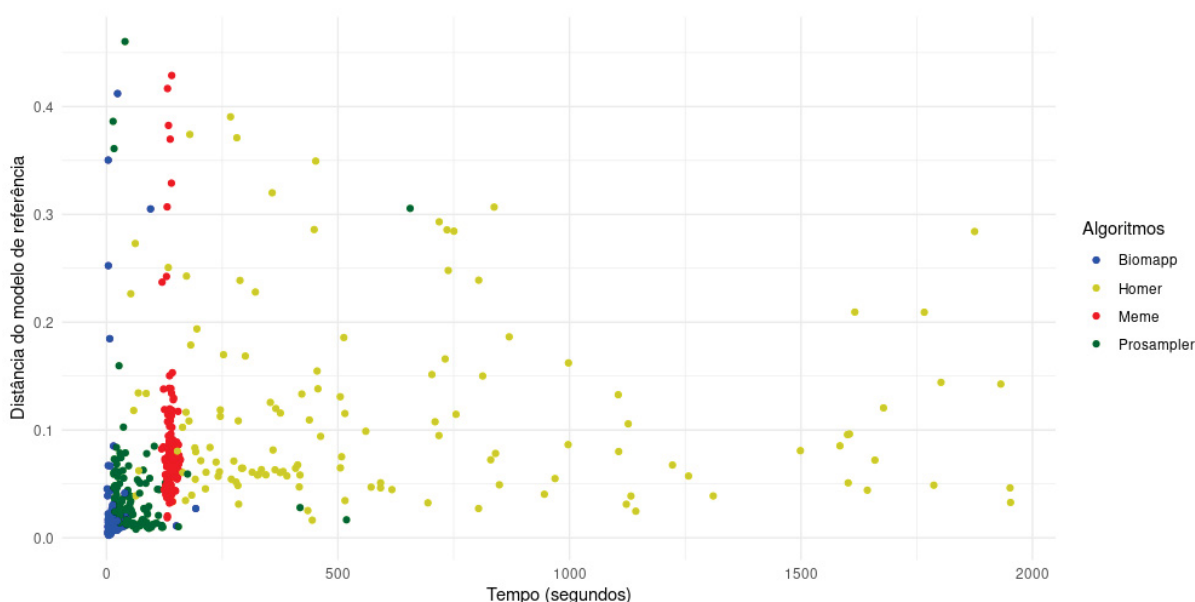


Figura 5.17: Comparação do desempenho dos quatro algoritmos em termos de consumo de tempo vs precisão, medida pela distância EUCLIDIANA do modelo de referência. Os pontos representam a performance individual de cada execução dos algoritmos BIOMAPP (azul), HOMER (amarelo), MEME (vermelho) e PROSAMPLER (verde). Fonte: o autor.

A Figura 5.18 exibe o comportamento dos algoritmos com relação ao consumo de RAM vs distância do modelo de referência. Podemos observar que há uma distribuição variável dos pontos que representam cada algoritmo no gráfico. Isso sugere diferenças no desempenho e eficiência de cada um em relação ao uso de memória RAM e precisão. O eixo x, que representa a memória RAM utilizada, mostra uma variação significativa entre as abordagens. O BIOMAPP e o MEME, por exemplo, usam menos RAM em comparação com o HOMER e o PROSAMPLER. Existe uma tendência de que, à medida que o uso de RAM aumenta, a distância do modelo de referência diminui, o que pode indicar uma melhor precisão. No entanto, essa tendência não é uniforme entre os métodos.

Ainda na Figura 5.18, vemos que o BIOMAPP (azul) parece oferecer uma ótima eficiência em termos de consumo de memória versus distância do modelo de referência, com muitos de seus pontos agrupados na região de menor uso de memória e menor distância. O HOMER (amarelo) mostra um uso de memória geralmente mais alto com uma distribuição mais ampla na distância

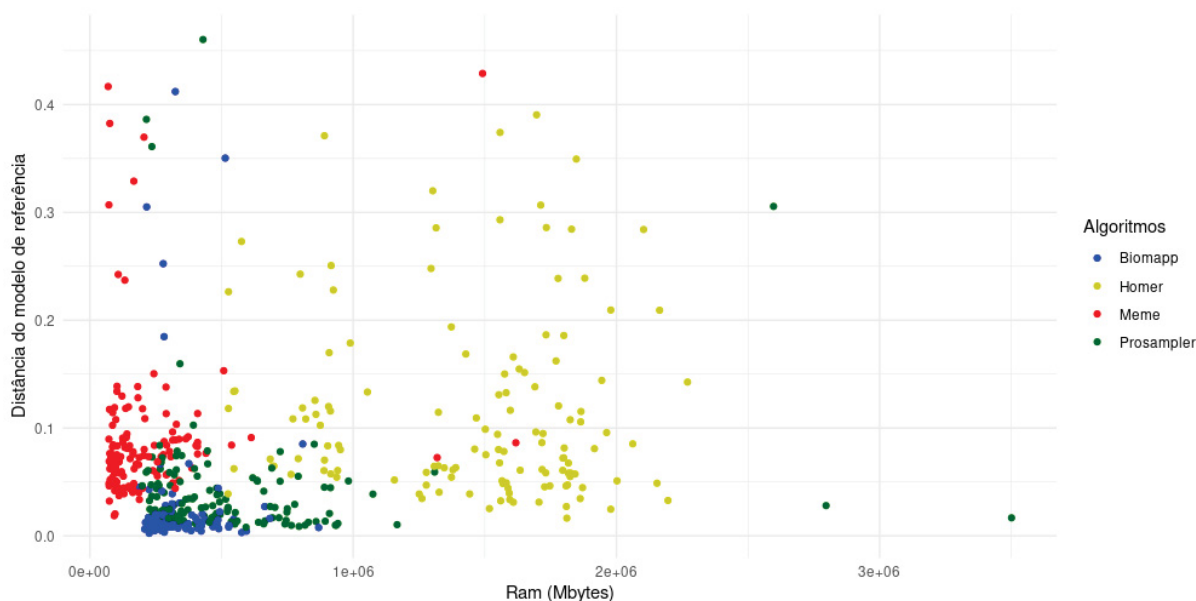


Figura 5.18: Comparação do desempenho dos quatro algoritmos em termos de consumo de RAM vs precisão, medida pela distância EUCLIDIANA do modelo de referência. Os pontos representam a performance individual de cada execução dos algoritmos BIOMAPP (azul), HOMER (amarelo), MEME (vermelho) e PROSAMPLER (verde). Fonte: o autor.

do modelo de referência, sugerindo uma variabilidade maior no desempenho. O MEME (vermelho) apresenta pontos concentrados em baixo consumo de RAM, mas com distâncias variáveis do modelo de referência. Por fim, o PROSAMPLER (verde) demonstra um perfil intermediário, com um consumo moderado de memória e distâncias próximas do modelo de referência.

5.3.6 Análise estatística

Na análise de desempenho de algoritmos, é imperativo empregar técnicas estatísticas rigorosas para determinar se as diferenças observadas nas diversas métricas são realmente significativas. A aplicação de testes estatísticos proporciona um meio de validar comparações entre diferentes abordagens ou cenários, garantindo que as conclusões tiradas são robustas e fiáveis.

