UNIVERSIDADE DO ESTADO DA BAHIA (UNEB)
DEPARTAMENTO DE CIÊNCIAS EXATAS E DA TERRA, CAMPUS I
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

ADEONITA DOS SANTOS SOUSA

FUNEXPRESSION: UM PIPELINE AUTOMATIZADO PARA ANÁLISE DE EXPRESSÃO DIFERENCIAL DE GENES DE FUNGOS UTILIZANDO DADOS BRUTOS DE BULK RNA-SEQ

SALVADOR
2024

ADEONITA DOS SANTOS SOUSA

FUNEXPRESSION: UM PIPELINE AUTOMATIZADO PARA ANÁLISE DE EXPRESSÃO DIFERENCIAL DE GENES DE FUNGOS UTILIZANDO DADOS BRUTOS DE BULK RNA-SEQ

Monografia apresentada ao Curso de Bacharelado em Sistemas de Informação do Departamento de Ciências Exatas e da Terra (DCET) - Campus I, da Universidade do Estado da Bahia (UNEB), como requisito à obtenção do grau de bacharel em Sistemas de Informação.
Área de concentração: Bioinformática

Orientador: Prof. Dr. Alexandre Rafael Lenz

SALVADOR
2024

AGRADECIMENTOS

Agradeço meu orientador, pela paciência e apoio, inclusive nas sextas à noite e aos finais de semana.

A Jair, meu companheiro de uma década por nunca ter soltado a minha mão.

Isso é tudo.

"Eu, um universo de átomos, um átomo no universo."
(Richard Feynman)

RESUMO

A análise de expressão diferencial de genes desempenha um papel crucial na compreensão de processos biológicos complexos e na identificação de potenciais alvos terapêuticos. Este trabalho apresenta o FunExpression, um pipeline desenvolvido com base nas ferramentas mais precisas para a análise de expressão diferencial de genes utilizando dados brutos de RNA-Seq bulk. O FunExpression possui uma arquitetura robusta, de fácil manutenção e alta reprodutibilidade, graças à utilização de contêineres Docker para gerenciamento e execução de ambientes computacionais. O pipeline foi projetado para ser intuitivo e acessível, permitindo que os usuários realizem análises complexas de forma simples por meio de uma interface web, independentemente de sua localização ou capacidade computacional. Ele é composto pelas seguintes etapas: corte com Trimmomatic, alinhamento com RNA STAR, contagem com HTSeq e, por fim, análise de expressão diferencial de genes com DESeq2. O desempenho do FunExpression foi avaliado em cinco conjuntos de dados de RNA-Seq extraídos de dois artigos de referência. Os resultados gerados pelo pipeline foram similares aos descritos nos artigos. No entanto, o FunExpression demonstrou ser mais minucioso em comparação com o pipeline utilizado em um dos artigos avaliados, revelando diversos DEGs que não foram identificados no estudo original. Este trabalho, portanto, oferece uma ferramenta útil e eficaz para pesquisadores interessados em explorar a expressão gênica em diferentes condições biológicas. O código-fonte do FunExpression está disponível em: https://github.com/Adeonita/funexpression.

Palavras-chave: FunExpression; RNA-Seq; análise de expressão diferencial; bulk; DEGs; *pipeline*.

ABSTRACT

The differential gene expression analysis plays a crucial role in understanding complex biological processes and identifying potential therapeutic targets. This work presents FunExpression, a pipeline developed using the most accurate tools for differential gene expression analysis with bulk RNA-Seq raw data. FunExpression features a robust architecture, easy maintenance, and high reproducibility, thanks to the use of Docker containers for computational environment management and execution. The pipeline was designed to be intuitive and accessible, enabling users to perform complex analyses effortlessly through a web interface, regardless of their location or computational capacity. It comprises the following steps: trimming with Trimmomatic, alignment with RNA STAR, counting with HTSeq, and, finally, differential gene expression analysis with DESeq2. FunExpression's performance was evaluated on five RNA-Seq datasets derived from two reference articles. The results generated by the pipeline were similar to those described in the articles. However, FunExpression proved to be more thorough compared to the pipeline used in one of the evaluated articles, uncovering several DEGs that were not identified in the original study. This work, therefore, provides a useful and effective tool for researchers interested in exploring gene expression under different biological conditions. The FunExpression source code is available at: https://github.com/Adeonita/funexpression.

Key-words: FunExpression; RNA-Seq; differential expression analysis; bulk; DEGs; pipeline.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DNA	Ácido Desoxirribonucleico
RNA	Ácido ribonucleico
EBI	Instituto Europeu de Bioinformática
RNA-Seq	Sequenciamento de RNA
DEGs	*Differentially Expressed Genes*
A	Adenina
T	Timina
G	Guanina
C	Citosina
U	Uracila
GEO	*Gene Expression Omnibus*
MIAME	*Minimum Information About a Microarray Experiment*
GTF	*Gene Transfer Format*
GFF	*General Feature Format*
TPM	*Transcripts Per Million*
RPKM	*Reads Per Kilobase of transcript per Million mapped reads*
FPKM	*Fragments Per Kilobase Million*
TMM	*Trimmed Mean of M-values*
DESeq2	*Differential Expression using Sequence data*
DE	Diferencialmente Expressos
edgeR	*Empirical Analysis of Digital Gene Expression in R*
WT	*Wild Type*
LFC	*Log Fold Change*
FC	*Fold Change*

log2FC	*Logarithm base 2 of the Fold Change*
NCBI	*National Center for Biotechnology Information*
AWS	*Amazon Web Service*
PubMed	*Public Medicine*
ACM	Association for Computing Machinery
DSR	*Design Science Research*
G2BC	Grupo de Pesquisa em Bioinformática e Biologia Computacional
SRA	Sequence Read Archive

SUMÁRIO

1 INTRODUÇÃO

Desde a década de 70, já existia a necessidade de analisar grandes volumes de dados biológicos, em especial sequências de DNA (Ácido Desoxirribonucleico), RNA (Ácido ribonucleico) e proteínas. Neste mesmo período, Needleman desenvolvia um método adaptável para encontrar semelhanças nas sequências de duas proteínas (Needleman S. B., 1970). Nesse cenário, a bioinformática surge como um campo interdisciplinar que combina biologia, informática e matemática, a fim de facilitar a análise e interpretação desses dados biológicos, a partir do desenvolvimento e utilização de algoritmos, ferramentas computacionais e softwares, (Luscombe N. M., 2001). E tornou-se essencial para diversas áreas como:

- a genômica, para a partir de análises de sequenciamento, identificar genes reguladores e variantes genéticas;
- a proteômica computacional, para o estudo da estrutura e função de proteínas;
- a biologia de sistemas, para modelar e entender a complexidade de sistemas biológicos;
- e a transcriptômica, para analisar dados de expressão gênica, permitindo a compreensão do padrão de transcrição em diferentes condições.

Com o passar do tempo, estes dados biológicos continuaram se multiplicando a uma taxa sem precedentes. Em agosto de 2000, o repositório do GenBank, banco de dados de sequências genéticas, possuía 8.214.000 entradas de sequências de DNA e o banco de dados SWISS-PROT possuía 88.166 sequências de proteínas. O Instituto Europeu de Bioinformática (EBI) em Hinxton, Reino Unido, parte do Laboratório Europeu de Biologia Molecular e um dos maiores repositórios de dados biológicos do mundo, em 2013, armazenava 20 petabytes[1] de dados e backups sobre genes, proteínas e pequenas moléculas (Marx, 2013). Atualmente, em 2024, o repositório do Genbank armazena mais de três bilhões e duzentos milhões de sequências; desde a sua criação, tem dobrado de tamanho aproximadamente a cada 18 meses (Information, 2024b).

A medida que esses dados avolumam, crescem os desafios para lidar com eles (Luscombe N. M., 2001). Desse modo, a bioinformática tem um papel fundamental de auxiliar nessas análises, pois ela permite a identificação de padrões, a previsão de estruturas e funções biológicas, e a compreensão das bases moleculares de doenças, possibilitando o desenvolvimento de novos tratamentos e terapias numa escala de exploração e compreensão que seria impraticável de serem realizadas sem os recursos da computação (Marx, 2013).

[1] 1 petabyte equivale a 10^{15} bytes

Para que seja possível transformar toda essa massa de dados brutos de sequenciamento em informações biológicas significativas, a bioinformática se utiliza da transcriptômica, que é o estudo do conjunto completo de transcritos [2], para um estado de desenvolvimento ou uma condição fisiológica específica. Os principais objetivos da transcriptômica são: catalogar todas as espécies de transcritos e quantificar o transcriptoma.

Para atingir os objetivos da transcriptômica, diversos métodos têm sido empregados. Existem diferentes maneiras de realizar a transformação desses dados. Dentre elas, tais como: *Microarray* [3], *Single Cell* [4], tecnologia de sequenciamento de RNA (RNA-Seq), conhecida também como método *Bulk* ou sequenciamento convencional. Segundo Sangket et al. (2022) RNA-Seq (Bulk) é considerada a técnica mais popular, robusta e adaptável para medir a expressão gênica e a ativação da transcrição em todo o genoma (Corchete et al., 2020). Nos últimos anos, a aplicação da tecnologia de RNA-Seq tornou-se mais extensa e o número de dados brutos disponíveis aumentou. Portanto, selecionar um fluxo de trabalho apropriado tornou-se uma questão importante para pesquisadores da área (Liu et al., 2022).

A escolha do fluxo de trabalho adequado envolve considerar e tratar de maneira eficiente as diversas dependências técnicas, como ferramentas responsáveis por subtarefas essenciais, incluindo controle de qualidade, corte, mapeamento e contagem de dados. Além disso, a complexidade do software utilizado nessas etapas torna sua instalação, configuração e implantação desafiadoras, exigindo atenção especial para garantir a precisão e a integridade dos resultados obtidos. Os métodos de análise de DEGs *(Differentially Expressed Genes)* são complexos e a maioria deles requer conhecimento prévio de uma linguagem de programação ou *shell*[5] de linha de comando, e os usuários que não possuem esse conhecimento precisam investir tempo e esforço para adquiri-los (Sangket et al., 2022). E, embora hajam diversas ferramentas para serem utilizadas em cada etapa do *pipeline* [6], não existe um consenso claro sobre os algoritmos e *pipelines* mais apropriados que devem ser usados para analisar dados de RNA-Seq (Corchete et al., 2020). Essa ausência de consenso pode impactar diretamente a escolha dos métodos, uma vez que diferentes algoritmos podem gerar resultados distintos, dependendo das características do experimento e das suposições subjacentes aos modelos estatísticos empregados.

Além disso, a implementação prática desses métodos enfrenta outro desafio: os cálculos precisam ser realizados em plataformas computacionais e arquiteturas de sistema heterogêneas. Essas variam desde pequenos *laptops* até *clusters* de alto desempenho, o

[2] É o material resultante do processo de transcrição celular.
[3] *Microarray* é uma técnica de sequenciamento de RNA que fixa fragmentos de ácido nucleico em um chip para análise laboratorial.
[4] Técnica que analisa a expressão gênica em células individuais
[5] Programa responsável por interpretar as instruções enviadas pelo usuário e seus programas ao sistema operacional
[6] Série de etapas ou processos automatizados aplicados para uma análise específica.

que exige uma adaptação das ferramentas e dos fluxos de trabalho às capacidades de processamento disponíveis. Assim, a combinação entre a falta de padronização teórica e as demandas práticas de execução torna o processo de análise de RNA-Seq ainda mais desafiador, pois é possível percebermos que há uma lacuna significativa no fluxo de trabalho para a montagem de um ambiente destinado à análise de expressão diferencial de genes.

A partir dessa constatação, este estudo apresenta a seguinte pergunta norteadora: "Como automatizar e facilitar o processo de análise de expressão diferencial de genes utilizando as ferramentas que estudos demonstraram ser mais precisas para construção de um *pipeline* de análise de dados brutos de RNA-Seq?"

Fazendo um contraponto ao modelo normalmente seguido na construção de um *pipeline*, onde as ferramentas são instaladas localmente, espera-se que a construção de um *pipeline* web para realização de análise de DEGs, a partir de dados brutos de RNA-Seq, contribua como uma solução para automatizar o processo. Pretende-se, com isso, que usuários sem conhecimento de linguagens de programação ou *shell* de comando realizem suas análises em um ambiente web, onde não é necessário preocupar-se com a configuração ou alto poder computacional. Dessa forma, o pipeline deve permitir análises de expressão diferencial especificamente para fungos, facilitando o acesso e a utilização dessa tecnologia por uma ampla gama de pesquisadores e profissionais. A ferramenta deve ser capaz de gerar DEGs de organismos não-modelo, cujos resultados obtidos beneficiarão diretamente esta área de estudo e poderão ser aplicados em outras áreas, com a integração em diversas outras ferramentas de análise de dados. Por exemplo, os DEGs resultantes podem servir de entrada para que a ferramenta FunRegulation [7] os utilize na criação de redes de regulação de genes.

A partir das lacunas e justificativas descritas, o objetivo geral deste projeto é implementar um *pipeline* automatizado destinado a análise de expressão diferencial de genes, a partir de dados brutos gerados pela técnica de *RNA-Seq (Bulk)* disponíveis publicamente no banco de dados GEO (*Gene Expression Omnibus*). Para alcançar o objetivo geral, foram estabelecidos os seguintes objetivos específicos:

1. Identificar as etapas do *pipeline* desde a obtenção dos dados brutos até a apresentação dos resultados.

2. Identificar as ferramentas mais adequadas utilizadas em cada etapa do *pipeline* através de uma revisão de literatura.

3. Desenvolver um ambiente de experimentação para execução de um *pipeline* que encontre os DEGs, a fim de gerar a lista de genes cuja a expressão se encontre aumentada (*upregulated*) ou diminuída (*downregulated*).

[7] Ferramenta desenvolvida pelo grupo de pesquisa G2BC para criação de redes de regulação de genes

4. Avaliar o *pipeline* estabelecido a partir de execução em ambiente local e a validação da implementação com a ferramenta finalizada em ambiente *Docker*[8].

Esta monografia está estruturada em sete capítulos. O Capítulo 1 introduz o tema central do trabalho, enquanto o Capítulo 2 apresenta o referencial teórico, discutindo a utilização da transcriptômica em aplicações de bioinformática voltadas para a análise de dados brutos de RNA-Seq, tanto sob uma perspectiva biológica quanto computacional. No Capítulo 3, são detalhadas as metodologias de pesquisa e desenvolvimento empregadas na concepção e validação do FunExpression.

O Capítulo 4 descreve os passos e ferramentas utilizadas na construção do FunExpression, incluindo sua arquitetura e o detalhamento do processo seguido para a realização da Avaliação 1. Já no Capítulo 5, aborda-se a validação da ferramenta web, com uma análise comparativa entre os resultados gerados por ela, os obtidos no ambiente Galaxy[9] e os apresentados nos artigos de referência.

Por sua vez, o Capítulo 6 explora as possibilidades de trabalhos futuros relacionados ao FunExpression, enquanto o Capítulo 7 apresenta as considerações finais, consolidando as conclusões e reflexões obtidas ao longo do trabalho.

[8] É uma plataforma que fornece a capacidade de empacotar e executar um aplicativo em um ambiente isolado chamado contêiner. (Docker, Inc., 2024)

[9] O Galaxy é uma plataforma web poderosa e de fácil utilização para análise de dados científicos. As etapas de uma análise são realizadas com ferramentas disponíveis na plataforma, que traduzem parâmetros de software de linha de comando para uma interface amigável.

2 DO DNA À EXPRESSÃO DIFERENCIAL

O DNA é uma biomolécula que contém informações genéticas que definem a identidade de cada organismo vivo. Ele é composto de uma fita dupla formada de nucleotídeos, cada uma das quais contém uma espinha dorsal de fosfato-desoxirribose e bases nucleicas: adenina (A), timina (T), guanina (G) e citosina (C) (Jeong et al., 2022). Já o ácido ribonucleico, o RNA, possui comumente uma estrutura de fita simples; suas bases nitrogenadas são formadas pela mesmas bases do DNA, com exceção de Timina, que é substituída pela uracila (U). Alguns vírus de RNA têm fita dupla e utilizam RNA como material genético, podendo causar várias doenças humanas (Wang; Farhana, 2024).

Diversos cientistas realizaram pesquisas nesta área, entretanto, foi Bruce Alberts Michael, nascido em 1938, na cidade de Chicago, que destacou-se por realizar estudos abrangentes dos complexos de proteínas, os quais permitem a replicação de cromossomos na ocasião da divisão de células vivas. Ele também foi autor-chefe do livro Biologia Molecular da Célula (do original em inglês, *Molecular Biology of the Cell*), publicação que se tornou referência na academia científica internacional tanto na disposição e divulgação de informações quanto na sua qualidade editorial (Ciências, 2024). Devido à relevância do trabalho de Alberts, o livro Biologia Molecular da Célula é utilizado como principal fonte do referencial teórico biológico desta pesquisa.

2.1 O dogma central da biologia molecular

Até a década de 50 era desconhecida a maneira como se davam os processos fundamentais de transferência de informação genética entre o DNA, RNA e proteínas. Era sabido que as proteínas possuíam uma estrutura tridimensional e sua finalidade era completamente dependente dessa estrutura. Entretanto, Crick focou seu olhar em como eram formadas as cadeias, e desta maneira ele transformou um problema tridimensional em unidimensional, e formulou o que até os dias de hoje é conhecido como o Dogma central da biologia molecular. Ele propôs que todos os conjuntos de proteínas provinham de um conjunto universal de vinte aminoácidos que eram utilizados em toda a natureza, e que a transferência de informação genética era dividida em três classes (Crick, 1970):

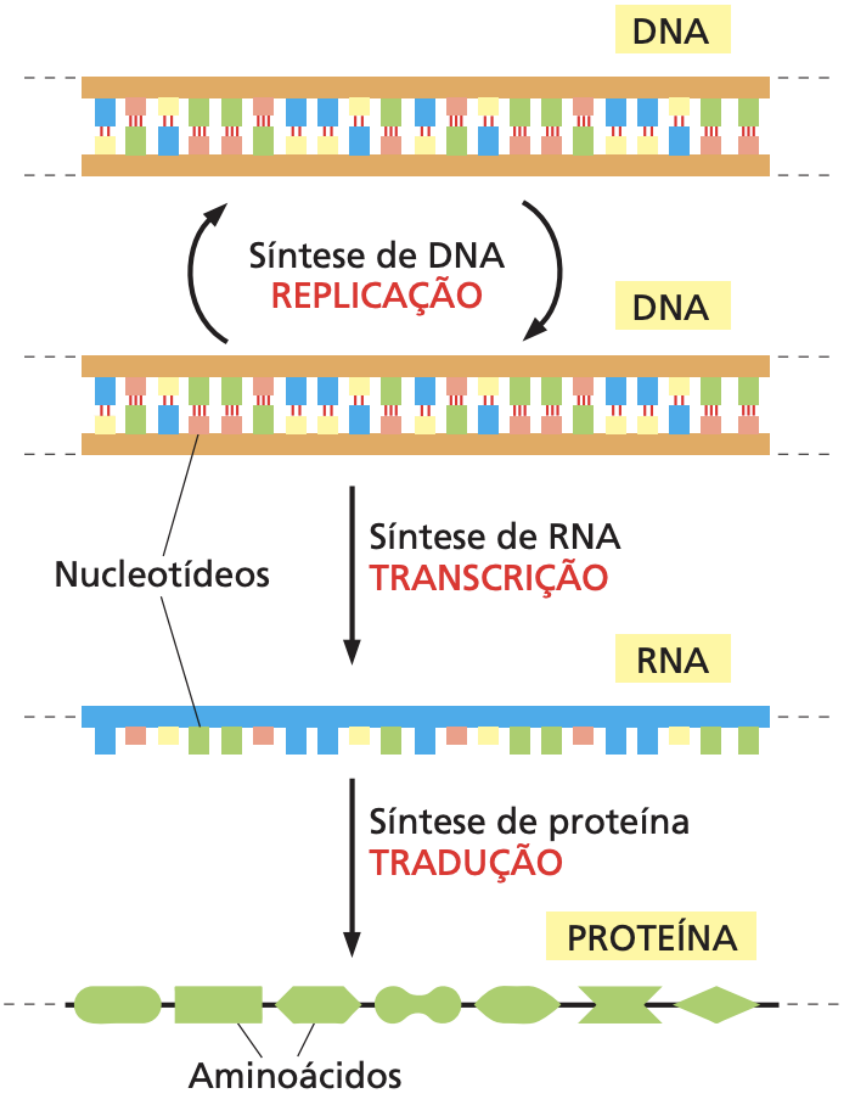
- Transferência geral:
  - DNA -> DNA
  - DNA -> RNA
  - RNA -> Proteína

- Transferência Especial:
  - RNA -> RNA
  - RNA -> DNA
  - DNA -> Proteína
- Transferência Desconhecida:
  - Proteína -> Proteína
  - Proteína -> RNA
  - Proteína -> DNA

A transferência geral é a forma que ocorre em todas as células, o fluxo de informações genéticas sempre flui do DNA para o RNA e deste para a proteína, e nunca no sentido contrário Figura 1. A segunda ocorre apenas em casos especiais, como, por exemplo quando a célula é infectada por vírus. Em relação à transferência desconhecida, o dogma central postula como impossível.

Ainda hoje, o dogma central permanece como pilar essencial da biologia molecular, guiando novas pesquisas e teorias, ao mesmo tempo que se adapta a novas descobertas e conhecimentos.

Figura 1 – Processo de sintetização do DNA a Proteína



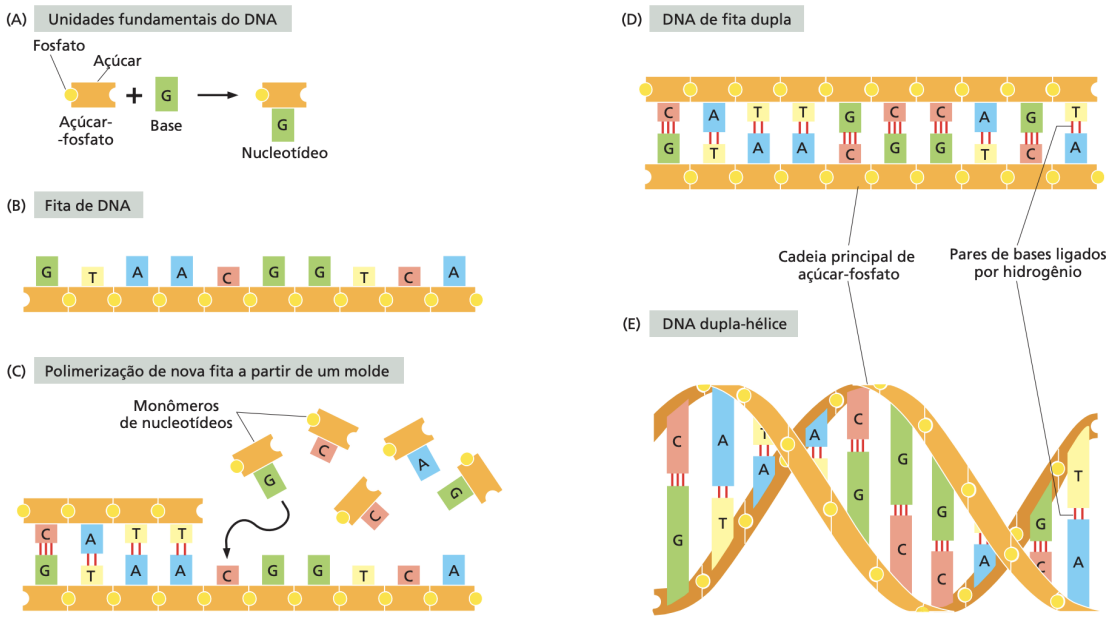
Fonte: Alberts (2017)

2.1.1 Célula, a forma mais simples de vida

Apesar de uma aparente diversidade, todo ser vivo é fundamentalmente semelhante em seu interior. Todos são compostos por pequenas unidades, delimitadas por membranas, preenchidas com uma solução aquosa concentrada de produtos químicos e dotadas da habilidade de criar cópias de si mesmas por meio de seu crescimento e, então, da sua autodivisão em duas (Alberts, 2017). As células contêm o código molecular comum no qual as especificações de todos os organismos vivos estão escritas, e, a partir dele, é possível ler, medir e decifrar informações sobre todas as formas de vida. O corpo humano é composto por mais de 10^{13} células; entretanto, tudo partiu da divisão de uma única célula, na qual

as informações que definem a espécie estão armazenadas em moléculas de DNA, conforme ilustrado na Figura 2 (Pauling; Corey, 1952).

