Pipelines automatizados para genômica e transcritômica

Tese de Doutorado

Christiane Nishibe

Orientação: Prof. Dr. Nalvo Franco de Almeida Jr.

Área de concentração: Bioinformática

Faculdade de Computação
Universidade Federal de Mato Grosso do Sul

Campo Grande, 29 de julho de 2015.

Agradecimentos

À minha família por todo apoio, dedicação, incentivo e carinho constante.

Ao meu orientador, Prof. Nalvo Franco de Almeida Junior, pela oportunidade de aprender um pouco sobre bioinformática, pela dedicação, paciência e ensinamentos que tornaram possível a conclusão de mais essa etapa.

Ao Flábio R. Araújo e a Tainá Raiol Alencar por fornecerem os dados que motivaram a execução do trabalho.

À Heidi Muniz Silva por me ensinar um pouco desse mundo biológico.

Aos professores e funcionários da FACOM por toda ajuda e apoio, que direta ou indiretamente, contribuíram para minha formação e consequentemente para o desenvolvimento deste trabalho.

Aos amigos que mesmo distantes proporcionaram ao longo desses anos excelentes momentos de desconcentração, alegrias e viagens inesquecíveis.

À CAPES pelo apoio financeiro.

Resumo

Avanços das tecnologias de genômica e transcritômica resultaram no aumento expressivo do volume de sequências geradas, tornando indispensável o uso de técnicas e ferramentas computacionais para analisar os dados e produzir uma melhor caracterização e compreensão dos organismos estudados. Entretanto, o grande número de programas disponíveis e a variedade de formatos de dados produzidos dificultam a escolha do conjunto de *softwares* apropriados para cada análise. Este trabalho apresenta *pipelines* automatizados para análises genômicas e transcritômicas. O primeiro objetiva a construção de *contigs* de bactérias, a partir de *reads paired-end*, a serem posteriormente anotados e analisados por um conjunto de programas que investigam diferentes aspectos do genoma. Esse *pipeline* foi utilizado para um conjunto de cepas de interesse da bactéria *Mycobacterium bovis*. Os resultados gerados propiciaram relevantes avanços no grande objetivo do projeto de genômica de *M. bovis*, que é a erradicação da Tuberculose Bovina, doença causada por essa bactéria, incluindo a determinação de um método mais eficiente para o diagnóstico da doença. O segundo *pipeline* visa determinar genes diferencialmente expressos entre diferentes amostras que possuem replicatas, tendo como estudos de caso três projetos de transcritoma, dois envolvendo linfócitos T humanos e um envolvendo células de camundongo sob efeito de infecção causada por fungo. Ambos os *pipelines* foram desenvolvidos com o objetivo de minimizar a participação do usuário nos passos intermediários e também minimizar a exigência de experiência no uso dos pacotes e programas utilizados.

Abstract

Advances in genomic and transcriptomic technologies led to an expressive increasing of the volume of generated sequences, making the use of computational tools absolutely necessary in order to analyze the data and to produce a better characterization and a better understanding of the organisms of interest. However, the large number of available programs and the variety of data formats make the choice of those tools difficult. This work presents automated pipelines for genomic and transcriptomic analyses. The first one aims to build bacterial contigs from paired-end reads, to be annotated and analyzed by a set of programs that investigate distinct aspects of the genome. This pipeline has been used to a set of strains of interest of the bacteria *Mycobacterium bovis*. Results lead to important advances to the main goal of the *M. bovis* Genome Project, that is to eradicate Bovine Tuberculosis, disease caused by this bacteria, including a more efficient method to diagnoses. The second one aims to find differentially expressed genes among different samples with replicates, having as case studies three transcriptome projects, two of them involving human T lymphocytes and another involving mouse cells under fungal infection. Both pipelines were developed having in mind to minimize both user interference in intermediate steps and experience in using packages and software included in the pipelines.

