Alinhamentos estruturais

By Laboratório de Bioinformática e Sistemas

12 de março de 2021

Alinhamentos estruturais: métodos de sobreposição de proteínas e outras moléculas

Vitor Pimentel dos Santos , Pedro Martins , Diego Mariano

Revisão: Lucianna Helene Santos

BIOINFO – Revista Brasileira de Bioinformática. Edição #01. Julho, 2021.

DOI: [10.51780/978-6-599-275326-07](https://doi.org/10.51780/978-6-599-275326-07)

Alinhamento estrutural consiste na tentativa de encontrar semelhanças entre duas estruturas tridimensionais de moléculas, como por exemplo, peptídeos, ligantes e proteínas [1]. A estrutura tridimensional de proteínas está estritamente ligada à sua atividade biológica em uma célula [2]. Portanto, conhecer a estrutura das proteínas e poder encontrar equivalências nessas estruturas é um fator crucial no campo da biologia estrutural [2]. Além disso, uma vez que estudos sugerem que a evolução tende a conservar a estrutura de macromoléculas, similaridades estruturais entre duas proteínas podem ser usadas como indícios de relações evolutivas ou funções comuns entre as estruturas sobrepostas [1]. Essas relações, por sua vez, são fundamentais na predição da interação entre proteínas [4]. Nesse contexto, a determinação entre similaridades entre estruturas se mostra de grande importância para uma gama de estudos em bioinformática estrutural.

No alinhamento estrutural, macromoléculas são sobrepostas visando encontrar regiões de similaridade. A Figura 1 ilustra o alinhamento estrutural entre duas enzimas beta-glicosidases classificadas como membros da família 1 das glicosídeo-hidrolases (GH1) na base de dados CAZy (*Carbohydrate-Active enZymes*; disponível em <http://www.cazy.org/>). Por serem proteínas de uma mesma família (e do mesmo clã GH-A), é esperado que beta-glicosidases GH1 possuam uma estrutura tridimensional parecida. Isso pode ser demonstrado sobrepondo as estruturas através de uma técnica de alinhamento estrutural, como na Figura 1.

Figura 1. Exemplo de alinhamento estrutural entre duas proteínas beta-glicosidase da família GH1. À esquerda vê-se a estrutura (cartoon verde) da beta-glicosidase do fungo *Humicola insolens* (PDB ID: 4MDP). À direita vê-se a estrutura (cartoon azul) da beta-glicosidase do cupim *Neotermes koshunensis* (PDB ID: 3VIK). Abaixo, vemos as duas estruturas sobrepostas por meio de alinhamento estrutural. Figura gerada usando PyMOL (<https://pymol.org/2/>). Fonte: próprio autor.

De fato, a família GH1 (e outras famílias pertencentes ao clã GH-A) é composta por enzimas que adotam uma estrutura de enovelamento denominada Barril TIM (β/α)⁸, em que oito fitas beta se intercalam com oito hélices-alfa dando a proteína a forma de um barril [12]. Quando comparamos as sequências (estrutura primária) dessas enzimas podemos perceber que elas apresentam menos de 40% de identidade, isto é, menos de 40% dos aminoácidos que compõem essas estruturas são similares (Figura 2). Isso demonstra uma das vantagens do alinhamento estrutural: encontrar macromoléculas com sequências pouco similares, mas com uma forma parecida.

Figura 2. Alinhamento entre seqüências da beta-glicosidase do fungo *Humicola insolens* (PDB ID: 4MDP) e da beta-glicosidase do cupim *Neotermes koshunensis* (PDB ID: 3VIK). Identidade de seqüência de 39%. Aminoácidos forma coloridos de acordo com o tipo. Gerado usando Clustal Omega (<https://www.ebi.ac.uk/Tools/msa/clustalo/>). Fonte: próprio autor.

Fundamentos do alinhamento estrutural

Fundamento 1 – considere o espaço 3D: partindo do princípio de que desejemos alinhar duas macromoléculas para determinar se possuem estruturas similares, devemos primeiramente considerar que estamos lidando com um ambiente tridimensional, isto é, composto por três dimensões (em geral, denominadas X, Y e Z). Considere que proteínas são compostas por aminoácidos, que por sua vez são compostos por átomos. Cada átomo é representado por uma coordenada tridimensional. Veja um exemplo de uma linha de um arquivo PDB que representa um átomo de nitrogênio da cadeia principal do aminoácido prolina de um peptídeo sintético semelhante ao colágeno (PDB ID: 1MDS):

```
ATOM 1 N PRO A 1 8.316 21.206 21.530 1.00 17.44 N
```

Os três números destacados representam as três posições X, Y e Z daquele átomo no espaço tridimensional. Logo, se desejamos sobrepor uma estrutura sobre a outra, precisamos alterar essas coordenadas.

Fundamento 2 – apenas uma estrutura é alterada: em um alinhamento par-a-par, apenas as coordenadas de uma estrutura devem ser alteradas. A outra estrutura é utilizada como referência para realização do alinhamento

(Figura 3). Caso tenha mais de duas estruturas para serem alinhadas, uma delas deverá ser utilizada como estrutura-referência, enquanto todas as outras serão alinhadas em relação a ela.

Figura 3. Alinhamento entre duas formas genéricas denominadas como A e B. Apenas B sofreu alterações, ou seja, em um alinhamento par-a-par, B é alinhado em comparação a A. Fonte: próprio autor.

Fundamento 3 – rotação e translação: no alinhamento estrutural, uma estrutura pode sofrer duas possíveis operações. São elas rotação e translação. No movimento de translação, um elemento é movido de um lugar para o outro sem alterar sua posição rotacional em relação a um referencial. No caso da rotação, esse movimento relaciona-se a girar determinado elemento, o que afeta outros elementos conectados. Se tratando de um ambiente tridimensional, esses conceitos podem parecer um pouco complexos (ainda mais quando os correlacionamos com interações moleculares).

Para entender o movimento de rotação, imagine um átomo representado como se fosse uma bola de futebol da marca BIOINFO (preste bastante atenção na marca). Imagine que a bola está parada, logo podemos rotacioná-la de seis formas possíveis, como na Figura 4:

Figura 4. Possíveis rotações que uma bola poderia sofrer. (1) rotação em sentido horário, (2) sentido anti-horário, (3) direita, (4) esquerda, (5) acima e (6) abaixo. Observe que todos os movimentos ocorrem com base em uma referência (no caso, o centro da bola). Fonte: próprio autor.

Observe que a bola se encontra é uma mesma posição, ou seja, apesar de estarmos rotacionando a bola, as coordenadas dela não estão sendo alteradas.

Agora, vamos supor que Joãozinho, um menino muito levado da rua, colou duas bolas de futebol usando supercola. Nesse caso, ao aplicar um movimento de rotação em qualquer uma das bolas, a outra também será afetada (Figura 5). Por exemplo, se rotacionarmos a bola 1, as coordenadas dela não serão alteradas (afinal ela se encontra no mesmo lugar). Entretanto, as coordenadas da bola 2 serão afetadas.

Figura 5. Ao rotacionar um elemento ligado a outro, a posição do segundo elemento que participa da interação será afetada. (A) Ilustração do movimento de rotação quando dois elementos estão unidos. Considere a bola 1 como elemento-referência. (B) Ao aplicar uma rotação a um elemento, ele continuará nas mesmas coordenadas, enquanto as coordenadas do outro serão afetadas. Por exemplo, considere que a bola 1 está presa ao chão (vamos considerar que a distância para o chão seja o eixo Y). Ao rotacioná-la, ela continuará presa ao chão (e.g., distância do eixo Y igual a 0 cm), mas a bola 2 não está mais (e.g., distância do eixo Y igual a 10 cm). Fonte: próprio autor.

Para entender a translação, observe o exemplo a seguir com um par de calçados (Figura 6). Na Figura 6A, dois calçados estão posicionados um ao lado do outro. A seguir, eles foram separados (Figura 6B), isto é, o chinelo à direita sofre um movimento de translação que amplia a distância para o chinelo à esquerda. Entretanto, note que a orientação não foi alterada (perceba como o chinelo à direita está levemente inclinado e mantém sua inclinação mesmo após a translação). Assim, podemos concluir que o movimento de translação desloca um elemento de um ponto a outro, mas sem alterar sua rotação (ou seja, apenas um dos eixos será alterado).

Figura 6. Ilustrando o movimento de translação usando calçados. (A) Chinelos estão próximos. (B) Chinelos são separados (i.e., o chinelo à direita realiza um movimento de translação). Note que a orientação dos chinelos não foi alterada. Apenas foi aplicada uma distância entre eles. Fonte: próprio autor.