Nesta subsecção foi utilizado o teste FRIEDMAN, um método não paramétrico empregado para identificar variações significativas nas medianas entre múltiplos grupos pareados. Este teste estatístico é especialmente relevante quando não são verificadas as condições de normalidade e homocedasticidade, pré-requisitos para a aplicação da ANOVA, como observado no presente estudo.

Após os testes principais, foram realizados testes POST HOC para análises de comparações múltiplas. O objetivo foi identificar quais grupos são estatisticamente diferentes entre si. Os resultados dos testes POST HOC são essenciais para estabelecer conclusões concretas sobre as métricas avaliadas.

Os dados exibidos na Tabela 5.10 mostram os resultados dos testes estatísticos FRIEDMAN e NEMENYI aplicados a diversas métricas para avaliar o desempenho dos algoritmos de descoberta de MOTIFS. As métricas avaliadas incluem EUC (distância EUCLIDIANA), MAN (distância MANHATTAN), PCC (correlação PEARSON), INFERNO (distância HELLINGER), SW (coeficiente SANDELIN-WASSERMAN) e BHA (distância BHATTACHARYYA).

Para cada métrica, o teste FRIEDMAN produziu P-VALORES altamente significativos, apontando para uma diferença considerável entre as abordagens. Todos os P-VALORES estão

Tabela 5.10: Resultados dos testes FRIEDMAN e NEMENYI para avaliação de algoritmos de descoberta de MOTIVOS. Cada linha representa uma métrica avaliada. As colunas χ^2 e p - value correspondem aos resultados do teste de FRIEDMAN. As colunas relativas a HOMER, MEME e PROSAMPLER exibem os P-VALUES obtidos pelo teste NEMENYI, todos comparados com o programa BIOMAPP::CHIP. A última coluna, chamada RESULTADO, indica se o BIOMAPP teve um desempenho melhor (+), inferior (-) em comparação com as outras abordagens ou (=) não estatisticamente significativo.

	χ^2	P-VALOR	HOMER	MEME	PROSAMPLER	RESULTADO
EUC	280,46	$1,69e - 60$	$< 2e - 16$	$< 2e - 16$	$4,01e - 09$	+
MAN	281,35	$1,08e - 60$	$< 2e - 16$	$< 2e - 16$	$8,58e - 10$	+
PCC	243,61	$1,57e - 52$	$< 2e - 16$	$2,94e - 14$	$7,42e - 13$	+
HELL	275,29	$2,21e - 59$	$< 2e - 16$	$< 2e - 16$	$1,50e - 12$	+
SW	261,30	$2,35e - 56$	$< 2e - 16$	$< 2e - 16$	$9,85e - 09$	+
BHA	271,19	$1,71e - 58$	$< 2e - 16$	$< 2e - 16$	$3,32e - 11$	+

muito próximos de zero, bem abaixo do nível de significância adotado de $\alpha = 0,05$, indicando que as diferenças são estatisticamente significativas. A coluna RESULT indica que o programa BIOMAPP::CHIP superou todas as outras abordagens para todas as métricas testadas, conforme indicado pelo símbolo (+). Os resultados do teste NEMENYI reforçam esta conclusão, com P-VALORES muito próximos de zero. Esses dados podem ser visualizados graficamente através da Figura 5.19.

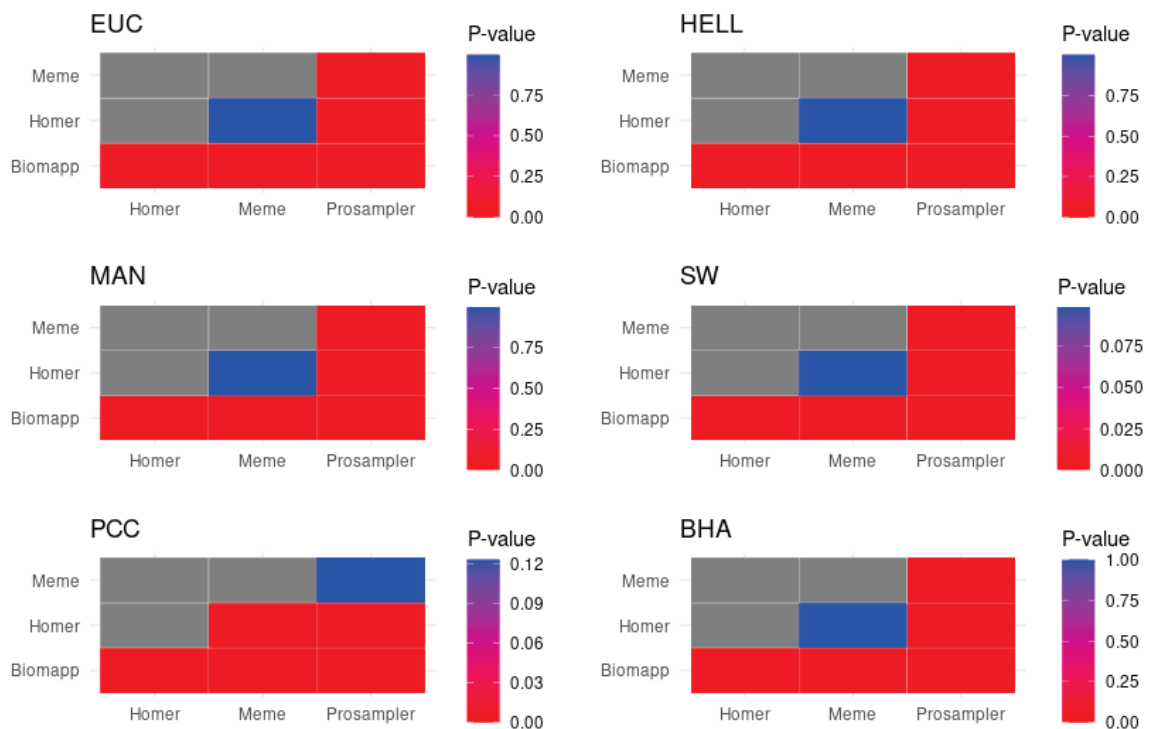


Figura 5.19: Os resultados dos P-VALUES derivados do teste NEMENYI indicam a presença de diferenças estatísticas significativas entre o programa BIOMAPP::CHIP e as demais abordagens testadas em todas as métricas avaliadas. Adicionalmente, nota-se que MEME não apresenta diferenças estatísticas significativas em relação a HOMER exceto na métrica PCC. Neste último, tanto MEME quanto PROSAMPLER foram estatisticamente semelhantes. Fonte: o autor.

Analisando esta figura, podemos verificar que os resultados obtidos no teste NEMENYI apontam para diferenças estatisticamente significativas entre o programa BIOMAPP::CHIP e as

demais metodologias avaliadas, abrangendo todas as métricas em questão. É relevante destacar que o MEME foi estatisticamente semelhante ao HOMER em todas as métricas de distância. Porém, na métrica PCC, o programa MEME se mostrou mais semelhante ao PROSAMPLER, não apresentando diferenças estatisticamente significativas entre eles neste quesito. A semelhança entre o MEME e o HOMER nas métricas de distância e a proximidade entre o MEME e o PROSAMPLER na métrica PCC podem ser atribuídas a vários fatores. Isso pode incluir a natureza dos algoritmos, a sensibilidade a valores discrepantes, peculiaridades dos dados ou até mesmo otimização para diferentes funções de perda. Cada métrica pode estar destacando diferentes aspectos das relações entre as abordagens, o que pode contribuir para esse comportamento.