Sumário

Agradecimentos **iii**

Resumo **v**

Abstract **vii**

1 Introdução **1**

2 Conceitos Básicos **5**

2.1 Biologia Molecular 5

2.1.1 Ácidos Nucleicos 6

2.1.2 Dogma Central da Biologia Molecular 8

2.2 Sequenciadores 15

2.2.1 Roche 454 15

2.2.2 Illumina 17

2.2.3 Ion Torrent 18

2.3 Genômica 20

2.3.1 Controle de Qualidade 22

2.3.2 Pré-processamento 22

2.3.3 Montagem e Mapeamento 22

2.3.4 Anotação 25

2.3.5 Análises 26

2.4 Transcritômica 30

2.4.1 Análise de Expressão Diferencial 31

3 *Pipeline* Automatizado para Genômica **39**

3.1 Visão Geral do *Pipeline* 39

3.2 Controle de Qualidade 41
3.3 Pré-processamento 42
3.4 Mapeamento 43
3.5 Obtenção dos *Contigs* 44
3.6 Preparação para a Anotação 45

4 *Pipeline* Automatizado para Análise de Expressão Diferencial 47
4.1 Visão Geral do *Pipeline* 47
4.2 Controle de Qualidade 50
4.3 Pré-processamento 50
4.4 Alinhamento 51
4.5 Contagem 52
4.6 Análise de Expressão Diferencial 53

5 Resultados Obtidos para Genômica 59
5.1 Genômica de *Mycobacterium bovis* 59
5.1.1 Sequenciamento e Filtragem 61
5.1.2 Montagem e Obtenção dos *Contigs* 61
5.1.3 Anotação 62
5.2 Análises 63

6 Resultados Obtidos para Transcritômica 75
6.1 Transcritoma de Pacientes Transplantados 75
6.2 Transcritoma de Linfócitos T tratados com Diferentes Anticorpos . . 80
6.3 Interação de Macrófagos Murinos e *Fonsecaea pedrosoi* 84

7 Conclusão 89

Referências Bibliográficas 93

A Pipeline User's Guide - Genomics 105

B Pipeline User's Guide - Differential Expression in Transcriptomics 113

Lista de Tabelas

2.1 Tabela de aminoácidos e seus códons correspondentes 14
2.2 Comparação dos principais sequenciadores utilizados atualmente . . . 19
2.3 Diferença na contagem das amostras 35

5.1 Número de *reads* obtidos em cada rodada de sequenciamento 61
5.2 Número de *contigs* obtidos em cada montagem 62
5.3 Número de elementos obtidos com a anotação feita pelo PGAP 63

6.1 Total de *reads* inicialmente sequenciados e depois da filtragem 77
6.2 Quantidade de genes diferencialmente expressos entre os grupos clínicos analisados 79
6.3 Quantidade de *reads* inicial e depois de filtrados em cada amostra . . 81
6.4 Quantidade de genes diferencialmente expressos resultantes da análise do perfil transcritômico de linfócitos T 83
6.5 Total de *reads* sequenciados a partir das diferentes formas do fungo *F. pedrosoi* 85
6.6 Total de *reads* após filtragem das diferentes formas do fungo *F. pedrosoi* 86
6.7 Quantidade de genes diferencialmente expressos resultantes da comparação de diferentes formas do fungo *F. pedrosoi* 87

Lista de Figuras

2.1 Representação do RNA, do DNA e das estruturas químicas das bases nitrogenadas 7
2.2 Representação do dogma central da biologia molecular 8
2.3 Esquema de replicação do DNA 9
2.4 Iniciação da transcrição 10
2.5 Fase de elongação da transcrição 11
2.6 Processamento do pré-mRNA 12
2.7 Estrutura de um ribossomo 13
2.8 Exemplo do formato FASTQ 20
2.9 Visão geral de um *pipeline* para anotação de genomas 21
2.10 Famílias encontradas na comparação dos genomas 1, 2, 3 e 4 29
2.11 Visão geral de um *pipeline* para análise de expressão diferencial . . . 32
2.12 Mapeamento de *spliced read* 33
2.13 Expressão de genes com diferentes tamanhos 34