Figura 2 – Unidades fundamentais do DNA

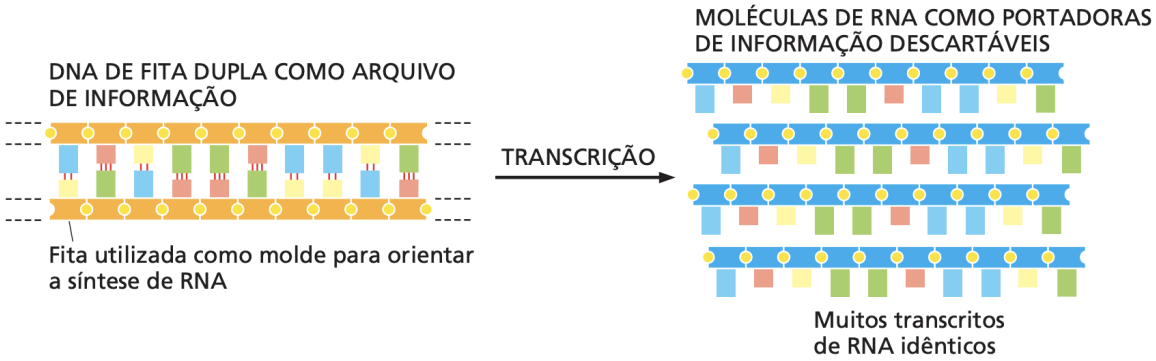


Fonte: Alberts (2017)

2.1.2 A transcrição

Para cumprir sua função de armazenar informações, o DNA deve não apenas se replicar, mas também expressar suas informações, orientando a síntese de outras moléculas na célula. Esse processo de sintetização é conhecido como transcrição, no qual, a partir de um molde, os segmentos de DNA são utilizados para promover a síntese de RNA. O resultado da transcrição é um conjunto de moléculas descartáveis idênticas, conforme a Figura 3, cuja sequência de nucleotídeos representa fielmente uma porção da informação genética da célula. A sua finalidade é intermediar a transferência de informações genéticas através do RNA mensageiro (mRNA) que, por sua vez, guiará a síntese de proteínas (Alberts, 2017).

Figura 3 – Sintetização do RNA a partir do DNA



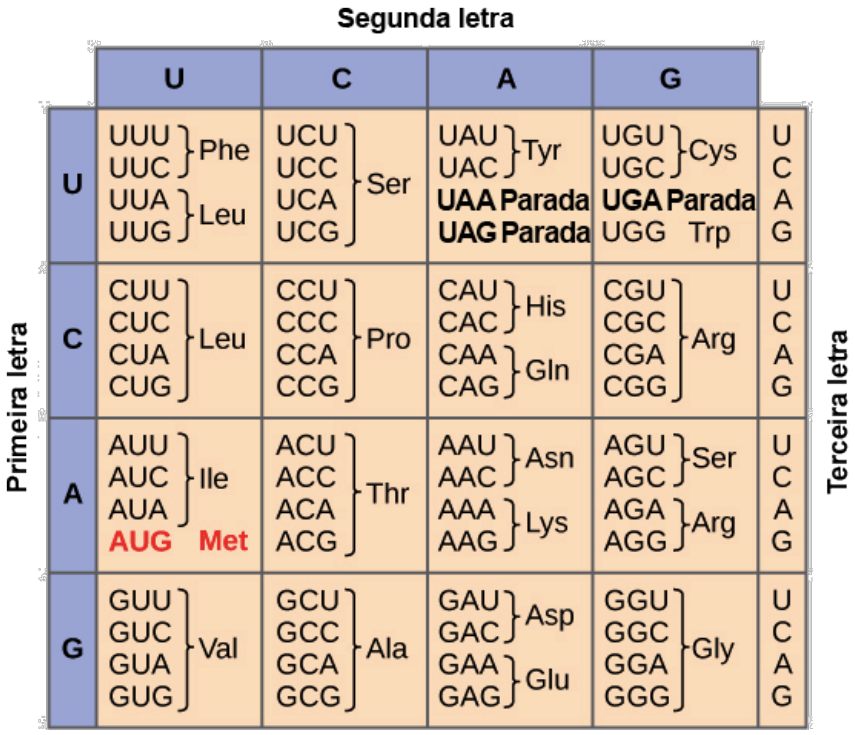
Fonte: Alberts (2017)

2.1.3 A tradução: quatro se transformam em vinte

Assim como no DNA e no RNA, as proteínas são oriundas do mesmo conjunto-padrão de unidades fundamentais monoméricas das moléculas. Elas são geradas a partir do processo de tradução, onde o RNA é utilizado para guiar a sua síntese.

Diferentemente do DNA e do RNA, os monômeros de uma proteína são conhecidos como aminoácidos, que possuem sua sequência especificada pelos polinucleotídeos presentes no mRNA e seu alfabeto é composto por 20 letras em vez de quatro, conforme a Figura 4. Quando ligadas a enzimas específicas, as proteínas desempenham um papel fundamental na catalisação de reações químicas e no rompimento de ligações covalentes. Além disso, elas também são capazes de realizar manutenção de estruturas, gerar movimentos, captar sinais, dentre outras funcionalidades. Cada molécula de proteína desempenhará um papel diferente conforme a sua sequência genética de aminoácidos, em que cada trinca de nucleotídeos, também conhecida como *códon*, codifica um aminoácido na proteína Alberts (2017).

Figura 4 – Alfabeto das proteínas



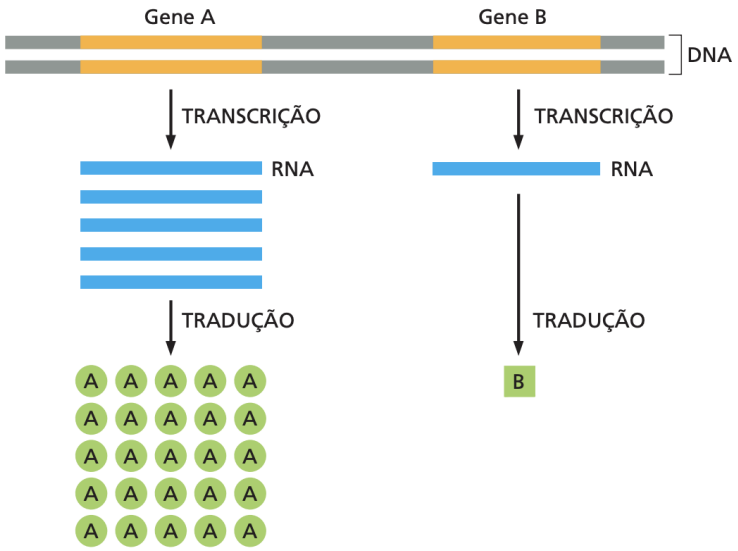
Fonte: CONNIE et al. (2016)

2.2 Transcrição: uma visão detalhada

A transcrição é o processo pelo qual o DNA transcreve o RNA, utilizando a mesma linguagem, ou seja, de nucleotídeos para nucleotídeos, conforme discutido na seção 2.1.2. Embora a maioria das células o utilize como intermediador para síntese de proteínas, alguns genes têm como produto final a própria molécula de RNA.

Durante o processo de tradução, várias cópias de um mesmo mRNA podem ser sintetizadas a partir de um mesmo gene, desta forma, cada molécula de RNA mensageiro pode sintetizar várias cópias idênticas de uma mesma proteína. Ou seja, quando uma célula precisa de uma determinada proteína em maior quantidade, mais moléculas de mRNA serão sintetizadas. Essa amplificação sucessiva permite que as células sintetizem rapidamente, e no momento necessário, um grande volume de proteínas (Alberts, 2017). Conforme ilustrado na Figura 5.

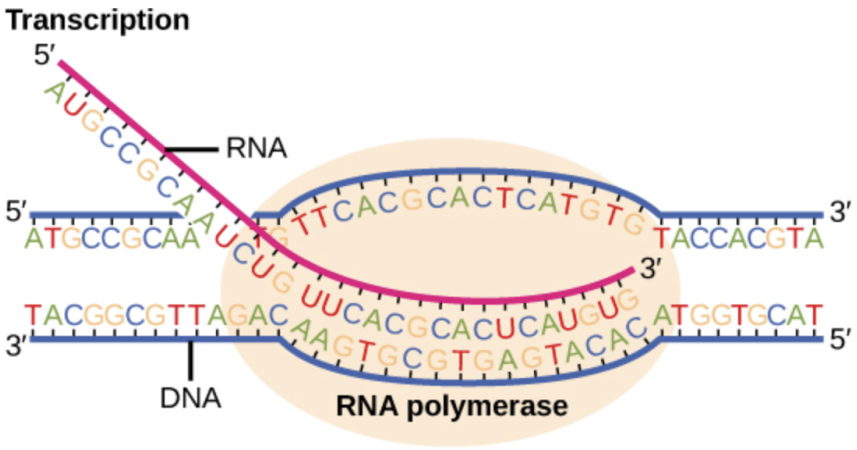
Figura 5 – Graus da expressão gênica



Fonte: Alberts (2017)

A transcrição é o primeiro passo no fluxo da informação genética, convertendo o DNA em RNA e preparando a informação para síntese de proteínas, que são essenciais para todas as funções celulares. Para que esse objetivo seja alcançado é necessário que a molécula de DNA seja submetida ao processo de transcrição e em seguida de tradução. A transcrição é necessária pois a síntese de proteínas é realizada no citoplasma, entretanto, devido ao seu tamanho, a molécula de DNA não consegue transitar livremente do núcleo para o citoplasma. Caso o organismo necessite de uma proteína específica, haverá no DNA um gene com a "receita" para a produção dessa proteína. Tudo se inicia com a liberação da enzima RNA-polimerase, ela é responsável por se conectar ao DNA e abri-lo na região portadora do material genético necessário para a síntese da proteína. Conforme a RNA-polimerase separa o DNA, é realizada a leitura da região e produção de uma cópia específica de RNA que contém a mensagem para produção de uma proteína, esse processo ocorre no sentido 5' 3', leia-se cinco linha, três linha. O RNA gerado é chamado de RNA mensageiro. A sua fita é simples e seu tamanho é menor que o do DNA. Desta forma ele consegue transitar através dos poros do núcleo, podendo chegar facilmente ao citoplasma, local onde ocorrerá a síntese de proteica. Uma vez que o RNA mensageiro é sintetizado, sua molécula se separa da cadeia de DNA , que reestabelece suas conexões de hidrogênio, e a dupla hélice é reconstituída, conforme ilustrado na Figura 6 (Alberts, 2017; CONNIE et al., 2016).

Figura 6 – Sintetização do RNA a partir do DNA



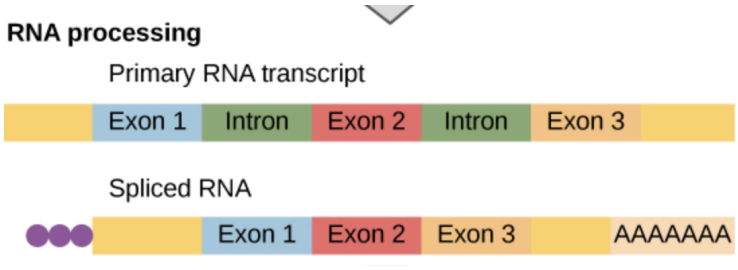
Fonte: CONNIE et al. (2016)

2.2.1 O Splicing

Antes que o RNA mensageiro deixe o núcleo celular, ele é submetido a um processamento chamado *splicing*. Esse processamento separa regiões do RNA mensageiro que são necessárias para a produção de proteínas, os *exons*, que vão para fora do núcleo, das regiões que ficam dentro do núcleo, os *íntrons*, conforme a Figura 7. Quando os *íntrons* se separam dos *exons*, dependendo da combinação gerada, um mesmo gene poderá sintetizar mRNA distintos a partir da mesma sequência de DNA.

Finalizado o *splicing*, os *exons* combinados compõem o RNA mensageiro final que, ao sair do núcleo, levará a informação da produção de proteínas ao RNA Ribossomal, ou simplesmente ribossomo.

Figura 7 – Remoção de *íntrons* do mRNA (*splicing*)



Fonte: CONNIE et al. (2016)

2.2.2 O Transcriptoma

O transcriptoma é o conjunto completo de todos os transcritos de RNA em uma célula, e suas quantidades, para um estágio de desenvolvimento específico ou condição fisiológica. A sua compreensão é fundamental para a interpretação dos elementos funci-

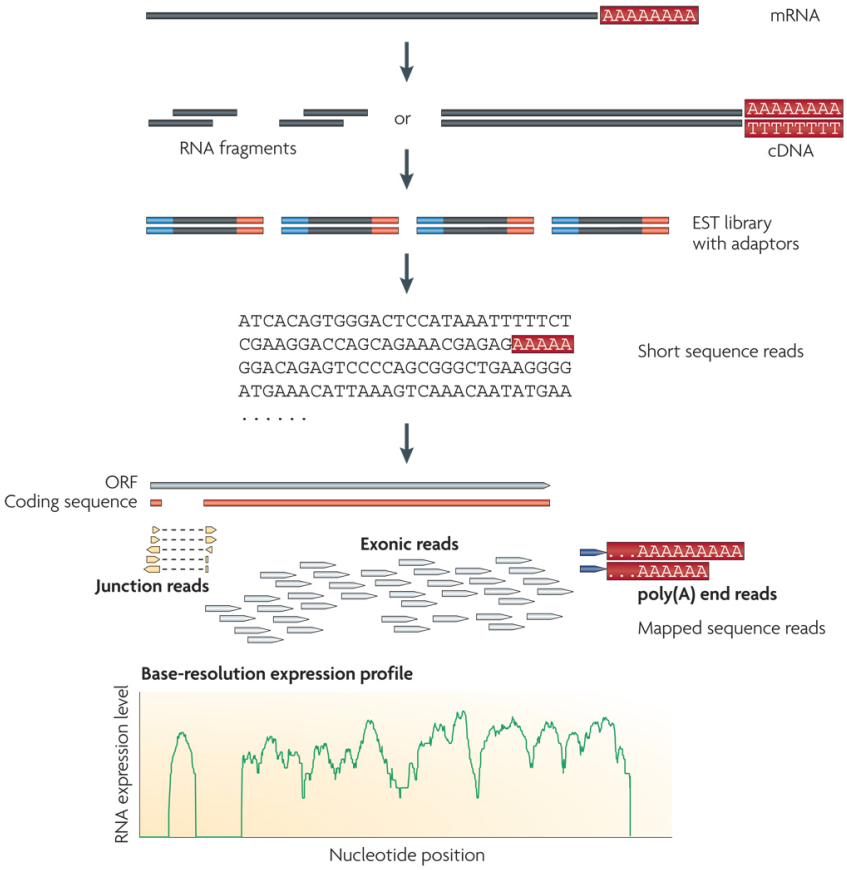
onais do genoma, expressão e regulação de genes, assim como para o entendimento do desenvolvimento de doenças. O seu estudo engloba tanto o RNA mensageiro, quanto os transportadores, os ribossômicos e até mesmo os não codificantes (Wang Zhong, 2009).

2.2.3 O RNA-Seq

RNA-Seq é uma abordagem desenvolvida para o sequenciamento de transcriptoma que utiliza tecnologias de sequenciamento profundo; ele oferece uma medição muito mais precisa dos níveis de transcritos do que outros métodos (Wang Zhong, 2009). Ele possui alto rendimento e fornece milhões de leituras de sequências curtas para mapeamento e quantificação de transcriptomas (Sangket et al., 2022). Quando comparado com outros métodos como *microarray*, ele se destaca por conseguir determinar diretamente a sequência do DNA complementar, o cDNA. Além disso, as abordagens por *microarrays* possuem um rendimento relativamente baixo, são caras e não são qualitativas, e só realizam o mapeamento quando há um genoma de referência (Wang Zhong, 2009).

O processo do sequenciamento envolve converter RNA em uma biblioteca de fragmentos de cDNA com adaptadores anexados, sequenciar esses fragmentos para obter leituras curtas e alinhar ou montar essas leituras, para produzir um mapa de transcrição do genoma. Diversos sistemas de sequenciamento, como Illumina, SOLiD e Roche 454, são utilizados para essa técnica. Após o sequenciamento, as leituras são alinhadas a um genoma ou transcrições de referência. Caso não haja a referência, elas são montadas *de novo* para que seja mapeada a sua estrutura transcricional e os níveis de expressão dos genes (Wang Zhong, 2009). Todo fluxo pode ser observado a partir da Figura 8.

Figura 8 – Fluxo do sequenciamento do RNA



Fonte: Wang Zhong (2009)

Assim, concluímos que o sequenciamento por meio da técnica de RNA-Seq resume-se a 3 etapas:

- processo de preparação da amostra: a partir da conversão de RNA em cDNA,
- sequenciamento de leituras: a fim de obter fragmentos curtos,
- alinhamento e montagem: que podem ser feitos a partir de um genoma de referência ou do método *de novo*.

O RNA-Seq é um método amplamente utilizado por sua capacidade de detectar transcritos sem sequências pré-existentes, não apresentando limites superiores, o que o torna altamente preciso na quantificação de altos níveis de expressão gênica, além de ter baixo custo.

2.2.4 A Expressão Diferencial

Nas seções subsequentes, discutiremos como se dá o fluxo de trabalho de análise de dados utilizando a técnica de RNA-Seq. Descreveremos em detalhes cada etapa de

um pipeline para análise de expressão diferencial de genes, abordando ferramentas mais comuns e considerações necessárias para o entendimento de como cada etapa pode ser aplicada neste trabalho.

2.2.5 GEO: São mais de duzentos e cinquenta milhões de sequências armazenadas

O GEO *(Gene Expression Omnibus)* é um repositório público funcional de dados de expressão genica que oferece suporte ao envio de dados em conformidade com o as informações mínimas sobre um experimento de *microarray* MIAME, do inglês *Minimum Information About a Microarray Experiment* e das Informações mínimas sobre um experimento de sequenciamento de próxima geração MINSEQE, do inglês *Minimum Information About a Next-generation Sequencing Experiment* (Information, 2024a). Ele arquiva e distribui dados de *microarrays* e sequenciamentos de alto rendimento que são continuamente enviados pela comunidade científica, além disso, ele fornece ferramentas para auxiliar os usuários a consultarem e baixarem experimentos e perfis de expressão gênica. No segundo semestre de 2024, ele possui mais de duzentos e cinquenta e um milhões de sequências e, devido à grande quantidade de dados disponibilizados, ele se torna fundamental para o desenvolvimento desta pesquisa, considerando que é necessária a obtenção de transcriptomas sequenciados para que seja possível realizar a análise de expressão diferencial de genes.

2.3 *Pipeline* de análise de expressão diferencial de genes

A busca por trabalhos correlatos descrita na sessão 2.5 identificou quais etapas são necessárias para a criação de um pipeline de análise de expressão diferencial de genes. Observou-se que este tipo de pipeline é composto por cinco etapas: corte, alinhamento, quantificação, normalização e a expressão diferencial.

2.3.1 O corte

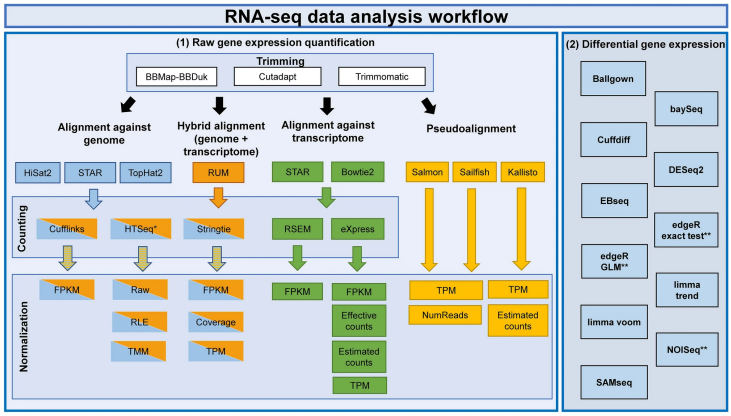
O corte é a etapa na qual as leituras de baixa qualidade e os adaptadores são removidos. Esta etapa é necessária, pois, durante a montagem da biblioteca de cDNA e o sequenciamento, podem ocorrer erros que tendem a diminuir a qualidade das leituras. As leituras podem ser obtidas em dois formatos principais: *single-end* e *paired-end*. No formato *single-end*, cada fragmento sequenciado é lido em apenas uma direção, enquanto no formato *paired-end*, cada fragmento é lido em ambas as direções, o que oferece maior precisão e qualidade para análises de alinhamento e quantificação.

Dentre os diversos tipos de corte, os experimentos que utilizam RNA-Seq possuem um número extremamente grande de leituras disponíveis. Portanto, recomenda-se o uso de

um corte modesto, ou seja, leituras que têm qualidades inferiores a 20 devem ser removidas. Isso porque o corte agressivo pode afetar negativamente a precisão das estimativas de expressão gênica. Além disso, nos experimentos *paired-end*, é fundamental que ambas as leituras de um par sejam tratadas de maneira consistente, para evitar perda de informações importantes durante o corte.

O corte é uma etapa opcional e, quando aplicado, medidas como a filtragem de comprimento devem ser consideradas no *pipeline* de pré-processamento para minimizar viés indesejado (Williams et al., 2016). Nesta etapa, são comumente utilizados programas como *BBMap-BBDuk*[1], *Cutadapt*[2] e o *Trimmomatic*[3], conforme demonstrado na Figura 9.