Por último, a Figura 5.20 apresenta os gráficos de distâncias críticas entre os diferentes algoritmos, proporcionando uma visualização intuitiva das posições relativas de cada método em termos de desempenho. A distância crítica é um valor-chave obtido a partir de testes estatísticos, como o teste post-hoc NEMENYI, e serve como um limite para determinar se as diferenças de desempenho entre vários algoritmos são estatisticamente significativas. Esta medida é importante porque fornece um critério objetivo para avaliar e comparar a eficácia de diferentes métodos. Se a diferença de desempenho entre dois algoritmos exceder a distância crítica, podemos concluir que um algoritmo é significativamente melhor que o outro com um certo nível de confiança.

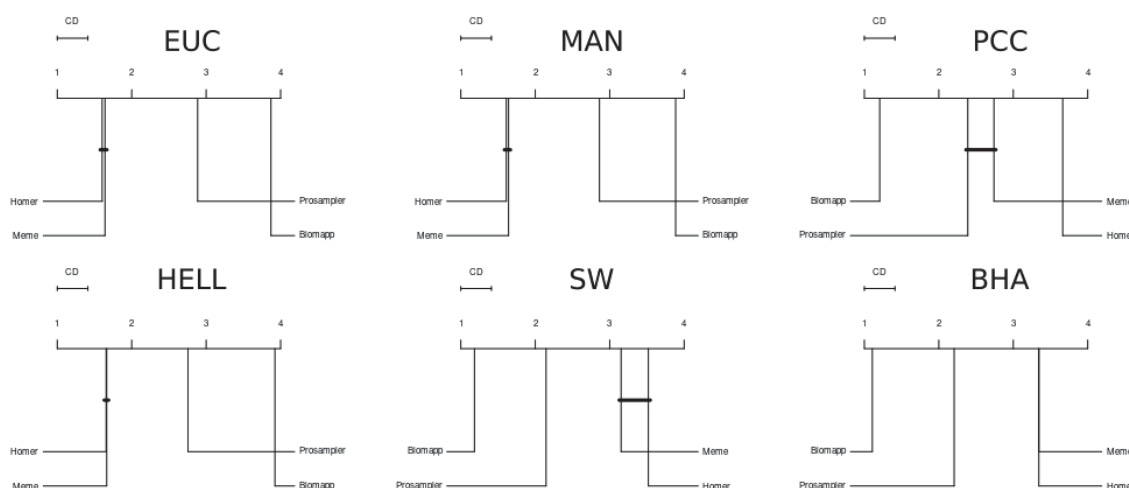


Figura 5.20: Comparação da distância crítica entre diferentes abordagens. A análise revela que o algoritmo BIOMAPP::CHIP se destaca, apresentando desempenho superior em relação às demais metodologias avaliadas. Fonte: o autor.

Observa-se claramente que o algoritmo BIOMAPP::CHIP obteve o melhor desempenho, demonstrando superioridade estatística em relação a todas as outras abordagens. Nota-se também uma estreita relação entre os algoritmos MEME e HOMER em diversas métricas, especialmente aquelas relacionadas à distância, corroborando a análise realizada na Figura 5.19. Da mesma forma, também é possível observar nesta figura que, na métrica PCC, o algoritmo MEME apresentou uma grande proximidade com o algoritmo PROSAMPLER, indicando que não há diferenças estatisticamente significativas entre eles nesta métrica específica.

5.4 CONCLUSÃO

Concluímos este capítulo reiterando a importância da contagem de k-mers na formulação de modelos probabilísticos robustos para a identificação de regiões genéticas com baixa variabilidade. Os resultados obtidos e as análises estatísticas realizadas corroboram firmemente

nossa hipótese inicial. Foi demonstrado que os algoritmos baseados na contagem de k-mers, particularmente SMT e SMT+EM desempenham um papel importante na detecção precisa de regiões genéticas de interesse e também estabelecem uma base consistente para a otimização subsequente.

6 CONCLUSÃO E CONSIDERAÇÕES FINAIS

Neste capítulo, apresentaremos as conclusões derivadas da pesquisa realizada ao longo do doutorado, enfatizando as principais contribuições para o campo da Ciência da Computação, Bioinformática e descoberta de MOTIFS biológicos. Discutiremos o impacto e a relevância dos resultados obtidos, refletindo sobre sua aplicabilidade prática e sugerindo possíveis direções para futuras investigações. Além disso, este capítulo incluirá uma seção dedicada às publicações geradas durante o período do doutorado, detalhando cada trabalho publicado ou aceito para publicação em revistas científicas e conferências.

6.1 CONCLUSÕES DE PESQUISA

Este trabalho apresentou o BIOMAPP::CHIP, um arcabouço inovador para a descoberta de MOTIFS biológicos em dados de CHIP-SEQ, que se destaca pelo uso eficiente da estrutura de dados SMT. A abordagem única do BIOMAPP::CHIP, combinando a contagem de k-mers através de árvores de sufixo com a geração de modelos PWM pré-otimizados, resultou em avanços significativos na identificação e análise de MOTIFS.

Nossa revisão bibliográfica abrangente detalhou o estado da arte e evidenciou as lacunas que o BIOMAPP::CHIP visou preencher. A metodologia empregada foi descrita com rigor, detalhando cada passo desde a contagem de k-mers até a formulação dos modelos PWM. A comparação do BIOMAPP::CHIP e SMT com outras ferramentas estabelecidas, como MEME, PROSAMPLER, HOMER, JELLYFISH e KANALIZE, foi meticulosamente conduzida utilizando tanto dados sintéticos quanto reais.

Os resultados obtidos são a confirmação da eficácia da SMT e do BIOMAPP::CHIP. Em comparação com o JELLYFISH e KANALIZE, a SMT contabilizou k-mers de maneira eficiente e também demonstrou uma capacidade superior na geração de modelos iniciais precisos e potencialmente mais próximos do ótimo global através do algoritmo KDIVE. Ademais, quando comparado com algoritmos de ponta, o BIOMAPP::CHIP exibiu desempenho superior, melhorando a identificação de MOTIFS biologicamente relevantes, que são importantes na compreensão das funções genômicas.

Para trabalhos futuros, entendemos que o aprimoramento contínuo das ferramentas e metodologias empregadas na descoberta de MOTIFS biológicos deverá ser continuado. Entre as direções possíveis, destacamos o desenvolvimento do K-FAST, uma versão avançada da SMT que estamos atualmente aperfeiçoando. O K-FAST visa oferecer uma abordagem ainda mais eficiente para a contagem de k-mers, otimizando o processamento de grandes volumes de dados.

Além disso, reconhecemos a necessidade do desenvolvimento de modelos probabilísticos mais sofisticados que incorporem a análise de gaps e as interações entre posições dentro dos MOTIFS. Esses modelos permitirão uma compreensão mais profunda das complexidades estruturais e funcionais dos MOTIFS, incluindo a influência de variações e mutações pontuais na atividade de ligação do fator de transcrição. A implementação de modelos mais avançados abrirá novas horizontes para a exploração das dinâmicas de interação DNA-proteína, contribuindo para a biologia molecular e a genômica funcional.