3.1 Visão geral do *pipeline* genômico 40
3.2 Exemplo do arquivo de configuração `genomic_pipeline.conf` 41
3.3 Gráfico de qualidade gerado pelo FastQC 42

4.1 Visão geral do *pipeline* para análise de expressão diferencial 48
4.2 Exemplo do arquivo de configuração `pipeline_de.conf` 49
4.3 Exemplo de um gráfico MDS gerado pela função `plotMDS` 55
4.4 Exemplo de um gráfico MA gerado pela função `plotSmear` 56

5.1 Tela inicial do Orthologsorter para os dados de 30 cepas de *M. bovis* . 64
5.2 Uma das famílias encontradas na comparação das 30 cepas de *M. bovis* pelo OrthoMCL 65
5.3 Arquivo FASTA com as proteínas da família identificada como 3.657 . 65
5.4 Alinhamento da família identificada como 3.657 66
5.5 Árvore resultante do alinhamento obtido pelo Orthologsorter 67
5.6 Curvas de *pan/core-genome* obtida com 29 cepas de *M. bovis* 68

5.7 Curva de novos genes para a análise de 29 cepas de *M. bovis* 69
5.8 Tela inicial para a escolha do genoma âncora e os outros genomas a serem comparados 70
5.9 Região do genoma MtH37Rv contendo proteínas de membrana Mce e não presentes nos outros 4 genomas 70
5.10 Trecho inicial da matriz de distância genômica, calculada entre cada par dos 30 genomas de *M. bovis* 71
5.11 Gráfico resultante da comparação genômica dos genomas Mb026316S6USA e Mb081930USA 72
5.12 Árvore gerada com o uso do pacote Phylip 72
5.13 Regiões anômalas encontradas pelo programa Alien Hunter no genoma MbAN5BRA 73
5.14 Gene do genoma MbAN5BRA 73

6.1 Gráficos MDS para as comparações entre os grupos clínicos analisados 78
6.2 Gráficos MA gerados para a análise dos diferentes grupos clínicos . . 79
6.3 Gráficos MDS para a análise do perfil transcritômico de linfócitos T . 82
6.4 Gráficos MA para a análise do perfil transcritômico de linfócitos T . . 83
6.5 Gráficos MDS para a análise de diferentes formas do fungo *F. pedrosoi* 86
6.6 Gráficos MA para a análise de diferentes formas do fungo *F. pedrosoi* 87

A.1 Overview of the pipeline for genomic analysis 106

B.1 Overview of the pipeline for differential expression analysis 114

Lista de Siglas

A	Adenina
CCD	*Charged-Coupled Device*
C	Citosina
DNA	Ácido desoxirribonucleico
FDR	*False Discovery Rate*
FPKM	*Fragments Per Kilobase of transcript per Million mapped*
GFF	*General Feature Format*
GO	*Gene Ontology*
GTF	*General Transfer Format*
G	Guanina
HACA	*Human Antichimeric Antibody*
HAMA	*Human Anti-Mouse Antibody*
ISFET	*Ion Sensitive Field Effect Transistor*
ISP	*Ion Spheres Particles*
lncRNA	RNA não codificador longo
MDS	*Multiple Dimensional Scaling*
miRNA	Micro RNA
mRNA	RNA mensageiro

