DESENVOLVIMENTO DE *COOKBOOKS* PARA O ENSINO DE BIOINFORMÁTICA EM CURSOS DE GRADUAÇÃO DE CIÊNCIAS DA VIDA

Lucas Sousa Palmeira
Bruno Silva Andrade

RESUMO

A bioinformática constitui-se de uma área interdisciplinar, abrangendo áreas como a Biologia Molecular, Bioquímica, Física, Matemática, Estatística e Ciências da Computação, com a finalidade de tratar e interpretar dados biológicos. Tendo em vista o grande volume de dados biológicos, surge a necessidade de capacitar pessoas aptas para interpretar esses dados com o uso de computação. Neste trabalho buscou-se elaborar uma documentação em Python na forma de *cookbooks* para o tratamento de dados biológicos como forma de introduzir conceitos de Bioinformática para professores e alunos de nível superior. A construção deste *cookbook* se dividiu na manipulação de sequências, alinhamento de sequências, busca de sequências homólogas, visualização de estruturas 3D de proteínas e modelagem molecular por homologia de proteínas. O *cookbook* conta com 6 notebooks, com uma licença para uso, modificação e redistribuição GPL 3.0 (*open source*), bem como se encontra hospedado no Github (https://github.com/lucaspalmeira/Bio-learning).

Palavras chave: Bioinformática, Ensino, *Cookbooks*.

1. INTRODUÇÃO

A Bioinformática constitui-se de uma área interdisciplinar, abrangendo áreas como a Biologia Molecular, Bioquímica, Física, Matemática, Estatística, Ciências da Computação, etc. Sendo o seu surgimento muito interligado com a própria Biologia Molecular, que com o uso da computação permitiu que muitas perguntas biológicas complexas fossem respondidas em meio ao crescente número de dados (VERLI, 2014; GIBAS; JAMBECK, 2001).

Na década de 1950, a elucidação da estrutura tridimensional da molécula de DNA mudou significativamente a nossa compreensão quanto às bases biológicas da vida, resultante de um esforço conjunto promovido pelos pesquisadores Rosalind Franklin, Maurice Wilkins, James

Watson e Francis Crick (VERLI, 2014). Mas além destes, outros dois importantes pesquisadores, Linus Pauling e Ramachandran contribuíram também de maneira significativa para a elucidação da estrutura tridimensional de proteínas, sendo responsáveis pelos estudos da conformação secundária da estrutura de alfa-hélice em proteínas (JASKOLSKI; DAUTER; WLODAWER, 2014; SUBRAMANIAN, 2001; RAMACHANDRAN; KARTHA, 1954).

Somando a esses feitos, a pesquisadora Margaret O. Dayhoff contribuiu para a elaboração dos primeiros algoritmos de alinhamento de sequências de proteínas (MARGARET OAKLEY DAYHOFF 1925–1983, 1984). Possibilitando assim, importantes avanços nos estudos de filogenias baseadas em caracteres moleculares. Em 1970, Saul B. Needleman e Christian D. Wunsch foram responsáveis pelo desenvolvimento do algoritmo de alinhamento global *Needleman-Wunsch* (NEEDLEMAN; WUNSCH, 1970). E mais tarde, na década de 1980, Temple Smith e Michael Waterman desenvolveram o algoritmo de alinhamento local de sequências conhecido como *Smith-Waterman* (SMITH; WATERMAN, 1981).

De 1980 a 2000, o volume de dados biológicos aumentou exponencialmente, especialmente os dados de sequenciamento de genoma e de cristalografia de proteínas (GIBAS; JAMBECK, 2001), alimentando assim importantes bancos de dados públicos como o *GenBank* (https://www.ncbi.nlm.nih.gov/genbank/) (SAYERS *et al*., 2019) e o *Protein Data Bank* (https://www.rcsb.org/) (BERMAN *et al*., 2000). Atualmente, o *GenBank* conta com mais de 2 bilhões de sequências de nucleotídeos depositadas, o que totaliza mais de 3 bilhões de pares de bases nucleotídicas (SAYERS *et al*., 2019), enquanto o *Protein Data Bank* conta com mais de 200 mil estruturas cristalográficas de proteínas depositadas até o presente momento deste estudo.

Visto tais importantes feitos para uma melhor compreensão da vida a nível molecular, iniciou-se uma discussão sobre a implementação de bioinformática em cursos de ciências da vida, principalmente em universidades dos Estado Unidos (MAGANA, 2017). O uso de atividades práticas em formas de cursos foi uma solução para incluir a bioinformática nas grades curriculares tanto em cursos de graduação quanto de pós-graduação (MAGANA, 2017; FEIG, JABRI, 2002; PHAM *et al*., 2008).

Estratégias de ensino têm sido exploradas, buscando capacitar os alunos para usarem importantes ferramentas e bases de dados da bioinformática, de maneira a prepará-los para usarem metodologias mais sofisticadas e apropriadas com a finalidade de resolver problemas biológicos mais complexos (BOYLE, 2004). Assim, um bioinformata deve estar preparado para usar diferentes linguagens de programação (*Python*, *Perl* ou *R*), apto para minerar dados e consequentemente conhecedor de linguagens adotadas para a construção de bancos de dados, bem como ser capaz de lidar com grandes volumes de dados (WELCH *et al*., 2014).

1.1 Justificativa

A necessidade de lidar com grandes volumes de dados requer profissionais aptos, especialmente quando se busca tratar dados biológicos. Já há algum tempo é discutida a inclusão da Bioinformática no fluxograma em cursos de graduação, em especial Biologia Molecular e Bioquímica, onde algumas metodologias já foram aplicadas e avaliadas quanto ao interesse dos alunos pela área (FEIG; JABRI, 2002; BOYLE, 2004; BADOTTI *et al*., 2014).

Entretanto, no Brasil, não vem sendo realizada essa discussão. Neste sentido, a fim de superar essa lacuna, torna-se importante a elaboração de soluções para o ensino de bioinformática. A elaboração de tutoriais, cadernos de prática (*cookbooks*) em uma linguagem de programação de alto nível e interpretada são interessantes formas de se aplicar o ensino de bioinformática em disciplinas ou cursos relacionados às ciências da vida.

1.2 Questão norteadora

Quais os desafios e qual a viabilidade para elaborar *cookbooks* com foco no ensino de Bioinformática em cursos de ciências da vida?

1.3 Objetivo geral

Elaborar uma documentação em linguagem de programação Python na forma de *cookbooks* com exemplos de código ou scripts para o tratamento de dados moleculares como sequências de DNA e proteínas de forma a introduzir conceitos de Bioinformática para professores e alunos das disciplinas relacionadas com Biologia Molecular, Genética e Bioquímica no nível superior.

2. REFERENCIAL TEÓRICO

Dificuldades no ensino de Biologia Molecular têm sido observadas quanto ao nível de abstração, a falta de equipamentos necessários e a falta destes para se manter um laboratório de Biologia Molecular. Somado a isso, o aumento do volume de dados biológicos tem crescido demasiadamente, levando assim ao surgimento de áreas como a Bioinformática para lidar com este grande volume. Assim, surgiu a necessidade de se discutir qual deve ser a formação para um

profissional de bioinformática, quais habilidades deve possuir e quais ferramentas ele deve ser apto para lidar com os problemas biológicos.

2.1 Dificuldades no ensino de Biologia Molecular

Um dos principais problemas para o ensino de Biologia Molecular é o quanto os conceitos são abstratos, o que se agrava ainda mais para alunos vindos do ensino médio. Onde a elaboração de modelos didáticos de estruturas proteicas e ou de DNA e RNA envolvidas em processos metabólicos como a replicação do DNA (etapa importante do dogma central da genética) tornam-se atraentes estratégias para melhor compreensão do conteúdo (FREITAS; MACIEL-CABRAL; SILVA, 2020).

Dificuldades como a falta de equipamentos ou tecnologias necessárias para a realização de práticas em laboratório são evidentes. Mas não somente isso, a formação acadêmica do professor também demonstra ser um problema, uma vez que não buscam se atualizar em meio ao crescente volume de dados biológicos e por não possuírem experiência de pesquisa a nível de pós-graduação (LINO, 2016; CAMARGO, 2007).