Além das fronteiras já exploradas, uma área de crescente interesse e desafiadora complexidade é a descoberta de MOTIFS em RNA. A natureza dinâmica e multifacetada do RNA, desde o m RNA até as variadas formas de RNA não-codificante, como mi RNA e lnc RNA, apresenta um conjunto singular de desafios que incluem a necessidade de compreender as

estruturas secundárias, as interações intramoleculares e intermoleculares, e o impacto dessas interações na regulação gênica e na expressão. A busca por MOTIFS em RNA envolve tanto a identificação de sequências quanto o entendimento da complexidade estrutural e funcional que essas sequências representam. Portanto, há uma necessidade premente de desenvolver ferramentas e abordagens computacionais específicas para o RNA que possam desvendar essas complexidades com precisão e eficiência. Essas ferramentas permitirão a exploração de novas camadas de regulação gênica e abrirão caminhos para terapias inovadoras baseadas em RNA, além de aprofundar nosso entendimento sobre os mecanismos moleculares subjacentes a diversas condições de saúde e doenças.

Em suma, o BIOMAPP::CHIP representa uma contribuição considerável para o campo da Ciência da Computação, Biologia Computacional e Bioinformática. Sua metodologia original e resultados promissores ampliaram o entendimento das complexas interações entre fatores de transcrição e DNA além de oferecer um ferramenta importante para o avanço científico na área da saúde. As implicações deste trabalho vão além da compreensão teórica, pois tem o potencial de impactar positivamente a cura de doenças ajudando no desenvolvimento de terapêuticas avançadas, incluindo novas drogas e vacinas.

6.2 PUBLICAÇÕES DECORRENTES DO DOUTORADO

Ao longo do doutorado, contribuições importantes foram feitas à área de descoberta de MOTIFS biológicos através de quatro publicações, refletindo avanços importantes na pesquisa e no desenvolvimento de ferramentas analíticas. Cada trabalho publicado representou um marco na trajetória da pesquisa conduzida, evidenciando a profundidade e a inovação das investigações realizadas.

O primeiro artigo publicado foi *Towards a Better Understanding of Heuristic Approaches Applied to the Biological Motif Discovery* na conferência BRACIS (*Brazilian Conference on Intelligent Systems*). Neste estudo, diversas técnicas de otimização foram exploradas e testadas no contexto do problema da descoberta de MOTIFS, tendo sido concluído que o algoritmo de Maximização da Expectativa (EM) destacou-se por apresentar resultados superiores, evidenciando sua promessa como técnica mais eficaz. Este estudo serviu como base para futuras investigações mais detalhadas na área.

Avançando na pesquisa, a combinação do ILS (*Iterated Local Search*) com EM e um algoritmo de mutação hierárquico foi apresentada no artigo *Expectation Maximization based algorithm applied to DNA sequence motif finder*, publicado no congresso CEC (*Congress on Evolutionary Computation*). A concepção do FAST-EM começou a ser delineada neste ponto, representando um avanço na abordagem para a descoberta de MOTIFS em sequências de DNA.

Em seguida, publicamos o artigo *SMT: a high-performance approach for counting kmers* na *17th International Joint Conference on Biomedical Engineering Systems and Technologies*, no qual a introdução da *Sparse Motif Tree* foi realizada. Ademais, foi neste trabalho que a ideia do K-FAST começou a ser desenvolvida, visando a otimização do processamento e análise de grandes volumes de dados genéticos.

Consolidando nossa série de contribuições, o artigo intitulado *Biomapp::chip: large-scale motif analysis* foi publicado na prestigiada revista *BMC Bioinformatics*. Este estudo detalha de maneira abrangente o framework que desenvolvemos, ressaltando os resultados mais impactantes obtidos e sublinhando a relevância de nossa pesquisa para o avanço do conhecimento na comunidade científica.

REFERÊNCIAS

- Aho, A. V. e Corasick, M. J. (1975). Efficient string matching: an aid to bibliographic search. *Communications of the ACM*, 18(6):333–340.
- Aho, A. V. e Hopcroft, J. E. (1974). *The design and analysis of computer algorithms*. Pearson Education India.
- Alon, U. (2007). Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8(6):450–461.
- Altschul, S. F. e Erickson, B. W. (1985). Significance of nucleotide sequence alignments: a method for random sequence permutation that preserves dinucleotide and codon usage. *Molecular biology and evolution*, 2(6):526–538.
- Altschul, S. F., Gish, W., Miller, W., Myers, E. W. e Lipman, D. J. (1990). Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410.
- Archbold, J. e Johnson, N. (1958). A construction for room's squares and an application in experimental design. *The Annals of Mathematical Statistics*, 29(1):219–225.
- Arnold, P., Erb, I., Pachkov, M., Molina, N. e van Nimwegen, E. (2012). Motevo: integrated bayesian probabilistic methods for inferring regulatory sites and motifs on multiple alignments of dna sequences. *Bioinformatics*, 28(4):487–494.
- Ashraf, F. B. e Shafi, M. S. R. (2020). Mfea: An evolutionary approach for motif finding in dna sequences. *Informatics in Medicine Unlocked*, 21:100466.
- Audano, P. e Vannberg, F. (2014). Kanalyze: a fast versatile pipelined k-mer toolkit. *Bioinformatics*, 30(14):2070–2072.
- Bailey, T. L. (2011). Dreme: motif discovery in transcription factor chip-seq data. *Bioinformatics*, 27(12):1653–1659.
- Bailey, T. L. (2021). Streme: accurate and versatile sequence motif discovery. *Bioinformatics*, 37(18):2834–2840.
- Bailey, T. L. e Elkan, C. (1995). Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine learning*, 21(1-2):51–80.
- Bailey, T. L., Elkan, C. et al. (1994). Fitting a mixture model by expectation maximization to discover motifs in bipolymers. *UCSD Technical Report CS94-351*.
- Bao, J., He, X. e Zhang, J. Z. (2021). Deepbsp—a machine learning method for accurate prediction of protein–ligand docking structures. *Journal of Chemical Information and Modeling*, 61(5):2231–2240.
- Benos, P. V., Bulyk, M. L. e Stormo, G. D. (2002). Additivity in protein–dna interactions: how good an approximation is it? *Nucleic acids research*, 30(20):4442–4451.
- Berg, O. G. e von Hippel, P. H. (1988). Selection of dna binding sites by regulatory proteins. *Trends in biochemical sciences*, 13(6):207–211.