Figura 9 – Fluxo da análise de dados de RNA-Seq



Fonte: Corchete et al. (2020)

2.3.2 O alinhamento

O alinhamento é a segunda etapa e ocorre logo após o corte das leituras. Ele é fundamental para que a estimativa da expressão gênica seja precisa, pois afeta profundamente a análise subsequente. Os alinhadores de transcriptoma simplificam o processo de alinhamento, pois correspondem às leituras de sequência às transcrições conhecidas. Por outro lado, os alinhadores de genoma alinham diretamente as leituras ao genoma e precisam lidar com leituras derivadas de *splicing* alternativo. Ferramentas como *STAR*[4] ou *PASSION*[5] podem ser utilizadas nesta etapa, pois, segundo um estudo comparativo, apresentaram resultados superiores às demais (Yang et al., 2015).

[1] https://jgi.doe.gov/data-and-tools/software-tools/bbtools/bb-tools-user-guide/bbduk-guide/
[2] https://cutadapt.readthedocs.io/en/stable/
[3] http://www.usadellab.org/cms/?page=trimmomatic
[4] https://github.com/alexdobin/STAR
[5] https://academic.oup.com/bioinformatics/article/28/4/479/213492

2.3.3 A quantificação

Uma vez que as leituras foram alinhadas, elas devem ser submetidas ao processo de contagem, utilizando como referência arquivos no formato de transferência de genes, conhecido pelo acrônimo em inglês GTF (*Gene Transfer Format*), ou no formato geral de características, conhecido como GFF (*General Feature Format*). Este processo é conhecido como contagem ou quantificação (Corchete et al., 2020). Com base nesses arquivos de anotação, é possível estimar a abundância de transcritos, calculando a quantidade de leituras atribuídas a cada gene. Um dos principais desafios na quantificação de transcrições a partir de dados de RNA-Seq é lidar com leituras que podem ser alinhadas a múltiplos genes. Essa questão torna-se particularmente relevante em quantificações baseadas em conjuntos de transcriptomas de novo, ou seja, em situações em que os genomas ainda não foram sequenciados previamente.

Ferramentas como HTSeq[6], StringTie[7], Cufflinks[8] e eXpress[9] encapsulam essas etapas e fornecem a expressão gênica já quantificada.

2.3.4 A normalização

A última etapa da quantificação da expressão gênica bruta é a normalização, ela é uma etapa crítica na análise de dados de RNA-Seq e tem um forte impacto na detecção de genes diferencialmente expressos (Risso et al., 2014). Seu objetivo é remover vieses que surgem da variabilidade de condições experimentais como coleta e preparação de amostras e, até mesmo, parametrização de máquinas (Yin et al., 2020; Yin et al., 2021). Nos últimos anos, várias estratégias de normalização foram propostas para corrigir diferenças entre amostras em contagens de leituras (Risso et al., 2014). Métodos como TPM (transcrição por milhão), RPKM, leituras por quilobase de transcrição por milhão de leituras mapeadas e FPKM, fragmentos por milhão de quilobases, vêm sendo empregados e são adequados para a expressão de transcrições dentro de uma única amostra. Entretanto, não são ideais para comparações entre diferentes amostras ou para normalizar a expressão diferencial, especialmente para genes pouco expressos, pois tendem a ter um desempenho ruim quando as distribuições de transcrições diferem entre as amostras. Métodos de normalização como TMM *(Trimmed Mean of M-values)* são mais robustos para lidar com diferentes tamanhos e composições de bibliotecas, pois eles normalizam o fator mais importante para comparação, que é a profundidade do sequenciamento, seja diretamente ou contabilizando o número de transcrições, os quais podem diferir significativamente (Zhao et al., 2021).

Logo, no RNA-Seq, fatores de normalização garantem que genes com o mesmo

[6] https://htseq.readthedocs.io/en/latest/
[7] https://github.com/gpertea/stringtie
[8] https://github.com/cole-trapnell-lab/cufflinks
[9] https://bioinformaticshome.com/tools/rna-seq/descriptions/eXpress.htmlgsc.tab=0

nível de expressão em duas amostras não sejam erroneamente detectados como DEGs. Um exemplo ilustrativo mostra que, se uma amostra A tem o dobro de genes expressos em relação à amostra B devido a genes adicionais não expressos em B, sem normalização, genes comuns teriam metade das leituras em A. Isso destaca a necessidade crucial da normalização para ajustar essas diferenças na produção total de RNA entre amostras sequenciadas na mesma profundidade (Robinson; Oshlack, 2010). Ferramentas como *edgeR*[10] e DeSeq2[11] podem ser utilizadas nesta etapa, pois ambas realizam a normalização a partir da profundidade do sequenciamento, desta forma ajustando as diferenças no tamanho das bibliotecas para normalizar amostras dentro de um conjunto de dados e utilizando uma amostra de referência para calcular mudanças na expressão de genes em relação a essa amostra.

2.3.5 A expressão diferencial de genes

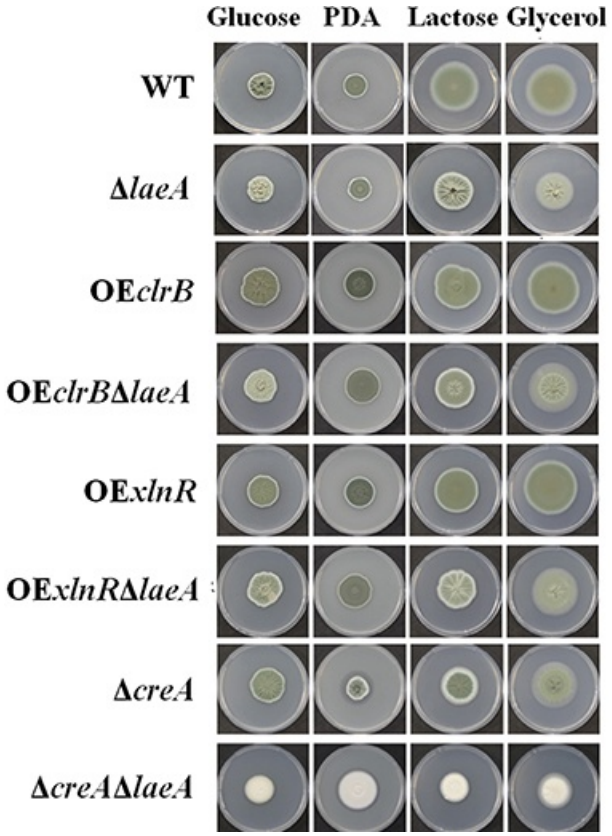
A análise diferencial de expressão gênica é utilizada em diversas áreas de pesquisa, nas ciências da saúde e da vida. Essas análises têm sido empregadas para a identificação de biomarcadores de câncer relacionados às manifestações clínicas da doença e para a identificação de genes responsáveis pela reprodução de animais economicamente importantes (Parvathareddy et al., 2021; Thepsuwan et al., 2021; Wang Zhong, 2009).

Sua aplicação envolve testes estatísticos e avaliações de diferenças significativas na expressão gênica entre duas ou mais condições experimentais (Sangket et al., 2022). Com ela, dado uma condição considerada "normal" para um organismo, é possível verificar quais genes estão *upregulated* ou *downregulated*.

Essa condição"normal" também é conhecida como condição de controle, e as condições experimentais são aquelas que podem ter sido alteradas por fatores externos, como condições de temperatura ou meio de cultivo da amostra, conforme demonstrado pela Figura 10. Portanto, a expressão diferencial é obtida a partir do resultado da comparação entre as duas condições. Para garantir a confiabilidade dos resultados, a análise diferencial de expressão gênica é frequentemente realizada com o uso de *triplicatas*, ou seja, três amostras biológicas independentes para cada condição experimental. O uso de triplicatas permite reduzir a variabilidade técnica e biológica, aumentando a precisão da análise e a confiabilidade das conclusões. Ao comparar as médias de expressão gênica entre as condições de controle e experimental, o uso de triplicatas também ajuda a minimizar o risco de erros decorrentes de flutuações aleatórias ou variações técnicas.

[10] https://bioconductor.org/packages/release/bioc/html/edgeR.html
[11] https://bioconductor.org/packages/release/bioc/html/DESeq2.html

Figura 10 – Cultivo do fungo selvagem (WT) de *Penicillium oxalicum* 114-2 e seus 7 mutantes em diferentes meios.



Fonte: Li et al. (2016)

Após a finalização da quantificação da expressão gênica bruta, o resultado gerado é utilizado para verificar as variações nos níveis de expressão de genes entre as amostras. A análise de expressão diferencial ocorre nesta etapa, com base em comparações estatísticas utilizando as contagens de expressão gênica entre as condições experimental e controle para realizar a classificação dos genes.A medida mais comum utilizada para quantificar a mudança na expressão de um gene entre duas condições é o *Log Fold Change* (LFC). Ela compara a condição experimental com a condição de controle. Se o gene em análise apresenta uma expressão maior em relação à condição de controle, significa que ele possui o LFC positivo; caso contrário, este valor é negativo. Considere o cenário hipotético onde a condição controle do gene A possui 100 unidades de expressão e a condição experimental possui 400 unidades de expressão. O cálculo do *Fold Change* (FC) seria a razão entre a condição experimental e a condição controle, conforme exemplificado por (Maza et al., 2013) na Equação 2.1.

$$FC = \frac{\text{expressão da condição experimento}}{\text{expressão da condição controle}} \tag{2.1}$$

Ou seja, 400/100, que resulta em 4. Portanto, o gene da condição experimental foi

expresso 4 vezes mais que na condição controle, conforme a Equação 2.2

$$FC(\frac{400}{100}) = 4 \tag{2.2}$$

O log2FC significa que este mesmo cálculo será realizado utilizando o logaritmo na base 2 do *fold change* obtido, desta forma, o log2FC deste gene seria 2, pois o logaritmo de 4 na base 2 é igual a 2, conforme expresso na Equação 2.3. Porém, o aumento da expressão gênica permanece 4 vezes maior, pois $2^2 = 4$.

$$log_2(4) = 2 \tag{2.3}$$

O mesmo ocorre quando o gene apresenta uma expressão diminuída. Considere que, neste caso, a condição de experimento possui 25 unidades de expressão, enquanto a condição de controle possui 100. O cálculo a ser realizado é 25/100 = 0,25, resultando em um log2FC igual a -2, porque $2^{-2} = 0,25$, conforme a Fórmula 2.4.

$$FC(\frac{25}{100}) = 0,25 \tag{2.4}$$

Contudo, apenas o log2FC não é suficiente para determinar a classificação do gene, os dados necessitam ser submetidos a um teste estatístico, utilizando o *p-value* para avaliar se a regulação do gene possui uma margem de erro dentro de um limite crítico, que costuma ser abaixo de 0,05. Para determinar a classificação definitiva de cada gene é necessário realizar primeiramente a filtragem apenas dos genes que possuem o *p-value* menor que 0,05 e então classificá-los a partir do log2FC com o limiar desejado, log2FC>=2 para genes *upregulated* e log2FC<=-2 para genes *downregulated*.

2.4 *Pipelines* e Aplicações web de bioinformática

Pipeline é um modelo de arquitetura no qual ocorre a divisão de um processamento sequencial em etapas. Neste modelo, cada fase normalmente é executada por um ou mais programas, onde os dados gerados em uma etapa servem de entrada para a próxima etapa e possuem caráter linear (Melo, 2009).

Diversas áreas da ciência utilizam este formato, entretanto, os *pipelines* de bioinformática geralmente processam dados biológicos provenientes de sequenciamento de DNA ou RNA (Oliveira, 2019). O processamento de sequenciamentos genéticos requer alto poder computacional, domínio de comandos em linha de terminal e enfrenta desafios significativos relacionados à instalação, configuração e implantação. Segundo Lataretu e Hölzer (2020), os *pipelines* de bioinformática vêm se tornando cada vez mais importantes, pois apresentam uma solução bastante portátil e viável, encapsulando todas as etapas necessárias para

uma determinada análise e podem ser executados nos mais diversos ambientes, tanto web quanto local.

2.4.1 Execução assíncrona

Os *pipelines* de bioinformática podem ser executados inúmeras vezes por diversos pesquisadores, utilizando várias ferramentas e parâmetros diferentes, gerando uma variedade de dados como resultado (Oliveira, 2019). Todavia, devido ao grande tamanho dos genomas, as aplicações de bioinformática podem levar algumas horas para realizar a execução dos *pipelines*. KingFungi[12], por exemplo, em sua primeira versão, levava cerca de 14h para executar seu fluxo completo, Souza (2024). Plasticome[13], leva cerca de 3 horas Ferreira (2024). Nesse sentido, o uso de ferramentas de processamento assíncrono podem ser de grande valia na construção da arquitetura de um *pipeline*, pois elas executam em um segundo plano e podem informar ao usuário quando as tarefas agendadas forem finalizadas.

2.4.2 Celery

O Celery é um sistema de filas de tarefas distribuído, ele possui natureza assíncrona, é flexível e confiável para processar grandes quantidades de mensagens, ao mesmo tempo em que fornece às operações as ferramentas necessárias para manter esse sistema. A sua fila de tarefas realiza tanto processamento em tempo real, como oferece suporte ao agendamento de tarefas (Celery, 2024).

Devido à estrutura de agendamento, tarefas que são extremamente demoradas podem ser agendadas para serem executadas em momento posterior. Isso é vantajoso pois o usuário final recebe um retorno imediato informando que o sistema recebeu sua solicitação. Em paralelo a isso, o Celery se encarrega de enviar a mensagem para um módulo intermediário chamado *broker*. Ele, por sua vez, é responsável por armazenar as mensagens em filas para que, quando acionados, os consumidores do Celery (*workers*) realizem as solicitações para o *broker* e iniciem o processamento das *tasks* agendadas (Morais, 2022). No fim deste processamento, o usuário solicitante da requisição pode ser notificado por meio do sistema de envio de mensagem cadastrado, que pode ser um e-mail ou um *push* de aplicativo.

2.4.3 RabbitMQ

O RabbitMQ é um servidor de mensageria e streaming confiável, maduro, fácil de implementar e de código aberto, sendo utilizado por milhões de pessoas ao redor do mundo.

[12] Aplicação desenvolvida para analisar cogumelos com potencial de absorção de micronutrientes
[13] Aplicação desenvolvida para analisar fungos com degradação de plástico

Utilizando os protocolos AMQP 1.0[14] e MQTT 5.0[15], ele é capaz de realizar comunicação assíncrona entre serviços e pode ser utilizado com a linguagem de programação de escolha do usuário. Seus principais componentes são o produtor, a fila, a *exchange* e o consumidor. O produtor é responsável por enviar mensagens; as *exchanges* atuam como roteadores, encaminhando cada mensagem para suas respectivas filas. O consumidor, por sua vez, consome as mensagens, retirando-as da fila. O RabbitMQ é um sistema bastante flexível, pois oferece diversas configurações para definir quais consumidores devem receber as mensagens. Além disso, ele realiza confirmação de entrega e replicação de mensagens (RabbitMQ, 2024).

2.4.4 Python

Python é uma linguagem de programação amplamente utilizada em aplicações web, desenvolvimento de software, ciência de dados e aprendizado de máquina. Os desenvolvedores a preferem por sua eficiência, facilidade de aprendizado e compatibilidade com diversas plataformas. Além disso, é uma linguagem gratuita, que se integra bem a diferentes sistemas e agiliza o desenvolvimento (AWS, 2024a). Devido a essas características, Python tem sido amplamente empregado no desenvolvimento de aplicações diversas, incluindo aquelas voltadas à bioinformática.

Na área da bioinformática, uma das bibliotecas mais utilizadas é o Biopython, mantida por uma associação internacional de desenvolvedores e disponibilizada gratuitamente. Essa biblioteca oferece um conjunto abrangente de pacotes, módulos e classes para a análise e anotação de sequências biológicas. Além disso, fornece métodos para acessar bancos de dados biológicos on-line, como, por exemplo, os bancos mantidos pelo NCBI (*National Center for Biotechnology Information*). O Biopython também inclui diversos módulos que permitem: análises de alinhamentos de sequências, estruturas de proteínas, genética de populações, filogenia, visualização de dados biológicos, identificação de motivos em sequências e, até mesmo, o uso de aprendizado de máquina (Mariano et al., 2016).

2.4.5 Reprodutibilidade, Interoperabilidade e Manutenibilidade

A reprodutibilidade é uma maneira de permitir que o conhecimento fique disponível para o público em geral, e é uma questão fundamental na produção do conhecimento científico. Uma contribuição científica é considerada valiosa se, entre outras coisas, outros pesquisadores são capazes de reproduzir seus resultados com sucesso (Oliveira, 2019). Entretanto, existem diversos tipos de processadores e sistemas operacionais disponíveis no mercado, e é inviável o desenvolvimento de um sistema particular para cada arquitetura.

[14] https://www.rabbitmq.com/tutorials/amqp-concepts
[15] https://aws.amazon.com/pt/what-is/mqtt/

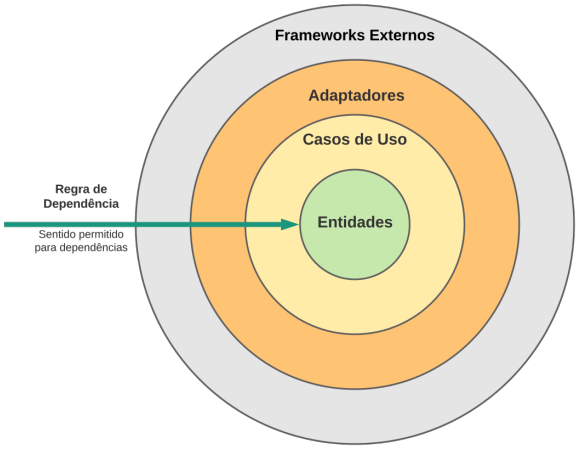
Para contornar este problema, é comumente utilizada uma tecnologia conhecida como Docker. O Docker é uma ferramenta que permite separar um aplicativo de sua infraestrutura, para que seja possível entregar software rapidamente, pois ele funciona como uma camada intermediária entre o sistema a ser implantado e a plataforma do hardware. Aproveitando as vantagens do Docker, é possível enviar, testar e implantar código rapidamente, reduzindo significativamente o atraso entre escrever o código e executá-lo em produção, além de facilitar a manutenção (Docker, 2024).

Sua estrutura é baseada em imagens e contêineres. As imagens são modelos somente leitura que contêm instruções, modelos das bibliotecas e dependências necessárias para que seja possível a criação de um contêiner. Por sua vez, os contêineres são a imagem em execução, suas estruturas permitem que os desenvolvedores empacotem software para execução em qualquer sistema de destino. Eles são altamente portáteis e podem ser executados em máquinas com configurações básicas. Por exemplo, uma aplicação corporativa pode ter centenas de microsserviços. Estes poderiam ser executados como contêineres em várias máquinas físicas e máquinas virtuais em um *datacenter* dedicado e na nuvem (AWS, 2024b).

2.4.6 A Arquitetura Limpa

Visando garantir que a aplicação seja facilmente testável e independente de ferramentas, de banco de dados ou de qualquer interferência externa, o desenvolvimento deste trabalho foi implementado utilizando a Arquitetura Limpa. Ela foi desenvolvida por Robert Cecil Martin, ou simplesmente *Uncle* Bob como é conhecido. Tio Bob atua como programador desde 1970, e já publicou dúzias de artigos em vários periódicos do setor de tecnologia e regularmente palestra em conferências internacionais e feiras (Martin, 2019). Nesta arquitetura, ele defende que nada que é externo à aplicação deve ter conhecimento do que está dentro, ou seja, qualquer coisa que não seja do domínio da aplicação não deve ser mencionado pelo código na sua parte interna. As dependências de código-fonte só podem apontar para dentro, isto inclui funções, classes, variáveis ou qualquer outra entidade de software. Esta regra é conhecida como Regra da Dependência, conforme demonstrado na Figura 11.

Figura 11 – Arquitetura limpa



Fonte: Adaptado de Martin (2019)

A arquitetura limpa é composta por quatro grandes círculos: as entidades, os casos de uso, os adaptadores e os *frameworkers* externos. As entidades são responsáveis por encapsular as regras de negócio, e são a parte menos propensa a mudar quando algo externo muda, logo, nenhuma mudança operacional deve afetar a camada de entidade. Os casos de uso contêm as regras específicas da aplicação, eles orquestram como ocorre o fluxo de dados para dentro das entidades para atingir os objetivos. As mudanças externas como banco de dados ou interface não devem afetar esta camada, no entanto, esperamos que mudanças na operação do aplicativo afetem os casos de uso. Se os detalhes de um caso de uso mudarem, algum código nesta camada certamente será afetado.

Os adaptadores de interface convertem dados do formato mais conveniente para os casos de uso e entidades para o formato mais conveniente para alguma agência externa, como *views*. Eles atuam como mediadores entre a camada mais externa da arquitetura (sistemas externos) e as camadas centrais (casos de uso e entidades). Os *frameworkers* externos pertencem à camada mais externa e geralmente são compostos por ferramentas como banco de dados ou bibliotecas. Esta camada é onde todos os detalhes estarão, o banco de dados é um detalhe, ferramentas utilizadas para corte ou obtenção de dados também são detalhes. Devemos mantê-las do lado de fora, onde podem causar pouco dano.

Seguir esta arquitetura garante que quando o banco de dados ou ferramentas como HTSeq se tornarem obsoletas ou precisarem ser substituídas será possível realizar a troca destes elementos com o mínimo de esforço e confusão.

2.5 Trabalhos correlatos

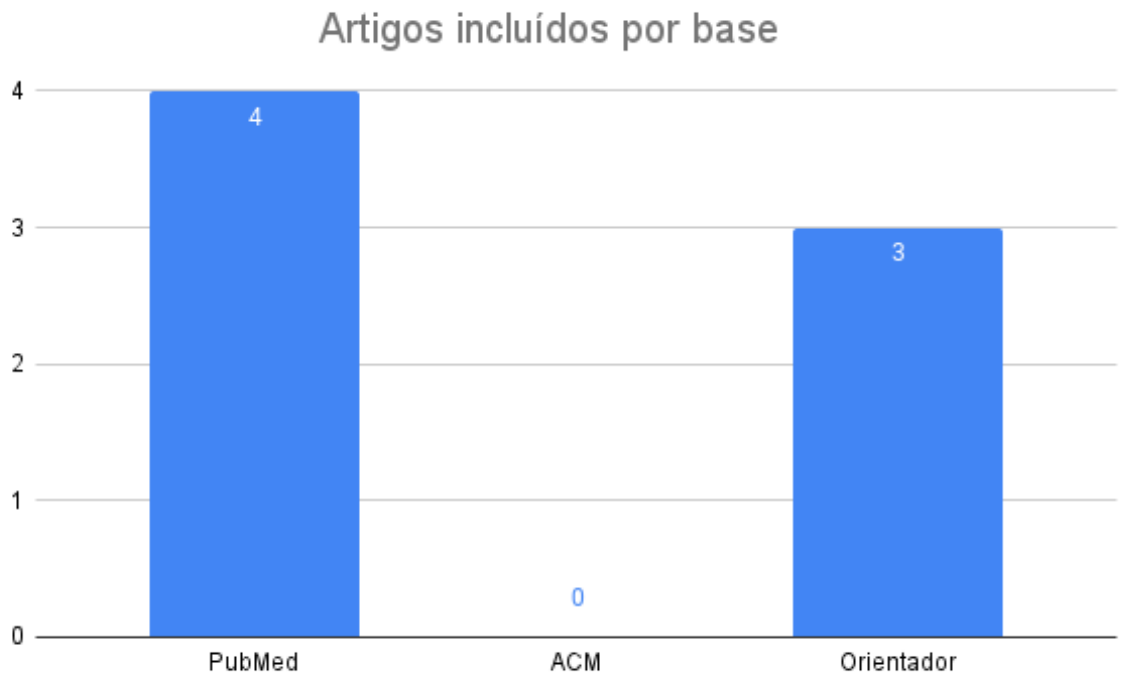
Através de revisão de literatura foram encontrados 7 trabalhos correlatos que abordam *pipelines* para análise de expressão diferencial de genes. A partir disto foi realizado o levantamento das ferramentas mais precisas utilizadas em cada etapa do *pipeline*. Para nortear o início das buscas pelos trabalhos correlatos, foram utilizadas quatro palavras-chave:

- RNA-Seq
- DEGs
- *"gene-expression"*
- *pipeline*

Uma vez definidas as palavras-chave, foi elaborada a cadeia de busca, que consiste em unir palavras-chave, em uma única expressão: RNA-Seq AND DEGs AND "*gene expression*"AND *pipeline*

Sob o total de artigos encontrados nas bases da PubMed e ACM (*Association for Computing Machinery*) foram aplicados os critérios de inclusão e exclusão, Tabela 1. Após isso, restaram 7 artigos, dos quais 4 foram da PubMed e 3 enviados pelo orientador, conforme a Figura 12.