2.2 Bioinformática no currículo em cursos de graduação de Ciências da Vida

A partir dos anos 2000, vem sendo discutida a inclusão da Bioinformática no currículo de cursos de graduação relacionados a Ciências Biológicas (FEIG, JABRI, 2002; PHAM *et al*., 2008). E verifica-se que no decorrer dos anos a Bioinformática tem sido aplicada em cursos de graduação e de pós-graduação (MAGANA, 2017).

Na universidade de Drake, nos Estados Unidos, houve a tentativa de implementar um curso de longa duração para o ensino de bioinformática, onde alunos das disciplinas de Biologia Celular e Molecular foram convidados a participar das atividades que envolviam programação e análise de dados biológicos. O estudo apresentou bons resultados, mas também demonstrou as dificuldades de implementação, onde problemas como a falta de habilidades de programação ou mmesmo a interpretação dos dados foram pontos observados (HONTS, 2003).

Em estudo de revisão, a bioinformática tem sido implementada em graduação ou pós-graduação na forma de cursos ou programas universitários e em disciplinas como genética e bioquímica ou de maneira interdisciplinar. Além disso, a principal metodologia nesses cursos é centrada na resolução de problemas biológicos, reforçando que o profissional de bioinformática deve estar apto para tais problemas. E por fim, percebeu-se que os resultados da implementação

de bioinformática no currículo foram positivos, pois permitiram a compreensão dos alunos em relação aos conceitos e a utilização de ferramentas para lidar com os dados biológicos (MAGANA *et al*., 2014).

Em estudo mais recente, observou-se que a implementação de um curso de bioinformática, voltado para genômica e transcriptômica, permitiu que os alunos adquirissem habilidades com sistema operacional Linux e em especial, com linhas de comando através do terminal para execução dos programas (PUCKER, 2019). Isso evidencia o fato de o profissional de bioinformática ter habilidades de lógica, programação e sistemas que utilizam kernel Linux (WELCH *et al*., 2014).

Entretanto, por se tratar de uma área interdisciplinar (VERLI, 2014; GIBAS; JAMBECK, 2001), surge a necessidade de descrever quais as competências que um bioinformata deve ter. Assim, um bioinformata deverá não somente possuir conhecimentos relacionados às Ciências da Computação, Matemática e Estatística, mas também relacionados à Genética e à Biologia Molecular (WELCH *et al*., 2014).

2.3 Habilidades a um bioinformata

Um dos desafios dentro da bioinformática é a compreensão por trás dos algoritmos, ou seja, a maneira como eles funcionam ou a lógica empregada para execução de determinada tarefa. Isso implica no fato de que porventura mais de um algoritmo pode ser utilizado para a realização de apenas um problema biológico e, assim, o bioinformata deverá estar apto para escolher o algoritmo que melhor satisfazer a sua necessidade para solucionar o problema biológico (HONTS, 2003).

A lógica de programação é essencial, além de, ter domínio em uma ou mais linguagens de programação como Python, R, Perl, Java, C e entre outras. Acessar, gerenciar e desenvolver bancos de dados fazem parte da rotina de um bioinformata, o que está intimamente relacionado com a análise de dados biológicos. Também ter conhecimento em áreas como matemática e implementação de modelos estatísticos são importantes para realização das análises (WELCH *et al*., 2014).

2.4 Tecnologias *Open Source* em Bioinformática

Programas *open source*, que do inglês significa “código aberto”, são códigos de programação na qual envolvem a participação de diferentes colaboradores para a construção de

um *software* ou projeto, onde cada colaborador do projeto contribui de forma voluntária. O termo "código aberto" está mais para um conceito filosófico, na qual permite a distribuição do código fonte do *software*, bem como não restringe a redistribuição deste, pois assim o programador terá a liberdade de modificá-lo de acordo com suas necessidades. Para que o *software* seja de código aberto, este deve possuir uma licença para uso, modificação e redistribuição (PIRHADI; SUNSERI; KOES, 2016). Assim, isso garante maior liberdade, criatividade (pois, mais pessoas estarão colaborando para o projeto), a comunidade garante a segurança e bom funcionamento do *software*, o que implica também no surgimento de novas tecnologias.

Dentre as tecnologias *open source* utilizadas em bioinformática, podemos destacar o Biopython, BioJava, Bioperl e RBioconductor. O BioPython é uma poderosa biblioteca gratuita e escrita em Python dedicada ao tratamento de dados da biologia molecular. Possuindo uma fácil sintaxe, seu código é aberto e possui uma grande comunidade que contribui para o desenvolvimento e melhorias do projeto (COCK *et al*., 2009).

O Bioperl é uma biblioteca gratuita escrita em linguagem de programação Perl que possui mais de 300 módulos e é mantida pela *Open Bioinformatics Foundation*. Esta biblioteca, assim como o Biopython, preocupa-se em lidar com dados moleculares, por exemplo, realizando alinhamento entre sequências, construção de árvores filogenéticas e acesso remoto a banco de dados (STAJICH *et al*., 2002).

A biblioteca BioJava é um projeto de código aberto, porém escrito em linguagem de programação Java. Assim como as demais, é focada em análise e tratamento de dados moleculares, como alinhamento de sequências, mas com uma função a mais, a possibilidade de manipular arquivos proteínas provenientes PDB podendo realizar alinhamento estrutural de proteínas ou estruturas tridimensionais. Da mesma forma que os outros projetos, possui uma comunidade ampla que dá suporte no desenvolvimento (HOLLAND *et al*., 2008).

O R/Bioconductor trata-se de uma biblioteca escrita em R. Esta linguagem é de alto nível e fácil de ser interpretada e o foco dela é em análises estatísticas. Da mesma forma que as outras, é uma linguagem de código aberto, mas a ideia principal desta biblioteca é a reprodutibilidade dos códigos, sem a necessidade de reinventar a roda (GENTLEMAN *et al*., 2004).

2.5 Utilização do Jupyter Lab e o gerenciador Bioconda para resolução de problemas biológicos

O Jupyter Notebook é uma ferramenta web, *open source* e gratuita que permite em um mesmo documento a execução de códigos em Python, a visualização da saída, a edição de textos

utilizando *markdown*, bem como a integração de multimídias. Assim, o pesquisador tem a possibilidade de organizar seu código em células, e ainda redigir textos que expliquem seu código, o que garante uma maior interpretabilidade dos scripts (PERKEL, 2018).

Já o Bioconda, trata-se de um gerenciador de pacotes com foco em análise de dados biológicos em Python, R, Java e Perl, por exemplo. O ambiente unificado é essencial para a reprodutibilidade do *software* bem como a manutenção de tantos pacotes, permitindo assim que o usuário ou pesquisador tenha uma maior facilidade para administrar seus projetos (GRÜNING *et al*., 2018).

3. METODOLOGIA

3.1 Desenvolvimento do cookbook.

Para todo o desenvolvimento dos cadernos (*notebooks*), utilizou-se o Jupyter Notebook versão 3.2.2 (PERKEL, 2018) bem como a linguagem de programação Python versão 3.9 (http://python.org).

3.1.1 Manipulação de sequências

Para o tratamento das sequências tanto de nucleotídeos quanto de aminoácidos utilizou-se a biblioteca Biopython (COCK *et al*., 2009). Como exemplo de sequências para serem manipuladas, foi feita uma busca de sequências no GenBank, onde foram extraídas as sequências de nucleotídeos e aminoácidos do gene XPRT (Xantina Fosforribosil Transferase) de *Leishmania donovani* (https://www.ncbi.nlm.nih.gov/nuccore/AF170105.1/) (JARDIM *et al*., 1999) e armazenadas em arquivo fasta para serem manipuladas e interpretadas pela biblioteca.

3.1.2 Alinhamento de sequências

Utilizou-se o algoritmo de alinhamento global Needleman-Wunsch (NEEDLEMAN; WUNSCH, 1970) para realizar o alinhamento par a par de sequências de aminoácidos também do gene XPRT de *L. donovani*. Com estas mesmas sequências, utilizou-se os algoritmos ClustalW (CHENNA, 2003) e Muscle (EDGAR, 2004) para realizar múltiplos alinhamentos entre sequências de aminoácidos de 5 enzimas da família das PRTases (Fosforribosil transferases) extraídas do PDB.