- Berger, M. F. e Bulyk, M. L. (2009). Universal protein-binding microarrays for the comprehensive characterization of the dna-binding specificities of transcription factors. *Nature protocols*, 4(3):393–411.
- Bi, C. (2012). Memetic algorithms for de novo motif-finding in biomedical sequences. *Artificial Intelligence in Medicine*, 56(1):1–17.
- Bishop, C. M. e Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.
- Blat, Y. e Kleckner, N. (1999). Cohesins bind to preferential sites along yeast chromosome iii, with differential regulation along arms versus the centric region. *Cell*, 98(2):249–259.
- Boland, A., Cherry, G. e Dickson, R. (2017). *Doing a systematic review: A student's guide*. Sage.
- Buhler, J. e Tompa, M. (2002). Finding motifs using random projections. *Journal of computational biology*, 9(2):225–242.
- Burge, C. e Karlin, S. (1997). Prediction of complete gene structures in human genomic dna. *Journal of molecular biology*, 268(1):78–94.
- Caporaso, J. G., Lauber, C. L., Walters, W. A., Berg-Lyons, D., Huntley, J., Fierer, N., Owens, S. M., Betley, J., Fraser, L., Bauer, M. et al. (2012). Ultra-high-throughput microbial community analysis on the illumina hiseq and miseq platforms. *The ISME journal*, 6(8):1621–1624.
- Castro-Mondragon, J. A., Riudavets-Puig, R., Rauluseviciute, I., Berhanu Lemma, R., Turchi, L., Blanc-Mathieu, R., Lucas, J., Boddie, P., Khan, A., Manosalva Pérez, N. et al. (2022). Jasp2022: the 9th release of the open-access database of transcription factor binding profiles. *Nucleic acids research*, 50(D1):D165–D173.
- Chen, C., Hou, J., Shi, X., Yang, H., Birchler, J. A. e Cheng, J. (2021a). Deepgrn: prediction of transcription factor binding site across cell-types using attention-based deep neural networks. *BMC bioinformatics*, 22(1):1–18.
- Chen, W.-C., OstroUCHOV, G., Schmidt, D., Patel, P., Yu, H., Heckendorf, C. e Ripley, B. (2021b). Package ‘pbdmpi’. *R Vignette*, URL <http://cran.r-project.org/package=pbdMPI>.
- Colombo, N. e Vlassis, N. (2015). Fastmotif: spectral sequence motif discovery. *Bioinformatics*, 31(16):2623–2631.
- Congdon, C. B., Fizer, C. W., Smith, N. W., Gaskins, H. R., Aman, J., Nava, G. M. e Mattingly, C. (2005). Preliminary results for gami: A genetic algorithms approach to motif inference. Em *Computational Intelligence in Bioinformatics and Computational Biology*, 2005. CIBCB'05. *Proceedings of the 2005 IEEE Symposium on*, páginas 1–8. IEEE.
- Conlon, E. M., Liu, X. S., Lieb, J. D. e Liu, J. S. (2003). Integrating regulatory motif discovery and genome-wide expression analysis. *Proceedings of the National Academy of Sciences*, 100(6):3339–3344.
- Consortium, E. P. et al. (2012). An integrated encyclopedia of dna elements in the human genome. *Nature*, 489(7414):57.
- Crick, F. (1970). Central dogma of molecular biology. *Nature*, 227(5258):561–563.

- Crooks, G. E., Hon, G., Chandonia, J.-M. e Brenner, S. E. (2004). Weblogo: a sequence logo generator. *Genome research*, 14(6):1188–1190.
- Cui, Y., Wang, Q., Stormo, G. D. e Calvo, J. M. (1995). A consensus sequence for binding of lrp to dna. *Journal of bacteriology*, 177(17):4872–4880.
- Dassi, E. e Quattrone, A. (2016). Dynamit: the dynamic motif integration toolkit. *Nucleic acids research*, 44(1):e2–e2.
- Deisenroth, M. P., Faisal, A. A. e Ong, C. S. (2020). *Mathematics for machine learning*. Cambridge University Press.
- Dempster, A. P., Laird, N. M. e Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.
- D’haeseleer, P. (2006a). How does dna sequence motif discovery work? *Nature biotechnology*, 24(8):959–961.
- D’haeseleer, P. (2006b). What are dna sequence motifs? *Nature biotechnology*, 24(4):423–425.
- Dhiman, G. e Kumar, V. (2018). Emperor penguin optimizer: A bio-inspired algorithm for engineering problems. *Knowledge-Based Systems*, 159:20–50.
- Ding, J., Dhillon, V., Li, X. e Hu, H. (2015). Systematic discovery of cofactor motifs from chip-seq data by siomics. *Methods*, 79:47–51.
- Ding, J., Hu, H. e Li, X. (2014). Siomics: a novel approach for systematic identification of motifs in chip-seq data. *Nucleic acids research*, 42(5):e35–e35.
- Drake, A. W. (1967). Discrete—state markov processes.
- Dudek, C.-A. e Jahn, D. (2022). Prodoric: state-of-the-art database of prokaryotic gene regulation. *Nucleic Acids Research*, 50(D1):D295–D302.
- DURBIN, R., EDY, S. R., KROGH, A. e MITCHISON, G. (1998). *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge University Press, UK, 17th edition.
- Evans, P. A., Smith, A. D. e Wareham, H. T. (2003). On the complexity of finding common approximate substrings. *Theoretical Computer Science*, 306(1-3):407–430.
- Fareed, H., Jang, K., Lee, W., Kim, I. S. e Han, S. (2022). Dehydroxylation-assisted self-crosslinking of mxene-based pervaporation membranes for treating high-salinity water. *Journal of Industrial and Engineering Chemistry*.
- Fischer, J., I, T., Köppl, D. e Sadakane, K. (2018). Lempel–ziv factorization powered by space efficient suffix trees. *Algorithmica*, 80:2048–2081.
- Fisher, R. A. (1922). On the interpretation of χ^2 from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society*, 85(1):87–94.
- Fitch, W. M. (1983). Random sequences. *Journal of Molecular Biology*, 163(2):171–176.

- Fogel, G. B., Weekes, D. G., Varga, G., Dow, E. R., Harlow, H. B., Onyia, J. E. e Su, C. (2004). Discovery of sequence motifs related to coexpression of genes using evolutionary computation. *Nucleic Acids Research*, 32(13):3826–3835.
- Furey, T. S. (2012). Chip-seq and beyond: new and improved methodologies to detect and characterize protein–dna interactions. *Nature Reviews Genetics*, 13(12):840–852.
- Garbelini, J. C., Kashiwabara, A. Y., Sanches, D. S. e Pozo, A. T. (2024). Smt: A high-performance approach for counting kmers. Em *In Proceedings of the 17th International Joint Conference on Biomedical Engineering Systems and Technologies*, páginas 545–552.
- Garbelini, J. M. C., Kashiwabara, A. Y. e Sanches, D. S. (2018). Sequence motif finder using memetic algorithm. *BMC bioinformatics*, 19(1):1–13.
- Garbelini, J. M. C., Sanches, D. S. e Pozo, A. T. R. (2022a). Expectation maximization based algorithm applied to dna sequence motif finder. Em *2022 IEEE Congress on Evolutionary Computation (CEC)*, páginas 1–8. IEEE.
- Garbelini, J. M. C., Sanches, D. S. e Pozo, A. T. R. (2022b). Towards a better understanding of heuristic approaches applied to the biological motif discovery. Em *Brazilian Conference on Intelligent Systems*, páginas 180–194. Springer.
- Gohardani, S. A., Bagherian, M. e Vaziri, H. (2019). A multi-objective imperialist competitive algorithm (moica) for finding motifs in dna sequences. *Mathematical biosciences and engineering*, 16(3):1575–1596.
- González-Álvarez, D. L., Vega-Rodriguez, M. A., Gomez-Pulido, J. A. e Sánchez-Pérez, J. M. (2011). Finding motifs in dna sequences applying a multiobjective artificial bee colony (moabc) algorithm. Em *European Conference on Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics*, páginas 89–100. Springer.
- Gumbel, E. J. (1935). Les valeurs extrêmes des distributions statistiques. Em *Annales de l'institut Henri Poincaré*, volume 5, páginas 115–158.
- Hamilton, W. (1861). *Discussions on Philosophy and Literature, Education and University Reform: Chiefly from the Edinburgh Review; Cor., Vindicated, Enl., in Notes and Appendices*. Harper & Brothers.
- Heinz, S., Benner, C., Spann, N., Bertolino, E., Lin, Y. C., Laslo, P., Cheng, J. X., Murre, C., Singh, H. e Glass, C. K. (2010). Simple combinations of lineage-determining transcription factors prime cis-regulatory elements required for macrophage and b cell identities. *Molecular cell*, 38(4):576–589.
- Hertz, G. Z. e Stormo, G. D. (1999). Identifying dna and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics*, 15(7):563–577.
- Hu, J., Hu, H. e Li, X. (2008). Mopat: a graph-based method to predict recurrent cis-regulatory modules from known motifs. *Nucleic acids research*, 36(13):4488–4497.
- Hu, J., Yang, Y. D. e Kihara, D. (2006). Emd: an ensemble algorithm for discovering regulatory motifs in dna sequences. *BMC bioinformatics*, 7(1):1–13.