Os movimentos de rotação e translação aplicados a átomos podem ser interpretados de forma similar. Átomos realizam interações químicas (no caso dos aminoácidos, eles estão interligados por ligações covalentes). Logo, ao aplicar um movimento de rotação em um dos átomos, as coordenadas de outros átomos interligados serão afetadas. Para ilustrar isso, observe a figura a seguir. Nela, dois aminoácidos triptofano em orientações e posições diferentes são representados por *sticks* verdes e azuis. Deseja-se sobrepor o triptofano azul sobre o triptofano verde. Observe como os movimentos de rotação e translação poderiam ser aplicados (Figura 7).

Figura 7. Alinhamento estrutural entre dois aminoácidos triptofano representados por *sticks* nas cores verde e azul ciano. (A) Aminoácidos em orientações e posições distintas. (B) Rotação dos eixos X e Y do triptofano azul visando se aproximar da posição final do triptofano verde. (C) Rotação considerando os eixos X e Z (observe que é difícil visualizar a representação do eixo Z, por essa se tratar de uma figura bidimensional). (D-E) Translação do triptofano azul para sobrepor o verde (em geral, a translação é realizada primeiro, mas para melhor ilustrar o processo, realizamos após a rotação). Figura gerada com PyMOL. Fonte: próprio autor.

Note que a sobreposição não é perfeita. Mesmo se tratando de um mesmo aminoácido, é natural que se encontre diferenças nas distâncias dos átomos devido às limitações dos métodos de determinação de estruturas, a própria natureza das ligações químicas e até mesmo a imprecisões dos algoritmos de alinhamento estrutural. Observe ainda que o processo de rotação e translação é bastante complexo quando se considera cada um dos átomos. Por exemplo, ao visar um melhor alinhamento entre os átomos de nitrogênio (em azul na figura), podemos ter um bom alinhamento considerando todo o aminoácido. Entretanto, os átomos remanescentes ainda não estarão bem alinhados. Digamos que a seguir, trasladamos e rotacionamos o átomo de oxigênio (em vermelho) para aperfeiçoar o posicionamento. Ao fazer isso, todas as coordenadas dos outros átomos serão afetadas, inclusive dos átomos de nitrogênio (que já haviam sido posicionadas corretamente). Assim, mais uma vez eles precisarão ser reorientados, o que irá mais uma vez afetar o alinhamento dos átomos de oxigênios (além de todos os outros). Perceba que não há uma solução perfeita para um alinhamento de estruturas. Logo, as ferramentas disponíveis usam diversos algoritmos e heurísticas para tentar obter uma boa sobreposição, balanceando precisão e desempenho.

Algoritmos de alinhamento estrutural

O problema do alinhamento entre estruturas tridimensionais possui um custo computacional maior do que o alinhamento de sequências. Frente a isso, algoritmos de modelagem e alinhamento estrutural eficientes são imprescindíveis dentro dessa área [3-4].

Por exemplo, dadas duas proteínas **A** e **B** de tamanhos m e n , o alinhamento estrutural visa encontrar subcadeias (vamos chamá-las aqui de **P** e **Q**), que tenham o mesmo tamanho, sejam similares e a correspondência seja a máxima possível [1]. A correspondência é definida como a associação de pares de átomos das duas proteínas que aparecem nas mesmas posições em suas respectivas subcadeias. O resultado de um alinhamento são subestruturas derivadas que tenham o maior grau de similaridade possível. Em geral, esse grau de similaridade é medido pela distância euclidiana entre os átomos correspondentes em cada estrutura [1].

Proteínas podem ser rotacionadas ou trasladadas sem ter sua estrutura afetada (no caso, considere as distâncias internas entre átomos). Então, pode-se fixar a proteína A e rotacionar a proteína B de forma a encontrar as maiores subcadeias P e Q que tenham alta similaridade e correspondência. Mais uma vez, devemos lembrar que, em um processo de alinhamento estrutural, é fundamental considerar a orientação e posição relativa em um espaço tridimensional das duas proteínas [1].

Sistemas de pontuação

Existe uma grande quantidade de algoritmos para alinhamento estrutural disponíveis. Assim, criar algoritmos novos e mais eficientes vem se tornando cada vez mais desafiador [4]. Em geral, tais algoritmos acompanham diversas métricas para medir o alinhamento. Uma métrica bastante utilizada é o *root-mean-square deviation* (RMSD), que é calculado após a superposição das estruturas [3], conforme visto na Equação 1:

$$\text{RMSD} = \sqrt{\frac{1}{N} \sum_{n=1}^N \delta_i^2}$$

(1)

em que δ é a distância entre átomos correspondentes de cada estrutura, e N é o número de pares de átomos correspondentes [13]. O RMSD é aplicável a qualquer subconjunto de uma proteína, pode ser calculado entre os carbonos-alfa ($C\alpha$) das estruturas ou apenas em resíduos de uma seção específica, ou pode ser calculado entre todos os átomos pesados [13].

Na prática quanto menor o valor de RMSD, mais similares são as estruturas. No exemplo a seguir, mostramos dois alinhamentos hipotéticos para duas estruturas exemplo (Figura 8). Observe que quão melhor uma estrutura sobrepõe a outra, menor o valor de RMSD.

Figura 8. Exemplo hipotético entre dois alinhamentos diferentes entre dois aminoácidos triptofano (verde e azul). Quanto mais bem sobrepostas, menor o RMSD. Gerado usando PyMOL. Fonte: próprio autor.

Algumas ferramentas podem apresentar diferentes métricas que vão além do RMSD, como por exemplo, o LS-Align que utiliza a métrica LS-Score [5] e a ferramenta TM-align, que utiliza o TM-Score [3]. Tanto o LS-Score quanto o TM-Score são o valor máximo de uma pontuação calculada com base no tamanho da estrutura sobre a qual será feita o alinhamento (no caso do TM-align, a primeira estrutura passada como parâmetro) e na soma de parâmetros obtidos a partir da distância entre os resíduos alinhados e de um fator de normalização. Por fim, ambas as métricas retornarão um valor entre 0 e 1 [3-5]. TM-score tem o valor no intervalo (0,1], onde 1 indica uma combinação perfeita entre duas estruturas. Nas próximas seções veremos detalhes de ferramentas que realizam alinhamento estrutural.

O alinhamento perfeito (RMSD = 0)

O melhor alinhamento estrutural possível teria um RMSD igual a zero. Podemos avaliar isso, alinhando uma estrutura com uma cópia idêntica a ela. No exemplo a seguir, usaremos a ferramenta PyMOL para realizar um alinhamento da lisozima 2LZM (em verde) contra uma cópia idêntica dela (em magenta). Note como as proteínas ficam perfeitamente sobrepostas (Figura 9).

Figura 9. Alinhamento da lisozima de PDB ID: 2LZM (exibida como um cartoon verde) com uma cópia idêntica dela (exibida como um cartoon na cor magenta). Imagem produzida usando PyMOL. Fonte: próprio autor.

Agora, observe o resultado do alinhamento exibido pelo PyMOL:

```
Executive: RMSD = 0.000 (162 to 162 atoms)
```

Note que todos os 162 átomos foram usados na comparação (foi usado apenas um átomo para cada resíduo). Como as coordenadas desses átomos são idênticas, o resultado final de RMSD será igual a zero, indicando um alinhamento perfeito.

Ferramentas para alinhamento estrutural

Na próxima seção, serão apresentadas algumas das principais ferramentas para realização de alinhamento estrutural:

- PyMOL
- TM-align
- Multiprot
- MUSTANG
- PROMALS3D

Alinhamento de estruturas 3D com PyMOL

PyMOL fornece diversos métodos para alinhamento estrutural de moléculas tanto por meio de sua interface simplificada quanto por meio de seu terminal de linhas comando. Para compreender como ele funciona na prática, vamos apresentar a seguir, uma série de estudos de caso utilizando-o para realizar alinhamentos de estruturas.

Estudo de caso 1: alinhamento entre duas beta-glicosidases

No exemplo a seguir, demonstraremos como utilizar a interface do PyMOL para realizar o alinhamento de duas proteínas. Para este estudo de caso, realizaremos o alinhamento da beta-glicosidase do fungo *Humicola insolens* (PDB ID: 4MDP) e da beta-glicosidase do cupim *Neotermes koshunensis* (PDB ID: 3VIK).

Inicialmente, acesse o terminal de comando da interface gráfica do PyMOL (Figura 10). A priori, utilizaremos o terminal apenas para fazer o *download* dos arquivos PDB que serão utilizados. Caso já os tenha em seu computador, você pode utilizar o menu *File > Open* para abrir manualmente os arquivos PDB.

Figura 10. Terminal de comandos do PyMOL. Comando *fetch* é usado para baixar estruturas. Fonte: próprio autor.

Para fazer o *download* dos arquivos utilize o comando *fetch* seguido do código PDB que se deseja carregar. Para o estudo de caso, execute os comandos:

```
fetch 4mdp  
fetch 3vik
```

Isso irá carregar a visualização tridimensional das duas estruturas das proteínas beta-glicosidase do fungo *Humicola insolens* (PDB ID: 4MDP), representada na cor verde, e de *Neotermes koshunensis* (PDB ID: 3VIK), representada na cor azul (Figura 11).