Figura 12 – Relação de artigos incluídos por base de dados



Fonte: O autor

Tabela 1 – Tabela de critérios de inclusão e exclusão

Identificador	**Critério**
I1	Artigo aborda pipelines para análise quantitativa de expressão de genes utilizando dados de RNA-Seq
I2	Artigo enviado pelo orientador
I3	Artigo aborda ferramentas computacionais para análise de expressão diferencial
E1	Artigo não aborda pipelines para análise quantitativa de expressão de genes
E2	O artigo não usa a técnica de RNA-Seq para sequenciamento de RNA

Identificador	**Critério**
E3	O artigo não agrega com softwares / soluções tecnológicas

Fonte: O autor

Cada trabalho e sua análise individual sobre o que cada um deles aborda pode ser visualizada na Tabela 2.

Tabela 2 – Tabela de artigos e análises

Artigo	**Análise**
Systematic comparison and assessment of RNA-seq procedures for gene expression quantitative analysis (Corchete et al., 2020)	O artigo faz uma comparação entre 192 pipelines, criando um ranking do top 10 das melhores combinações mais precisas e utiliza rtPCR como validador dos resultados.
Best practices on the differential expression analysis of multi-species RNA-seq. (CHUNG et al., 2021)	O artigo discute a importância da abordagem transcriptômica multi espécies, listando as ferramentas utilizadas. Não utilizou a etapa de normalização.
Identifying suitable tools for variant detection and differential gene expression using RNA-seq data, Genomics (Dharshini; Taguchi; Gromiha, 2020)	O artigo avalia várias ferramentas para análise de expressão diferencial versus a análise de variantes. Foi possível identificar as ferramentas utilizadas em cada pipeline.

Artigo	**Análise**
bestDEG: a web-based application automatically combines various tools to precisely predict differentially expressed genes (DEGs) from RNA-Seq data (Sangket et al., 2022).	O foco principal é a análise comparativa entre as ferramentas de análise de expressão diferencial. A comparação é feita entre DESeq2, edgeR, NOISeq e EBSeq, com DESeq2 apresentando melhor resultado.
A comparison of transcriptome analysis methods with reference genome. BMC Genomics (Liu et al., 2022).	O foco principal é a comparação entre 5 ferramentas de quantificação e 6 para análise de expressão diferencial. HTSeq apresentou melhor resultado para quantificação e limma para expressão diferencial, ratificando o resultado obtido por (Corchete et al., 2020)
An Effective and Simple RNA-Seq Differential Gene Expression Pipeline Using Nextflow. Genes (Basel) (Lataretu; Hölzer, 2020).	O artigo explica detalhadamente o processo para construção do pipeline, demonstrando inclusive a estrutura dos diretórios e link para o projeto no github.
Transcriptome software results show significant variation among different commercial pipelines (Thawng; Smith, 2023).	Realiza análise comparativa de expressão diferencial, concluindo que o pipeline formado por DNAstar-Deseq2 produz resultados mais adequados para estudos de expressão gênica mediana.

Fonte: O autor

A realização da leitura destes trabalhos correlatos mostrou que embora hajam diversas ferramentas para realização da análise DEGs, ainda não existe um consenso sobre as ferramentas, e que o resultado da análise ainda é vinculado a escolha das ferramentas. Do ponto de vista de avaliação quantitativa de cada etapa, no pré-processamento, também

conhecida como corte, a ferramenta mais utilizada foi a *Trimmomatic*. No alinhamento as mais utilizadas foram *STAR* e *HISAT2*, e na quantificação as que mais apareceram foram *HTSeq* e *featureCounts*. Já para a etapa de normalização a mais utilizada foi a *TMM*, e para etapa de análise de DEGs foi a *DESeq2*.

Entretanto, o estudo de Corchete et al. realizou uma comparação com mais de 192 pipelines montados com a combinação das mais populares ferramentas utilizadas para análise de DEGs, e o *pipeline* mais preciso e exato foi a combinação entre Trimmomatic + RUM + HTSeq + TMM. Mas embora esse tenha sido a melhor combinação, as ferramentas que fizeram a maior diferença foi HTSeq para contagem e TMM para normalização, pois faziam parte de 10 dos 10 principais pipelines. Por outro lado, os *pipelines* consistentes em alinhamentos brutos, a cobertura de Stringtie e os NumReads de Salmon ocuparam a maioria das últimas posições no ranking, confirmando a importância de uma combinação adequada de métodos de contagem e normalização.

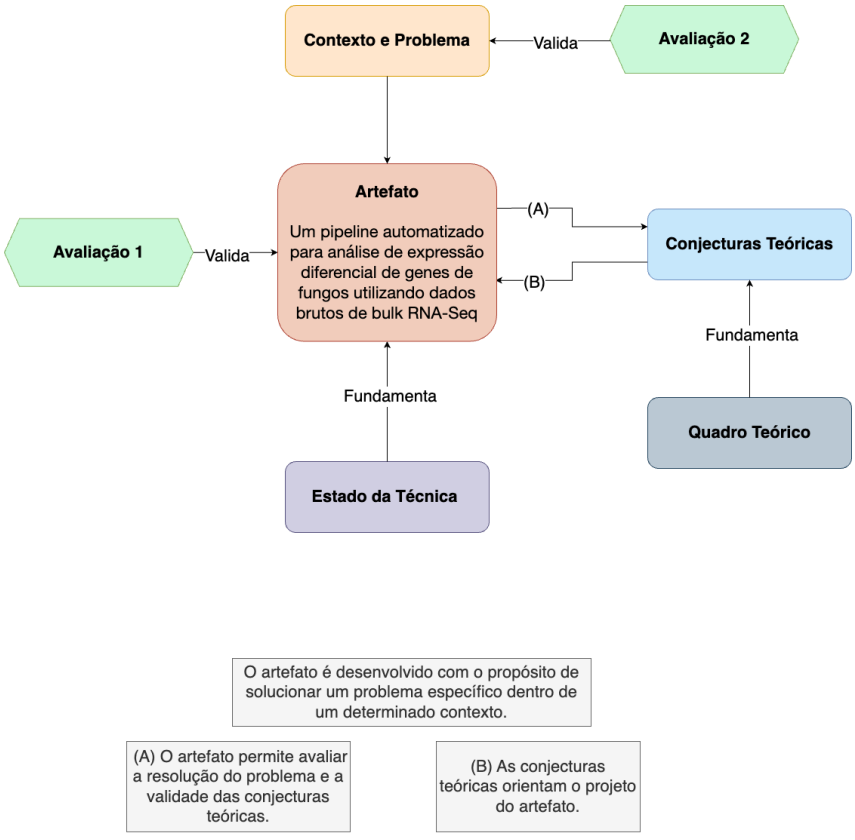
Desta forma, a construção do *pipeline* seguirá a combinação entre Trimmomatic + RNA Star + HTSeq + DeSeq2.

3 METODOLOGIA

3.1 Metodologia de Pesquisa

Neste estudo, foi adotada a metodologia de pesquisa *Design Science Research* (DSR). O DSR é uma abordagem amplamente reconhecida que visa criar, desenvolver e avaliar soluções práticas para problemas complexos e relevantes no campo da ciência da computação e outras disciplinas relacionadas (Brocke; Hevner; Maedche, 2020). O DSR segue um processo estruturado, cujo objetivo é orientar o pesquisador no desenvolvimento e avaliação dos artefatos que resolverão problemas práticos. Ele compreende as etapas de identificação do problema, definição dos resultados esperados, projeto e desenvolvimento, demonstração, avaliação e comunicação, conforme a Figura 13. É possível realizar uma melhoria contínua ao longo de todo o processo de pesquisa, pois as etapas são iterativas. Neste projeto, o objetivo principal é desenvolver um artefato para facilitar e automatizar a análise de expressão diferencial de genes a partir de dados brutos de RNA-Seq.

Figura 13 – Estrutura Geral do DSR aplicada ao objeto de pesquisa



Fonte: O autor

Neste projeto, o DSR foi aplicado da seguinte forma:

Contexto e problema: a montagem de um *pipeline* para análise de expressão diferencial apresenta desafios que vão além das questões técnicas relacionadas à programação e instalação de *softwares*. Embora o conhecimento avançado em programação e habilidades *shell* de comando sejam frequentemente necessários, a análise de RNA-Seq exige uma abordagem multidisciplinar. Essa tarefa engloba etapas que requerem profundo entendimento em biologia molecular e genômica, para interpretar as sequências de RNA e compreender os processos biológicos subjacentes; estatística e bioinformática, essenciais para o desenvolvimento e aplicação de métodos analíticos robustos; e, por fim, a interpretação biológica dos resultados, indispensável para correlacionar os dados obtidos com fenômenos biológicos relevantes. Adicionalmente, a diversidade de ferramentas e algoritmos disponíveis para análise de RNA-Seq pode tornar a escolha das melhores soluções uma tarefa desafiadora, exigindo discernimento técnico e biológico.

Para cada objetivo listado na introdução, é esperado que haja um resultado associado a ele:

1. etapas do *pipeline* estejam descritas;
2. ferramentas mais adequadas identificadas;
3. código do *pipeline* disponibilizado no Github [2];
4. resultados da validação estejam compatíveis com os encontrados em artigos.

O estado da técnica refere-se ao conhecimento atual sobre as ferramentas e tecnologias relevantes para este projeto. As principais áreas abordadas incluem a documentação das ferramentas utilizadas em cada etapa do *pipeline*, bem como a construção de aplicações web e *pipelines* de bioinformática. A conjectura teórica para este projeto está fundamentada na hipótese de que a automação do processo de análise de RNA-Seq pode ser alcançada de forma eficiente utilizando-se um *pipeline* estruturado em Docker, com a integração de ferramentas de bioinformática amplamente utilizadas. A falta de conhecimento em programação dos profissionais de biologia, de pode dificultar o uso de métodos de análise de DEGs (genes diferencialmente expressos), tornando-os inacessíveis para muitos usuários. Isso exige um significativo investimento de tempo para aprender as habilidades necessárias gerando problemas de acessibilidade e barreiras de conhecimento.

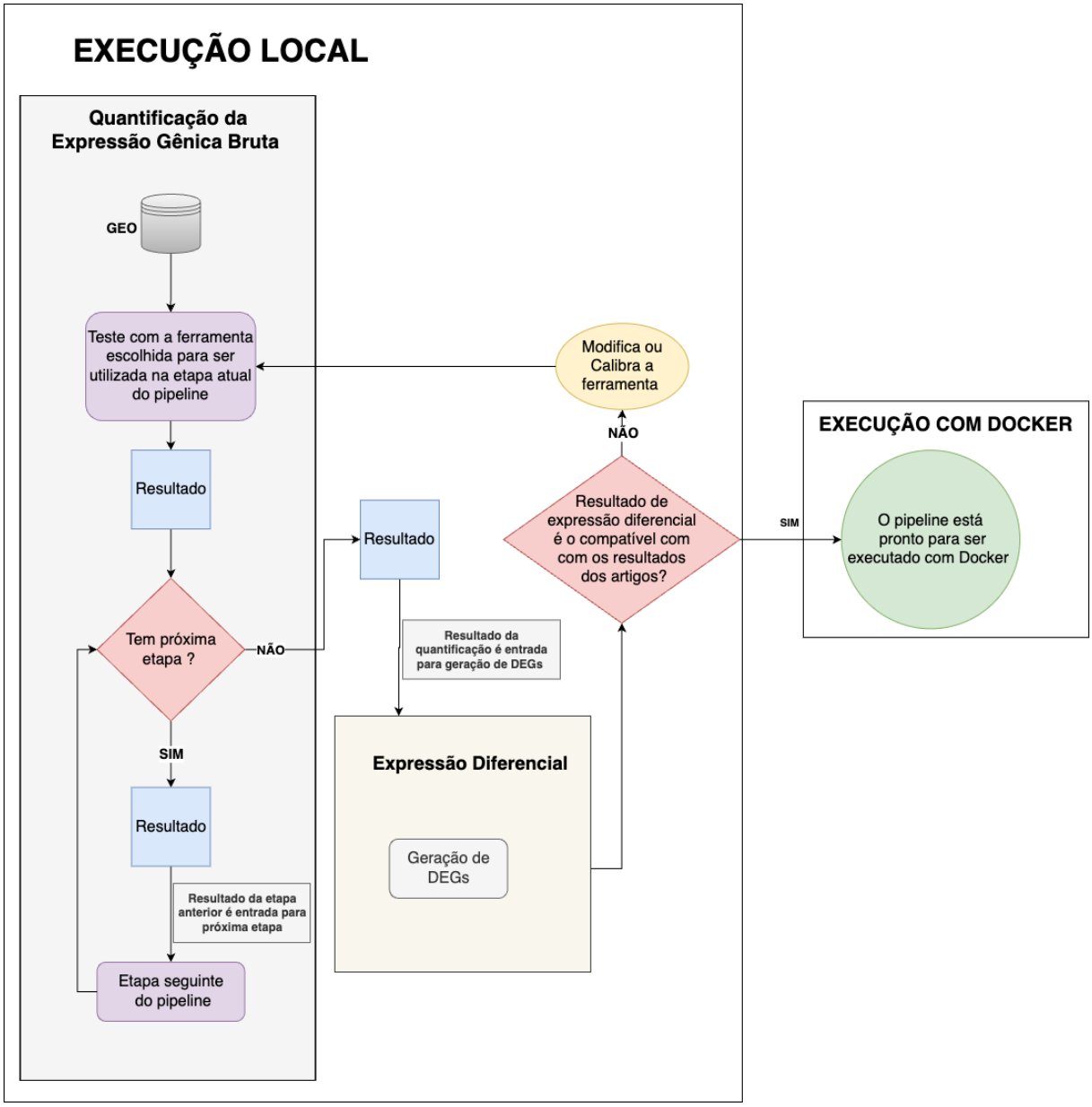
A avaliação será realizada em duas etapas. Na primeira avaliação, será utilizado o ambiente Galaxy, conforme ilustrado na Figura 14. Nesta etapa, serão realizados testes das ferramentas utilizando dados experimentais, com o objetivo de validar se os resultados obtidos são os esperados antes de proceder à implementação utilizando o Docker. A segunda avaliação consiste em executar o *pipeline* automatizado no ecossistema Docker, conforme

[2] GitHub é uma plataforma de hospedagem de código-fonte e arquivos com controle de versão usando o Git

ilustrado na Figura 15. Foi utilizada uma massa de testes com 5 conjuntos de dados coletados do GEO, que foram comparados com artigos publicados a fim de validar o funcionamento do *pipeline*. Para ambas as avaliações, caso os resultados não sejam compatíveis com os encontrados em artigos publicados, a ferramenta utilizada deverá ser alterada ou calibrada novamente. Embora os resultados do pipeline não sejam necessariamente idênticos aos reportados no artigo de referência, eles apresentam consistência suficiente para demonstrar a robustez e a confiabilidade do método utilizado, pois, conforme discutido no trabalho de (Corchete et al., 2020) diferentes abordagens de análise podem levar a variações nos resultados. Além disso, caso a aplicação identifique artigos adicionais ou diferentes em relação aos resultados esperados, isso não representa necessariamente um problema, desde que os dados identificados sejam coerentes e biologicamente relevantes. Esse comportamento pode até ser benéfico, pois indica que a ferramenta possui sensibilidade suficiente para detectar resultados que outras abordagens podem ter deixado de identificar.

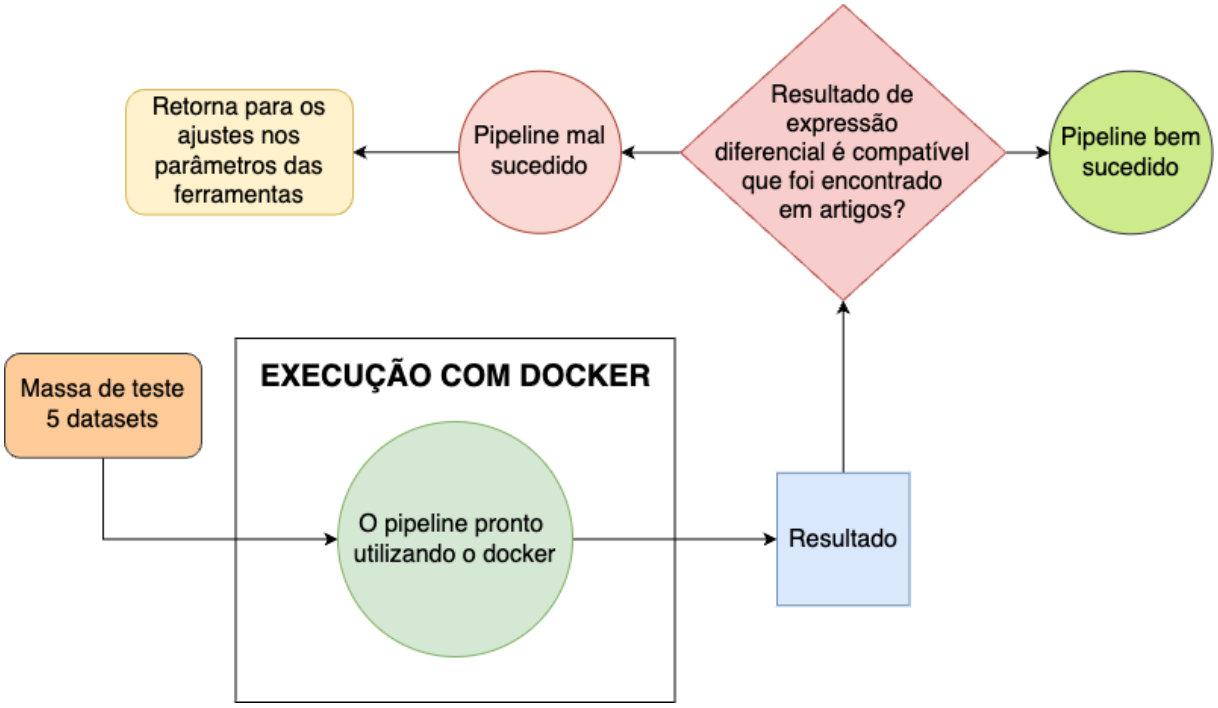
A comunicação dos resultados será realizada por meio da publicação da monografia e da apresentação do projeto para a banca avaliadora.

Figura 14 – Primeira etapa de avaliação do pipeline em ambiente Galaxy



Fonte: O autor

Figura 15 – Segunda etapa de avaliação do artefato



Fonte: O autor

3.2 Metodologia de desenvolvimento

O desenvolvimento da etapa de codificação deste trabalho foi embasado em uma combinação adaptada dos métodos ágeis Scrum e Kanban. Essa abordagem foi escolhida devido à necessidade de organizar e gerenciar de forma eficiente as tarefas do projeto, permitindo priorizar demandas críticas e ajustar as etapas conforme o progresso e os desafios encontrados. A flexibilidade do Kanban, aliada à estrutura iterativa do Scrum, possibilitou um controle eficaz sobre o fluxo de trabalho e garantiu a entrega contínua de funcionalidades incrementais. Tal combinação foi particularmente útil em um cenário de desenvolvimento acadêmico, onde os requisitos podem evoluir à medida que as pesquisas avançam e os testes de validação geram novas demandas.

O Scrum é *framework* ágil de gestão de projetos que auxilia equipes a gerenciar projetos complexos de maneira estruturada, entregando resultados de alto valor em curtos períodos chamados *sprints*. Focado no desenvolvimento contínuo, trabalho colaborativo entre equipes multifuncionais e avaliação e adaptação frequentes, o Scrum divide o projeto em *sprints*, permitindo identificar e resolver problemas rapidamente. Cada *sprint*, com duração de uma a quatro semanas, transforma itens do *backlog*[1] em incrementos no *software*, promovendo um progresso contínuo e eficiente (Educação, 2024).

[1] É a lista de todas as tarefas que estão pendentes de serem feitas.

Kanban é um método de gestão visual baseado em princípios *Lean*, originalmente desenvolvido pela Toyota para processos de manufatura. Posteriormente, foi adaptado para o desenvolvimento de software com o intuito de melhorar a eficiência e a transparência do fluxo de trabalho. Seu principal objetivo é aprimorar a gestão de processos e a entrega contínua, enfatizando a visibilidade dos itens de trabalho e limitando o trabalho em progresso para otimizar o fluxo. Diferentemente do Scrum, o Kanban não possui prazos definidos, permitindo uma maior flexibilidade e ajuste contínuo às demandas do projeto (Ahmad; Markkula; Oivo, 2013).

Combinado com o quadro *Kanban* cada tarefa a ser entregue fica disposta na ferramenta *trello* [2] que possui três estados: a fazer, em andamento e feito. As *sprints* acontecem em ciclos semanais e quando finalizadas as tarefas, elas são enviadas para o *github*.

Estas duas metodologias, quando usadas juntas, garantem que o entregável está conforme o esperado, e caso não esteja, ela é capaz de identificar se existe tempo hábil para retornar do começo e realizar os ajustes necessários.

[2] Ferramenta web de gestão de projetos baseada em quadro *Kanban*.

4 FUNEXPRESSION: UM *PIPELINE* PARA ANÁLISE DE EXPRESSÃO DIFERENCIAL DE GENES DE FUNGOS

O foco deste capítulo é discutir as ferramentas, a arquitetura, o fluxo de dados e as avaliações de FunExpression.

4.1 Passos e Ferramentas que compõem o *pipeline*

A seleção das ferramentas para a construção do pipeline foi baseada nos trabalhos correlatos mencionados na Seção 2.5.