- Hughes, J. D., Estep, P. W., Tavazoie, S. e Church, G. M. (2000). Computational identification of cis-regulatory elements associated with groups of functionally related genes in *saccharomyces cerevisiae*. *Journal of molecular biology*, 296(5):1205–1214.
- Illumina, I. (2015). An introduction to next-generation sequencing technology. *Illumina, Inc.*
- Jacob, F. e Monod, J. (1961). Genetic regulatory mechanisms in the synthesis of proteins. *Journal of molecular biology*, 3(3):318–356.
- Jensen, S. T., Liu, X. S., Zhou, Q. e Liu, J. S. (2004). Computational discovery of gene regulatory binding motifs: a bayesian perspective. *Statistical Science*, 19(1):188–204.
- Jia, C., Carson, M. B., Wang, Y., Lin, Y. e Lu, H. (2014). A new exhaustive method and strategy for finding motifs in chip-enriched regions. *PLoS One*, 9(1):e86044.
- Jia, C., Carson, M. B. e Yu, J. (2013). A fast weak motif-finding algorithm based on community detection in graphs. *BMC bioinformatics*, 14(1):1–14.
- Jolma, A., Yan, J., Whittington, T., Toivonen, J., Nitta, K. R., Rastas, P., Morgunova, E., Enge, M., Taipale, M., Wei, G. et al. (2013). Dna-binding specificities of human transcription factors. *Cell*, 152(1-2):327–339.
- Jones, N. C. e Pevzner, P. A. (2004). *An introduction to bioinformatics algorithms*. MIT press.
- Kaya, M. (2009). Mogamod: Multi-objective genetic algorithm for motif discovery. *Expert Systems with Applications*, 36(2):1039–1047.
- Keich, U. e Pevzner, P. A. (2002a). Finding motifs in the twilight zone. Em *Proceedings of the sixth annual international conference on Computational biology*, páginas 195–204.
- Keich, U. e Pevzner, P. A. (2002b). Subtle motifs: defining the limits of motif finding algorithms. *Bioinformatics*, 18(10):1382–1390.
- Kelil, A., Dubreuil, B., Levy, E. D. e Michnick, S. W. (2014). Fast and accurate discovery of degenerate linear motifs in protein sequences. *PLoS One*, 9(9):e106081.
- Klinck, R., Serup, P., Madsen, O. D. e Jørgensen, M. C. (2008). Specificity of four monoclonal anti-nkx6-1 antibodies. *Journal of Histochemistry & Cytochemistry*, 56(4):415–424.
- Kulakovskiy, I. V., Vorontsov, I. E., Yevshin, I. S., Sharipov, R. N., Fedorova, A. D., Rumynskiy, E. I., Medvedeva, Y. A., Magana-Mora, A., Bajic, V. B., Papatsenko, D. A. et al. (2018). Hocomoco: towards a complete collection of transcription factor binding models for human and mouse via large-scale chip-seq analysis. *Nucleic acids research*, 46(D1):D252–D259.
- Ladunga, I. (2010). *Computational Biology of Transcription Factor Binding*. Springer.
- Landt, S. G., Marinov, G. K., Kundaje, A., Kheradpour, P., Pauli, F., Batzoglou, S., Bernstein, B. E., Bickel, P., Brown, J. B., Cayting, P. et al. (2012). Chip-seq guidelines and practices of the encode and modencode consortia. *Genome research*, 22(9):1813–1831.
- Lawrence, C. E., Altschul, S. F., Boguski, M. S., Liu, J. S., Neuwald, A. F. e Wootton, J. C. (1993). Detecting subtle sequence signals: a gibbs sampling strategy for multiple alignment. *science*, 262(5131):208–214.

- Lawrence, C. E. e Reilly, A. A. (1990). An expectation maximization (em) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *Proteins: Structure, Function, and Bioinformatics*, 7(1):41–51.
- Le, T., Altman, T. e Gardiner, K. (2010). Higeda: a hierarchical gene-set genetics based algorithm for finding subtle motifs in biological sequences. *Bioinformatics*, 26(3):302–309.
- Lee, T.-Y., Chen, Y.-J., Lu, T.-C., Huang, H.-D. e Chen, Y.-J. (2011a). Snosite: exploiting maximal dependence decomposition to identify cysteine s-nitrosylation with substrate site specificity. *PloS one*, 6(7):e21849.
- Lee, T.-Y., Lin, Z.-Q., Hsieh, S.-J., Bretaña, N. A. e Lu, C.-T. (2011b). Exploiting maximal dependence decomposition to identify conserved motifs from a group of aligned signal sequences. *Bioinformatics*, 27(13):1780–1787.
- Lesurf, R., Cotto, K. C., Wang, G., Griffith, M., Kasaian, K., Jones, S. J., Montgomery, S. B., Griffith, O. L. e Consortium, O. R. A. (2016). Oreganno 3.0: a community-driven resource for curated regulatory annotation. *Nucleic acids research*, 44(D1):D126–D132.
- Li, M., Ma, B. e Wang, L. (2002). Finding similar regions in many sequences. *Journal of Computer and System Sciences*, 65(1):73–96.
- Li, Y., Ni, P., Zhang, S., Li, G. e Su, Z. (2019). Prosampler: an ultrafast and accurate motif finder in large chip-seq datasets for combinatory motif discovery. *Bioinformatics*, 35(22):4632–4639.
- Liang, J., Yu, L., Yin, J. e Savage-Dunn, C. (2007). Transcriptional repressor and activator activities of sma-9 contribute differentially to bmp-related signaling outputs. *Developmental biology*, 305(2):714–725.
- Liu, F. F., Tsai, J. J., Chen, R.-M., Chen, S. e Shih, S. (2004). Fmga: finding motifs by genetic algorithm. Em *Bioinformatics and Bioengineering, 2004. BIBE 2004. Proceedings. Fourth IEEE Symposium on*, páginas 459–466. IEEE.
- Liu, J. e Stormo, G. D. (2005). Combining selex with quantitative assays to rapidly obtain accurate models of protein–dna interactions. *Nucleic acids research*, 33(17):e141–e141.
- Liu, J. S., Neuwald, A. F. e Lawrence, C. E. (1995). Bayesian models for multiple local sequence alignment and gibbs sampling strategies. *Journal of the American statistical Association*, 90(432):1156–1170.
- Liu, X., Brutlag, D. L., Liu, J. S. et al. (2001). Bioprospector: discovering conserved dna motifs in upstream regulatory regions of co-expressed genes. Em *Pacific symposium on biocomputing*, volume 6, página 38.
- Liu, X. S., Brutlag, D. L. e Liu, J. S. (2002). An algorithm for finding protein–dna binding sites with applications to chromatin-immunoprecipitation microarray experiments. *Nature biotechnology*, 20(8):835–839.
- Luo, C., Tsementzi, D., Kyrpides, N., Read, T. e Konstantinidis, K. T. (2012). Direct comparisons of illumina vs. roche 454 sequencing technologies on the same microbial community dna sample. *PloS one*, 7(2):e30087.