Figura 11. Visualizações das proteínas beta-glicosidase do fungo *Humicola insolens* (PDB ID: 4MDP) – cartoon verde – e da beta-glicosidase do cupim *Neotermes koshunensis* (PDB ID: 3VIK) – cartoon azul – na interface do PyMOL. Fonte: próprio autor.

Para alinhar as estruturas, podemos utilizar o painel à direita para selecionar uma ação de alinhamento (Figura 12). Vamos escolher uma das estruturas (no caso, selecionamos 3VIK) e clicar sobre o botão com a letra A (*action*) ao lado do nome da estrutura. Isso irá carregar um menu com diversas ações que podem ser aplicadas a estrutura desejada. Vamos até a opção *align* (alinhamento), depois em “to molecule (*/*CA)” e por fim em *4mdp* (pois iremos sobrepor 3vik a 4mdp).

Figura 12. Alinhamento estrutural usando a interface do PyMOL. Fonte: próprio autor.

Por padrão, o PyMOL realiza um alinhamento de sequências para determinar regiões similares e depois um refinamento é realizado usando as posições

dos carbonos-alfa de cada resíduo de aminoácido. Como resultado, pode-se observar as estruturas sobrepostas (Figura 13).

Figura 13. Estruturas sobrepostas de 3vik e 4mdp. Fonte: próprio autor.

PyMOL permite ainda visualizar o alinhamento de seqüências na interface acima do alinhamento estrutural. Para isso, basta clicar no botão com a letra S no canto inferior direito (Figura 14).

Figura 14. Alinhamento estrutural e de seqüências na interface do PyMOL. Fonte: próprio autor.

PyMOL permite ainda ampliar a visualização de uma região específica. Para isso, selecione a região desejada com base na visualização das seqüências alinhadas, ou clique sobre os aminoácidos desejados (você pode pressionar

SHIFT para selecionar mais de um aminoácido). A seguir, clique com o botão direito sobre a sequência selecionada e vá em *zoom* (Figura 15).

Figura 15. Ampliando uma região (*zoom*) na interface do PyMOL. Fonte: próprio autor.

O PyMOL irá gerar uma visualização mais próxima da região desejada (Figura 16). Você pode ainda usar o mesmo processo para centralizar a região desejada (no caso, clique na opção *center*). Ou ainda pode esconder a região clicando em *hide*, exibir os rótulos com nomes dos aminoácidos e átomos clicando em *label*, ou alterar as cores clicando em *color*.

Figura 16. Zoom em uma região alinhada no PyMOL. Fonte: próprio autor.

Estudo de caso 2: alinhamento entre duas lisozimas

Para compreendermos melhor como funciona o algoritmo de alinhamento padrão do PyMOL, neste novo exemplo, realizaremos um alinhamento da estrutura de cristal de uma lisozima (PDB ID: 2LZM) com uma estrutura cuja diferença se trata de um aminoácido não natural p-iodo-L-fenilalanina na posição 153 (PDB ID 1T6H).

Para realizar o alinhamento das estruturas, PyMOL inicialmente utiliza programação dinâmica para realizar um alinhamento de sequências. Na prática, ele realiza um alinhamento usando a ferramenta BLAST com base na matriz de pontuação BLOSUM62 (não precisamos entrar em detalhes, pois esses são parâmetros-padrão de execução do BLAST). Podemos simular essa primeira etapa executando o alinhamento das sequências usando a ferramenta BLAST web (disponível em <https://blast.ncbi.nlm.nih.gov>). Observe o resultado do alinhamento:

Alinhamento 2LZM x 1T6H									
Query	1	MNIFEMLRID	EGLRLKIYK	DEGYTIG	IHLTKSP	SLNAAKSEL	DKAIGRNC	NGVITK	60
Sbjct	1	MNIFEMLRID	EGLRLKIYK	DEGYTIG	IHLTKSP	SLNAAKSEL	DKAIGRNC	NGVITK	60
Query	61	DEAEKLFNQ	DVAAVRG	ILRNAKL	KPVYDSL	DAVRRCA	LNNVFQMG	ETGVAGFT	120
Sbjct	61	DEAEKLFNQ	DVAAVRG	ILRNAKL	KPVYDSL	DAVRRCA	LNNVFQMG	ETGVAGFT	120
Query	121	LQQRNDEA	AVNLAKSR	WYNQTP	NAKRVI	TTFTGT	TWDAYKNL		164
Sbjct	121	LQQRNDEA	AVNLAKSR	WYNQTP	NAKRVI	TTXRTG	TWDAYKNL		164

Aqui podemos ver que ambas as sequências possuem 164 aminoácidos com uma identidade superior a 99% (163 dos 164 resíduos coincidem). Veja que o alinhamento nos dá uma tabela de correspondência entre resíduos. Por exemplo, a metionina na posição 1 (resíduo M1) de 2LZM é equivalente a

metionina na posição 1 de 1T6H. O mesmo vale para N2, I3, F4, E5 e assim sucessivamente. De fato, essa regra valerá para quase todos os resíduos (exceto F153 de 2LZM). Por ter estruturas bastante similares, esse exemplo poderá facilitar a compreensão da primeira etapa da estratégia de alinhamento do PyMOL.

Agora que temos a tabela de correspondentes, vamos para a próxima etapa. A seguir, PyMOL coleta as coordenadas dos carbonos-alfa de cada um dos resíduos. Lembre-se que cada resíduo de aminoácido terá um único carbono-alfa localizado em sua cadeia principal e cada átomo desses terá três valores de coordenadas de posicionamento (aqui denominadas X, Y e Z). Na Figura 17 apresentamos uma visualização da estrutura de 2LZM (Figura 17A) e 1T6H (Figura 17D). Em Figura 17 B,E são apresentados apenas os carbonos-alfa de cada uma dessas proteínas e na Figura 17C,F uma linha interligando esses átomos foi inserida para destacar a forma da proteína.

Figura 17. Estruturas de 2LZM e 1T6H. (A) Visualização em cartoon e linhas azuis de 2LZM. (B) Visualização dos carbonos-alfa de 2LZM como esferas azuis. (C, F) Linhas que interligam os carbonos-alfa vizinhos foram utilizadas para destacar a forma da proteína. (D) Visualização em cartoon e linhas amarelas de 1T6H. (E) Visualização dos carbonos-alfa de 1T6H como esferas amarelas. Figura gerada com PyMOL. Fonte: próprio autor.

PyMOL irá utilizar apenas as posições das coordenadas desses átomos para a realização da sobreposição. É importante levar em consideração que apesar das proteínas serem bastante parecidas, as coordenadas dos átomos são bastante diferentes. Na Figura 18 podemos ver que, quando as duas estruturas são plotadas em um mesmo espaço conformacional, isto é, quando abertas em uma mesma sessão do PyMOL, os átomos estão localizados em posições diferentes.

Figura 18. Estruturas de 2LZM (azul) e 1T6H (amarelo) quando plotadas em um mesmo espaço conformacional. Fonte: próprio autor.

Para que possamos visualizar como uma estrutura se sobrepõe a outra, devemos alterar as coordenadas geográficas de uma das estruturas utilizando as regras de rotação e translação (lembre-se que se o algoritmo decidir rotacionar um dos átomos, todos os outros da mesma estrutura serão afetados). Assim, PyMOL utilizará a matriz de correspondência de resíduos para reposicionar os átomos da segunda proteína (no caso, escolhemos reposicionar os átomos de 1T6H).

Na Figura 19, vemos dois pares de resíduos correspondentes usados como exemplo. O primeiro deles é R125 de 2LZM, que obviamente corresponde a R125 de 1T6H. Para alinhar esses dois resíduos, a ferramenta poderia fazer um movimento de translação, movendo a posição de R125 de 1T6H para próximo da posição do resíduo equivalente em 2LZM. Perceba que ao fazer isso, todos os outros átomos seriam afetados. Para tentar melhorar esse processo, a ferramenta poderia efetuar um processo de rotação usando um segundo resíduo (no exemplo, E10). E assim, a ferramenta deveria realizar esse processo para outros resíduos, mas como ela saberia que o alinhamento está bom? Basta calcular a distância entre pontos equivalentes. Se a distância de R125 de 2LZM para R125 de 1T6H é, por exemplo, 50 Å (1 Å é equivalente a 10^{-10} m) e após a translação/rotação, essa distância reduziu para 1 Å, podemos concluir que o alinhamento está melhor. Após rodadas de translação e rotação, esse mesmo cálculo é feito para todos os átomos e depois é calculada uma média (chamamos isso de RMSD ou *Root-mean-square deviation of atomic positions* que pode ser traduzido para “desvio quadrático médio das posições atômicas”).

Figura 19. Compreendendo os movimentos de rotação e translação no algoritmo de alinhamento estrutural do PyMOL. Fonte: próprio autor.