MUMs *Maximal Unique Matches*

NCBI *National Center for Biotechnology Information*

ncRNA RNA não codificador

NGS *Next-Generation Sequencing*

OLC *Overlap Layout Consensus*

ORF *Open Reading Frame*

PCR *Polymerase chain reaction*

PE *Paired-end*

PGAP *Prokaryotic Genome Annotation Pipeline*

PGM *Personal Genome Machine*

PPD *Purified Protein Derivative*

RNA-Seq *RNA Sequencing*

RNA Ácido ribonucleico

RPKM *Reads Per Kilobase of transcript per Million mapped*

rRNA RNA ribossomal

SAM *Sequence Alignment/Mapping*

SE *Single-end*

SFF *Standard Flowgram Format*

siRNA RNA de interferência pequeno

snoRNA RNA nucleolar pequeno

SNP *Single Nucleotide Polymorphism*

snRNA RNA nuclear pequeno

SSBP *Single Strand Binding Proteins*

TMM *Trimmed Means of M values*

tRNA	RNA transportador
T	Timina
U	Uracila

Capítulo 1

Introdução

O genoma de um organismo é formado por longas cadeias de ácido desoxirribonucleico (DNA). Essas moléculas contêm instruções para a construção e a manutenção das células. Para que estas instruções cumpram seu papel, o DNA precisa ser transcrito em moléculas de ácido ribonucleico (RNA) a partir de regiões transcricionalmente ativas, chamadas de genes. Existem diversos tipos de RNAs. O principal tipo, o RNA mensageiro (mRNA), desempenha um papel vital na síntese de proteínas, mas há também RNAs que não codificam proteínas, chamados de RNAs não codificadores (ncRNAs), como os RNAs transportadores (tRNAs), os RNAs ribossomais (rRNAs) e os micro RNAs (miRNAs), entre outros [115].

Os avanços da Biologia Molecular, em especial dos métodos de sequenciamento de DNA e RNA, tornaram possível a identificação da sequência de bases que formam esses ácidos, possiblitando o sequenciamento completo de animais, plantas e vírus, o resequenciamento de diversos organismos e o sequenciamento de RNAs, gerando avanços nas áreas da genômica comparativa, da metagenômica e da transcritômica [87].

A redução dos custos e do tempo de sequenciamento, bem como o aumento na qualidade e na quantidade dos dados produzidos, se comparado ao sequenciamento pelo método Sanger, utilizado durante muitos anos, são alguns dos fatores que possibilitaram avanços nessas áreas [109]. Por outro lado, isso também gerou a necessidade de desenvolver ferramentas computacionais mais eficientes e robustas para armazenar, analisar e gerenciar esses dados, visando a obtenção de informações biologicamente

relevantes. Essas análises envolvem várias etapas e para cada etapa há uma grande diversidade de programas que podem ser utilizados, dependendo do organismo analisado, do tipo de sequenciamento e do objetivo da análise, dificultando assim todo o processo.

Este trabalho insere-se nesse contexto na medida em que propõe *pipelines* automatizados que podem ser usados em dois tipos de projetos envolvendo o tratamento de dados biológicos: genômica e transcritômica. No contexto deste trabalho entende-se por *pipeline* uma sequência ordenada de tarefas tais que a execução de uma tarefa tem como entrada a saída da tarefa anterior na sequência.

Um projeto genoma, em especial de organismos procariotos, envolve três grandes etapas: sequenciamento e montagem, anotação e análise. O sequenciamento e a montagem consistem na determinação da sequência exata de nucleotídeos que forma o DNA de um determinado organismo. A anotação prediz a posição de cada gene no genoma, bem como sua função. A análise tenta obter caracterizações mais detalhadas e pode incluir a construção de árvores filogenéticas utilizadas na classificação de espécies, a comparação de diversos genomas a fim de entender a estrutura, a organização e a evolução dos mesmos.

O transcritoma de um organismo, ou de um tecido, ou ainda de uma célula, consiste no conjunto de todos os seus transcritos em uma determinada condição. Diferentemente do genoma, o transcritoma altera-se continuamente, tanto em termos qualitativos (o que está ou não sendo transcrito) quanto em termos quantitativos (com que intensidade determinados genes estão sendo transcritos) [15]. Dessa forma, a análise do transcritoma estuda o nível de expressão dos genes de maneira qualitativa, identificando os genes que são expressos e os que não são expressos em um determinado momento, e de forma quantitativa, medindo a variação de expressão dos diferentes genes em diferentes situações.