3.1.3 Busca de sequências homólogas

Utilizou-se a ferramenta Blast (CAMACHO *et al*, 2009) para realizar a busca de sequências homólogas na base de dados do próprio NCBI (https://pubmed.ncbi.nlm.nih.gov/). Para isso, as mesmas sequências exemplares das etapas anteriores foram utilizadas para realizar a busca. O algoritmo Blast permite que diferentes formas de busca sejam realizadas, então foram escritos scripts para cada uma das variações de busca no *cookbook*, sendo elas: Blastn, uma consulta ao banco de dados de nucleotídeos utilizando uma sequência de consulta também de nucleotídeos; Blastp, uma consulta ao banco de dados de proteínas utilizando uma sequência de consulta de aminoácidos; Blastx, uma consulta ao banco de dados de proteínas utilizando uma sequência de consulta de nucleotídeos; tBlastn, uma consulta ao banco de dados de proteínas utilizando uma sequência de nucleotídeos traduzida e tBlastx uma consulta ao banco de dados de nucleotídeos traduzidos usando uma sequência consulta de nucleotídeo traduzida.

3.1.4 Visualização de estruturas tridimensionais de proteínas

Nesta etapa, foram escritos scripts utilizando da biblioteca NGLview (NGUYEN; CASE; ROSE, 2018). A proteína escolhida para visualização foi a de código PDB 1PZM (https://www.rcsb.org/structure/1pzm) que é a estrutura cristalográfica da Hipoxantina-Guanina Fosforribosil Transferase (HGPRT) de *Leishmania tarentolae* (MONZANI *et al*., 2007). Buscou-se representá-la em formato *cartoon* evidenciando a estrutura bidimensional de α-hélices e folhas-β.

3.1.5 Modelagem molecular por homologia de proteínas

Para esta última etapa do *cookbook*, utilizou-se o *software* Modeller 10.4 (WEBB; SALI, 2016) para realizar a modelagem molecular por homologia da sequência de aminoácidos da proteína XPRT de *L. donovani* (https://www.ncbi.nlm.nih.gov/nuccore/AF170105.1/). A visualização da estrutura modelada foi realizada utilizando a biblioteca NGLview.

4. RESULTADOS E DISCUSSÃO

O projeto foi escrito em linguagem de programação Python versão 3.9 contendo 6 notebooks, sendo um notebook de introdução a linguagem de programação Python, um notebook para trabalhar manipulação de sequências de nucleotídeos e aminoácidos, um notebook para alinhamento de sequências, um notebook para execução do algoritmo Blast, um notebook para visualização tridimensional de proteínas e por último um notebook para modelagem de proteínas por homologia.

Todo o projeto se encontra hospedado no Github (https://github.com/lucaspalmeira/Bio-learning), ao qual utiliza uma licença GPL 3.0 que confere ao projeto a possibilidade de ser distribuído, modificado e redistribuído (PIRHADI; SUNSERI; KOES, 2016). Isso é importante, pois pode dar liberdade a professores e alunos para adaptá-los de acordo com as necessidades de cada um, ou seja, acrescentar mais exercícios, modificar os exercícios já existentes, etc.

O primeiro notebook dedica-se a ser um tutorial para iniciantes em Python. Inicialmente é ensinado como realizar a instalação da linguagem de programação na versão 3.9, uma versão mais estável e que consequentemente possui menos *bugs* (falhas no *software*). Também o notebook possui uma orientação para instalar o Anaconda Python, um gerenciador de pacotes e bibliotecas escritas em Python. Por padrão o Jupyter Notebook vem incluso nesse gerenciador, o que evita o usuário de ter que instalar os pacotes individualmente.

O Jupyter Notebook funciona como uma aplicação web e ambiente de desenvolvimento, na qual permite que o projeto seja organizado e executado em células (figura 1). As células podem ser utilizadas para escrita de *scripts* ou para marcação de texto (*markdown*) e, além disso, o usuário tem a possibilidade de explicar cada linha de código utilizando hashtags, algo essencial para o entendimento do código. Assim, essas características deste ambiente de desenvolvimento podem ser melhores para um iniciante na programação (DAVIES *et al*., 2020).

Operações matemáticas também foram implementadas para a compreensão dos tipos de dados que o usuário irá lidar. Também foram implementadas operações com texto (*strings*) bem como manipulação de arquivos. Assim, ao lidar com números os usuários estará lidando com dados do tipo inteiro (*int*) ou de ponto flutuante (*float*) e quando for dados de texto então estará lidando com dados do tipo *string*.

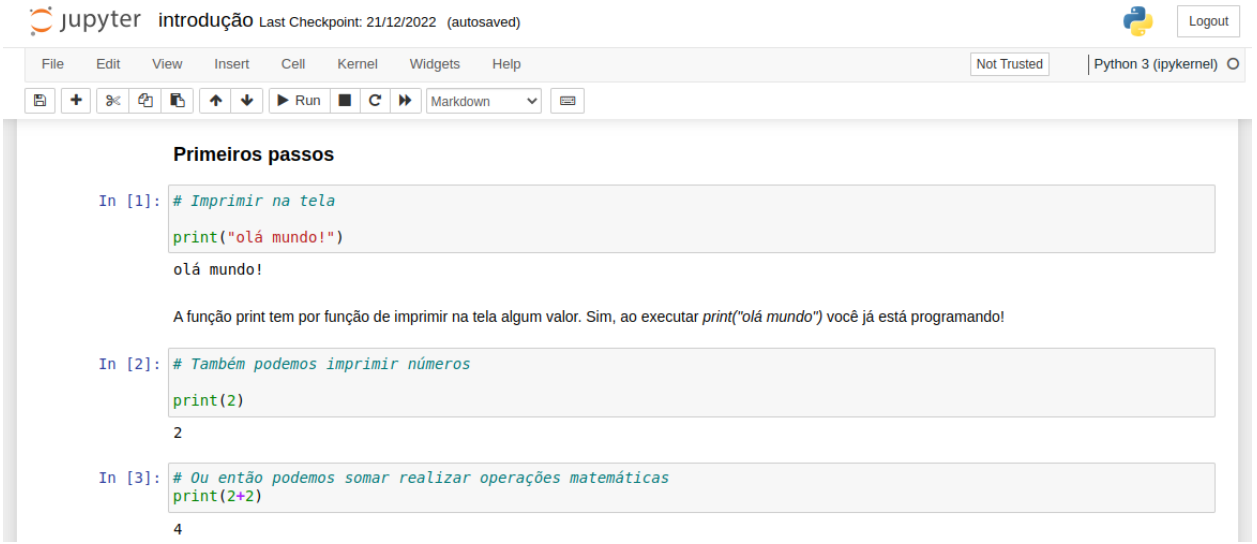

Figura 1. Organização dos primeiros passos para imprimir uma saída no Jupyter Notebook. Fonte: autoria própria.

No segundo notebook, é introduzido o conteúdo acerca do Biopython, um projeto e também uma biblioteca gratuita de código aberto dedicada à análise de dados biológicos por meio da própria linguagem Python (COCK *et al*., 2009). Durante a construção do notebook, a instalação ocorreu sem erros, sendo executada de maneira fácil por uma simples linha de código (figura 2). Foi instalada a versão mais recente da biblioteca (biopython-1.81), onde foi destacada a opção de instalação tanto para sistemas operacionais Windows quanto Linux.