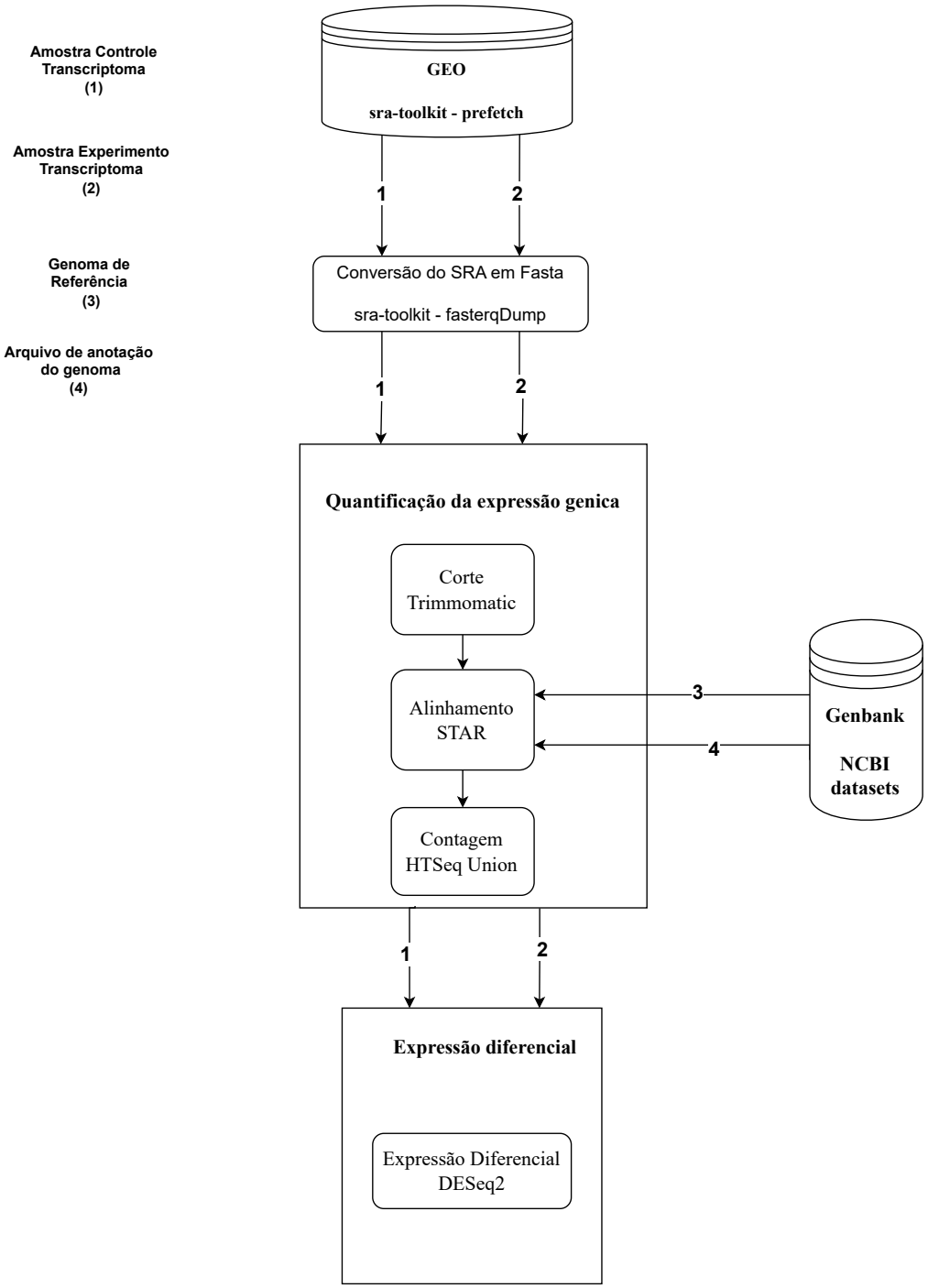
O fluxo se inicia com a obtenção das amostras controle e experimento do transcriptoma, assim como do genoma de referência que foram inseridas pelo usuário. Através do sra-toolkit é possível realizar o download do transcriptoma por meio do comando *prefetch*. Já conversão dos arquivos no formato SRA para FASTQ é feita com o comando *fasterq-dump* da mesma ferramenta. Enquanto os arquivos FASTA e GTF do genoma de referência são obtidos através do NCBI Datasets.

Na etapa de pré-processamento, também conhecida como corte, a ferramenta mais utilizada foi o Trimmomatic. Para o alinhamento, as ferramentas mais empregadas foram o STAR e o HISAT2, enquanto, na quantificação, as mais frequentes foram o HTSeq e o featureCounts. Finalmente, na etapa de análise de DEGs, destacou-se o uso do DESeq2. Entretanto, o estudo *Systematic comparison and assessment of RNA-seq procedures for gene expression quantitative analysis* (Corchete et al., 2020) realizou uma comparação abrangente de mais de 192 pipelines, formados a partir de combinações das ferramentas mais populares para a análise de DEGs. De acordo com esse estudo, a combinação mais precisa e exata foi Trimmomatic + RUM + HTSeq Union. Apesar disso, o elemento que mais influenciou na precisão dos resultados foi a utilização do HTSeq para a contagem, o qual estava presente nos 10 pipelines mais bem avaliados. Em contrapartida, pipelines baseados em alinhamentos brutos, como aqueles que utilizavam a cobertura do StringTie e os valores de NumReads do Salmon, ocuparam as últimas posições no ranking, reforçando a relevância de uma combinação adequada de métodos de contagem e normalização.

Inicialmente, a ferramenta escolhida para a etapa de alinhamento foi o RUM, conforme recomendado no estudo de Corchete et al.. No entanto, ao tentar realizar o alinhamento, a função destinada a prefixar o organismo modelo apresentou um erro ao verificar se o valor inserido era um dígito. Após uma busca no repositório da ferramenta para solucionar o problema, constatou-se que o mesmo estava sem manutenção desde 2022. Além disso, uma discussão aberta no GitHub revelou que os mantenedores recomendavam o uso de alinhadores mais modernos, como o STAR, para essa etapa. Assim, no desenvolvimento

do *FunExpression*, a etapa de alinhamento foi adaptada, substituindo o RUM pelo RNA STAR (Gapped-read mapper for RNA-Seq data). A Figura 16 demonstra diagrama do pipeline completo proposto para FunExpression.

Figura 16 – Pipeline completo



Fonte: O autor

4.1.1 Avaliação 1

O objetivo desta avaliação é validar, antes do desenvolvimento, as ferramentas identificadas na revisão sistemática para a construção do pipeline. Essa validação foi realizada utilizando o artigo de (Li et al., 2015), que aborda a regulação da expressão gênica de celulases em fungos filamentosos do organismo Penicillium oxalicum.

O referido artigo investiga a rede de regulação transcricional responsável pela produção de celulases, utilizando uma biblioteca de mutantes com deleção de genes que codificam fatores de transcrição. Para esta análise, foi utilizado o genoma de referência GCA_000346795.1, com as seguintes amostras: para a condição controle, SRR2042782, SRR2042783 e SRR2042784; e para a condição experimento, SRR2042785, SRR2042786 e SRR2042787. As tabelas com os resultados de genes *upregulated* e *downregulated* podem ser encontradas em https://doi.org/10.1371/journal.pgen.1005509.s014 e https://doi.org/10.1371/journal.pgen.1005509.s013, respectivamente.

O pipeline de validação foi implementado utilizando a plataforma Galaxy. O Galaxy permite a montagem de um pipeline personalizado, onde o usuário seleciona as ferramentas necessárias para cada etapa.

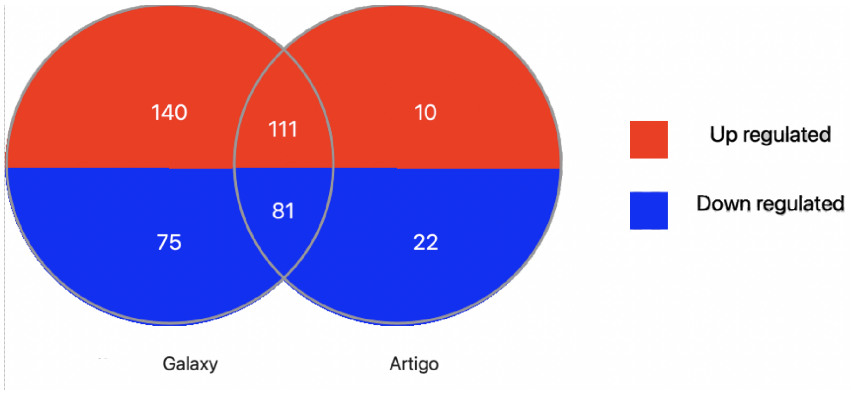
A montagem do pipeline incluiu as seguintes etapas:

Download dos transcriptomas: Utilizando os números de acesso fornecidos no artigo, os dados foram importados para a plataforma através da opção "Download and Extract Reads in FASTQ". Pré-processamento: O corte foi realizado com o software Trimmomatic, utilizando os seguintes parâmetros: *Perform initial ILLUMINACLIP step: yes Adapter sequence: TruSeq2 (single-ended, for Illumina GAII) Average quality: 0.* Os demais parâmetros foram mantidos como padrão. Alinhamento: Realizou-se o download do arquivo do genoma no formato FASTA e do arquivo de anotação (GTF), necessários para o alinhamento. O alinhamento foi configurado para gerar contagens por gene e saídas no formato BAM (TranscriptomeSAM GeneCounts). Contagem de leituras: A ferramenta HTSeq foi utilizada para gerar a contagem de genes. Os arquivos `mapped.bam` e o GTF foram carregados, utilizando os parâmetros padrão. Análise de expressão diferencial: O DeSeq2 foi empregado para comparar as condições controle e experimento. Foram enviadas as triplicatas de cada condição, mantendo os parâmetros padrão da ferramenta. O DeSeq2 produziu um gráfico vulcano e um arquivo no formato TSV contendo os IDs dos genes, valores de log2FC, taxa de erro, *p-value* e *p-adj.* Com os resultados obtidos pelo Galaxy e os dados do artigo, iniciou-se a comparação dos genes identificados como *upregulated* e *downregulated.* Um código desenvolvido no GitHub foi utilizado para essa análise que pode ser acessado através do link `https://github.com/Adeonita/funexpression_reviews/tree/main/avaliacao_1`. Para isso, aplicou-se o filtro $p\text{-}adj < 0{,}05$, reduzindo a probabilidade de falsos positivos para 5%. Foram considerados *upregulated* os genes com

log2FC > 1 e *downregulated* aqueles com log2FC < -1.

Os resultados do pipeline FunExpression em ambiente Galaxy identificaram 140 genes *upregulated* e 75 *downregulated* exclusivamente para o Galaxy. Exclusivamente no artigo foram encontrados 10 genes *upregulated* e 22 *downregulated*, enquanto a interseção revelou 111 *upregulated* e 81 *downregulated*, conforme ilustrado na Figura 17.

Figura 17 – Diagrama de *venn* com a distribuição da regulação dos genes a partir do pipeline proposto



Fonte: O autor

Embora os resultados obtidos não tenham sido idênticos aos reportados no artigo de referência, eles foram considerados satisfatórios para prosseguir com a etapa seguinte. As diferenças nos resultados podem ser atribuídas às especificidades da plataforma Galaxy, que utiliza versões adaptadas ou desatualizadas de algumas ferramentas em comparação com as versões mais recentes disponíveis nos repositórios. Por exemplo, a ferramenta STAR utilizada no Galaxy estava em uma versão anterior 2.7.11a, enquanto a versão mais recente disponível no momento do desenvolvimento deste trabalho era a 2.7.11b. Diferenças de versão semelhantes também foram observadas em outras ferramentas, como HTSeq e DESeq2, que apresentam variações de desempenho entre as versões testadas. Ainda assim, os genes diferencialmente expressos identificados apresentaram uma sobreposição significativa com os resultados descritos no artigo de referência, conforme mostrado nos diagramas de Venn. Essa sobreposição demonstra a capacidade do *pipeline* de identificar genes biologicamente relevantes. Portanto, embora os resultados obtidos com o Galaxy não sejam idênticos aos apresentados no artigo original, a diferença entre eles não é altamente discrepante. Além disso, nesta avaliação, o pipeline demonstrou uma sensibilidade superior, identificando um maior número de genes em comparação ao estudo original. Dessa forma, é viável sua implementação em um ambiente Docker, que oferece maior controle sobre as configurações e reprodutibilidade do processo.

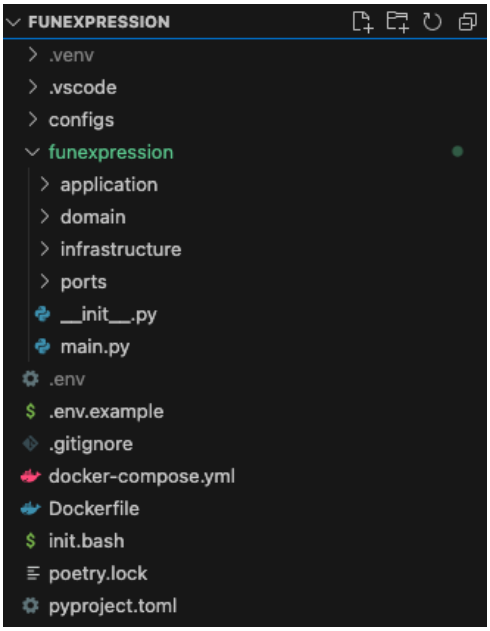
4.2 A arquitetura do FunExpression

Nesta sessão, discutiremos a estrutura arquitetural do código, a construção dos *contaienrs* no Docker, o fluxo dos dados, a construção das filas e dos *workers* e a interface com o usuário.

4.3 A estrutura arquitetural do código

Todo o código foi construído a partir dos conceitos da Arquitetura Limpa. Desta maneira, a sua estrutura de diretórios ficou dividida em: *application*, *domain* e *infrastructure*, conforme a Figura 18. A camada de aplicação contém o conteúdo necessário para o contrato da aplicação, a camada de domínio possui os casos de uso de cada etapa do *pipeline* e a camada de infraestrutura o código necessário para a execução das ferramentas em si.

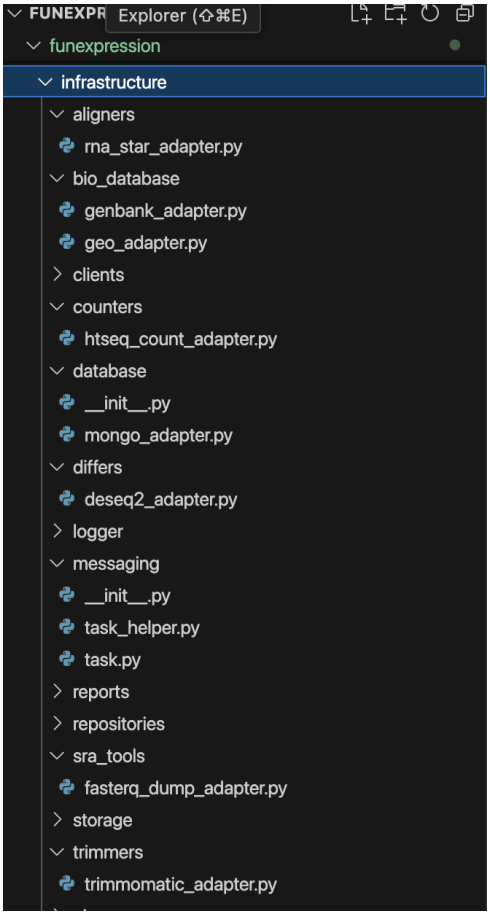
Figura 18 – Estrutura de diretórios de FunExpression



Fonte: O autor

Na camada de infraestrutura, cada ferramenta escolhida para ser utilizada em cada etapa do pipeline possui o seu *adapter*, conforme a Figura 19. Isso deixa o sistema altamente desacoplado e flexível a mudanças. Pois, caso haja necessidade de alterar a ferramenta utilizada, apenas a parte mais externa precisará ser alterada, uma vez que as dependências são injetadas de fora para dentro.

Figura 19 – Estrutura de diretórios da infraestrutura



Fonte: O autor

4.4 A construção dos containers

A criação dos containers do sistema foi estruturada com base no conceito de *multistage builds* do Docker, o que garante reprodutibilidade, eficiência e organização. Essa abordagem utiliza uma imagem inicial denominada *builder*, ilustrada na Figura 20, para realizar a instalação de dependências e a preparação do ambiente de execução.

Figura 20 – Container para instalar as dependências

```
FROM python:3.11-buster AS builder

RUN pip install poetry==1.8.3

ENV POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache

WORKDIR /app

COPY pyproject.toml poetry.lock ./

RUN poetry install --no-dev --no-root && rm -rf $POETRY_CACHE_DIR
```

Fonte: O autor

As imagens correspondentes às etapas do pipeline reutilizam o ambiente previamente configurado pela imagem *builder*, adicionando apenas as dependências específicas necessárias. Para isso, elas utilizam como base a imagem *python:3.11-slim-buster*[1], uma versão compacta e otimizada do Python 3.11 baseada no Debian Buster, que mantém apenas os pacotes essenciais para otimizar recursos e facilitar a personalização, conforme a Figura 21.

Figura 21 – Imagem Docker para o corte reutilizando o contexto da imagem *builder*

```
FROM python:3.11-slim-buster AS trimming_worker

RUN apt-get update
RUN apt-get install -y wget unzip
RUN apt-get install -y openjdk-11-jre-headless

RUN wget http://www.usadellab.org/cms/uploads/supplementary/Trimmomatic/Trimmomatic-0.39.zip
RUN unzip Trimmomatic-0.39.zip
RUN mv Trimmomatic-0.39 trimmomatic

ENV VIRTUAL_ENV=/app/.venv \
    JAVA_HOME=/usr/lib/jvm/openjdk-11-jre-headless/jre/bin/java \
    PATH="/app/.venv/bin:/app/java/bin:$PATH"

WORKDIR /funexpression

COPY --from=builder ${VIRTUAL_ENV} ${VIRTUAL_ENV}

ARG TRANSCRIPTOME_TRIMMING_WORKER_CONCURRENCY
ENV TRANSCRIPTOME_TRIMMING_WORKER_CONCURRENCY_VALUE=$TRANSCRIPTOME_TRIMMING_WORKER_CONCURRENCY

ENTRYPOINT celery -A infrastructure.messaging.task worker -l info --pool=threads --queues=trimming_transcriptome --
concurrency=$TRANSCRIPTOME_TRIMMING_WORKER_CONCURRENCY_VALUE
```

Fonte: O autor

[1] https://hub.docker.com/_/python

A imagem de download SRA inclui o SRA Toolkit na versão 3.1.1[2], uma ferramenta amplamente usada para manipulação de dados do Sequence Read Archive (SRA). Para realizar o download dos arquivos de sequenciamento, foi utilizado o comando *prefetch*, que apresenta vantagens significativas no gerenciamento de downloads. Esse comando é otimizado para lidar com falhas durante o processo, sendo capaz de retomar transferências interrompidas devido a problemas de rede, além de evitar downloads redundantes ao verificar se os arquivos já foram previamente baixados. A ferramenta e o comando foram integrados ao ambiente configurado pela imagem *builder*.

Já a imagem de conversão de SRA para FASTA utiliza o comando *fasterq-dump*, que também faz parte do SRA Toolkit previamente configurado. Esse comando é otimizado para eficiência, utilizando múltiplas *threads* para realizar a conversão de maneira rápida e paralela.

Para o pré-processamento de dados FASTA, foi criada a imagem de *trimming*, que incorpora o Trimmomatic na versão 0.39 (Bolger; Lohse; Usadel, 2014) e o OpenJDK 11, necessário para sua execução.

A imagem destinada ao download de genomas utiliza a ferramenta NCBI Datasets (O'Leary et al., 2024), obtida diretamente do repositório oficial, para manipulação de dados genômicos. De forma semelhante, a imagem para alinhamento de transcriptomas inclui o STAR Aligner na versão 2.7.11b (Dobin et al., 2013), amplamente utilizado para alinhamento de sequências RNA-Seq.

Para a geração de índices genômicos, foi desenvolvida uma imagem que também utiliza o *STAR Aligner* 2.7.11b, permitindo a criação eficiente dos índices necessários para etapas subsequentes do pipeline. A imagem destinada à contagem de transcriptomas utiliza a biblioteca *HTSeq* na versão 2.0.9[3] para realizar as contagens gênicas. Por fim, a imagem de análise diferencial de expressão gênica incorpora a biblioteca PyDESeq2 na versão 0.4.12[4].

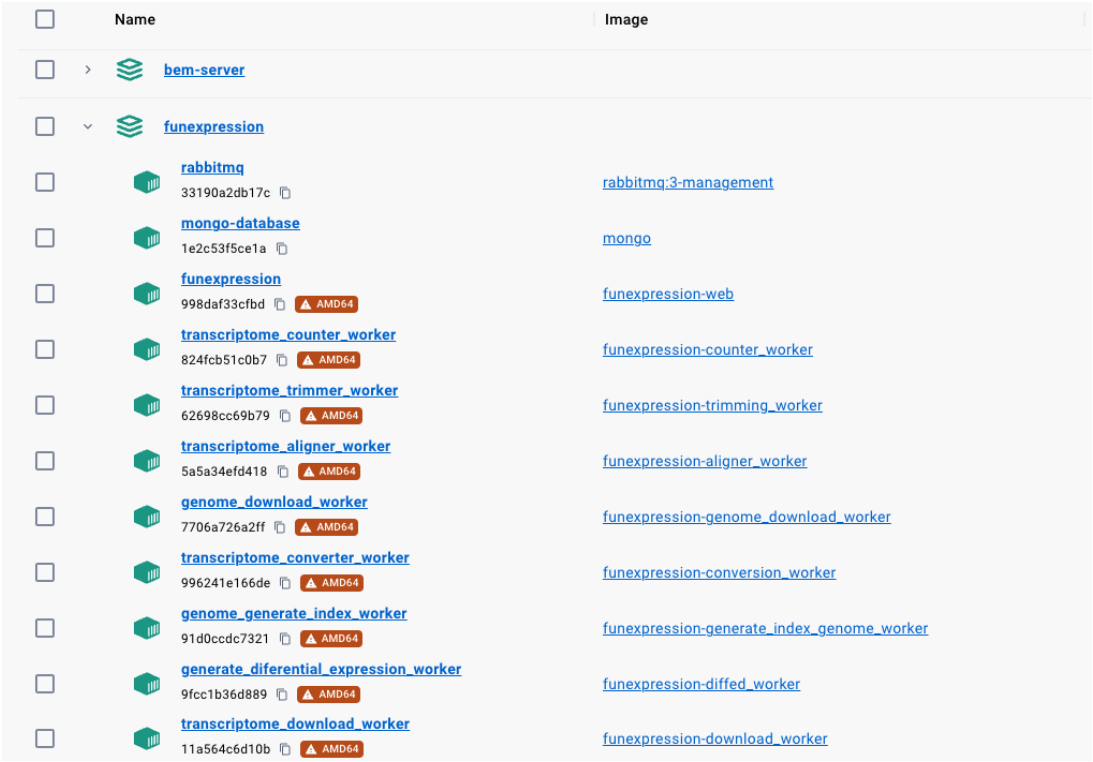
Um diferencial importante na construção das imagens é a configuração da concorrência para o Celery, que pode ser ajustada dinamicamente por meio de variáveis de ambiente. Essa flexibilidade permite adaptar a quantidade de tarefas simultâneas processadas por cada *worker* de acordo com a demanda e os recursos disponíveis, otimizando o desempenho do sistema.