Na prática, o algoritmo de sobreposição é um pouco mais complicado do que o exemplo apresentado. Entretanto, o objetivo aqui é apresentar de forma didática como ele funciona de maneira geral, focando principalmente na compreensão dos movimentos de rotação e translação. Ao fim, em um alinhamento bom, as coordenadas de átomos dos carbonos-alfa irão se sobrepor e isso permitirá que possamos visualizar as proteínas sobrepostas (como visto na Figura 20).

Figura 20. Sobreposição entre carbonos-alfa (esquerda) de 2LZM (azul) e 1T6H (amarelo). À direita podemos ver as estruturas completas sobrepostas representadas em forma de cartoon. Fonte: próprio autor.

Estudo de caso 3: alinhamento entre proteínas com sequências pouco idênticas

Nos exemplos anteriores, demonstramos a visualização de um alinhamento entre duas proteínas com estrutura conservada. Entretanto, o que aconteceria se alinhássemos duas proteínas com estruturas bastante diferentes. Para ilustrar esse hipotético problema, vamos utilizar dois PDBs escolhidos aleatoriamente: **1mdb** (uma ligase) e **2lzm** (uma lisozima de um bacteriófago).

Primeiro, vamos carregar os arquivos PDB de **1mdb** e **2lzm** usando o comando *fetch*:

fetch 1mdb
fetch 2lzm

E a seguir, realizar o alinhamento das duas estruturas usando a interface do PyMOL (Figura 21).

Figura 21. Alinhamento estrutural de 1mdb e 2lzm. Fonte: próprio autor.

Ao tentar alinhar duas proteínas com estruturas pouco similares, PyMOL tentará encontrar a maior quantidade de regiões com sequências parecidas e tentará de alguma forma alinhá-los. Em estruturas pouco similares, isso irá gerar um alinhamento ruim. Podemos ver isso, observando as linhas amarelas que conectam pontos equivalentes em que houve a tentativa de alinhamento (Figura 22).

Figura 22. Alinhamento estrutural de 1mdb e 2lzm. Linhas amarelas representam pontos em que houve uma tentativa de alinhamento. Fonte: próprio autor.

Observe que não há praticamente nenhuma sobreposição entre estruturas secundárias. Isso se deve ao fato do PyMOL, por padrão, usar uma estratégia baseada em alinhamento de sequências. Como as sequências apresentam uma alta discrepância (Figura 23), os poucos resíduos alinhados levarão a um alinhamento estrutural ruim.

Figura 23. Alinhamento de sequências de 1mdb e 2lzm. Percebam que há poucos resíduos similares. Fonte: próprio autor.

Podemos obter resultados melhores usando os outros métodos de alinhamento presentes no PyMOL, mas para isso precisamos introduzir o uso de alinhamentos por linhas de comando.

Alinhando estruturas por linha de comando com PyMOL

Pymol é uma ferramenta de visualização de moléculas com uma grande quantidade de recursos em sua interface. Dentre esses recursos, se encontra o painel do terminal de linha de comandos. Por meio desse terminal é possível realizar inúmeras análises, como por exemplo, alinhamento de estruturas. Esse tipo de alinhamento pode ser realizado pela interface gráfica, entretanto, usando o terminal podemos facilmente realizar um alinhamento usando o comando **align**.

Terminal PyMOL: comando *align*

A sintaxe do comando é a seguinte:

```
align objeto1, objeto2
```

onde objeto1 e objeto2 são os dois nomes dos objetos que armazenam as estruturas que se deseja alinhar (os nomes são exibidos no painel à direita).

O comando *align* usa o método padrão de alinhamento do PyMOL. Sucintamente, ele utiliza um algoritmo de superposição baseado em duas etapas:

- alinhamento de sequências utilizando BLAST e a matriz BLOSUM62 para detecção de resíduos correspondentes (por padrão, apenas os átomos da cadeia principal são usados);
- superposição dos átomos seguida por até cinco ciclos de refinamento iterativo. Átomos que não atingem os critérios de corte estabelecidos são excluídos das análises. Por fim, o número de átomos restantes é exibido junto com o valor de RMS (*root mean square*, que pode ser traduzido como “raiz do valor quadrático médio” ou “valor eficaz” como também é conhecido).

No exemplo a seguir, vamos utilizar o comando *align* para alinhar as estruturas do PDB de ID 2lzm com o PDB de ID 1alm (Figura 24).

Figura 24. Alinhamento de sequências de 1mdb e 2lzm usando o comando align do PyMOL.
Fonte: próprio autor.

Observe que os resultados do alinhamento são exibidos no painel do terminal de linha de comandos (na parte superior da imagem). Observe ainda que inicialmente é realizado um alinhamento entre a estrutura com 282 resíduos e a outra com 891. Detectou-se uma pontuação de alinhamento de 41. Com base nos resíduos alinhados (não exibidos no painel), realizou-se uma tentativa de sobreposição de 873 átomos. A seguir, cinco rodadas de novos alinhamentos estruturais são realizadas tentando reduzir o RMSD. Perceba que nessa etapa de refinamento do alinhamento, alguns átomos são desconsiderados, uma vez que o algoritmo considera que mantê-los reduz a precisão do resultado final. Por fim, obtêm-se o RMSD final e o total de átomos usados no cálculo. No exemplo apresentado, o RMSD foi de ~22. Lembre-se que valores altos de RMSD indicam uma sobreposição ruim entre estruturas. Assim, podemos confirmar pela visualização das estruturas sobrepostas que o alinhamento foi ruim.

Para mais informações sobre o comando *align*, como variações de parâmetros, acesse: <https://pymolwiki.org/index.php/Align>. Para realizar um alinhamento independente da sequência, recomenda-se o uso do comando *super*.

Terminal PyMOL: comando *super*

O terminal de comando da ferramenta PyMOL permite a realização de alinhamentos entre estruturas usando o comando *align* (que utiliza a mesma metodologia do alinhamento realizado pela interface gráfica). Entretanto, o comando *align* é recomendado para estruturas com uma identidade maior do que 30%. Para realizar um alinhamento que considera uma melhor

sobreposição entre as estruturas de proteínas com sequências pouco similares é recomendado o uso do comando *super*.

O comando *super* realiza alinhamento estrutural independente da sequência (ao contrário de *align*) usando uma estratégia baseada em programação dinâmica, seguido por uma série de ciclos de refinamento para remover átomos *outliers* (etapa similar a última parte do algoritmo usado por *align*). Por isso, o comando *super* é considerado mais robusto do que o comando *align* para proteínas com baixa similaridade de sequência.

No exemplo a seguir, vamos utilizar o comando *super* para alinhar as estruturas do PDB de ID 2lzm com o PDB de ID 1mdb (Figura 25).

Figura 25. Alinhamento de sequências de 1mdb e 2lzm usando o comando *super* do PyMOL. Fonte: próprio autor.

Note que as sequências de 1mdb e 2lzm são pouco similares. Vemos também uma baixa similaridade na estrutura; entretanto, ao utilizar o comando *super*, PyMOL encontra partes da estrutura secundária que se coincidem (vemos por exemplo, as hélices-alfa que se sobrepõe). Isso demonstra a maior eficácia do comando *super* para detecção de estruturas secundárias mais similares quando comparado ao método tradicional (comando *align*).

Para mais informações sobre o comando *super*, como variações de parâmetros, acesse: <https://pymolwiki.org/index.php/Super>.

Alinhamento de estruturas 3D com TM-align

TM-align é um algoritmo desenvolvido para identificar o melhor alinhamento possível entre um par de proteínas através de uma matriz de rotação construída com o TM-Score e programação dinâmica [3]. Ele é descrito como mais rápido, com maior precisão e cobertura que algoritmos conceituados como DALI [7] e CE [8].

Figura 26. Logotipo do TM-align. Fonte: <https://zhanglab.ccmb.med.umich.edu/TM-align/>.

O algoritmo realiza três etapas de alinhamentos iniciais rapidamente calculados. Na primeira etapa, as estruturas secundárias das proteínas são alinhadas usando programação dinâmica. A seguir, o algoritmo constrói uma matriz binária (0 ou 1), que indica se há uma correspondência na estrutura secundária do resíduo em questão [3]. Na segunda etapa, o alinhamento é baseado na correspondência contínua das duas estruturas, a estrutura menor e continuamente posicionada sobre a estrutura maior, e o alinhamento com melhor TM-Score é selecionado. O terceiro alinhamento também é obtido por programação dinâmica, mas a matriz de pontuação é obtida das matrizes de pontuação dos dois alinhamentos anteriores.

O TM-align pode ser utilizado tanto online (*webtool*) quanto em ambiente *desktop* (neste caso, a instalação deve ser feita em ambiente Linux por linha de comando). Para utilizar a ferramenta web ou realizar o *download* do código-fonte para a instalação local, acesse <https://zhanglab.ccmb.med.umich.edu/TM-align/>.