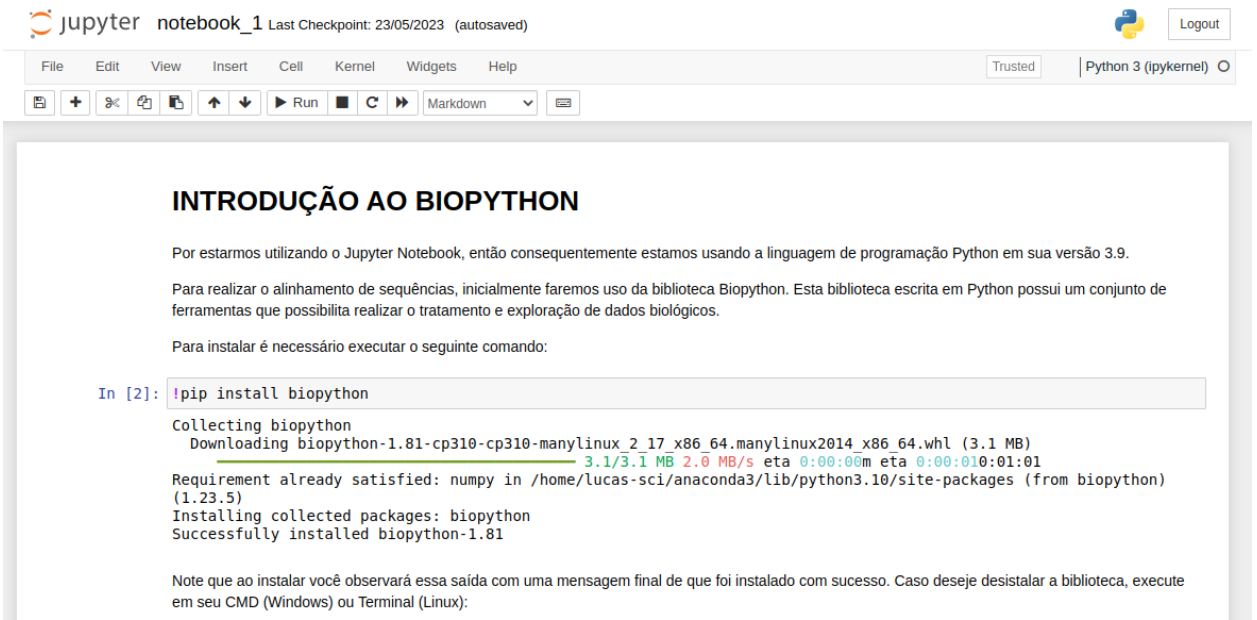

Figura 2. Notebook com instruções para instalação da biblioteca Biopython bem como manipulação de sequências de nucleotídeos e aminoácidos. Fonte: autoria própria.

Ainda no segundo notebook, foi implementada uma introdução à análise de sequências (Figura 3). Esta análise foi realizada através do módulo *Bio.Seq*, onde foi calculado o comprimento da sequência de nucleotídeos (CATGTAGACTAG), bem como a sequência complementar, a sequência reversa e a sequência de aminoácidos traduzida. A implementação desta manipulação de sequências é básica, entretanto faz-se necessário entender quais bibliotecas ou módulos são necessárias para realizá-la, pois faz parte da rotina de um bioinformata (WELCH *et al*., 2014).

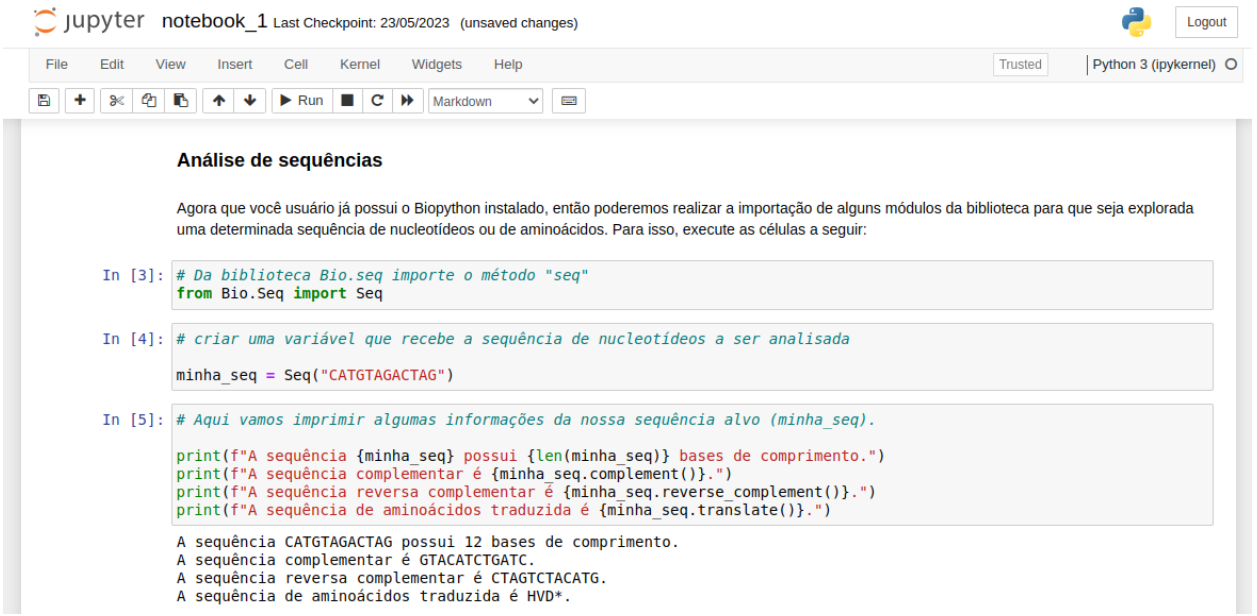

Figura 3. Análise de sequências de nucleotídeos e aminoácidos utilizando da classe *Bio.seq* o módulo *Seq*. Fonte: autoria própria.

Destaca-se ainda neste notebook, a escrita do código que permite a leitura de arquivos do tipo *fasta*. Este tipo de arquivo tem por finalidade de guardar uma sequência de nucleotídeos ou de aminoácidos e informações quanto ao ID da sequência (que poderá ser usada na busca em alguma base de dados) nome do gene, proteína, etc. Neste tutorial, foi utilizada a sequência do gene XPRT (Xantina Fosforribosil Transferase) de *Leishmania donovani* com o ID AF170105.1 (https://www.ncbi.nlm.nih.gov/nuccore/AF170105.1/).

Neste exemplo do *cookbook*, a sequência foi obtida através do NCBI (Centro Nacional de Informações sobre Biotecnologia) (https://www.ncbi.nlm.nih.gov/). O professor ou aluno poderá acessar a base de dados do NCBI e realizar uma busca mais completa com base no ID desta sequência. É importante ressaltar que uma das competências do Bioinformata é a capacidade de lidar com bases de dados, sendo esta habilidade pertencente ao perfil do Cientista de

Bioinformática, sendo que geralmente os biólogos ou graduando em biologia irão apresentar esse tipo de perfil (WELCH *et al*., 2014).

Seguindo a ordem, o terceiro notebook apresentou uma certa dificuldade para sua implementação. Este notebook foi dedicado a realização de alinhamento de sequências, sendo assim necessário realizar a instalação de outros *softwares* como o ClustalW e o Muscle, ambos algoritmos de alinhamento para sequências tanto de nucleotídeos quanto de aminoácidos (CHENNA, 2003; EDGAR, 2004).

Neste notebook, ressalto a importância da saída após ter sido realizada o alinhamento múltiplo de sequências (figura 4), onde professor e aluno ao interpretar a saída em formato de arquivo (extensão .aln) podem chegar a conclusões relacionadas ao grau de similaridade entre as sequências e até mesmo realizar inferências evolutivas. Na saída do alinhamento, a presença de caracteres como asteriscos (*) representam a correspondência entre os nucleotídeos ou aminoácidos, dois pontos (:) apontam que os nucleotídeos ou aminoácidos são semelhantes em sua estrutura química, pontos (indicam que não há correspondência), enquanto os os hifens representam a deleção de um nucleotídeo ou aminoácido na sequência (VERLI, 2014).

Figura 4. Resultado de alinhamento múltiplo entre sequências realizado através do algoritmo ClustalW. Fonte: próprio autor. Fonte: autoria própria.

Outro algoritmo de alinhamento também foi explorado, o Blast (Ferramenta básica de pesquisa de alinhamento local), ao qual foi implementado no quarto notebook do projeto. Este algoritmo permite que seja realizada uma consulta a um determinado banco de dados (local ou *web*) a partir de uma sequência de consulta (*query*) (VERLI, 2014). Uma das principais

dificuldades enfrentadas ao implementá-lo é a demora para se obter a saída quando realizado através do servidor NCBI Blast utilizando internet. Sendo indicado a instalação do *software* Blast+ no desktop do usuário para que seja realizado de forma local sem a necessidade de internet.