Por meio da comparação dos transcritomas de diferentes tipos de células, ou de células expostas a diferentes condições, é possível entender a constituição de um tipo específico de célula, como essa célula normalmente funciona e como mudanças no nível expressão do gene podem refletir ou contribuir para mudanças fenotípicas. Além disso, é possível ter uma visão completa dos genes que estão ativos em uma célula específica ou ainda revelar quantos mRNAs diferentes podem ser gerados a partir de um gene específico, pois alguns genes codificam múltiplos mRNAs [63].

A análise de expressão diferencial dos genes tem se tornado um importante instrumento de diagnóstico em certas áreas da Medicina. O exame do perfil de expressão gênica de um tumor canceroso, por exemplo, pode ajudar no diagnóstico do tipo de tumor, na determinação da probabilidade de metástase e na estratégia de tratamento mais eficaz [15]. Esse tipo de análise tornou-se muito popular a partir do uso de sequenciadores de alto desempenho para esse fim.

Essas análises, tanto de genômica quanto de transcritômica, envolvem várias etapas computacionais, que precisam lidar com diferentes formatos de arquivos, grandes volumes de dados, diferentes projetos, mas que possuem características comuns que se repetem. Por isso, foram desenvolvidos dois *pipelines* que utilizam um conjunto de ferramentas disponíveis para realizar as análises genômicas e transcritômicas de modo mais simples, ágil e automático.

Como motivação prática e estudo de caso, o *pipeline* de análise genômica foi utilizado na montagem e análise de 17 cepas de *Mycobacterium bovis*, em colaboração com a Embrapa Gado de Corte. O sequenciamento dessas cepas faz parte de um grande projeto de genômica de *M. bovis*, envolvendo o Intituto Nacional de Tecnologia Agropecuária da Argentina (INTA), o Departamento de Agricultura dos Estados Unidos (USDA), o Ministério da Agricultura Pecuária e Abastecimento do Brasil (MAPA), o Instituto Bioilógico de São Paulo (IB) e a Universidade Federal de Mato Grosso do Sul (UFMS).

Os resultados obtidos com esses genomas geraram as publicações [10, 11, 22, 84]. Uma publicação envolvendo a comparação dessas cepas está em preparação [12], além de outra, também em preparação, direcionada a aspectos relacionados a genes de virulência [23]. Os resultados gerados propiciaram relevantes avanços no grande objetivo do projeto de genômica de *M. bovis*, que é a erradicação da Tuberculose Bovina, doença causada por essa bactéria. O principal resultado apresentado em [10], por exemplo, consiste em um método mais eficiente para o diagnóstico da doença.

O *pipeline* de transcritômica, por sua vez, foi utilizado em três projetos do Laboratório de Biologia Molecular do Instituto de Biologia da UnB e envolve transcritomas humanos e de camundongos. Esses projetos encontram-se em fase final.

O texto está organizado da seguinte forma. O Capítulo 2 apresenta conceitos de Biologia Molecular, métodos de sequenciamento e uma visão geral das diversas etapas

envolvidas na análise genômica e na expressão diferencial de genes. No Capítulo 3 é apresentado o *pipeline* genômico para a análise de genomas bacterianos. Em seguida, no Capítulo 4, o *pipeline* automatizado para análise de expressão diferencial é descrito. No Capítulo 5, apresentamos os resultados obtidos na anotação de diferentes cepas de *M. bovis*. No Capítulo 6 apresentamos os resultados obtidos com a utilização do *pipeline* de expressão diferencial em três projetos de transcritomas diferentes. Por fim, no Capítulo 7, apresentamos as contribuições e algumas sugestões de trabalhos futuros. Os apêndices A e B trazem os manuais de utilização dos *pipelines* genômico e transcritômico, respectivamente.