Executando o TM-align pela Internet

Executar o TM-align pelo **servidor web oficial** é a forma mais simples de utilização. Na ferramenta web (Figura 27), deve-se utilizar como entrada dois arquivos no formato PDB ou PDBx/mmCIF (pode-se enviar os arquivos separadamente ou apenas colar o texto correspondente dentro das caixas de texto).

Quick & Accurate Structural Alignment

TM-align is an algorithm for sequence-independent protein structure comparison. For two protein structures of unknown relationship, TM-align first generates optimized residue-to-residue alignment based on structural similarity using heuristic dynamic programming function. An optimal superposition of the two structures built on the structural alignment, as well as the **TM-scores** value which scales the structural similarity, will be returned. TM-scores has the value in (0,1), where 1 indicates a perfect match between two structures. Following strict statistics of structures in the PDB, scores below 0.2 correspond to randomly chosen unrelated proteins while those higher than 0.5 assume generally the same fold in SCOP/CATH.

Notes: TM-align now allows for input structure with either **PDB** or **PDB-formatCIF** format. Meanwhile **C++ version of TM-align** is now officially released.

TM-align on-line (view an example of output)

Note: This server is only for pair-wise structure comparison. If you want to match one protein structure with all proteins in the PDB library, you can do it in **COPACOS Server**.

• **Input Structure 1 in PDB format or PDB-formatCIF format (mandatory)**
Please copy and paste your structure file here. [Sample Input](#)

```

HEADER          TRANSFER
TITLE          3D6M: HUMAN PROTEINASE FROM M-DUTY, ESTABLISHED AT 100
COMPND          MOL_ID: 1;
SOURCE          3D6M: HUMAN PROTEINASE FROM M-DUTY, ESTABLISHED AT 100
EXPRES          3D6M: HUMAN PROTEINASE FROM M-DUTY, ESTABLISHED AT 100
SEQUENCE        3D6M: HUMAN PROTEINASE FROM M-DUTY, ESTABLISHED AT 100

```

Or upload the structure file
Escalator arquivo: [3d6m.pdb](#)

• **Input Structure 2 in PDB format or PDB-formatCIF format (mandatory)**
Please copy and paste your structure file here. [Sample Input](#)

```

HEADER          TRANSFER
TITLE          3D6M: HUMAN PROTEINASE FROM M-DUTY, ESTABLISHED AT 100
COMPND          MOL_ID: 1;
SOURCE          3D6M: HUMAN PROTEINASE FROM M-DUTY, ESTABLISHED AT 100
EXPRES          3D6M: HUMAN PROTEINASE FROM M-DUTY, ESTABLISHED AT 100
SEQUENCE        3D6M: HUMAN PROTEINASE FROM M-DUTY, ESTABLISHED AT 100

```

Or upload the structure file
Escalator arquivo: [3d6m.pdb](#)

• **Input Email (optional, where results will be sent to)**

• **Run TM-align** [Clear form](#)

TM-align download

• [Click TM-align](#) | [or TM-align.cxx](#) to download the newest version of the TM-align source code coded in Fortran77 (or C++). You can compile the program in your Linux computer (if you can ignore the "static" option for some machines, such as Mac, which does not

Figura 27. Exemplo de entrada de dados na página inicial do TM-align. Fonte: <https://zhanglab.ccmb.med.umich.edu/TM-align/>.

Opcionalmente, os resultados podem ser enviados para o seu e-mail (nesse caso, você deve preencher o campo *Input Email*). Por fim, execute o alinhamento clicando em *Run TM-align*. Quando o processamento for concluído, será exibida uma página com um resumo da execução e os arquivos de resultados disponíveis para *download*.

A seguir, será apresentado um exemplo de uso. Para este estudo de caso, realizaremos o alinhamento da beta-glicosidase do fungo *Humicola insolens* (PDB ID: 4MDP) e da beta-glicosidase do cupim *Neotermes koshunensis* (PDB ID: 3VIK). O primeiro passo foi acessar o site oficial usando um navegador de internet (foi usado o Google Chrome). A seguir, realizamos o *upload* dos arquivos de entrada no formato PDB (Figura 28).

Falsifying strict statistics of structures in the PDB, scores below 0.2 correspond to randomly chosen unrelated proteins while those higher than 0.5 assume generally the same fold in SCOP/CATH.

Notes: TM-align now allows for input structure with either **PDB** or **PDB-formatCIF** format. Meanwhile **C++ version of TM-align** is now officially released.

TM-align on-line (view an example of output)

Note: This server is only for pair-wise structure comparison. If you want to match one protein structure with all proteins in the PDB library, you can do it in **COPACOS Server**.

• **Input Structure 1 in PDB format or PDB-formatCIF format (mandatory)**
Please copy and paste your structure file here. [Sample Input](#)

Or upload the structure file
Escalator arquivo: [3vik.pdb](#)

• **Input Structure 2 in PDB format or PDB-formatCIF format (mandatory)**
Please copy and paste your structure file here. [Sample Input](#)

Or upload the structure file
Escalator arquivo: [4mdp.pdb](#)

• **Input Email (optional, where results will be sent to)**

Figura 28. Executando o TM-align para as entradas 3vik e 4mdp. Os arquivos PDB foram baixados em <https://www.rcsb.org/structure/3vik> e <https://www.rcsb.org/structure/4mdp>, respectivamente. Fonte: próprio autor.

A seguir, executamos as análises clicando em *Run TM-align*. Após o processamento, a ferramenta web retornou um resumo do log de execução com a referência bibliográfica para citação do uso da ferramenta, os dados

das estruturas de entrada, como nomes e tamanho de sequências, o valor do TM-score e o alinhamento das sequências (Figura 29)

TM-align Results

```
*****
*              TM-align (Version 20190802)              *
* An algorithm for protein structure alignment and comparison *
* Based on statistics:                                     *
* 0.5 < TM-score < 0.30, random structural similarity      *
* 0.5 < TM-score < 1.00, in about the same fold           *
* Reference: Y. Zhang and J. Skolnick, Nucleic Acids Res 33, 2302-9 (2005) *
* Please email your comments and suggestions to: zhanglab@umich.edu *
*****

Name of Chain_1: A089693
Name of Chain_2: B089699
Length of Chain_1: 472 residues
Length of Chain_2: 477 residues

Aligned length: 349, RMSD: 1.62, Seq_ID%_identical/n_aligned: 0.396
TM-score: 0.92239 (if normalized by length of Chain_1)
TM-scores: 0.91291 (if normalized by length of Chain_2)
(You should use TM-score normalized by length of the reference protein)

("." denotes aligned residue pairs of d < 3.0 Å, "." denotes other aligned residues)
TVYTFPEPFLGAWTASVQIEGAGDGGSPILLDTLTPEFVVDGATGDLGGDHYLVKRWLLKELGQVYFSSHWRLPEGHQ-HIVNDGGDIYVVALINELLVNGIEPMYTHMHDLPQALDQ-LGGVPA-LVLKYSSEYVARN
-SHSLPPDFKGFATAVQIEGAGDGGSPILLDTLTPEFVVDGATGDLGGDHYLVKRWLLKELGQVYFSSHWRLPEGHQ-HIVNDGGDIYVVALINELLVNGIEPMYTHMHDLPQALDQ-LGGVPA-LVLKYSSEYVARN
```

Figura 29. Resumo do alinhamento de 3vik com 4mdp. Fonte: <https://zhanglab.ccmb.med.umich.edu/TM-align/>.

Além disso, a ferramenta retornou uma visualização tridimensional das estruturas diretamente no navegador usando a biblioteca Jsmol (Figura 30), além de *links* para o *download* dos arquivos de entrada e saída, e os arquivos **pdb** para visualizar a sobreposição na ferramenta Rasmol (disponível em <http://www.openrasmol.org/>).

Visualization (Protein-1 in blue and Protein-2 in red)

Document downloads

- Click [A089693.pdb](#) to download the first structure (3vik.pdb) that you submitted.
- Click [B089699.pdb](#) to download the second structure (4mdp.pdb) that you submitted.
- Click [C389693.pdb](#) to download a Rasmol script file to show the superposed Ca-traces in the aligned regions (To view the structure superposition, please run "rasmol -script C389693").
- Click [D188693.pdb](#) to download a Rasmol script file to show the superposed Ca-traces of entire chain (To view the structure superposition, please run "rasmol -script D188693").
- Click [E386693.pdb](#) to download a Rasmol script file to show the superposed full-atom structure in the aligned region (To view the structure superposition, please run "rasmol -script E386693").
- Click [F586693.pdb](#) to download a Rasmol script file to show the superposed full-atom structure of the entire chain (To view the structure superposition, please run "rasmol -script F586693").
- Click [G589693.pdb](#) to download a Rasmol script file to show the superposed full-atom structure of the entire chain with ligands/solvents (To view the structure superposition, please run "rasmol -script G589693").

All above files will be removed from the server after one month.