Outra dificuldade enfrentada na construção deste notebook é o fato da saída ser um pouco confusa. O fato da sequência consulta ter sido comparada com várias sequências depositadas no banco de dados de sequências não redundantes do GenBank gerou uma saída muito grande e com muitas informações (figura 5) para serem assimiladas. Se o usuário não tiver um conhecimento maior acerca de cada uma das informações apresentadas como o ID da sequência, nome do gene ou proteína, métricas como o E-value e identidade, então um usuário leigo poderá se sentir perdido em meio às informações. Como sugestão, futuramente estes notebooks poderão ser melhorados e uma saída mais organizada poderá ser implementada a fim de diminuir a abstração dos resultados.

```
****Alinhamento****
Sequência:  ref|XP_003392448.1| xanthine phosphoribosyltransferase [Leishmania infantum JPCM5] >gb|AAD50967.1| x
anthine phosphoribosyltransferase XPRT [Leishmania donovani] >emb|CAC9485907.1| xanthine_phosphoribosyltransfera
se [Leishmania infantum] >gb|AYU78560.1| xanthine phosphoribosyltransferase [Leishmania donovani] >emb|CAC542983
3.1| xanthine_phosphoribosyltransferase|GeneDB:LmjF.21.0850 [Leishmania donovani] >emb|CBZ08611.1| xanthine phos
phoribosyltransferase [Leishmania infantum JPCM5]
Tamanho do alinhamento:  241
Score:  654.0
Identidade:  124
E-value:  3.21488e-84
****Alinhamento****
Sequência:  ref|XP_003860616.1| xanthine phosphoribosyltransferase [Leishmania donovani] >gb|TPP49324.1| Phospho
ribosyl transferase domain family protein [Leishmania donovani] >emb|CBZ33911.1| xanthine phosphoribosyltransfer
ase [Leishmania donovani]
Tamanho do alinhamento:  241
Score:  649.0
Identidade:  123
E-value:  2.09021e-83
```

Figura 5. Saída ou resultado após a realização de uma busca de sequências homólogas para a sequência consulta utilizando o Blast (Blastp). Fonte: autoria própria.

O algoritmo Blast, de forma semelhante ao ClustalW e Muscle, permite que sejam realizadas inferências evolutivas. Onde na análise da saída no notebook, observou-se a presença de informações como identidade e E-value. A identidade se refere ao grau de semelhança entre as sequências (consulta versus a sequência encontrada no banco de dados), enquanto que o E-value corresponde a uma métrica estatística que informa se a sequência alinhada foi encontrada de forma aleatória ou não. Assim, quanto maior a identidade e mais próximo de 0 for o valor de E-value significa que as sequências podem ser homólogas (VERLI, 2014).

Nestes dois últimos notebooks, foram apresentados 3 *softwares* de alinhamento (ClustalW, Muscle e Blast). Nesse sentido, como é sugerido em um estudo de implementação e avaliação de um programa de graduação em Bioinformática e Biologia Molecular, o professor

tem a oportunidade de mostrar aos alunos as vantagens e desvantagens no uso de diferentes *softwares*, o que pode agregar numa maior experiência para os alunos, visto que diferentes problemas biológicos requerem diferentes ferramentas a serem utilizadas (PHAM *et al*., 2008).

Em continuação, o quinto notebook explora a visualização tridimensional de proteínas. Sua construção foi feita utilizando a biblioteca NGLview, uma biblioteca em Python para visualização de gráficos moleculares através de *widgets* que podem ser exibidos no Jupyter Notebook. Essa biblioteca suporta a visualização de pequenas moléculas que podem ser implementadas pelo RDKit (biblioteca em Python dedicada manipulação de pequenas moléculas), bem como extrair dados diretamente do Protein Data Bank, facilitando assim a visualização de proteínas (NGUYEN; CASE; ROSE, 2018).

No decorrer da construção deste notebook, alguns erros foram obtidos, especialmente para a instalação desta biblioteca (Figura 6), ao qual requer que outras bibliotecas estejam instaladas no *desktop*, como por exemplo, a ipywidgets (https://ipywidgets.readthedocs.io/en/stable/). Para superar isso, foi necessário recorrer a fóruns ou comunidades de programadores e bioinformatas para tentar solucionar a instalação das bibliotecas, como por exemplo o Stack Overflow (https://stackoverflow.com/) e o Biostars (https://www.biostars.org/), respectivamente.

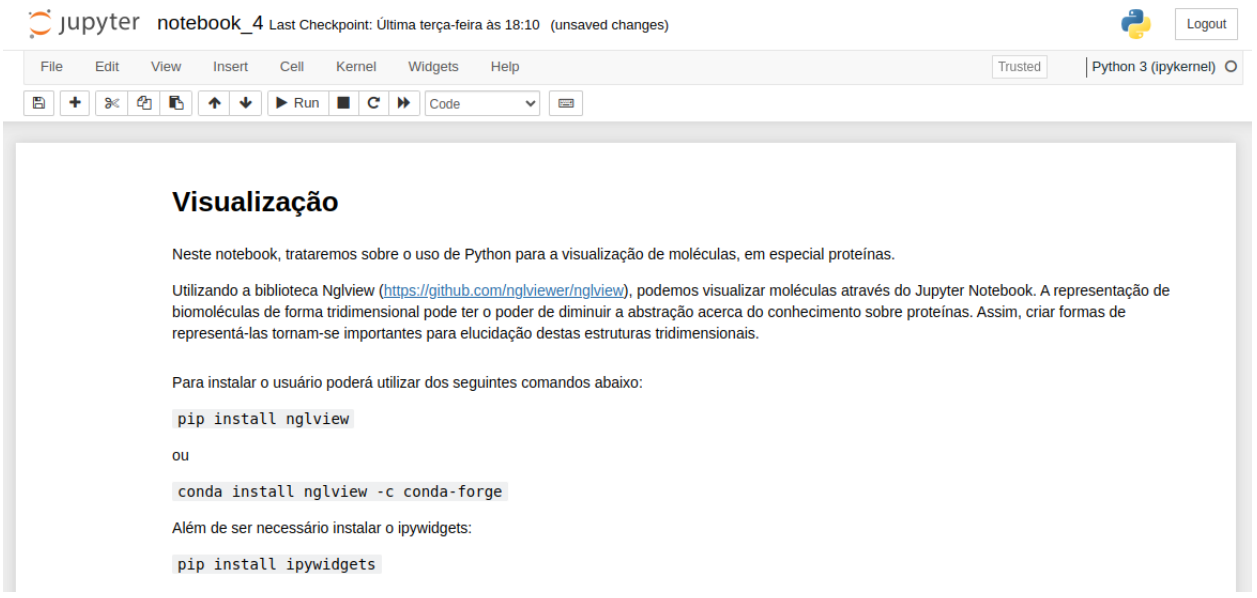

Figura 6. Linhas de comando para instalação das bibliotecas NGLview e Ipywidgets necessárias para a visualização de moléculas no Jupyter Notebook.

Como foi dito, a Biblioteca NGLview permite que estruturas cristalográficas possam ser baixadas diretamente do Protein Data Bank (PDB). O PDB já conta com mais de 200 mil estruturas de proteínas depositadas em sua base de dados. Apesar dos erros durante a instalação

da biblioteca, suas funcionalidades permaneceram. Assim, foi possível extrair a proteína de código 1PZM (HGPRT) do PDB e então visualizá-la em formato *cartoon*, exibindo suas α-hélices e folhas-β (Figura 7). Outras formas de visualização também foram exploradas no notebook, por exemplo, em formato de superfície e esferas e bastões (*balls and sticks*)

Tanto α-hélices quanto folhas-β são formadas a partir de interações entre as cadeias laterais de aminoácidos, geralmente por ligações de hidrogênio (VERLI, 2014). Durante a visualização da proteína, é possível notar a presença de três α-hélices e 7 folhas-β, além disso, é importante dizer que esta proteína é um dímero, ou seja, possui 2 cadeias (A e B) (MONZANI *et al*., 2007). Pois, ao se deparar com outras estruturas no PDB, o aluno precisa estar ciente que as proteínas podem estar em conformações de dímeros, trímeros, tetrâmeros, pentâmeros ou mais (VERLI, 2014).