Capítulo 2

Conceitos Básicos

Neste capítulo apresentamos conceitos relacionados à Biologia Molecular, necessários para o entendimento do trabalho. Conceitos básicos são apresentados na Seção 2.1. Alguns métodos de sequenciamento atualmente disponíveis no mercado são vistos na Seção 2.2. Além disso, descrevemos alguns aspectos sobre a análise genômica na Seção 2.3 e a análise de transcritos na Seção 2.4.

2.1 Biologia Molecular

A Biologia Molecular estuda os princípios moleculares que regem e regulam os processos biológicos. Estes processos biológicos envolvem a existência, interação e regulação de milhares de genes e suas proteínas correspondentes. Os conceitos de Biologia Molecular aqui apresentados, necessários para a leitura do texto, foram extraídos de [17, 20, 63, 115, 118].

A natureza hereditária de todo organismo vivo é definido por seu genoma, composto por longas sequências de ácido desoxirribonucleico (DNA), que fornecem toda a informação genética carregada pelo organismo em suas células. Por meio de uma complexa série de interações e regulações, o DNA codifica todas as proteínas do organismo em momentos e locais apropriados. As proteínas possuem diversas funções nos organismos. Elas fazem parte da estrutura dos organismos e realizam reações metabólicas necessárias à vida.

Fisicamente, o genoma pode estar dividido em várias moléculas de DNA ou cro-

mossomos. Funcionalmente, o genoma está dividido em genes. Cada gene é uma sequência do DNA que codifica diferentes tipos de ácido ribonucleico (RNA).

Apesar da similaridade existente entre as células que constituem os seres vivos, os organismos mantêm diferenças fundamentais em nível celular, podendo ser classificados em dois grandes grupos: os procariotos e os eucariotos. Os organismos procariotos são unicelulares e mais simples em sua organização, embora possam ocorrer associados a grupos, formando colônias com alguma diferenciação de funções. As células de organismos procariotos não possuem núcleo. Seu material genético está concentrado em uma região chamada **nucleóide**, mas não há uma membrana que separa essa região do restante da célula. Fazem parte desse grupo as bactérias e as arqueas.

Os organismos eucariotos são mais complexos e incluem não somente plantas pluricelulares, animais e fungos, mas também protozoários e alguns organismos unicelulares, como leveduras e algas verdes. As células eucariotas possuem um núcleo, delimitado por uma membrana nuclear e separado do citoplasma, que armazena o material genético. Ambos os organismos dividem importantes processos celulares e biológicos que são mediados por meio de ácidos nucleicos e proteínas.

2.1.1 Ácidos Nucleicos

Os ácidos nucleicos são moléculas de grande importância biológica e encontradas em todos os organismos vivos. A partir dos ácidos nucleicos as células recebem as informações sobre quais proteínas sintetizar, qual a sequência de aminoácidos de sua estrutura e qual a função dessas moléculas. Eles são, portanto, as moléculas que armazenam e transmitem a informação genética na célula. Toda essa informação fica em unidades gênicas, localizadas nos cromossomos das células. Tal informação é decifrada por meio do código genético, cuja tradução resulta na síntese proteica.

Existem dois tipos de ácidos nucleicos: o ácido desoxirribonucleico (DNA) e o ácido ribonucleico (RNA), mostrados na Figura 2.1. Ambos são macromoléculas formadas a partir de unidades mais simples e menores chamadas **nucleotídeos**.

Cada nucleotídeo é composto por um açúcar (pentose), um grupo fosfato e uma base nitrogenada. A molécula de açúcar é formada por cinco átomos de carbono (numerados de 1' até 5'). O grupo fosfato conecta o carbono 3' da molécula de

açúcar de um nucleotídeo com o carbono 5' da molécula de açúcar do próximo nucleotídeo, fomando uma cadeia ou fita. Essa fita possui uma orientação que inicia na extremidade 5' e acaba na extremidade 3'.