Reference:

Y. Zhang, J. Skolnick, TM-align: A protein structure alignment algorithm based on TM-score, Nucleic Acids Research, 2005 33, 2302-2309.

[Back to TM-align Homepage](#)

Figura 30. Visualização do alinhamento de 3vik com 4mdp. Um exemplo de resultado similar (mas com outras estruturas de entrada) pode ser encontrado em <https://zhanglab.ccmb.med.umich.edu/TM-align/example/873772.html>. Fonte: próprio autor.

Execução local do TM-align

Caso não deseje enviar os seus dados para o servidor web, você pode realizar o *download* do código-fonte do TM-align e instalá-lo em sua máquina. A versão em C++ do TM-align pode ser obtida em

<https://zhanglab.ccmb.med.umich.edu/TM-align/TMalign.cpp>. Entretanto,

antes de utilizá-la, você terá que realizar a compilação do código. Para isso, execute no terminal Linux os comandos:

```
1. mkdir tm
2. wget https://zhanglab.ccmb.med.umich.edu/TM-align/TMalign.cpp
3. g++ TMalign.cpp -o tmalgin
4. ./tmalign protein1.pdb protein2.pdb
```

Na linha 1, o comando cria uma pasta para armazenar o arquivo. A linha 2 utiliza o comando `wget` para fazer o download do arquivo do código-fonte (você pode baixá-lo manualmente). Na linha 3 é feita a compilação do arquivo para gerar um executável chamado **tmalign**. Na linha 4 é finalmente realizada a execução da ferramenta, em que **protein1.pdb** e **protein2.pdb** são as estruturas que serão alinhadas.

O algoritmo irá rotacionar e transladar a primeira proteína para alinhá-la com a segunda (que será a referência). O resultado será mostrado na saída do terminal. A execução padrão do programa pode ser alterada usando certos parâmetros, como por exemplo, o parâmetro `-o`, que gera arquivos que podem ser visualizados em ferramentas como Rasmol e no PyMOL. Outro parâmetro interessante é o `-fast`, que gera um alinhamento mais rápido. Entretanto, isso gera uma redução na precisão do resultado. Existem outros parâmetros que podem ser aplicados. Para obtê-los, consulte a documentação do programa ou execute no terminal o comando:

```
./tmalign -h
```

Para ilustrar o uso local do TM-align, vamos reproduzir o estudo de caso apresentado anteriormente. Neste caso, os arquivos PDB foram adicionados ao mesmo diretório do arquivo executável **tmalign**. A seguir foi executado o comando:

```
./tmalign 3vik.pdb 4mdp.pdb -o 3VIK_x_4MDP/Out
```

Isso fará com que o TM-align alinhe as duas proteínas de exemplo e escreva o resultado dentro da pasta 3VIK_x_4MDP. Além disso, os arquivos de saída serão iniciados com a palavra "Out" (Figura 31), sendo "Out.pdb" o arquivo correspondente à proteína rotacionada escrita em formato PDB.

Out	20/03/2020 12:07	Arquivo	80 KB
Out	20/03/2020 12:07	Arquivo PDB	777 KB
Out	20/03/2020 12:07	PyMOL Script	1 KB
Out_all	20/03/2020 12:07	Arquivo	84 KB
Out_all	20/03/2020 12:07	PyMOL Script	1 KB
Out_all_atm	20/03/2020 12:07	Arquivo	419 KB
Out_all_atm	20/03/2020 12:07	PyMOL Script	1 KB
Out_all_atm_lig	20/03/2020 12:07	Arquivo	476 KB
Out_all_atm_lig	20/03/2020 12:07	PyMOL Script	1 KB
Out_atm	20/03/2020 12:07	Arquivo	396 KB
Out_atm	20/03/2020 12:07	PyMOL Script	1 KB

Figura 31. Arquivos resultantes da execução do TM-align. Fonte: próprio autor.

Como vantagem do uso do TM-align por linha de comando, podemos destacar que a realização de alinhamentos pode ser automatizada através de qualquer linguagem de programação que tenha acesso aos comandos do sistema. Por exemplo, a linguagem Python possui a biblioteca “os” que fornece métodos de acesso a funcionalidades do sistema operacional. Para executar o TM-align seria possível usar a função `os.system()` da seguinte forma:

```
import os
os.system("./talign protein1.pdb protein2.pdb")
```

Alinhamento de estruturas 3D com o software Multiprot

Multiprot é um software automatizado e eficiente para detecção de múltiplos alinhamentos estruturais de proteínas. Seu princípio básico visa encontrar os núcleos geométricos comuns entre as moléculas de entrada. Isso não exige que todas as moléculas de entrada participem do alinhamento, o que permite que o algoritmo seja mais rápido e eficiente quando comparada a outros alinhadores estruturais, permitindo ainda alinhar várias estruturas ao mesmo tempo [9]. Multiprot pode ser executado pelo site <http://bioinfo3d.cs.tau.ac.il/MultiProt> ou pode ser instalado localmente e executado via linha de comando.

Executando o Multiprot localmente

Multiprot pode ser baixado para execução em servidor Linux pelo endereço: http://bioinfo3d.cs.tau.ac.il/MultiProt/index_v1.6.html. Os arquivos do Multiprot estão compactados em formato **tar**. Após a extração (utilize o comando **tar -xvf multiprot[versão-atual].tar** para descompactar), execute o arquivo **multiprot.Linux**, que está dentro da pasta extraída:

```
./multiprot.Linux protein1.pdb protein2.pdb protein3.pdb  
protein4.pdb
```

Em que **protein1.pdb**, **protein2.pdb**, **protein3.pdb** e **protein4.pdb** são os caminhos para os arquivos em formato PDB das proteínas a serem alinhadas.

O programa gera três arquivos de saída:

- **log_multiprot.txt**, que contém os parâmetros utilizados no alinhamento e a saída do programa;
- **n_sol.res** (em que “n” é o número de moléculas alinhadas), que contém o registro das equivalências. O programa pode gerar mais de uma possível solução para as equivalências. Elas são numeradas e descritas nesse arquivo;
- **n_sets.res** (em que “n” é o número de moléculas alinhadas), que contém uma lista com os fragmentos contíguos multiplamente alinhados.

A seguir, será apresentado um exemplo de uso. Para este estudo de caso, realizaremos o alinhamento da beta-glicosidase do fungo *Humicola insolens* (PDB ID: 4MDP) e da beta-glicosidase do cupim *Neotermes koshunensis* (PDB ID: 3VIK). Os arquivos PDB de 3VIK e 4MDP foram baixados em <https://www.rcsb.org/structure/3vik> e <https://www.rcsb.org/structure/4mdp>, respectivamente. Eles foram movidos para o mesmo diretório em que se encontra o executável do Multiprot, que foi executado via terminal pelo comando:

```
./multiprot.Linux 3VIK.pdb 4MDP.pdb
```

Esse comando gerou como saída o arquivo **log_multiprot.txt** (Figura 32A), que contém o log de execução com parâmetros e informações, como por exemplo, o tempo de execução e o tamanho da maior solução de alinhamento proposta. A principal saída é o arquivo **2_sol.res** (Figura 32B), que contém os pares de átomos alinhados com os equivalentes na outra proteína. Além disso, é criado o arquivo **2_sets.res** (Figura 32C), que contém uma lista dos arquivos de entrada e algumas configurações usadas na execução do programa.

```
Open [icon] log_multi
~/Downloads/t

##### Program Parameters #####
BioCore (Int) 3
BioCoreRadius (Float) 3
ChainEq (Int) 0
ClusterSeqBlockRMSDthr (Float) -1
Debug (Str) 0
ExtndTrials (Int) 1
FullSet (Int) 1
IJStep (Int) 1
IJshift (Int) 1000
MaxDels (Int) 150
MaxIns (Int) 150
MaxRFP (Int) 4500
OnlyRefMol (Int) 0
PointType (Int) 0
ResNum (Int) 1
RobustFPCLustering (Int) 0
Scoring (Int) 0
SeqBlockMinSize (Int) 5
SeqBlockOverlapRatio (Float) 0.8
SeqBlockRMSDthr (Float) 3
SeqOrder (Int) 0
StartSeq (Int) 0
Step (Int) 1
TransEqualRMSD (Float) 1
#####
Loading Molecule: ../3vik.pdb
Molecule ../3vik.pdb size : 472
Loading Molecule: ../4mdp.pdb
Molecule ../4mdp.pdb size : 477

Num Of Mols: 2 Largest Solution: 434

Multiple fragment alignment:
Sets - Mols: 2 Longest (contiguous) Solution: 165

Overall Run Time: 00:00:10.70
```

(A)

```

Molecules:
Mol-0 : ../3vik.pdb
Mol-1 : ../4mdp.pdb

Results:

Solution Num : 0

Hit Corres Score : 434
Reference Molecule : 0
Molecule : 1
Trans : -0.264769 -0.990266 -2.7785 -50.8398 121.704 22.729!
ISD : 0.999547

Hit List (Chain_ID.Res_Type.Res_Num) : 434
V.28 A.M.1
T.30 A.L.472
F.31 A.L.3
P.32 A.P.4
D.33 A.P.5
E.34 A.D.6
F.35 A.F.7
K.36 A.K.8
L.37 A.W.9
G.38 A.G.10
A.39 A.F.11

```

(B)

```

Molecules:
Mol-0 : ../3vtk.pdb
Mol-1 : ../4mdp.pdb

Results:

Solution Num : 0

Reference Molecule : 0
Length : 165
Reference Fragment : 28 192
Molecule : 1
0- 0 164 : 28 192 0 164 : 2.99625 : -0.310431 -1.00466 -2.04638 -49.4704 121.734 23.9405 : 0 164

```

(C)

Figura 32. Resultados do Multiprot. (A) Arquivo `log_mutiprot.txt`. (B) Fragmento do arquivo gerado **2_sol.res**. (C) Arquivo gerado **2_sets.res**. Fonte: próprio autor.