Também foi evidenciado o sítio ativo desta proteína apenas mudando as cores das regiões onde se encontram os aminoácidos envolvidos na interação com o substrato natural, a guanina monofosfato ou GPM (MONZANI *et al*., 2014). Entretanto, não foi possível escrever um código que possibilitasse também a visualização do GPM (figura 8). Desta forma, os aminoácidos foram representados nas cores verde, azul e vermelho que fazem parte das alças (*loops*) 1, 3 e 4, respectivamente. A alça 2 não foi possível de ser visualizada, mas não por um erro no *script* implementado, e sim pela falta dos aminoácidos no arquivo PDB extraído.

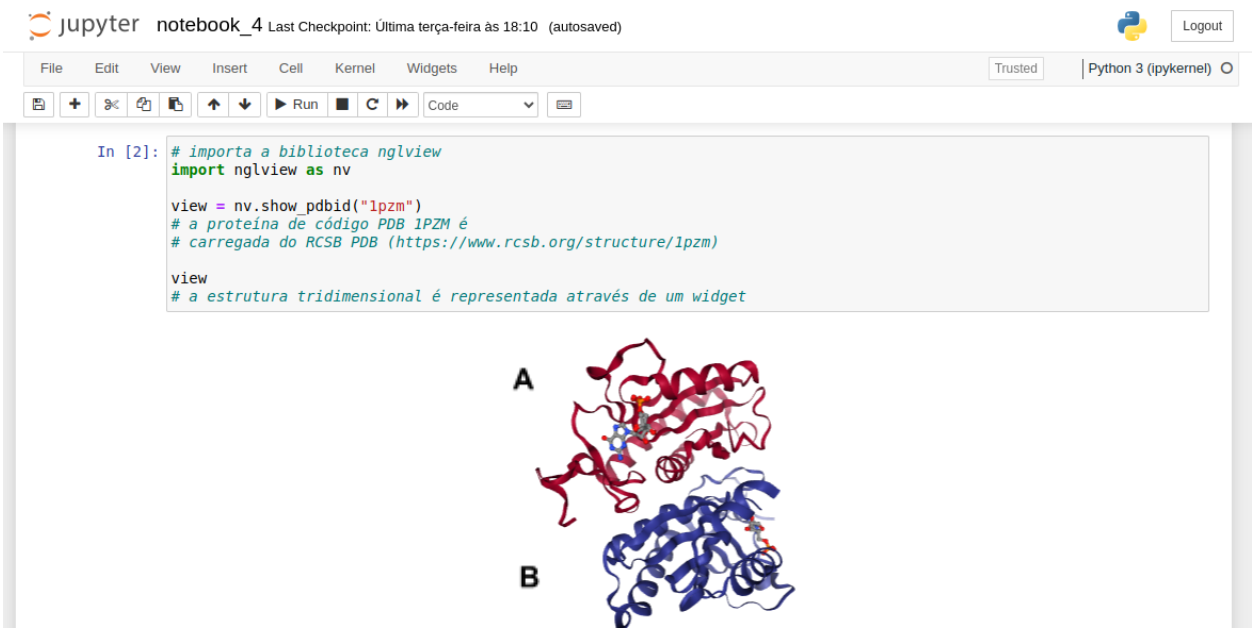

Figura 7. Visualização da estrutura cristalográfica da proteína Hipoxantina-Guanina Fosforribosil Transferase de *Leishmania tarentolae* complexada com o substrato natural de GMP (guanosina monofosfato). Em vermelho e azul as cadeias A e B das proteínas, respectivamente. Fonte: autoria própria.

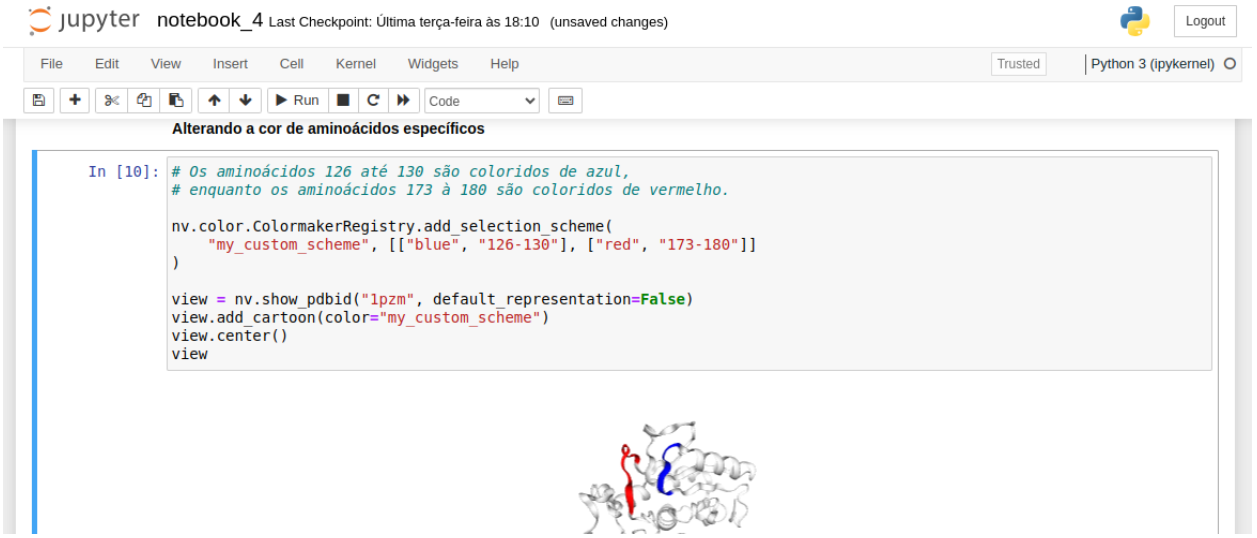

Figura 8. O sítio ativo da proteína evidenciado através dos *Loops* 1 (verde), 3 (azul) e 4 (vermelho).

Por último, foi elaborado o notebook acerca da técnica de modelagem molecular por homologia. Ela permite que seja construída a estrutura tridimensional de uma proteína a partir de sua sequência de aminoácidos e molde cristalográfico de proteína homóloga. Se observarmos a quantidade de dados de sequências gênicas depositadas no Genbank e compararmos com a quantidade de proteínas depositadas no Protein Data Bank, notamos uma enorme diferença. Como alternativa para superar isso, são empregadas as técnicas de modelagem molecular de proteínas.

Na construção deste notebook, dificuldades foram encontradas para realizar a instalação do *software* Modeller 10.4, sendo este *software* responsável para realizar a modelagem molecular, pois o pacote não era encontrado pelo gerenciador Anaconda ao tentar instalar pelo terminal Linux. Para superar isso, foi necessário realizar o download do pacote manualmente diretamente do site do Anaconda para então ser instalado. Essas dificuldades relacionadas à instalação de pacotes são descritas em estudo de montagem e anotação do genoma em sala de aula, sendo uma das desvantagens de se utilizar programas, pacotes ou bibliotecas *open source* (JUNG *et al*., 2020).

Para realizar a modelagem da proteína alvo a Xantina Fosforribosil Transferase de *Leishmania donovani* (ID Genbank: AF170105.1), foi necessário introduzir algumas informações e conceitos, como por exemplo, como é a estrutura de um arquivo com extensão *ali* (arquivo de entrada contendo a sequência alvo para ser modelada) e também o conceito de *Bitscore*. Além disso, neste notebook os conceitos de Identidade e *E-value* também foram retomados, tendo em vista que esse último notebook possui uma dificuldade maior até mesmo para ser construído.