- Mahony, S. e Benos, P. V. (2007). Stamp: a web tool for exploring dna-binding motif similarities. *Nucleic acids research*, 35(suppl_2):W253–W258.
- Marçais, G. e Kingsford, C. (2011). A fast, lock-free approach for efficient parallel counting of occurrences of k-mers. *Bioinformatics*, 27(6):764–770.
- Mathelier, A. e Wasserman, W. W. (2013). The next generation of transcription factor binding site prediction. *PLoS Comput Biol*, 9(9):e1003214.
- Matys, V., Kel-Margoulis, O. V., Fricke, E., Liebich, I., Land, S., Barre-Dirrie, A., Reuter, I., Chekmenev, D., Krull, M., Hornischer, K. et al. (2006). Transfac® and its module transcompel®: transcriptional gene regulation in eukaryotes. *Nucleic acids research*, 34(suppl_1):D108–D110.
- McLaughlin, W. A. e Berman, H. M. (2003). Statistical models for discerning protein structures containing the dna-binding helix-turn-helix motif. *Journal of molecular biology*, 330(1):43–55.
- McLaughlin, W. A., Kulp, D. W., De La Cruz, J., Lu, X.-J., Lawson, C. L. e Berman, H. M. (2005). A structure-based method for identifying dna-binding proteins and their sites of dna-interaction. *Journal of structural and functional genomics*, 5(4):255–265.
- Moses, A. M., Chiang, D. Y. e Eisen, M. B. (2003). Phylogenetic motif detection by expectation-maximization on evolutionary mixtures. Em *Biocomputing 2004*, páginas 324–335. World Scientific.
- Mount, D. W. (2008). Comparison of the pam and blosum amino acid substitution matrices. *Cold Spring Harbor Protocols*, 2008(6):pdb–ip59.
- Nakato, R. e Sakata, T. (2021). Methods for chip-seq analysis: a practical workflow and advanced applications. *Methods*, 187:44–53.
- Neil, A. J., Liang, M. U., Khristich, A. N., Shah, K. A. e Mirkin, S. M. (2018). Rna–dna hybrids promote the expansion of friedreich’s ataxia (gaa) n repeats via break-induced replication. *Nucleic acids research*, 46(7):3487–3497.
- Ngo, V., Wang, M. e Wang, W. (2019). Finding de novo methylated dna motifs. *Bioinformatics*, 35(18):3287–3293.
- Norvig, P. e Russell, S. (2013). *Inteligência Artificial*. Elsevier, USA, 3th edition.
- Park, P. J. (2009). Chip–seq: advantages and challenges of a maturing technology. *Nature reviews genetics*, 10(10):669–680.
- Pavesi, G., Mauri, G. e Pesole, G. (2001). An algorithm for finding signals of unknown length in dna sequences. *Bioinformatics*, 17(suppl 1):S207–S214.
- Pepke, S., Wold, B. e Mortazavi, A. (2009). Computation for chip-seq and rna-seq studies. *Nature methods*, 6(11):S22–S32.
- Ptashne, M. e Hopkins, N. (1968). The operators controlled by the lambda phage repressor. *Proceedings of the National Academy of Sciences*, 60(4):1282–1287.
- Quail, M. A., Kozarewa, I., Smith, F., Scally, A., Stephens, P. J., Durbin, R., Swerdlow, H. e Turner, D. J. (2008). A large genome center’s improvements to the illumina sequencing system. *Nature methods*, 5(12):1005–1010.

- Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.
- Reddy, C. K., Weng, Y.-C. e Chiang, H.-D. (2006). Refining motifs by improving information content scores using neighborhood profile search. *Algorithms for Molecular Biology*, 1(1):1–14.
- Reid, J. E. e Wernisch, L. (2014). Steme: a robust, accurate motif finder for large data sets. *PloS one*, 9(3):e90735.
- Riza, L. S., Dhiba, T. F., Setiawan, W., Hidayat, T. e Fahsi, M. (2019). Parallel random projection using r high performance computing for planted motif search. *Telkomnika*, 17(3):1352–1359.
- Robertson, G., Hirst, M., Bainbridge, M., Bilenky, M., Zhao, Y., Zeng, T., Euskirchen, G., Bernier, B., Varhol, R., Delaney, A. et al. (2007). Genome-wide profiles of stat1 dna association using chromatin immunoprecipitation and massively parallel sequencing. *Nature methods*, 4(8):651–657.
- Robison, K., McGuire, A. M. e Church, G. M. (1998). A comprehensive library of dna-binding site matrices for 55 proteins applied to the complete escherichia coli k-12 genome. *Journal of molecular biology*, 284(2):241–254.
- Sabo, P. J., Humbert, R., Hawrylycz, M., Wallace, J. C., Dorschner, M. O., McArthur, M. e Stamatoyannopoulos, J. A. (2004). Genome-wide identification of dnasei hypersensitive sites using active chromatin sequence libraries. *Proceedings of the National Academy of Sciences*, 101(13):4537–4542.
- Sagot, M.-F. (1998). Spelling approximate repeated or common motifs using a suffix tree. Em *LATIN 98: Theoretical Informatics*, páginas 374–390. Springer.
- Sanderson, C. e Curtin, R. (2016). Armadillo: a template-based c++ library for linear algebra. *Journal of Open Source Software*, 1(2):26.
- Sayers, E. W., Bolton, E. E., Brister, J. R., Canese, K., Chan, J., Comeau, D. C., Farrell, C. M., Feldgarden, M., Fine, A. M., Funk, K. et al. (2023). Database resources of the national center for biotechnology information in 2023. *Nucleic acids research*, 51(D1):D29–D38.
- Schmidt, D., Wilson, M. D., Spyrou, C., Brown, G. D., Hadfield, J. e Odom, D. T. (2009). Chip-seq: using high-throughput sequencing to discover protein–dna interactions. *Methods*, 48(3):240–248.
- Schneider, T. D. e Stephens, R. M. (1990). Sequence logos: a new way to display consensus sequences. *Nucleic acids research*, 18(20):6097–6100.
- Shashidhara, H., Joseph, P. e Srinivasa, K. (2010). Improving motif refinement using hybrid expectation maximization and random projection. Em *Proceedings of the International Symposium on Biocomputing*, páginas 1–7.
- Shendure, J. e Ji, H. (2008). Next-generation dna sequencing. *Nature biotechnology*, 26(10):1135–1145.
- Siddharthan, R. (2010). Dinucleotide weight matrices for predicting transcription factor binding sites: generalizing the position weight matrix. *PloS one*, 5(3):e9722.