Nesse exemplo do alinhamento de 3VIK com 4MDP (Figura 32B), vemos que o Multiprot considerou que o resíduo V28 da cadeia A de 3VIK está em uma posição equivalente ao resíduo M1 da cadeia A de 4MDP. Vemos ainda outras correspondências como T30 com L472, F31 com L3, P32 com P4, D33 com P5, E34 com D6 e assim por diante. Perceba que os resíduos não precisam ser iguais para que Multiprot indique uma correspondência. Para que haja correspondência, Multiprot detecta o resíduo mais próximo da referência quando as proteínas forem sobrepostas. Parâmetros podem ser utilizados ainda para definir qual será o ponto de referência (em geral, é utilizada a coordenada do carbono-alfa). Para conferir quais parâmetros podem ser modificados, consulte a documentação.

Alinhamento de estruturas 3D com o software MUSTANG

MUSTANG (*M*Ultiple *p*rotein *S*Tructural *A*lign*M*ent *a*l*G*orith*m*) [10] é um software desenvolvido para realizar alinhamento estrutural múltiplo de proteínas (Figura 33). Possui código-aberto desenvolvido em C++. Sua estratégia de alinhamento sobrepõe as estruturas a partir das posições espaciais dos carbonos alfa ($C\alpha$) dos aminoácidos. O seu algoritmo utiliza uma heurística progressiva em pares e passos de refinamentos são realizados para otimizar o resultado, a fim de encontrar bons alinhamentos.

Figura 33. Logotipo da ferramenta Mustang. Fonte: <https://lcb.infotech.monash.edu/mustang/>.

Instalação do MUSTANG

A versão atual da ferramenta MUSTANG pode ser baixada através da página: <https://lcb.infotech.monash.edu/mustang/>. Para instalar o MUSTANG no sistema operacional Linux, com distribuições baseadas no Debian (Ubuntu por exemplo), basta executar o comando:

```
>> sudo apt-get install mustang
```

Para sistemas operacionais Windows e demais distribuições Linux, é necessário realizar o *download* do código-fonte e compilá-lo no computador. Para isso, o computador deve possuir o compilador *g++* que irá compilar o código-fonte e irá criar o executável do MUSTANG. Os passos a seguir descreverão esta etapa.

Para Windows, baixe e instale **MinGW**. Após a instalação, irá surgir uma janela para gerenciar os compilares a serem instalados. Nela, clique em "All Packages" e marque os pacotes "*mingw32-gcc-v3-g++-bin*" e "*mingw32-make-bin*" (Figura 34).

Figura 34. Instalando os pacotes necessários para compilar o MUSTANG. Fonte: próprio autor.

Utilizando o MUSTANG por linha de comando

Para utilizar o Mustang via linha de comando, deve-se informar o comando **mustang** seguido do parâmetro **-i** e os arquivos de entrada, como por exemplo:

```
>> mustang -i proteina1.pdb proteina2.pdb
```

No exemplo a seguir, é realizado o alinhamento entre os arquivos PDB de código **1a1m** e **1a1n** (Figura 35). Para alinhar as duas estruturas de proteínas foi usada a opção **-i**: **1a1m_C_A.pdb** e **1a1n_C_A.pdb**. Ainda foi requisitado o RMSD (opção **-r ON**) e o arquivo de saída contendo as estruturas sobrepostas foi descartado (opção **-s OFF**).

```
pmartins@pop-os:~$ mustang -i 1a1m_C_A.pdb 1a1n_C_A.pdb -r ON -s OFF

MUSTANG (v3.2.3): A Multiple STuructural AlignNment alGorithm.
Authors:      A S Konagurthu, J Whisstock, and P J Stuckey, A M Lesk.
Reference:    A S Konagurthu et al. Proteins 64(3):559-574, 2006

Parsing the PDB file: 1a1m_C_A.pdb (...contains 287 residues.) [ OK ]
Parsing the PDB file: 1a1n_C_A.pdb (...contains 284 residues.) [ OK ]
Calculating Distance Matrices... [ OK ]
Pairwise Comparisons: [ OK ]
Extending Edge-Weights thro' intermediate structures... [ OK ]
Final Alignment using the extended weights... [ OK ]
Superposing structures on the core: [ OK ]
Writing rotation matrices and RMSDs:(...writing results.rms_rot) [ OK ]
Output alignment in html: (...writing results.html) [ OK ]

All Done! Thank you for using MUSTANG.
Running Time : 0.78 s.

--o0o--
/ Visit http://lcb.infotech.monash.edu.au for other supporting programs /
/ Some suggested utilities for you to try out: /
/ SST: A Bayesian approach to assigning secondary structures. /
/ MMLigner: Pairwise structural alignment using information & compression. /
/ Super: Rapidly screen entire PDB for well-superposable fragments. /
--o0o--
```

Figura 35. Exemplo de execução do MUSTANG, alinhando duas estruturas de proteínas (opção **-i**: **1a1m_C_A.pdb** e **1a1n_C_A.pdb**), retornando o resultado do RMSD (opção **-r ON**) e descartando o arquivo de saída contendo as estruturas sobrepostas (opção **-s OFF**). Fonte: próprio autor.

Como resultado, obteve-se um arquivo no formato HTML (Figura 36). Por ele é possível ver o alinhamento estrutural simplificado em suas sequências. O formato HTML permite que o arquivo seja visualizado em navegadores, como

Chrome e Firefox, e com isso, destacando com cores a natureza química dos aminoácidos.

```
#####
# Program: MUSTANG v3.2.3: A Multiple structural alignment algorithm
# Authors: A. S. Konagurthu, J. C. Whisstock, and P. J. Stuckey, A. M. Lesk
# Rundate: Mon Mar 23 16:49:08 2020
# Report file: results.html
#####
#
# Aligned structures: 2
# 1: 1alm_C_A.pdb
# 2: 1aln_C_A.pdb
#
# Length: 288
# Identity: 272/288 ( 94.4%) (Calculated as the percentage of conserved columns in the alignment.)
# Similarity: 272/288 ( 94.4%) (Calculated as the percentage of semi-conserved columns in the alignment)
# Gaps: 5/288 ( 1.7%) (Calculated as the percentage of columns with atleast one gap.)
#=====ALIGNMENT START=====

1alm_C_A.pdb      1  GSHSMRYFYTAMSRPGRGEPFIAVGYYDDTFVRFDSDAASPRTEPRPPNIEQEGPEYW  60
1aln_C_A.pdb      1  GSHSMRYFYTAMSRPGRGEPFIAVGYYDDTFVRFDSDAASPRTEPRPPNIEQEGPEYW  60
    GSHSMRYFYTAMSRPGRGEPFIAVGYYDDTFVRFDSDAASPRTEPRPPNIEQEGPEYW

1alm_C_A.pdb      61  DRNTQIFKTNITQTYRENLRALRYYNQSEAGSHIQRMYGCDLGPDRLLRGNDQSAYDG  120
1aln_C_A.pdb      61  DRNTQIFKTNITQTYRENLRALRYYNQSEAGSHIQRMYGCDLGPDRLLRGNDQSAYDG  120
    DRNTQIFKTNITQTYRE LR      YYNQEAGSHIQRMYGCDLGPDRLLRGNDQSAYDG

1alm_C_A.pdb      121  KDYIALNEDLSSMTAADTAAQITQKWEAARVAEQLRAYLEGLCVENLRRYLENGKETLQ  180
1aln_C_A.pdb      121  KDYIALNEDLSSMTAADTAAQITQKWEAARVAEQLRAYLEGLCVENLRRYLENGKETLQ  180
    KDYIALNEDLSSMTAADTAAQITQKWEAARVAEQLRAYLEGLCVENLRRYLENGKETLQ

1alm_C_A.pdb      181  RADPPKTHVTHHPVSDHEATLRCAWGFYPAEITLTWQRDGEDQTQDTLVEVTRPAGDRT  240
1aln_C_A.pdb      181  RADPPKTHVTHHPVSDHEATLRCAWGFYPAEITLTWQRDGEDQTQDTLVEVTRPAGDRT  240
    RADPPKTHVTHHPVSDHEATLRCAWGFYPAEITLTWQRDGEDQTQDTLVEVTRPAGDRT

1alm_C_A.pdb      241  FQKWAAVVVPVGEEQRYTCHVQHEGLPKPLTLRWEPHHTPYDINQML-  287
1aln_C_A.pdb      241  FQKWAAVVVPVGEEQRYTCHVQHEGLPKPLTLRWEV--VPLRPMT--Y  284
    FQKWAAVVVPVGEEQRYTCHVQHEGLPKPLTLRWEV P

#=====ALIGNMENT END=====
#LEGEND:
#
# Colours indicate the chemical nature of the amino acid:
# Red      = small hydrophobic including aromatic,{A,F,I,L,M,P,V,W}
# Blue     = Acidic,{D,E}
# Magenta  = Basic,{K,R} and
# Green    = Basic amino acids with hydroxyl groups and/or amine groups {C,G,H,N,Q,S,T,Y}.
#
# The "markup row" below each stretch of the multiple alignment is used to mark completely conserved
# residue (denoted in UPPERCASE) and semi-conserved residue (denoted in lowercase) in a column of the alignment.
#
#####EOF#####
```