A abordagem da construção das imagens específicas foi adotada pois, ao realizar o download das ferramentas diretamente de seus repositórios oficiais, evita-se a dependência de imagens desatualizadas ou sem manutenção, assegurando o uso das versões mais recentes e estáveis de cada ferramenta.

[2] https://github.com/ncbi/sra-tools
[3] https://htseq.readthedocs.io/
[4] https://pydeseq2.readthedocs.io/

Na Figura 22, é possível observar todos os containers criados e em execução para o FunExpression.

Figura 22 – Containers da aplicação FunExpression na interface gráfica do Docker



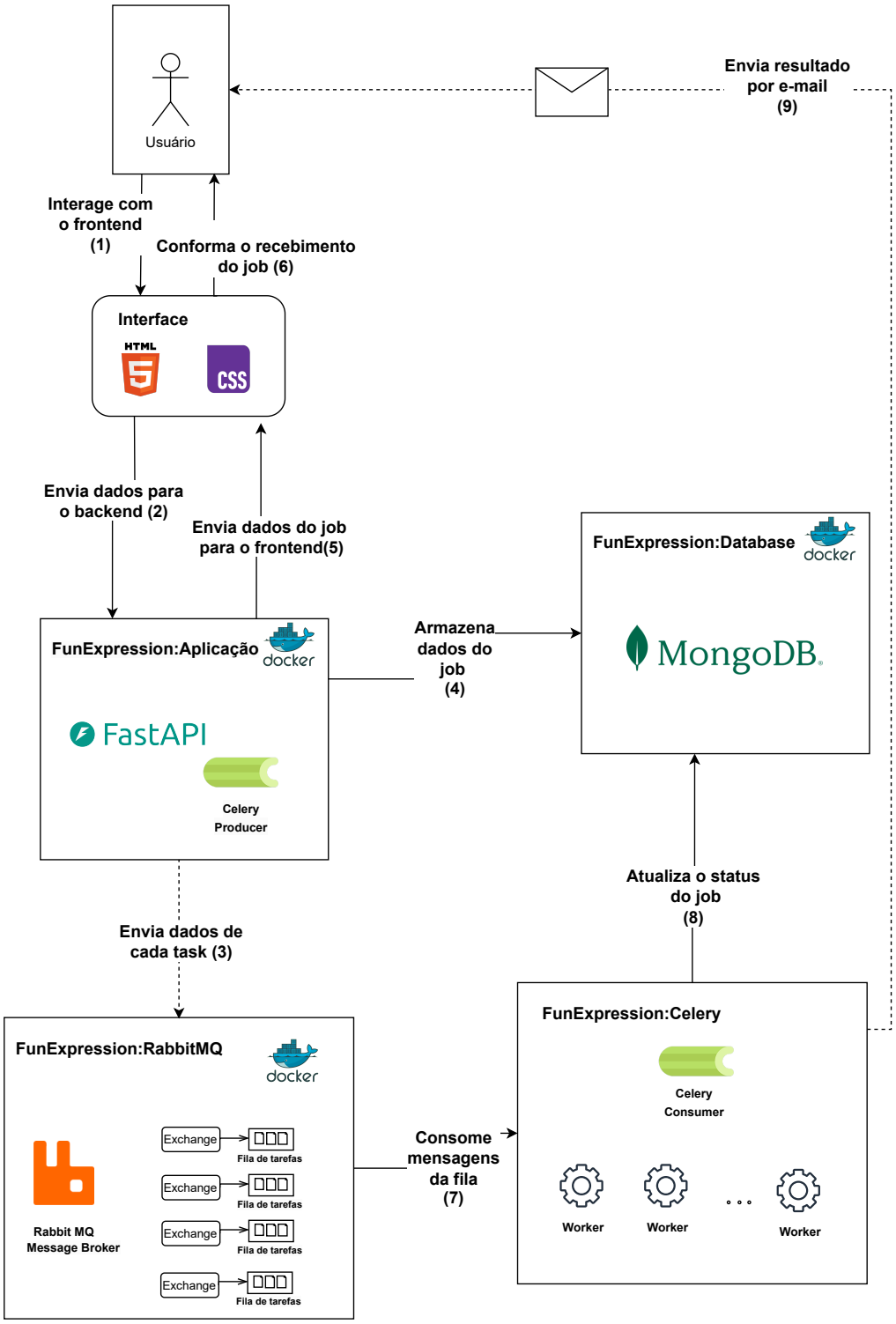
Fonte: O autor

O uso da interface gráfica do Docker facilita a visualização e o gerenciamento das imagens construídas.

4.4.1 O fluxo de dados

Além da arquitetura em que o código foi estruturado, é necessário entender como a informação é processada dentro dessa arquitetura. Na Figura 23, é demonstrado o processamento da análise de uma expressão diferencial, desde a requisição do usuário até o envio por e-mail:

Figura 23 – Arquitetura FunExpression



Fonte: O autor

No item (1), o usuário insere na interface os dados das amostras a serem analisadas, são submetidos 1 número de acesso correspondente ao genoma de referência armazenado no GenBank; 3 números de acesso, correspondentes à triplicata do transcriptoma da condição de controle; e 3 números de acesso correspondentes à triplicata do transcriptoma da condição de experimento, ambos armazenados no GEO. Além disso, o usuário informa seu nome e e-mail, que posteriormente servirão para o envio dos resultados da análise de DEGs.

Em seguida, no item (2), a interface da aplicação se comunica com o *backend* repassando as informações enviadas pelo usuário. A aplicação, por meio do Celery, realizará o envio das *tasks* contendo os dados das sequências para as filas do RabbitMq, onde as mensagens ficarão armazenadas até que sejam processadas pelo Consumidor, item (3).

Os dados de número de acesso de cada *sra_id*, o id do *pipeline*, *job_id*, o *run_id*, que é uma combinação do id do pipeline e do e-mail do usuário, e-mail do usuário, também são armazenados no banco de dados, item (4). Na Figura 24 é possível visualizar a representação de um pipeline no banco de dados.

Figura 24 – Representação do *job* no banco de dados

Fonte: O autor

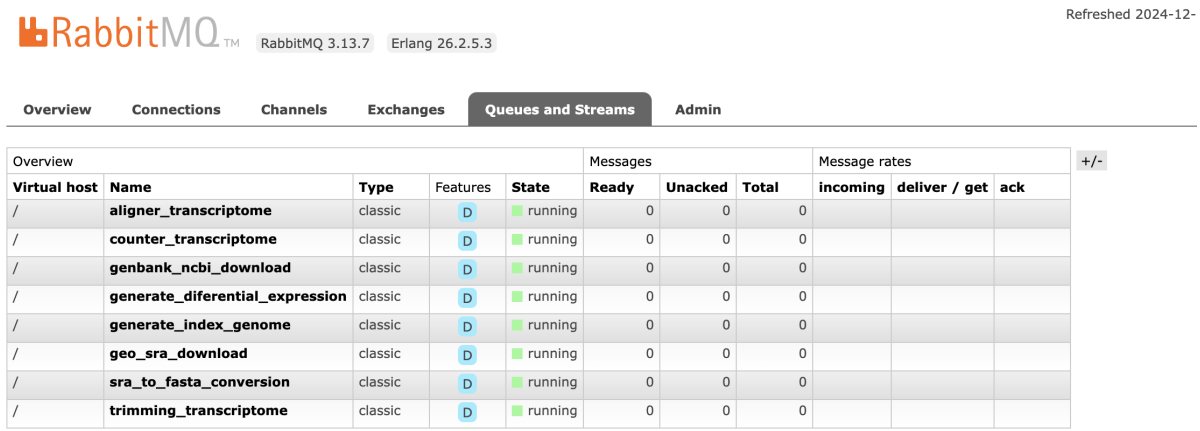
Após o agendamento e a persistência dos dados no banco de dados MongoDB, o *backend* envia os dados do *job* para a interface da aplicação (5), que exibirá uma mensagem ao usuário informando que os dados foram recebidos e que ele deve aguardar pelo processamento (6).

Em seguida, o Celery entra em ação novamente, conforme o item (7), desta vez como consumidor dos dados nas filas. Cada *worker* consome as mensagens armazenadas no RabbitMQ, persiste o resultado de cada etapa na pasta de arquivos correspondente àquela análise e atualiza o status do *job* (8). Ao final da execução, o sistema dispara um e-mail para o usuário solicitante com o resultado da análise da expressão diferencial (9).

4.4.2 Filas e Workers

A arquitetura do *pipeline* utiliza filas e workers, conforme ilustrado na Figura 25 para garantir a execução eficiente e sequencial das etapas do processamento de dados.

Figura 25 – Visualização das filas na interface do RabbitMq



Fonte: O autor

Cada fila do RabbitMQ é responsável por armazenar as mensagens que indicam o estado de uma etapa específica, e cada worker consome as mensagens dessas filas, realizando as ações correspondentes. A definição dessas filas e workers permite que o processo de análise de dados seja distribuído e escalável, garantindo que as etapas sejam executadas de maneira independente, mas de forma sincronizada quando necessário. A seguir, são detalhados os papéis de cada fila e worker no processo. Na implementação do *pipeline*, oito filas do RabbitMQ foram definidas em:

- *genome_download*: armazena o id do *pipeline* e o número de acesso do genoma de referência;
- *generate_genome_index*: armazena o id do *pipeline*, o número de acesso do genoma de referência, o caminho para o arquivo de anotação e o arquivo do genoma, e o caminho onde o index do genoma deve ser armazenado;

- *transcriptome_download*: armazena o id do *pipeline*, o número de acesso do sra e o grupo da triplicata do transcriptoma;
- *transcriptome_converter*: armazena o id do *pipeline*, o número de acesso do sra e o grupo da triplicata do transcriptoma;
- *transcriptome_trimmer*: armazena o id do *pipeline*, o número de acesso do sra, o grupo da triplicata do transcriptoma, o tipo do corte que por padrão é o *single end*, o caminho do arquivo convertido e o caminho onde será armazenado o arquivo cortado;
- *transcriptome_aligner*: armazena o id do *pipeline*, o número de acesso do sra, o grupo da triplicata do transcriptoma, o caminho do index do genoma, o caminho do arquivo cortado e o caminho onde será armazenado o arquivo alinhado;
- *transcriptome_counter*: armazena o id do *pipeline*, o número de acesso do sra, o grupo da triplicata do transcriptoma, o caminho do arquivo de anotação do genoma, o caminho do arquivo alinhado e o caminho onde será armazenado o arquivo contado;
- *generate_diferential_expression*: armazena o id do *pipeline* e uma lista com o caminho de todos os arquivos sra contados.

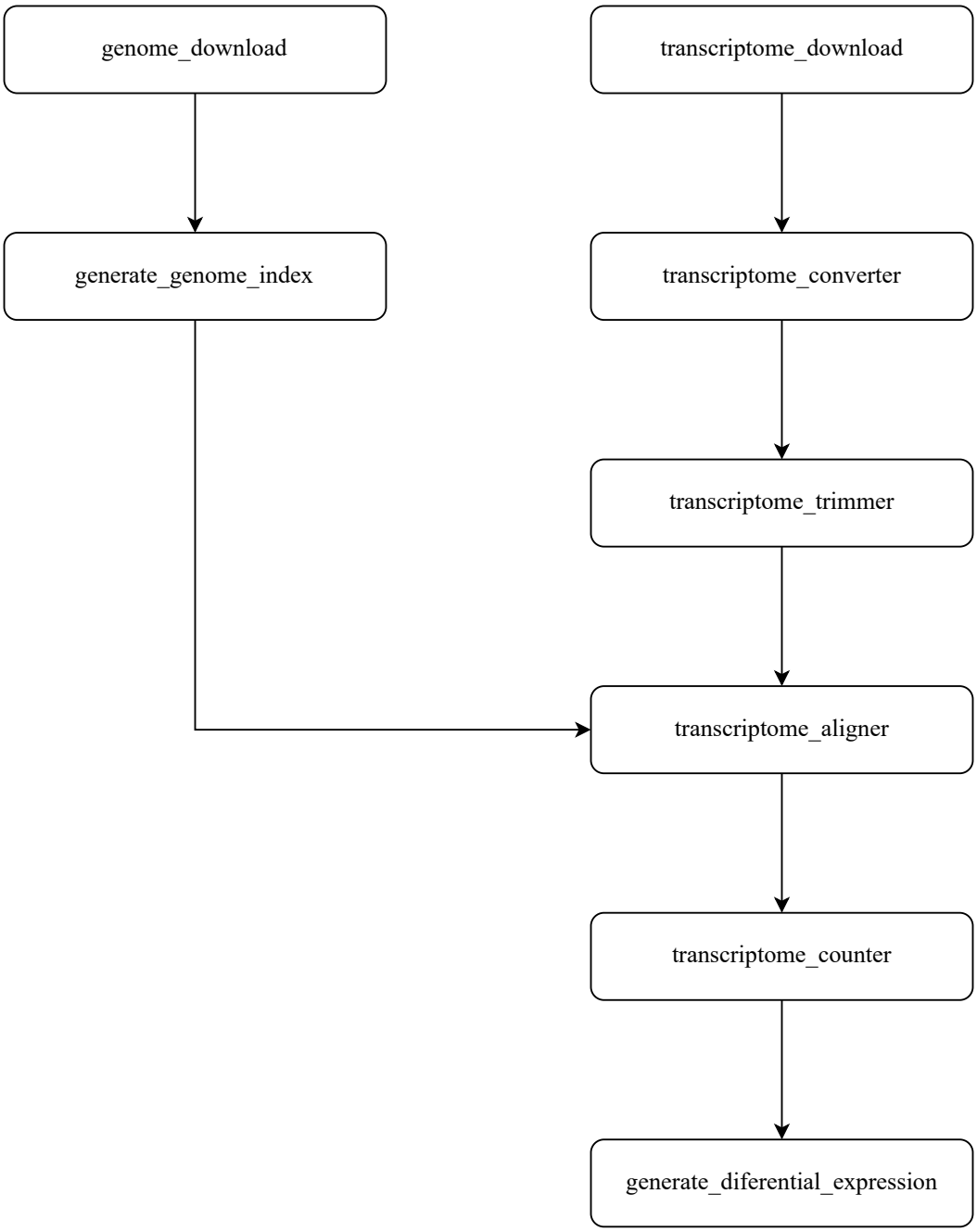
E oito workers, um para cada fila:

- *genome_download_worker*: realiza download dos arquivos fasta e gtf do genoma no NCBI;
- *genome_generate_index_worker*: gera o index do genoma utilizando os arquivos fasta e gtf do genoma;
- *transcriptome_download_worker*: realiza o download do arquivo sra do transcriptoma no GEO;
- *transcriptome_converter_worker*: realiza a conversão do arquivo sra do transcriptoma em um arquivo fasta;
- *transcriptome_trimmer_worker*: realiza o corte do transcriptoma em formato fasta;
- *transcriptome_aligner_worker*: realiza o alinhamento do transcriptoma a partir do index do genoma, gerando como saída um arquivo no formato bam;
- *transcriptome_counter_worker*: realiza a contagem dos transcritos utilizando o arquivo alinhado e o arquivo de anotação do genoma, gerando como saída um arquivo no formato txt;

- *generate_diferential_expression_worker*: realiza a análise da expressão diferencial de genes a partir dos arquivos de contagem de transcritos da condição experimento em relação à condição controle.

As etapas do pipeline devem acontecer de maneira sequencial; portanto, ao finalizar uma etapa, o sistema envia a mensagem para a etapa seguinte, conforme demonstrado na Figura 26.

Figura 26 – Fluxo das mensagens através dos *workers*



Fonte: O autor

A execução desses *workers* ocorre de maneira independente, pois é possível realizar o corte da triplicata 1 da amostra controle enquanto a triplicata 3 da amostra experimento está sendo contada. Entretanto, a etapa que gera a expressão diferencial é a única que necessita que todos os arquivos estejam contados para ser iniciada.

Para possibilitar o controle em nível de sistema, foi criado um mecanismo de status de arquivo e status de *pipeline*, denominado no código como *SRAFileStatusEnum*, conforme ilustrado na Figura 27. Cada arquivo da triplicata é inicialmente salvo no banco de dados com o status *pending*, sendo que, a cada conclusão de etapa, esse estado é atualizado.

Com base nesse controle, foi possível implementar as tentativas de reprocessamento em caso de falhas durante as etapas. O programa possui um controle, configurável via variável de ambiente, para definir o valor máximo de tentativas consecutivas, com intervalos de 180 segundos entre cada tentativa. Caso o erro persista, a mensagem é descartada, e o *Job* tem seu status atualizado para *failed*.

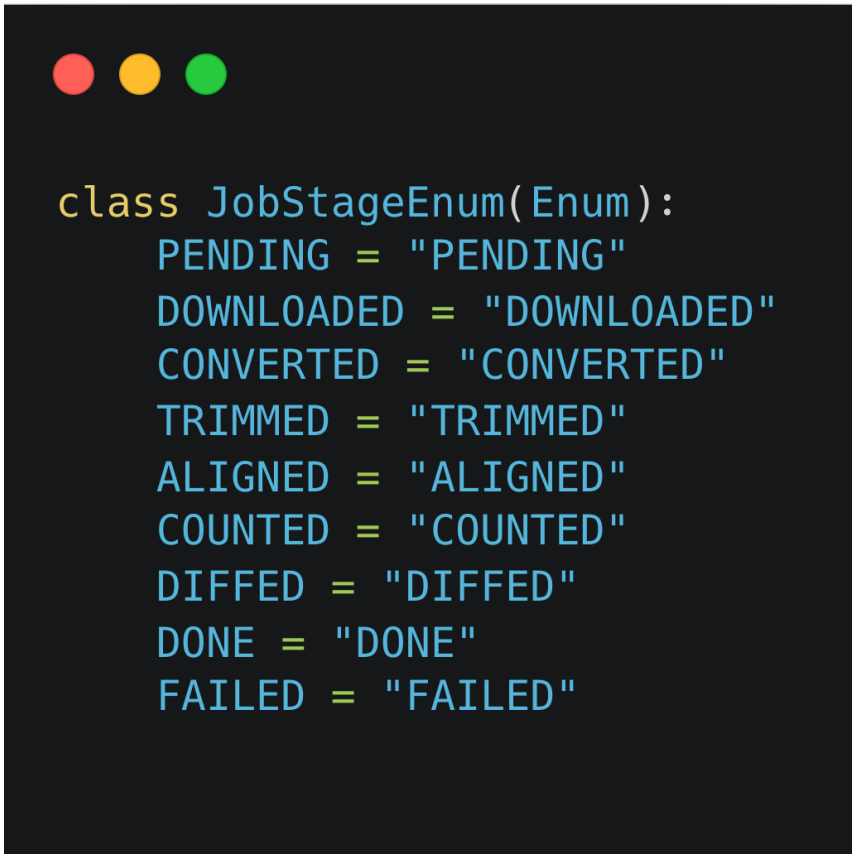
Figura 27 – Status de cada arquivo *sra* da triplicata

```
class SRAFileStatusEnum(StrEnum):
    PENDING = "PENDING"
    DOWNLOADED = "DOWNLOADED"
    CONVERTED = "CONVERTED"
    TRIMMED = "TRIMMED"
    ALIGNED = "ALIGNED"
    COUNTED = "COUNTED"
    DIFFED = "DIFFED"
```

Fonte: O autor

Um controle similar acontece para cada status do *pipeline*, que só é alterado quando todos os arquivos sra daquela etapa estiverem atualizados, na Figura 28 é possível visualizar cada status do *pipeline*.

Figura 28 – Status do *job*



```
class JobStageEnum(Enum):
    PENDING = "PENDING"
    DOWNLOADED = "DOWNLOADED"
    CONVERTED = "CONVERTED"
    TRIMMED = "TRIMMED"
    ALIGNED = "ALIGNED"
    COUNTED = "COUNTED"
    DIFFED = "DIFFED"
    DONE = "DONE"
    FAILED = "FAILED"
```

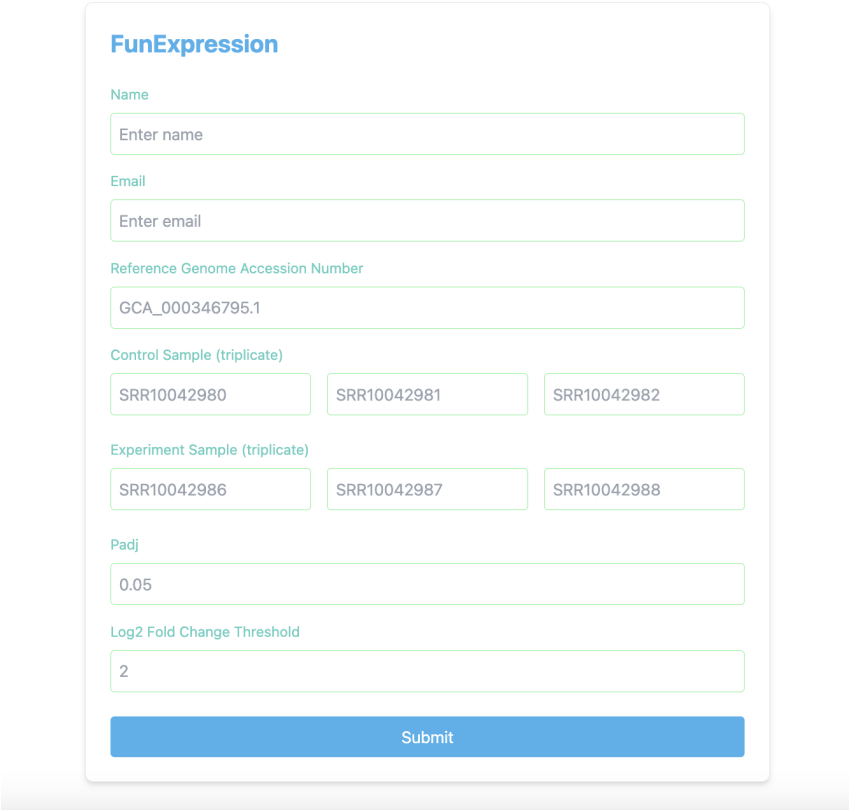
Fonte: O autor

Os arquivos de genoma fasta, gtf e index também possuem controle de status divididos em *pending*, *downloaded* e *generated*.

4.4.3 A interface com o usuário

A interação com o usuário acontece em dois momentos distintos, através do formulário de envio de dados por parte do usuário e ao fim do processamento do *pipeline* por meio do envio dos resultados por e-mail. A interface da aplicação pode ser visualizada na Figura 29 é composta por nome do usuário, e-mail, número de acesso de genoma de referência, número de acesso da triplicata da amostra controle e número de acesso da triplicata de experimento, *p-ajd* e limiar do log2FC. É importante ressaltar que a aplicação funciona apenas para transcriptomas registrados no GEO e genomas registrados no GenBank. Caso seja fornecido um número de acesso inexistente para qualquer um desses itens, o processamento da análise de DEGs não será realizado.

Figura 29 – Formulário para envio de dados



Fonte: O autor

Após o envio dos dados, o sistema exibe uma mensagem confirmando a submissão bem-sucedida e informando que o resultado será enviado por e-mail. Se o mesmo job for enviado mais de uma vez, o sistema alerta que já existe um job em execução e exibe seu status. No entanto, jobs com status *failed* ou *done* não geram alertas.