O primeiro passo para realizar a modelagem molecular por homologia é a busca de sequências homólogas em relação à sequência da proteína alvo. Assim, foram retornadas 45 sequências de proteínas depositadas no Protein Data Bank com identidades entre 31% e 55%. O Modeller considera que duas sequências podem ser homólogas quando há no mínimo 25% de identidade, ou seja, 25% de correspondência entre os aminoácidos de ambas as sequências (WEBB; SALI, 2016). Além disso, a saída após esse passo também é dividida em colunas, trazendo os valores de *Bitscore*, *E-value* e identidade (figura 9).

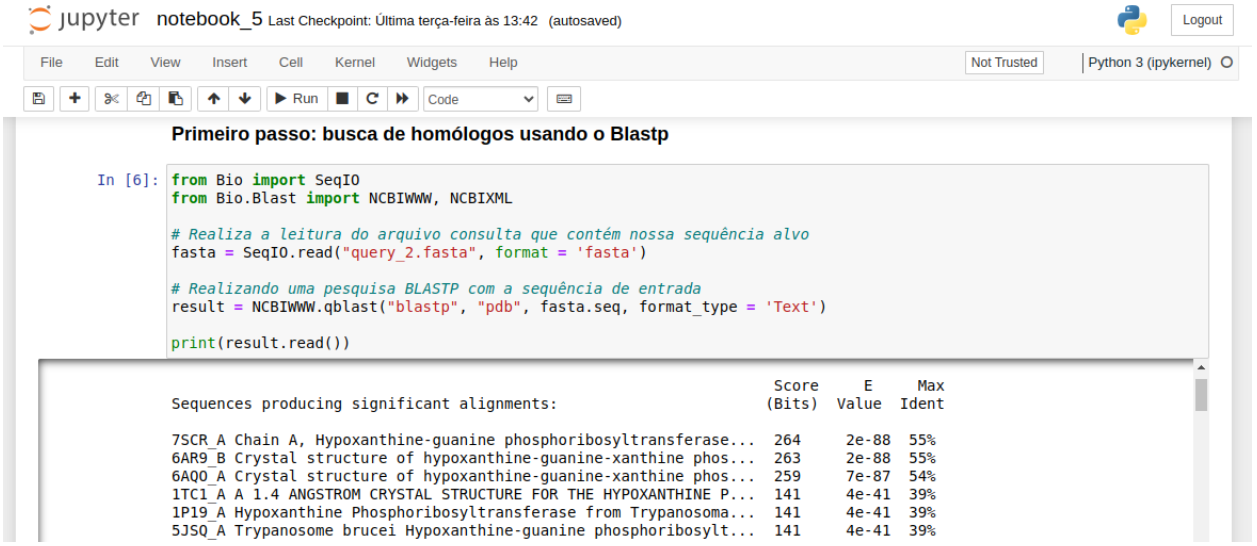

Figura 9. Célula contendo o script para execução do Blast e sua saída após execução, exibindo as sequências homólogas encontradas no PDB.

Para este *cookbook*, a sequência homóloga escolhida foi a de código PDB 7SCR, pois é a que apresenta o maior valor de *Bitscore* e também por apresentar o menor valor de *E-value* e a maior identidade. Além disso, levou-se em consideração também a resolução cristalográfica da proteína, que é dada em Angstrons (Å), onde a proteína 7SCR apresentou uma resolução de 2.12 Å. Em um curso de bioinformática aplicado a estudantes de graduação e pós-graduação é descrito que uma boa resolução deve ser igual ou menor que 3 Å (BADOTTI *et al*., 2014).

As dificuldades encontradas para realizar a modelagem molecular estão mais relacionadas à escrita dos scripts. Apesar de poderem ser encontrados no manual do *software*, eles requerem um nível maior de conhecimento na sintaxe da linguagem Python e no decorrer do desenvolvimento deste notebook alguns erros de espaçamento e sintaxe levaram a um maior custo de tempo para realizar a modelagem molecular.

Para a construção do modelo tridimensional da proteína alvo, foi optado por construir apenas 1 modelo (figura 10). Por padrão o Modeller permite que sejam construídos 5 modelos,

mas a escolha de apenas um se deu pelo fato da saída ser muita extensa. O que iria ocasionar na criação de muitos arquivos de saída bem como uma possível dificuldade para o usuário interpretar a saída.

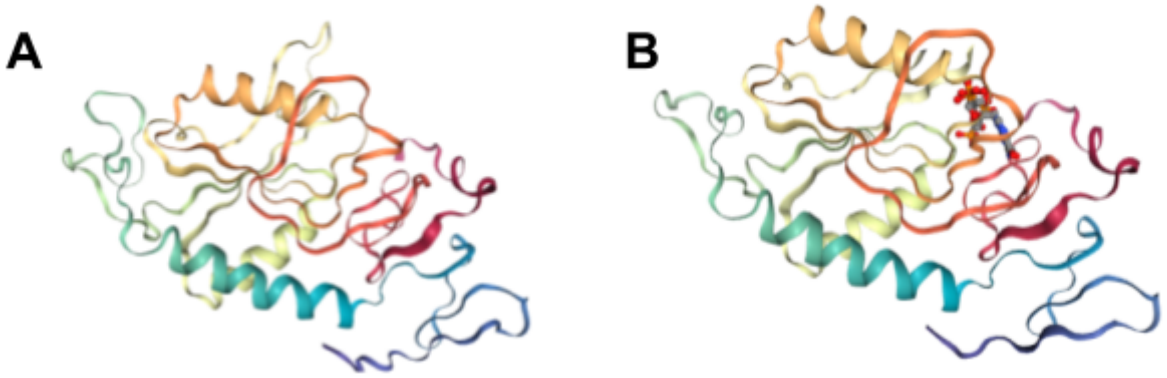

Figura 10. Estrutura tridimensional da proteína alvo (A) e seu modelo cristalográfico de código PDB 7SCR (B).

O *cookbook* se encerra com a modelagem molecular por homologia. No decorrer da construção dos notebooks, notou-se as dificuldades para lidar com diferentes bibliotecas ou pacotes. Também se nota que os notebooks devem ser escritos com grande acurácia na sintaxe dos *scripts*, para que sejam evitados possíveis erros ao serem executados. É sugerido que sejam realizados testes constantes e que haja suporte técnico para manutenção destes notebooks, tendo em vista que um público mais leigo poderá ter acesso a esse tipo de material (DAVIES *et al*., 2020).

Além disso, é possível que alunos possam sentir dificuldades quanto à interpretação e escrita dos *scripts* presentes em cada um dos notebooks. Isso foi observado em trabalho para incorporar a Bioinformática no currículo do curso de Biologia na Universidade de Drake nos Estados Unidos, onde alunos relataram que tiveram dificuldades para realizar as tarefas de programação. Como sugestão, o estudo trás que essas atividades quando feitas em grupo podem ser divididas as tarefas de modo que aqueles que apresentarem mais dificuldades na programação realizem atividades que não envolvam a programação diretamente (HONTS, 2003).

CONCLUSÕES

Este projeto em formato de *cookbooks* tem por finalidade servir como um aporte básico para que alunos e professores de Biologia Molecular possam desenvolver práticas de Bioinformática empregando conceitos da Biologia Molecular. Um dos aspectos observados durante a construção deste material é a dificuldade para lidar com a instalação de diferentes

pacotes e bibliotecas, mesmo havendo gerenciadores para isso como o Anaconda Python. Então, o ideal é que seja realizada uma leitura prévia da documentação das ferramentas, *softwares*, bibliotecas e gerenciadores utilizados nesse projeto para que erros sejam evitados durante a execução de cada um dos notebooks. Além disso, outros recursos para superar essas adversidades é o acesso a fóruns como o Stack Overflow (https://stackoverflow.com/) e o Biostars (https://www.biostars.org/).

REFERÊNCIAS BIBLIOGRÁFICAS

LINO, G. P. M.; BARATELI, L. O.; SANTOS, R. S. A formação de professores de biologia e a prática docente: As principais dificuldades para o ensino de genética dentro de uma concepção pedagógica. **Revista CTS IFG Luziânia**, v. 2, n. 1, 2016

BADOTTI, F. *et al*. Comparative modeling of proteins: A method for engaging students' interest in bioinformatics tools. **Biochemistry and Molecular Biology Education**, v. 42, n. 1, p. 68–78, 2014.