- Sinha, S. e Tompa, M. (2003). Ymf: a program for discovery of novel transcription factor binding sites by statistical overrepresentation. *Nucleic acids research*, 31(13):3586–3588.
- Sloan, C. A., Chan, E. T., Davidson, J. M., Malladi, V. S., Strattan, J. S., Hitz, B. C., Gabdank, I., Narayanan, A. K., Ho, M., Lee, B. T. et al. (2016). Encode data at the encode portal. *Nucleic acids research*, 44(D1):D726–D732.
- Smit, A. F., Hubley, R. e Green, P. (1996). Repeatmasker.
- Staden, R. (1989). Methods for discovering novel motifs in nucleic acid sequences. *Bioinformatics*, 5(4):293–298.
- Stine, M., Dasgupta, D. e Mukatira, S. (2003). Motif discovery in upstream sequences of coordinately expressed genes. Em *The 2003 Congress on Evolutionary Computation, 2003. CEC'03.*, volume 3, páginas 1596–1603. IEEE.
- Stormo, G. D. (2000). Dna binding sites: representation and discovery. *Bioinformatics*, 16(1):16–23.
- Stormo, G. D. e Fields, D. S. (1998). Specificity, free energy and information content in protein–dna interactions. *Trends in biochemical sciences*, 23(3):109–113.
- Stormo, G. D. e Hartzell, G. W. (1989). Identifying protein-binding sites from unaligned dna fragments. *Proceedings of the National Academy of Sciences*, 86(4):1183–1187.
- Tatusov, R. e Lipman, D. (1996). Dust, in the ncbi. *Toolkit available at <http://blast.wustl.edu/pub/dust>*.
- Teixeira, A. A. e Queirós, A. S. (2016). Economic growth, human capital and structural change: A dynamic panel data analysis. *Research policy*, 45(8):1636–1648.
- Teixeira, M. C., Monteiro, P., Jain, P., Tenreiro, S., Fernandes, A. R., Mira, N. P., Alenquer, M., Freitas, A. T., Oliveira, A. L. e Sá-Correia, I. (2006). The yeasttract database: a tool for the analysis of transcription regulatory associations in *saccharomyces cerevisiae*. *Nucleic acids research*, 34(suppl_1):D446–D451.
- Temin, H. M., Mizutami, S. et al. (1970). Rna-dependent dna polymerase in virions of rous sarcoma virus. *Nature*, 226:1211–1213.
- Thijs, G., Lescot, M., Marchal, K., Rombauts, S., De Moor, B., Rouze, P. e Moreau, Y. (2001a). A higher-order background model improves the detection of promoter regulatory elements by gibbs sampling. *Bioinformatics*, 17(12):1113–1122.
- Thijs, G., Lescot, M., Marchal, K., Rombauts, S., De Moor, B., Rouze, P. e Moreau, Y. (2001b). A higher-order background model improves the detection of promoter regulatory elements by gibbs sampling. *Bioinformatics*, 17(12):1113–1122.
- Thompson, J. A. e Congdon, C. B. (2014). Gami-crm: using de novo motif inference to detect cis-regulatory modules. Em *2014 IEEE Congress on Evolutionary Computation (CEC)*, páginas 1022–1029. IEEE.

- Tierrafría, V. H., Rioualen, C., Salgado, H., Lara, P., Gama-Castro, S., Lally, P., Gómez-Romero, L., Peña-Loredo, P., López-Almazo, A. G., Alarcón-Carranza, G. et al. (2022). Regulondb 11.0: Comprehensive high-throughput datasets on transcriptional regulation in *Escherichia coli* K-12. *Microbial Genomics*, 8(5):000833.
- Tompa, M. (1999). An exact method for finding short motifs in sequences, with application to the ribosome binding site problem. Em *ISMB*, volume 99, páginas 262–271.
- Tompa, M., Li, N., Bailey, T. L., Church, G. M., De Moor, B., Eskin, E., Favorov, A. V., Frith, M. C., Fu, Y., Kent, W. J. et al. (2005). Assessing computational tools for the discovery of transcription factor binding sites. *Nature biotechnology*, 23(1):137–144.
- Wang, L. e Jiang, T. (1994). On the complexity of multiple sequence alignment. *Journal of computational biology*, 1(4):337–348.
- Wang, R., Wang, Z., Wang, J. e Li, S. (2019). SpliceFinder: ab initio prediction of splice sites using convolutional neural network. *BMC bioinformatics*, 20:1–13.
- Wang, X. e Miao, Y. (2014). Gaem: a hybrid algorithm incorporating GA with EM for planted edited motif finding problem. *Current Bioinformatics*, 9(5):463–469.
- Wasserman, W. W. e Sandelin, A. (2004). Applied bioinformatics for the identification of regulatory elements. *Nature Reviews Genetics*, 5(4):276–287.
- Watson, J. (2012). *The double helix*. Hachette UK.
- Wei, Z. e Jensen, S. T. (2006). Game: detecting cis-regulatory elements using a genetic algorithm. *Bioinformatics*, 22(13):1577–1584.
- Weirauch, M. T., Yang, A., Albu, M., Cote, A. G., Montenegro-Montero, A., Drewe, P., Najafabadi, H. S., Lambert, S. A., Mann, I., Cook, K. et al. (2014). Determination and inference of eukaryotic transcription factor sequence specificity. *Cell*, 158(6):1431–1443.
- Whitaker, J. W., Chen, Z. e Wang, W. (2015). Predicting the human epigenome from DNA motifs. *Nature methods*, 12(3):265–272.
- Wingender, E., Dietze, P., Karas, H. e Knüppel, R. (1996). Transfac: a database on transcription factors and their DNA binding sites. *Nucleic acids research*, 24(1):238–241.
- Wingender, E., Schoeps, T., Haubrock, M., Krull, M. e Dönitz, J. (2018). TFCClass: expanding the classification of human transcription factors to their mammalian orthologs. *Nucleic acids research*, 46(D1):D343–D347.
- Yang, Y., Robertson, J. A., Guo, Z., Martinez, J., Coghlan, C. e Heath, L. S. (2019). Mcat: Motif combining and association tool. *Journal of Computational Biology*, 26(1):1–15.
- Yu, Q., Huo, H., Chen, X., Guo, H., Vitter, J. S. e Huan, J. (2015). An efficient algorithm for discovering motifs in large DNA data sets. *IEEE transactions on nanobioscience*, 14(5):535–544.
- Zang, C., Schones, D. E., Zeng, C., Cui, K., Zhao, K. e Peng, W. (2009). A clustering approach for identification of enriched domains from histone modification ChIP-seq data. *Bioinformatics*, 25(15):1952–1958.

- Zhang, M. e Marr, T. G. (1993). A weight array method for splicing signal analysis. *Computer applications in the biosciences: CABIOS*, 9(5):499–509.
- Zhang, Y., Liu, T., Meyer, C. A., Eeckhoute, J., Johnson, D. S., Bernstein, B. E., Nusbaum, C., Myers, R. M., Brown, M., Li, W. et al. (2008). Model-based analysis of chip-seq (macs). *Genome biology*, 9(9):1–9.
- Zvelebil, M. J. e Baum, J. O. (2007). *Understanding bioinformatics*. Garland Science.