Ao fim da execução é enviado um e-mail para o usuário com os dados da análise enviada por ele, conforme a Figura 30 o gráfico *volcano* conforme ilustrado na Figura 31 e o arquivo csv com os genes classificados entre *upregulated* e *downregulated* conforme nas figuras Figura 32.

Figura 30 – Mensagem de resultado enviada por e-mail ao usuário

[🧬 FUNEXPRESSION]: Resultados da Análise de Expressão Diferencial

g2bc@uneb.br
para mim

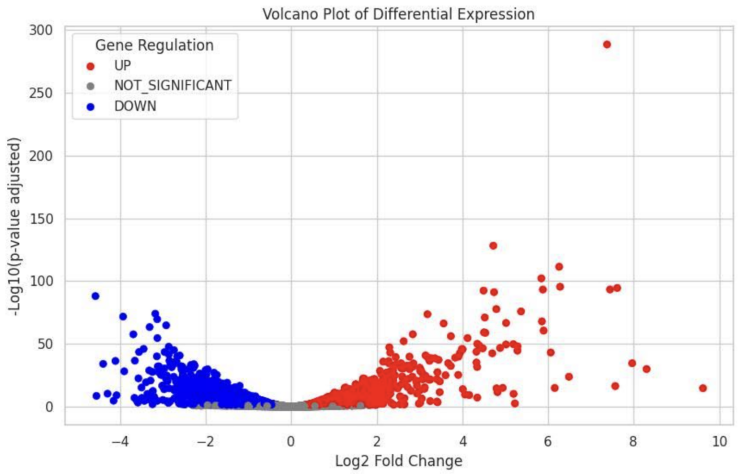
Olá adeonita,
Seguem informações referentes à análise da sua expressão diferencial

Você solicitou uma análise de expressão diferencial de genes, segue em anexo seus resultados.
run_id: adeonita-426c8cba-77d7-4347-b0b8-a90900357b40
Arquivos de entrada:
Controle: SRR28810692, SRR28810693, SRR28810694
Experimento: SRR28810689, SRR28810690, SRR28810691
Genoma: GCA_000143535.4
Parâmetros:
Threshold Padj: 0.05
Threshold Log2FC: ± 0.0
Anexos:
1. Gráfico volcano com os genes diferencialmente expressos classificados como UP/DOWN regulated
2. Planilha com resultados detalhados de expressão diferencial de genes

Esperamos que esta análise tenha sido divertida para você!
[🍄 FUNEXPRESSION by G2BC]

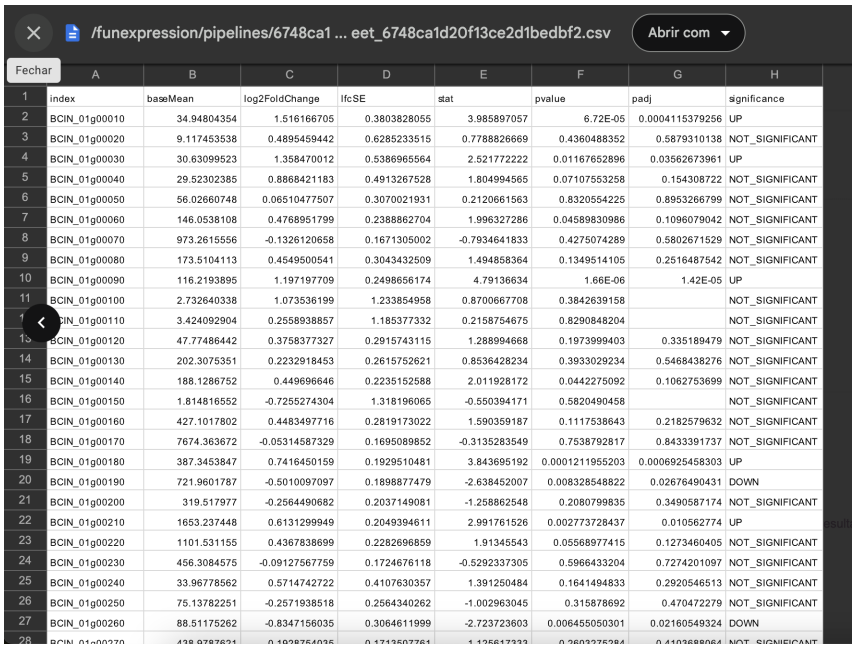
Fonte: O autor

Figura 31 – Anexo do e-mail de resultados: Gráfico Volcano



Fonte: O autor

Figura 32 – Anexo do de resultados: Arquivo com os dados dos genes classificados



	index	baseMean	log2FoldChange	lfcSE	stat	pvalue	padj	significance
2	BCIN_01g00010	34.94804354	1.516166705	0.3803828055	3.985897057	6.72E-05	0.0004115379256	UP
3	BCIN_01g00020	9.117453538	0.4895459442	0.6285233515	0.7788826669	0.4360488352	0.5879310138	NOT_SIGNIFICANT
4	BCIN_01g00030	30.63099523	1.358470012	0.5386965564	2.521772222	0.01167652896	0.03562673961	UP
5	BCIN_01g00040	29.52302385	0.8868421183	0.4913267528	1.804994565	0.07107553258	0.154308722	NOT_SIGNIFICANT
6	BCIN_01g00050	56.02660748	0.06510477507	0.3070021931	0.2120661563	0.8320554225	0.8953266799	NOT_SIGNIFICANT
7	BCIN_01g00060	146.0538108	0.4768951799	0.2388862704	1.996327286	0.04589830986	0.1096079042	NOT_SIGNIFICANT
8	BCIN_01g00070	973.2615556	-0.1326120658	0.1671305002	-0.7934641833	0.4275074289	0.5802671529	NOT_SIGNIFICANT
9	BCIN_01g00080	173.5104113	0.4549500541	0.3043432509	1.494858364	0.1349514105	0.2516487542	NOT_SIGNIFICANT
10	BCIN_01g00090	116.2193895	1.197197709	0.2498656174	4.79136634	1.66E-06	1.42E-05	UP
11	BCIN_01g00100	2.732640338	1.073536199	1.233854958	0.8700667708	0.3842639158		NOT_SIGNIFICANT
12	BCIN_01g00110	3.424092904	0.2558938857	1.185377332	0.2158754675	0.8290848204		NOT_SIGNIFICANT
13	BCIN_01g00120	47.77486442	0.3758377327	0.2915743115	1.288994668	0.1973999403	0.335189479	NOT_SIGNIFICANT
14	BCIN_01g00130	202.3075351	0.2232918453	0.2615752621	0.8536428234	0.3933029234	0.5468438276	NOT_SIGNIFICANT
15	BCIN_01g00140	188.1286752	0.449696646	0.2235152588	2.011928172	0.0442275092	0.1062753699	NOT_SIGNIFICANT
16	BCIN_01g00150	1.814816552	-0.7255274304	1.318196065	-0.550394171	0.5820490458		NOT_SIGNIFICANT
17	BCIN_01g00160	427.1017802	0.4483497716	0.2819173022	1.590359187	0.1117538643	0.2182579632	NOT_SIGNIFICANT
18	BCIN_01g00170	7674.363672	-0.05314587329	0.1695089852	-0.3135283549	0.7538792817	0.8433391737	NOT_SIGNIFICANT
19	BCIN_01g00180	387.3453847	0.7416450159	0.1929510481	3.843695192	0.0001211955203	0.0006925458303	UP
20	BCIN_01g00190	721.9601787	-0.5010097097	0.1898877479	-2.638452007	0.008328548822	0.02676490431	DOWN
21	BCIN_01g00200	319.517977	-0.2564490682	0.2037149081	-1.258862548	0.2080799835	0.3490587174	NOT_SIGNIFICANT
22	BCIN_01g00210	1653.237448	0.6131299949	0.2049394611	2.991761526	0.002773728437	0.010562774	UP
23	BCIN_01g00220	1101.531155	0.4367838699	0.2282696859	1.91345543	0.05568977415	0.1273460405	NOT_SIGNIFICANT
24	BCIN_01g00230	456.3084575	-0.09127567759	0.1724676118	-0.5292337305	0.5966433204	0.7274201097	NOT_SIGNIFICANT
25	BCIN_01g00240	33.96778562	0.5714742722	0.4107630357	1.391250484	0.1641494833	0.2920546513	NOT_SIGNIFICANT
26	BCIN_01g00250	75.13782251	-0.2571938518	0.2564340262	-1.002963045	0.315878692	0.470472279	NOT_SIGNIFICANT
27	BCIN_01g00260	88.51175262	-0.8347156035	0.3064611999	-2.723723603	0.006455050301	0.02160549324	DOWN
28	BCIN_01g00270	438.9787621	0.1928764035	0.1713507761	1.125617332	0.2603275284	0.4103688064	NOT_SIGNIFICANT

Fonte: O autor

5 AVALIAÇÃO 2 - VALIDAÇÃO DA FERRAMENTA WEB

Após o desenvolvimento, FunExpression foi submetido à avaliação que será discorrida nesta seção.

5.1 Avaliação 2

A avaliação dois tem o objetivo de validar os resultados do pipeline do FunExpression executando em ambiente Docker. Ela foi conduzida utilizando o software *FunExpression* para a análise de expressão diferencial, com base em cinco conjuntos de dados distintos, identificados como *id1*, *id2*, *id3*, *id4* e *id5*. Esses dados, extraídos de dois artigos selecionados, conforme detalhado na Tabela 3, incluíam amostras de controle e experimento, processadas de acordo com o genoma de referência correspondente. A análise concentrou-se na identificação de genes diferencialmente expressos, classificados como genes *upregulated* e *downregulated*. Os resultados foram organizados considerando os registros encontrados exclusivamente no software *FunExpression*, nos artigos de referência e na interseção entre ambos.

Tabela 3 – Tabela de resultados avaliados nos artigos

Id	**Triplicata Controle**	**Triplicata Experimento**	**Genoma de Referência**	**log2FC**	**padj**	**Artigo**
1	SRR28810692, SRR28810693, SRR28810694	SRR28810689, SRR28810690, SRR28810691	GCA_-000143535.4	0	$< 0{,}05$	(Rascle Bastien Malbert; Poussereau, 2024)
2	SRR28810692, SRR28810693, SRR28810694	SRR28810686, SRR28810687, SRR28810688	GCA_-000143535.4	0	$< 0{,}05$	(Rascle Bastien Malbert; Poussereau, 2024)
3	SRR28810689, SRR28810690, SRR28810691	SRR28810683, SRR28810684, SRR28810685	GCA_-000143535.4	0	0,05	(Rascle Bastien Malbert; Poussereau, 2024)

Id	Triplicata Controle	Triplicata Experimento	Genoma de Referência	log2FC	padj	Artigo
4	SRR28810686, SRR28810687, SRR28810688	SRR28810683, SRR28810684, SRR28810685	GCA_-000143535.4	0	0,05	(Rascle Bastien Malbert; Poussereau, 2024)
5	SRR2042782, SRR2042783, SRR2042784	SRR2042785, SRR2042786, SRR2042787	GCA_-000346795.1	>1 e < -1	0,05	(Li et al., 2015)

Fonte: O autor

O estudo de (Rascle Bastien Malbert; Poussereau, 2024) investiga as alterações transcriptômicas em um fungo fitopatogênico causador do mofo cinzento, avaliando como a deleção do gene *pacC* afeta a expressão gênica sob condições de pH ácido e neutro. A pesquisa foi realizada com dados do organismo *Botrytis cinerea B05.10*, sendo o sequenciamento realizado na plataforma Illumina HiSeq 2500, utilizando leituras de 125 bp em modo *single-end*. Para o processamento dos dados, foram empregados os programas Trimmomatic (v0.33) para o pré-processamento das sequências, STAR (v2.6) para o alinhamento das leituras, e FeatureCounts (Subread v2.0.2) para a quantificação dos genes. A análise DEGs foi conduzida utilizando o método *DiCoExpress*, considerando como diferencialmente expressos os genes cujo valor ajustado de *p-value* foi inferior a 0,05.

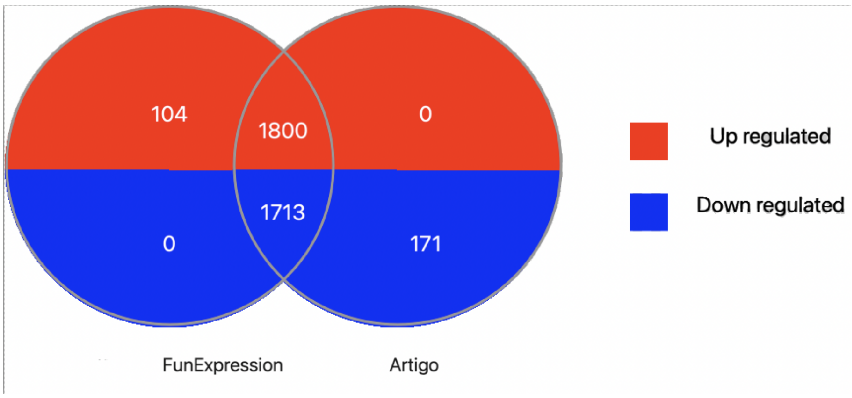
Já o estudo de (Li et al., 2015) analisou a rede de regulação transcricional envolvida na produção de celulases por meio de uma biblioteca de mutantes com deleção de genes codificadores de fatores de transcrição. Reguladores como ClrB, CreA, XlnR e AmyR demonstraram controle sinérgico e dependente de dose na expressão de genes de celulases. O sequenciamento foi realizado na plataforma Illumina HiSeq 2000, utilizando leituras de 50 bp, também em modo *single-end*. Diferentemente do primeiro artigo, neste estudo não foi realizada qualquer etapa de *trimming* das leituras. O alinhamento das sequências foi feito com o software SOAP2, enquanto a quantificação dos genes baseou-se nas abordagens RPKM, DESeq e NOIseq v2.10. Para a análise de DEGs, os genes foram considerados diferencialmente expressos quando o valor absoluto do *log2FC* era superior a 1.

O fluxo do processo de avaliação inicia-se com o envio dos dados por meio da interface, na qual são submetidos os números de acesso das amostras de controle e experimento dos transcriptomas, o genoma de referência, o e-mail do usuário, o valor desejado para o *log2FC* e o *p-value*. Após o processamento, os resultados da análise são enviados por e-mail e, em seguida, analisados utilizando o Jupyter Notebook[1]. Nesse ambiente, foram gerados os códigos necessários para a análise dos resultados produzidos pelo pipeline do *FunExpression*. Os resultados da avaliação dois podem ser acessados por meio do endereço: `https://github.com/Adeonita/funexpression_reviews`.

Para o conjunto de dados *id1*, os resultados indicaram que 1.713 genes *downregulated* estavam presentes tanto no *FunExpression* quanto no artigo, enquanto 171 genes foram exclusivos do artigo, e nenhum foi identificado apenas no *FunExpression*. Para os genes *upregulated*, 1.800 foram encontrados em ambos os contextos, enquanto 104 foram exclusivos do *FunExpression*, e nenhum foi exclusivo do artigo, conforme a Figura 33.

[1] Documento compartilhável que combina código de computador, descrições em linguagem simples, dados, visualizações gráficas como modelos 3D, gráficos, tabelas e figuras, além de controles interativos (Team, 2024).

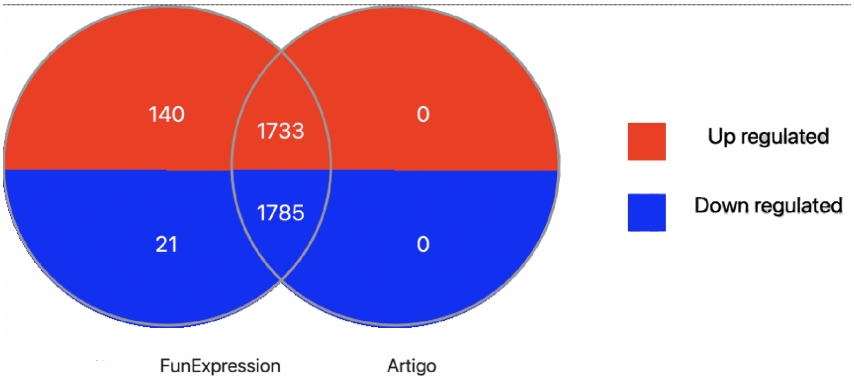
Figura 33 – Diagrama de venn com a distribuição da regulação dos genes do artigo 1



Fonte: O autor

No conjunto *id2*, os resultados demonstraram que 1.785 genes *downregulated* estavam presentes em ambas as plataformas, sendo 21 genes exclusivos do *FunExpression* e nenhum exclusivo do artigo. Para os genes *upregulated*, 1.733 foram encontrados em ambos os contextos, enquanto 140 foram exclusivos do *FunExpression*, e nenhum foi exclusivo do artigo, conforme a Figura 34.

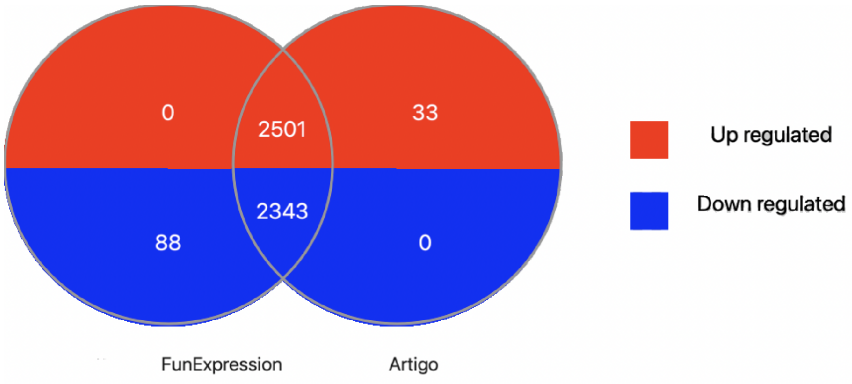
Figura 34 – Diagrama de venn com a distribuição da regulação dos genes do artigo 2



Fonte: O autor

No conjunto *id3*, observou-se que 2.343 genes *downregulated* estavam presentes em ambas as plataformas, enquanto 88 foram exclusivos do *FunExpression* e nenhum foi exclusivo do artigo. Para os genes *upregulated*, 2.501 estavam presentes em ambos os contextos, nenhum foi exclusivo do *FunExpression*, e 33 foram exclusivos do artigo, conforme a Figura 35.

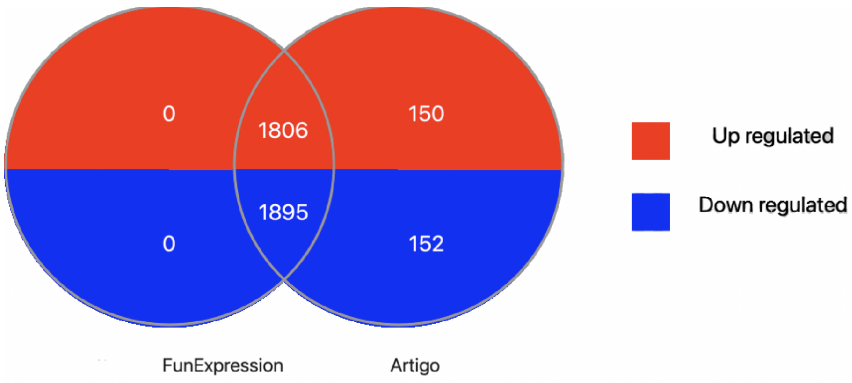
Figura 35 – Diagrama de venn com a distribuição da regulação dos genes do artigo 3



Fonte: O autor

No conjunto *id4*, 1.806 genes *downregulated* estavam presentes em ambas as plataformas, nenhum foi exclusivo do *FunExpression*, e 150 foram exclusivos do artigo. Para os genes *upregulated*, 1.895 estavam presentes em ambos os contextos, nenhum foi exclusivo do *FunExpression*, e 152 foram exclusivos do artigo, conforme a Figura 36.

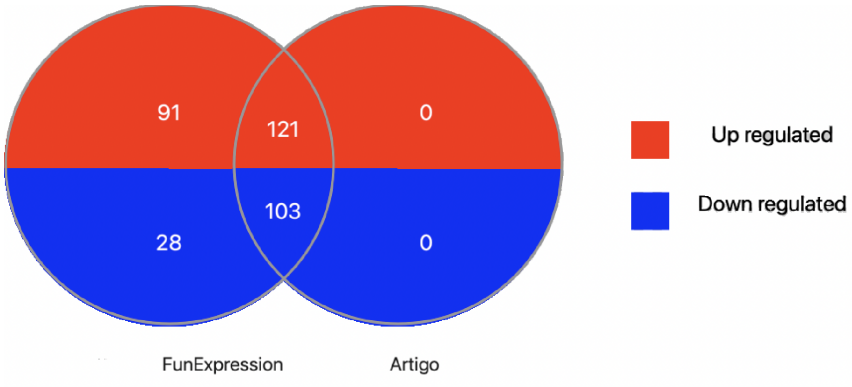
Figura 36 – Diagrama de venn com a distribuição da regulação dos genes do artigo 4



Fonte: O autor

Por fim, no conjunto *id5*, 103 genes *downregulated* estavam presentes em ambas as plataformas, enquanto 28 foram exclusivos do *FunExpression* e nenhum foi exclusivo do artigo. Para os genes *upregulated*, 121 estavam presentes em ambos os contextos, sendo 91 exclusivos do *FunExpression*, e nenhum foi exclusivo do artigo, conforme a Figura 37.

Figura 37 – Diagrama de venn com a distribuição da regulação dos genes do artigo 5



Fonte: O autor

5.1.1 Discussão dos resultados

Conforme discutido ao longo deste trabalho, diferentes ferramentas geram resultados distintos. Nesta seção, abordaremos os resultados encontrados.

5.1.1.1 Resultados diferentes para o mesmo artigo em ambientes diferentes

A Avaliação 1 apresentou resultados diferentes, pois foi realizada em ambiente Galaxy, que utiliza versões específicas de softwares adaptadas ou empacotadas para integração com a plataforma. Essas versões podem incluir modificações no código-fonte ou ajustes na compatibilidade com os workflows do Galaxy, o que pode impactar diretamente os resultados obtidos. Além disso, as versões utilizadas no Galaxy são, em alguns casos, mais antigas do que as empregadas no FunExpression. Por exemplo, o STAR utilizado no Galaxy é a versão 2.7.11a, enquanto o FunExpression faz uso da versão mais recente, 2.7.11b. Para o HTSeq, o Galaxy utiliza a versão 2.0.5, enquanto o FunExpression adota a versão 2.0.9. Quanto ao DESeq2, a versão nativa em R utilizada no Galaxy é a 2.11.40.8, enquanto o FunExpression utiliza o PyDESeq2, que corresponde à versão 0.4.12, uma adaptação do DESeq2 em Python baseada na versão 1.34.0 da ferramenta. Conforme apresentado na Tabela 4, que demonstra as versões utilizadas nos artigos, no *pipeline* em ambiente Galaxy e no *pipeline* do FunExpression em ambiente Docker.

Tabela 4 – Ferramentas utilizadas em cada etapa do *pipeline* por ambiente.