BERMAN, H. M. *et al*. The Protein Data Bank. **Nucleic Acids Research**, v. 28, n. 1, p. 235–242, 2000.

BOYLE, J. A. Bioinformatics in undergraduate education: Practical examples. **Biochemistry and Molecular Biology Education**, v. 32, n. 4, p. 236–238, 2004.

CAMARGO, S. S.; INFANTE-MALACHIAS, M. E.; AMABIS, J. M. O ENSINO DE BIOLOGIA MOLECULAR EM FACULDADES E ESCOLAS MÉDIAS DE SÃO PAULO. **Revista de Ensino de Bioquímica**, v. 5, n. 1, p. 1, 2007.

CHENNA, R. Multiple sequence alignment with the Clustal series of programs. **Nucleic Acids Research**, v. 31, n. 13, p. 3497–3500, 1 jul. 2003.

COCK, P. J. A. *et al.* Biopython: freely available Python tools for computational molecular biology and bioinformatics. **Bioinformatics**, v. 25, n. 11, p. 1422–1423, 1 jun. 2009.

DAVIES, A. *et al.* Using interactive digital notebooks for bioscience and informatics education. **PLOS Computational Biology**, v. 16, n. 11, p. e1008326, 5 nov. 2020.

EDGAR, R. C. MUSCLE: multiple sequence alignment with high accuracy and high throughput. **Nucleic Acids Research**, v. 32, n. 5, p. 1792–1797, 8 mar. 2004.

FREITAS, X. M. S.; MACIEL-CABRAL, H. M.; SILVA, C. C. DA. O ensino do dogma central da biologia molecular: dificuldades e desafios. **EDUCA - Revista Multidisciplinar em Educação**, v. 7, n. 17, p. 452, 27, maio de 2020.

GENTLEMAN, R. C. *et al.* Bioconductor: open software development for computational biology and bioinformatics. **Genome biology**, v. 5, n. 10, p. R80, 2004.

GIBAS, C.; JAMBECK, P. **Desenvolvendo Bioinformática:** ferramentas de software para aplicações em biologia. Tradução: MIlarepa Ltda. 1. ed. Rio de Janeiro: Campus, 2001. 12 p.

GRÜNING, B. *et al.* Bioconda: sustainable and comprehensive software distribution for the life sciences. **Nature Methods**, v. 15, n. 7, p. 475–476, 2 jul. 2018.

HOLLAND, R. C. G. *et al.* BioJava: an open-source framework for bioinformatics. **Bioinformatics**, v. 24, n. 18, p. 2096–2097, 15 set. 2008.

HONTS, J. E. Evolving Strategies for the Incorporation of Bioinformatics Within the Undergraduate Cell Biology Curriculum. **Cell Biology Education**, v. 2, n. 4, p. 233–247, dez. 2003.

JUNG, H. *et al.* Twelve quick steps for genome assembly and annotation in the classroom. **PLOS Computational Biology**, v. 16, n. 11, p. e1008325, 12 nov. 2020.

JARDIM, A. *et al.* Xanthine Phosphoribosyltransferase from Leishmania donovani. **Journal of Biological Chemistry**, v. 274, n. 48, p. 34403–34410, nov. 1999.

JASKOLSKI, M.; DAUTER, Z.; WLODAWER, A. A brief history of macromolecular crystallography, illustrated by a family tree and its Nobel fruits. **The FEBS Journal**, v. 281, n. 18, p. 3985–4009, 2014.

MAGANA, A. J. *et al.* A Survey of Scholarly Literature Describing the Field of Bioinformatics Education and Bioinformatics Educational Research. **CBE—Life Sciences Education**, v. 13, n. 4, p. 607–623, dez. 2014.

Margaret Oakley Dayhoff 1925–1983. **Bulletin of Mathematical Biology**, v. 46, n. 4, p. 467–472, 1984.

MONZANI, P. S. *et al.* Crystal structure of Leishmania tarentolae hypoxanthine-guanine phosphoribosyltransferase. **BMC Structural Biology**, v. 7, n. 1, p. 59, 25 dez. 2007.

NGUYEN, H.; CASE, D. A.; ROSE, A. S. NGLview–interactive molecular graphics for Jupyter notebooks. **Bioinformatics**, v. 34, n. 7, p. 1241–1242, 1 abr. 2018.

NEEDLEMAN, S. B.; WUNSCH, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. **Journal of molecular biology**, v. 48, n. 3, p. 443–453, 1970.

PERKEL, J. M. Why Jupyter is data scientists' computational notebook of choice. **Nature**, v. 563, n. 7729, p. 145–146, 30 nov. 2018.

PHAM, D. Q. D. *et al*. Implementation and assessment of a molecular biology and bioinformatics undergraduate degree program. **Biochemistry and Molecular Biology Education**, v. 36, n. 2, p. 106–115, 2008.

PIRHADI, S.; SUNSERI, J.; KOES, D. R. Open source molecular modeling. **Journal of Molecular Graphics and Modelling**, v. 69, p. 127–143, set. 2016.

PUCKER, B.; SCHILBERT, H. M.; SCHUMACHER, S. F. Integrating Molecular Biology and Bioinformatics Education. **Journal of Integrative Bioinformatics**, v. 16, n. 3, 25 set. 2019.

RAMACHANDRAN, G. N.; KARTHA, G. Structure of Collagen. **Nature**, v. 174, n. 4423, p. 269–270, 1954.

SAYERS, E. W. *et al*. GenBank. **Nucleic Acids Research**, v. 47, n. D1, p. D94–D99, 2019.

SMITH, T. F.; WATERMAN, M. S. Identification of common molecular subsequences. **Journal of Molecular Biology**, v. 147, n. 1, p. 195–197, 1981.

STAJICH, J. E. *et al.* The Bioperl Toolkit: Perl Modules for the Life Sciences. **Genome Research**, v. 12, n. 10, p. 1611–1618, 1 out. 2002.

SUBRAMANIAN, E. G.N. Ramachandran. **Nature Structural Biology**, v. 8, n. 6, p. 489–491, 2001.

VERLI, H. (Org). **Bioinformática da Biologia à flexibilidade molecular**. 1. ed. São Paulo: SBBq, 2014. p. 3.

WELCH, L. *et al.* Bioinformatics Curriculum Guidelines: Toward a Definition of Core Competencies. **PLoS Computational Biology**, v. 10, n. 3, p. e1003496, 6 mar. 2014.

WEBB, B.; SALI, A. Comparative Protein Structure Modeling Using MODELLER. **Current protocols in bioinformatics**, v. 54, p. 5.6.1, 2016.

AGRADECIMENTOS

Em primeiro lugar, agradeço a Deus pela oportunidade que me concedeu de estar estudando e por me permitir conseguir conquistar os meus objetivos de acordo com a vontade d'Ele. Em segundo, agradeço também aos meus pais Adivaldo da Costa Palmeira e Ivanusia de Sousa Santana pelo apoio incondicional durante toda minha caminhada acadêmica. Agradecer também ao meu irmão Vinícius Sousa Palmeira que também se fez presente e que sempre me motivou a continuar firme em meus objetivos.

Quero agradecer aos meus familiares também por todo apoio, por toda ajuda e incentivo ao longo desta graduação, também por acreditarem e apostarem em mim. Sem dúvidas foram

muito importantes para que eu me mantivesse na universidade estudando. Agradeço também aos amigos que ao longo desta graduação estiveram sempre presentes, ajudando, contribuindo e se divertindo para que a vida não ficasse somente resumida a Universidade.

Agradeço ao meu orientador Dr. Bruno Silva Andrade pela amizade, parceria e pela oportunidade que me deu de poder fazer pesquisa e de sonhar com uma carreira de pesquisador e cientista. Agradeço também aos meus amigos e colegas de laboratório, que sempre estiveram presentes para contribuir com conhecimento. Sou grato também a todos que direta ou indiretamente fizeram parte da minha formação, aos colegas de turma e de curso que de alguma forma contribuíram comigo.

Agradeço a Universidade Estadual do Sudoeste da Bahia e a todos os professores que fizeram parte da minha trajetória pela graduação. Sou grato por todo conhecimento transmitido até a mim e que possibilitou que eu chegasse até aqui.