Figura 36.: Exemplo do arquivo de saída **results.html**. Fonte: próprio autor.

Alinhamentos híbridos com PROMALS3D

Ao acessar a página do PROMALS3D, o usuário encontrará a seguinte interface (Figura 37). PROMALS3D permite usar como entrada sequências no formato FASTA. Opcionalmente, o usuário pode utilizar como entrada arquivos de estruturas de proteínas ou apenas informar o código identificador no PDB. Para executar, basta clicar no botão “Submit”.

PROMALS3D multiple sequence and structure alignment server

PROMALS3D constructs alignments for **multiple protein sequences and/or structures** using information from sequence database searches, secondary structure prediction, available homologs with 3D structures and user-defined constraints. [\[Documentation\]](#)

DATA INPUT

Input can be either protein sequences, protein structures, or both sequences and structures.

Enter protein sequences in [FASTA](#) format: Clear sequences

Or upload a file Escolher arquivo Nenhum arquivo selecionado

[Enter protein structures](#) (optional)
Sequences will be extracted from structure files and added to the above input sequences.

Structure file 1:	Escolher arquivo Nenhum arquivo selecionado	or pdb id: <input type="text"/>	chain id: <input type="text"/>
Structure file 2:	Escolher arquivo Nenhum arquivo selecionado	or pdb id: <input type="text"/>	chain id: <input type="text"/>
Structure file 3:	Escolher arquivo Nenhum arquivo selecionado	or pdb id: <input type="text"/>	chain id: <input type="text"/>
Structure file 4:	Escolher arquivo Nenhum arquivo selecionado	or pdb id: <input type="text"/>	chain id: <input type="text"/>
Structure file 5:	Escolher arquivo Nenhum arquivo selecionado	or pdb id: <input type="text"/>	chain id: <input type="text"/>

[Click here to enter more structures](#)

[Click here to enter user-defined constraints](#) (help)

DATA SUBMIT

Enter **email** to receive the result ([recommended](#)):

Enter a **job name** ([recommended](#)): Submit Reset

Figura 37. Visão geral da interface do servidor web do PROMALS3D. Fonte: próprio autor.

Após o processamento dos dados, PROMALS3D retornará a seguinte página de resultados (Figura 38). Essa página irá permitir o download dos dados de alinhamento em diversos formatos, como o formato padrão do PROMALS3D, o formato CLUSTAL e o formato FASTA.

PROMALS3D results for job name:

Alignment results

COLORED alignment
(with **predicted secondary structure** and **conservation** information)

- **sequence order:** input ☐ aligned ☒
- **block size:** or show each sequence in one line ☐
- **show starting and ending residue numbers:** ☒
- **show conservation indices equal to or above:**
- **consensus level:**

CLUSTAL format alignment

- **sequence order:** input ☐ aligned ☒
- **block size:** or show each sequence in one line ☐

FASTA format alignment

- **sequence order:** input ☐ aligned ☒

Information about **alignment formats** is [here](#).

Inputs and intermediate results

Input sequences: none

Input structures are [here](#).

PSI-BLAST and **PSIPRED** results are [here](#).

The guide tree (based on a crude distance measure) for some sequences is [here](#).

Alignment parameters are [here](#).

[PROMALS3D homepage](#) [Documentation](#)

Your feedback is welcome. jpei@chop.swmed.edu

Figura 38. Página de resultado do PROMALS3D. Fonte: próprio autor.

O principal resultado do PROMALS3D pode ser obtido clicando no botão "show" da aba "Alignment results". Para demonstrar um exemplo real, realizamos o alinhamento da beta-glicosidase do fungo *Humicola insolens* (PDB ID: 4MDP) e da beta-glicosidase do cupim *Neotermes koshunensis* (PDB ID: 3VIK). Usamos como entrada apenas os códigos PDB ID (4MDP e 3VIK). Na Figura 39, vemos os alinhamento das duas sequências.

[illegible]

Perceba que acima de cada bloco de linhas dos alinhamentos, temos uma marcação de valor que indica se a posição é conservada ou não (valores inteiros entre 0 e 9, com 9 correspondendo à maior conservação). Abaixo de cada bloco temos a estrutura secundária predita, sendo a letra “e” (com fundo salmão) utilizada para indicar fitas-beta e a letra “h” (com fundo azul) utilizada para indicar hélices-alfa. As cores de cada resíduo também indicam a estrutura secundária, sendo vermelho indicando que o resíduo se encontra em uma hélice-alfa e azul indicando que se encontra em uma fita-beta.

Referências do capítulo

- [1] Approximate protein structural alignment in polynomial time. Rachel Kolodny, Nathan Linial Proceedings of the National Academy of Sciences Aug 2004, 101 (33) 12201–12206; DOI: doi.org/10.1073/pnas.0404383101
- [2] Antczak, M., Kasprzak, M., Lukasiak, P. *et al.* Structural alignment of protein descriptors – a combinatorial model. *BMC Bioinformatics* 17, 383 (2016).
<https://doi.org/10.1186/s12859-016-1237-9>
- [3] Yang Zhang, Jeffrey Skolnick, TM-align: a protein structure alignment algorithm based on the TM-score, **Nucleic Acids Research**, Volume 33, Issue 7, 1 April 2005, Pages 2302–2309, <https://doi.org/10.1093/nar/gki524>
- [4] John Rozewicki, Songling Li, Karlou Mar Amada, Daron M Standley, Kazutaka Katoh, MAFFT-DASH: integrated protein sequence and structural alignment, *Nucleic Acids Research*, Volume 47, Issue W1, 02 July 2019, Pages W5–W10, <https://doi.org/10.1093/nar/gkz342>
- [5] Jun Hu, Zi Liu, Dong-Jun Yu, Yang Zhang, LS-align: an atom-level, flexible ligand structural alignment algorithm for high-throughput virtual screening, **Bioinformatics**, Volume 34, Issue 13, 01 July 2018, Pages 2209–2218, <https://doi.org/10.1093/bioinformatics/bty081>
- [6] <https://zhanglab.ccmb.med.umich.edu/TM-align/> (arquivo README e interface)
- [7] HOLM, Liisa; SANDER, Chris, Protein structure comparison by alignment of distance matrices, *Journal of molecular biology*, v. 233, n. 1, p. 123–138, 1993.
DOI: <https://doi.org/10.1006/jmbi.1993.1489>
- [8] I N Shindyalov, P E Bourne, Protein structure alignment by incremental combinatorial extension (CE) of the optimal path., **Protein Engineering, Design and Selection**, Volume 11, Issue 9, Sep 1998, Pages 739–747,
<https://doi.org/10.1093/protein/11.9.739>
- [9] Shatsky M., Nussinov R., Wolfson H.J. (2002) MultiProt — A Multiple Protein Structural Alignment Algorithm. In: Guigó R., Gusfield D. (eds) *Algorithms in Bioinformatics*. WABI 2002. Lecture Notes in Computer Science, vol 2452. Springer, Berlin, Heidelberg Doi https://doi.org/10.1007/3-540-45784-4_18
- [10] http://bioinfo3d.cs.tau.ac.il/MultiProt/index_v1.6.html (arquivo README)

[11] Konagurthu, Arun S., et al. "MUSTANG: a multiple structural alignment algorithm." *Proteins: Structure, Function, and Bioinformatics* 64.3 (2006): 559-574.

[12] Wierenga RK (March 2001). "The TIM-barrel fold: a versatile framework for efficient enzymes". *FEBS Letters*. 492 (3): 193-8. doi:10.1016/s0014-5793(01)02236-0. PMID 11257493. S2CID 42044123.