Etapa do *pipeline*	**Rascle Bastien Malbert e Poussereau (2024)**	**Li et al. (2015)**	***Pipeline* Galaxy**	***Pipeline* FunExpression**
Corte	Trimmomatic (v0.33)	Sem corte	Galaxy Version 0.39+galaxy2	Trimmomatic (v0.39)
Alinhamento	STAR (v2.6)	SOAP2	Galaxy Version 2.7.11a+galaxy1	RNA Star (v2.7.11b)
Contagem	FeatureCounts (Subread v2.0.2)	RPKM	Galaxy Version 2.0.5+galaxy0	HTSeq (v2.0.9)
DE	DiCoExpress	DESeq and NOIseq v2.10	Galaxy Version 2.11.40.8+galaxy0	PyDeseq2 (v0.4.12)

Fonte: O autor

Considerando que as versões do FunExpression são mais atualizadas, é possível que as melhorias de performance entre uma versão e outra tenham impactado diretamente a melhoria dos resultados do artigo de id 5 nos diferentes cenários das avaliações 1 e 2.

5.1.1.2 Discussão sobre os diferentes métodos de contagem, análise e normalização

Na análise de expressão diferencial utilizando RNA-Seq, as etapas de contagem e normalização são fundamentais. Os pipelines dos artigos utilizados nas avaliações e o FunExpression empregaram ferramentas distintas para a contagem de genes em dados de RNA-Seq, como HTSeq, FeatureCounts (Subread v2.0.2). Essas ferramentas possuem abordagens distintas, o que pode impactar os resultados obtidos, especialmente na etapa de contagem de genes, que é essencial para a análise de expressão diferencial. A ferramenta HTSeq adota uma abordagem baseada na contagem de leituras mapeadas em regiões genômicas específicas, utilizando um arquivo de anotações de genes (geralmente em formato GFF ou GTF). Ela oferece grande flexibilidade na contagem de leituras, sendo capaz de lidar com diferentes tipos de contagens. Essa flexibilidade e sua precisão na associação de leituras a genes ou regiões específicas tornam o HTSeq altamente eficaz em cenários com dados complexos ou em experimentos onde a precisão da contagem é essencial. Contudo, o HTSeq pode ser computacionalmente mais lento em grandes datasets, mas oferece maior precisão, o que o torna ideal para análises rigorosas. O FeatureCounts (Subread v2.0.2), por sua vez, é uma ferramenta eficiente e amplamente utilizada para contagem de genes, desenvolvida para lidar com grandes volumes de dados. Ela se destaca pela velocidade e pela capacidade de processar grandes conjuntos de dados, sendo frequentemente utilizada em experimentos com um grande número de amostras. Embora eficiente, o FeatureCounts pode apresentar limitações em cenários de sobreposição de transcritos e no gerenciamento de leituras que se alinham a múltiplas regiões genômicas. Em comparação com o HTSeq, o FeatureCounts pode ser menos preciso em situações que exigem uma contagem mais refinada, especialmente em genes com múltiplas isoformas.

Na etapa de análise de expressão diferencial, foram utilizadas as ferramentas DESeq2, DiCoExpress e NOIseq v2.10. Cada uma adota abordagens distintas para identificar genes diferencialmente expressos, o que pode influenciar significativamente os resultados obtidos. O DESeq2 utiliza um modelo estatístico baseado na distribuição binomial negativa para normalização dos dados e cálculo dos testes de significância. Essa abordagem se destaca por sua robustez e precisão, especialmente em experimentos com um número maior de amostras e replicações. Já o RPKM (Reads Per Kilobase per Million mapped reads) é uma métrica de normalização aplicada após a contagem de leituras, ajustando as contagens brutas com base no comprimento do gene e no total de leituras mapeadas na amostra. O RPKM permite a comparação relativa da expressão de genes dentro de uma mesma amostra, mas pode ser influenciado por variações no número total de leituras mapeadas. Embora útil para análises exploratórias, o RPKM não é ideal para estudos comparativos entre amostras ou condições experimentais devido à falta de ajuste para a composição das sequências. O DiCoExpress adota uma abordagem baseada em agrupamento *k-means* para lidar com a grande variabilidade biológica observada em algumas amostras. Embora eficiente em cenários com alta heterogeneidade, sua metodologia não realiza uma normalização tão refinada quanto a do DESeq2, o que pode impactar a precisão dos resultados em estudos com menos variabilidade. Já o NOIseq v2.10, com uma abordagem não paramétrica baseada em permutação, é particularmente útil em cenários com poucas amostras, mas pode ser limitado em comparação com métodos mais sofisticados, como o DESeq2.

6 TRABALHOS FUTUROS

FunExpression demonstrou um resultado satisfatório e superior aos pipelines utilizados na avaliação 2, entretanto algumas questões permanecem abertas e representam oportunidades para investigações futuras.

Análise de dados *paired-end*: atualmente, as análises foram realizadas utilizando leituras em modo *single-end*, embora o código esteja preparado para receber o tipo de corte; ajustes na estrutura de dados do alinhamento são necessários. A inclusão de dados *paired-end* permitirá uma análise mais robusta, dado que este formato oferece maior precisão no alinhamento e na identificação de transcritos, especialmente em regiões genômicas complexas.

Avaliações em plataformas de sequenciamento distintas do Illumina: este trabalho foi baseado exclusivamente em dados gerados pela tecnologia Illumina. Estudos futuros podem permitir que o usuário escolha o tipo de sequenciamento como PacBio ou Oxford Nanopore, ampliando a aplicabilidade da ferramenta.

Impacto da variação na quantidade de amostras biológicas: a análise foi conduzida com base em dados encontrados em revisão sistemática e em 100% dos casos os experimentos eram realizados com triplicatas. Entretanto, na busca por massa de dados para testes, foi encontrado diferentes abordagens de replicatas, como duplicatas.

Adição do parâmetro *average quality* no *frontend*: esta adição permitirá que o usuário ajuste a qualidade do corte de acordo com suas necessidades, melhorando a precisão dos resultados ao filtrar leituras de baixa qualidade, especialmente em dados com variação na qualidade das sequências.

Desenvolvimentos futuros podem implementar a flexibilidade da análise, permitindo duplicatas, triplicatas, quadruplicatas e até mesmo amostras maiores. Essas direções visam não apenas aumentar a abrangência dos resultados obtidos, mas também consolidar a versatilidade e eficácia da ferramenta em contextos experimentais mais amplos.

7 CONSIDERAÇÕES FINAIS

O desenvolvimento de FunExpression alcançou com êxito a totalidade os objetivos traçados:

1. Identificar as etapas do pipeline: as etapas do pipeline foram identificadas e são divididas entre: corte, alinhamento, contagem e análise da expressão diferencial;
2. Identificar as ferramentas mais adequadas para cada etapa: através da revisão de literatura foi possível encontrar a combinação de Trimmomatic, RNA Star, HTSeq e DESeq2;
3. Desenvolver um ambiente de experimentação para execução do pipeline: utilizando o ecossistema Docker foi desenvovido um ambiente de experimentação altamente reprodutível e portátil tanto para execução local quanto para implantação em diferentes servidores independentes de sistema operacional;
4. Avaliar o pipeline: através das avaliações 1 e 2 foi possível validar os resultados de FunExpression

Desta forma podemos afirmar que a construção do *pipeline* com as ferramentas encapsuladas remove a necessidade de instalação e decisão de escolha de ferramentas na análise de expressão diferencial de fungos. Contribuindo desta forma para que a comunidade científica preocupe-se apenas em concentrar suas energias na análise dos resultados da expressão diferencial, e não com a escolha de ferramentas, configuração de ambiente ou poder de processamento. Os próximos desafios são referentes ao aumento da flexibilidade para diferentes formatos de dados e filtros específicos de ferramentas a fim de que o usuário final possa aumentar as configurações de parâmetros de entrada dos seus experimentos.

REFERÊNCIAS

AHMAD, M. O.; MARKKULA, J.; OIVO, M. Kanban in software development: A systematic literature review. In: **2013 39th Euromicro Conference on Software Engineering and Advanced Applications**. [S.l.]: IEEE, 2013. Citado na página 47.

ALBERTS, B. **Molecular Biology of the Cell**. [S.l.]: ISBN 9781317563754, 2017. Citado nas páginas 19, 20, 21, 22 e 23.

ANDERS, S.; PYL, P.; HUBER, W. Htseq — a python framework to work with high-throughput sequencing data. **Bioinformatics (Oxford, England)**, v. 31, 09 2014. Citado na página 85.

AWS. **O que é python**. 2024. https://aws.amazon.com/pt/what-is/python/. Acesso em: 01 de julho de 2024. Citado na página 34.

AWS. **"Qual é a diferença entre imagens e contêineres do Docker?"**. 2024. https://aws.amazon.com/pt/compare/the-difference-between-docker-images-and-containers/. Acesso em: 28 junho 2024. Citado na página 35.

BOLGER, A. M.; LOHSE, M.; USADEL, B. Trimmomatic: a flexible trimmer for illumina sequence data. **Bioinformatics**, v. 30, n. 15, p. 2114–2120, Aug 2014. Disponível em: https://academic.oup.com/bioinformatics/article/30/15/2114/2390096. Citado nas páginas 56 e 85.

BROCKE, J. v.; HEVNER, A.; MAEDCHE, A. Introduction to design science research. **Design Science Research: Cases**, Springer, p. 1–13, 2020. Citado na página 42.

CELERY. **"Celery - Fila de tarefas distribuídas"**. 2024. https://docs.celeryq.dev/en/stable/. Acesso em: 23 junho 2024. Citado na página 33.

CHUNG, M. et al. Best practices on the differential expression analysis of multi-species rna-seq. **Genome Biology**, v. 22, n. 1, p. 121, abr 2021. Citado na página 39.

CIÊNCIAS, A. B. de. **Bruce alberts**. 2024. http://www.abc.org.br/membro/bruce-alberts/. Acesso em: 19 junho 2024. Citado na página 17.

CONNIE, R. et al. Biology. 2016. Citado nas páginas 22, 23 e 24.

CORCHETE, L. A. et al. Systematic comparison and assessment of rna-seq procedures for gene expression quantitative analysis. **Scientific Reports**, v. 10, p. 14088, 2020. Citado nas páginas 14, 28, 29, 39, 40, 41, 44 e 48.

CRICK, F. Cental dogma of molecular biology. **Nature, 227(5258), 561-563**, 1970. Citado na página 17.

DHARSHINI, S. A. P.; TAGUCHI, Y.-H.; GROMIHA, M. M. Identifying suitable tools for variant detection and differential gene expression using rna-seq data. **Genomics**, v. 112, n. 3, p. 2166–2172, 2020. ISSN 0888-7543. Disponível em: https://www.sciencedirect.com/science/article/pii/S0888754319307517. Citado na página 39.

DOBIN, A. et al. STAR: ultrafast universal RNA-seq aligner. **Bioinformatics**, v. 29, n. 1, p. 15–21, Jan 2013. Disponível em: https://doi.org/10.1093/bioinformatics/bts635. Citado nas páginas 56 e 85.

DOCKER, I. **"Get Docker"**. 2024. https://docs.docker.com/get-docker/. Acesso em: 28 junho 2024. Citado na página 35.

Docker, Inc. **Docker Overview**. 2024. Accessed: 2024-11-06. Disponível em: https://docs.docker.com/get-started/docker-overview/. Citado na página 16.

EDUCAçãO, G. 2024. https://g4educacao.com/portal/metodologia-scrum. Acesso em: 02 de julho de 2024. Citado na página 46.

FERREIRA, E. S. Plasticome: um método computacional para mineração em genomas fúngicos a fim de enconrar potenciais enzimas que degradam plástico. 2024. Citado na página 33.

INFORMATION, N. C. for B. **Gene Expression Omnibus**. 2024. https://www.ncbi.nlm.nih.gov/geo/. Acesso em: 16 junho 2024. Citado na página 27.

INFORMATION, N. C. for B. **Genomic Data Science**. 2024. https://www.ncbi.nlm.nih.gov/genbank/statistics/. Acesso em: 21 maio 2024. Citado na página 13.

JEONG, S. H. et al. Biomolecules as green flame retardants: Recent progress, challenges, and opportunities. **Journal of Cleaner Production**, v. 368, p. 133241, 2022. ISSN 0959-6526. Disponível em: https://www.sciencedirect.com/science/article/pii/S0959652622028281. Citado na página 17.

LATARETU, M.; HÖLZER, M. RNAflow: An effective and simple RNA-Seq differential gene expression pipeline using nextflow. **Genes (Basel)**, Switzerland, v. 11, n. 12, dez. 2020. Citado nas páginas 32 e 40.

LEINONEN, R. et al. The sequence read archive. **Nucleic Acids Research**, v. 39, n. Database issue, p. D19–D21, January 2011. Disponível em: https://doi.org/10.1093/nar/gkq1019. Citado na página 85.

LI, Y. et al. The different roles of penicillium oxalicum laea in the production of extracellular cellulase and -xylosidase. **Frontiers in Microbiology**, v. 7, 2016. ISSN 1664-302X. Disponível em: https://www.frontiersin.org/journals/microbiology/articles/10.3389/fmicb.2016.02091. Citado na página 31.

LI, Z. et al. Synergistic and dose-controlled regulation of cellulase gene expression in *Penicillium oxalicum*. **PLOS Genetics**, v. 11, n. 9, p. e1005509, 2015. Disponível em: https://doi.org/10.1371/journal.pgen.1005509. Citado nas páginas 51, 71, 72 e 76.

LIU, X. et al. A comparison of transcriptome analysis methods with reference genome. **BMC Genomics**, v. 23, n. 1, p. 232, mar. 2022. Citado nas páginas 14 e 40.

LUSCOMBE N. M., G. D. . G. M. What is bioinformatics? a proposed definition and overview of the field. **Methods of Information in Medicine, 40(4), 346-358**, 2001. Citado na página 13.

MARIANO, D. et al. **Introdução à Programação para Bioinformática com Biopython**. 3ª edição. ed. Belo Horizonte, Brasil: Amazon, 2016. Citado na página 34.

MARTIN, R. C. **Clean Architecture: A Craftsman's Guide to Software Structure and Design**. Boston, MA, USA: Prentice Hall, 2019. ISBN 978-0134494166. Citado nas páginas 35 e 36.

MARX, V. Biology: The big challenges of big data. **Nature, 498(7453), 255-260**, 2013. Citado na página 13.

MAZA, E. et al. Comparison of normalization methods for differential gene expression analysis in rna-seq experiments. **Communicative & Integrative Biology**, Taylor & Francis, v. 6, n. 6, p. e25849, 2013. PMID: 26442135. Disponível em: https://doi.org/10.4161/cib.25849. Citado na página 31.

MELO, H. V. F. Desenvolvimento e um pipeline para análise genômica e transcriptomica com base em web services. **Biblioteca comunitaria da UFSCar**, 2009. Citado na página 32.

MORAIS, W. d. **"Entendendo Task Queue com Django, Celery e RabbitMQ"**. 2022. https://dev.to/wesleymorais/entendendo-task-queue-com-django-celery-e-rabbitmq-174b. Acesso em: 27 junho 2024. Citado na página 33.

MUZELLEC, B. et al. Pydeseq2: a python package for bulk rna-seq differential expression analysis. **Bioinformatics**, v. 39, n. 9, p. btad547, 09 2023. ISSN 1367-4811. Disponível em: https://doi.org/10.1093/bioinformatics/btad547. Citado na página 86.

NEEDLEMAN S. B., . W. C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. **Journal of Molecular Biology**, 1970. Citado na página 13.

O'LEARY, N. A. et al. Exploring and retrieving sequence and metadata for species across the tree of life with ncbi datasets. **Scientific Data**, v. 11, n. 1, p. 732, Jul 2024. Disponível em: https://www.nature.com/articles/s41597-024-03571-y. Citado na página 56.

OLIVEIRA, P. W. de. Gerenciamento de proveniência de dados de workflows de bioinformática em ambiente de nuvens federadas. **Biblioteca comunitaria da UFSCar**, 2019. Citado nas páginas 32, 33 e 34.

PARVATHAREDDY, S. K. et al. Differential expression of PD-L1 between primary and metastatic epithelial ovarian cancer and its clinico-pathological correlation. **Sci. Rep.**, Springer Science and Business Media LLC, v. 11, n. 1, p. 3750, fev. 2021. Citado na página 30.

PAULING, L.; COREY, R. A proposed str ucture for the nucleic a cids. **Nature**, 1952. Citado na página 20.

RABBITMQ. **"Why RabbitMQ ?"**. 2024. https://www.rabbitmq.com/. Acesso em: 23 junho 2024. Citado na página 34.

RASCLE BASTIEN MALBERT, I. G. M. C. C. B. C.; POUSSEREAU, N. Transcriptomic changes in the pacc transcription factor deletion mutant of the plant pathogenic fungus *Botrytis cinerea* under acidic and neutral conditions. **BMC Genomics Data**, v. 25, p. 87, 2024. Disponível em: https://doi.org/10.1186/s12863-024-01257-3. Citado nas páginas 70, 71, 72 e 76.

RISSO, D. et al. Normalization of rna-seq data using factor analysis of control genes or samples. **Nature Biotechnology**, v. 32, p. 896–902, 2014. Disponível em: https://doi.org/10.1038/nbt.2931. Citado na página 29.

ROBINSON, M. D.; OSHLACK, A. A scaling normalization method for differential expression analysis of rna-seq data. **Genome Biology**, v. 11, p. R25, 2010. Disponível em: https://doi.org/10.1186/gb-2010-11-3-r25. Citado na página 30.

SANGKET, U. et al. bestDEG: a web-based application automatically combines various tools to precisely predict differentially expressed genes (DEGs) from RNA-Seq data. **PeerJ**, United States, v. 10, p. e14344, nov. 2022. Citado nas páginas 14, 25, 30 e 40.

SOUZA, R. B. A. Kingfungi: um pipeline para prospecção de cogumelos com capacidade de acúmulo de micronutrientes. 2024. Citado na página 33.

TEAM, J. D. **What is a Notebook?** 2024. Acesso em: 7 dez. 2024. Disponível em: https://docs.jupyter.org/en/latest/#what-is-a-notebook. Citado na página 72.

THAWNG, C. N.; SMITH, G. B. Transcriptome software results show significant variation among different commercial pipelines. **BMC Genomics**, v. 24, n. 1, p. 662, nov. 2023. Citado na página 40.

THEPSUWAN, T. et al. Long non-coding RNA profile in banana shrimp, fenneropenaeus merguiensis and the potential role of lncPV13 in vitellogenesis. **Comp. Biochem. Physiol. A Mol. Integr. Physiol.**, Elsevier BV, v. 261, n. 111045, p. 111045, nov. 2021. Citado na página 30.

WANG, D.; FARHANA, A. **Biochemistry, RNA Structure**. Treasure Island (FL): StatPearls Publishing, 2024. [Updated 2023 Jul 29]. Disponível em: https://www.ncbi.nlm.nih.gov/books/NBK558999/. Citado na página 17.

WANG ZHONG, G. M. S. M. RNA-Seq: a revolutionary tool for transcriptomics. **Nat. Rev. Genet.**, Springer Science and Business Media LLC, v. 10, n. 1, p. 57–63, jan. 2009. Citado nas páginas 25, 26 e 30.

WILLIAMS, C. R. et al. Trimming of sequence reads alters RNA-Seq gene expression estimates. **BMC Bioinformatics**, v. 17, n. 1, p. 103, fev. 2016. Citado na página 28.

YANG, C. et al. The impact of RNA-seq aligners on gene expression estimation. In: **Proceedings of the 6th ACM Conference on Bioinformatics, Computational Biology and Health Informatics**. New York, NY, USA: ACM, 2015. Citado na página 28.

YIN, S. et al. Mixnorm: normalizing rna-seq data from formalin-fixed paraffin-embedded samples. **Bioinformatics**, v. 36, n. 11, p. 3401–3408, June 2020. Disponível em: https://doi.org/10.1093/bioinformatics/btaa153. Citado na página 29.

YIN, S. et al. Smixnorm: Fast and accurate rna-seq data normalization for formalin-fixed paraffin-embedded samples. **Frontiers in Genetics**, v. 12, p. 650795, 2021. Disponível em: https://doi.org/10.3389/fgene.2021.650795. Citado na página 29.

ZHAO, Y. et al. Tpm, fpkm ou contagens normalizadas? um estudo comparativo de medidas de quantificação para a análise de dados de rna-seq do repositório de modelos derivados de pacientes do nci. **Journal of Translational Medicine**, v. 19, p. 269, 2021. Disponível em: https://doi.org/10.1186/s12967-021-02936-w. Citado na página 29.

ANEXO A – GLOSSÁRIO DE FERRAMENTAS

SRAToolkit: O SRA Toolkit é um conjunto de ferramentas desenvolvidas pelo NCBI para acessar e manipular dados de SRA. Ele permite o download de dados de sequenciamento depositados no repositório SRA, convertendo-os para formatos que podem ser utilizados em pipelines de bioinformática. Dois comandos desta ferramenta que foram utilizados neste trabalho são: prefetch que é responsável por fazer o download de arquivos do SRA para o armazenamento local, garantindo a disponibilidade dos dados para processamentos futuros e fastq-dump que realiza a conversão dos arquivos no formato SRA para o formato FASTQ, que é amplamente utilizado em pipelines de análise (Leinonen et al., 2011).

Trimmomatic: O Trimmomatic é uma ferramenta de bioinformática amplamente utilizada para realizar o pré-processamento de dados de sequenciamento de RNA-Seq. Sua principal função é a remoção de sequências de baixa qualidade e de adaptadores presentes nas leituras. Ele utiliza algoritmos específicos para identificar e cortar regiões com qualidade insuficiente, garantindo dados mais limpos para as etapas subsequentes. O Trimmomatic suporta dados em formatos FASTQ e é compatível com sequenciamentos de leitura curta, tanto em modos pareados quanto não pareados (Bolger; Lohse; Usadel, 2014).

RNA STAR: O RNA STAR (Spliced Transcripts Alignment to a Reference) é uma ferramenta de alinhamento de alta performance projetada para dados de RNA-Seq. Ela realiza o mapeamento das leituras de RNA sequenciado ao genoma de referência, identificando regiões de splicing com precisão. O STAR é conhecido por sua eficiência em lidar com grandes volumes de dados e por gerar resultados de alinhamento que incluem informações detalhadas sobre emparelhamento e áreas de divergência (Dobin et al., 2013).

HTSeq: O HTSeq é uma ferramenta desenvolvida para a contagem de reads alinhadas em nível de genes ou regiões genômicas. Ele é amplamente usado em pipelines de RNA-Seq para gerar contagens baseadas no alinhamento obtido em etapas anteriores. A contagem é feita comparando as coordenadas das reads alinhadas com as anotações do genoma de referência, garantindo a quantificação precisa da expressão gênica (Anders; Pyl; Huber, 2014).

DESeq2: O DESeq2 é um pacote estatístico para a análise de expressão diferencial em dados de RNA-Seq. Ele utiliza modelos lineares generalizados baseados em distribuição binomial negativa para identificar genes diferencialmente expressos entre diferentes condições experimentais. Além disso, o DESeq2 realiza normalização dos dados, considerando fatores como o tamanho das bibliotecas e a variabilidade entre amostras (Muzellec et al.,

2023).