A base nitrogenada pode ser: adenina (`A`), citosina (`C`), guanina (`G`), timina (`T`) ou uracila (`U`). Adenina e guanina pertencem ao grupo das **purinas**, enquanto citosina, timina e uracila pertencem ao grupo das **pirimidinas**.

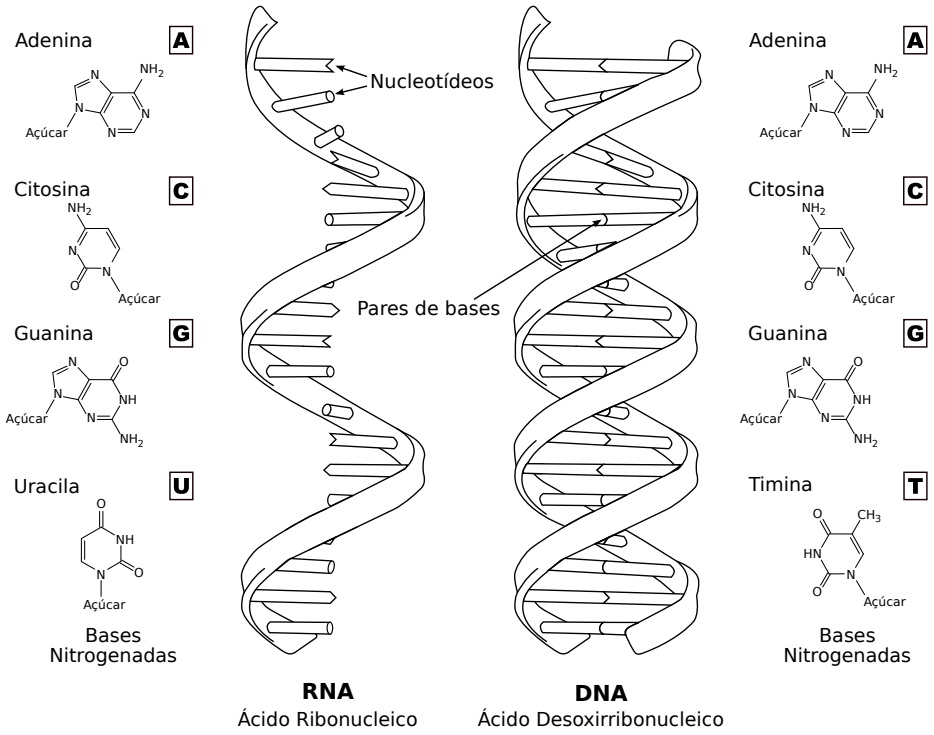


Figura 2.1: Representação do RNA, do DNA e das estruturas químicas das bases nitrogenadas. Figura adaptada de [20].

Quando o DNA forma sua dupla hélice, um nucleotídeo do grupo das purinas é ligado com um nucleotídeo do grupo das pirimidinas na outra fita. Elas são ligadas por pontes de hidrogênio formando pares de bases, sendo que adenina é pareada com timina (`A-T`) e citosina com guanina (`C-G`). Os pares `A-T` e `C-G` são denominados **pares de bases complementares**, ou simplesmente **pares de bases**.

Algumas diferenças entre os dois tipos de ácido nucleico residem no tipo de açúcar e na composição de bases nitrogenadas da molécula. No DNA a pentose é sempre a desoxirribose e, no RNA, a ribose. Adenina, citosina e guanina estão presentes tanto no DNA como no RNA, mas a timina ocorre apenas no DNA, enquanto que a uracila está presente apenas no RNA. Entretanto, a principal diferença entre DNA

e RNA é a presença de diferentes tipos de RNAs, classificados de acordo com suas localizações e suas funções na célula.

Além dos RNAs envolvidos diretamente na síntese de proteínas, detalhados na Seção 2.1.2, como o *messenger* RNA (RNA mensageiro - mRNA), *ribosomal* RNA (RNA ribossomal - rRNA) e *transfer* RNA (RNA transportador - tRNA), a descoberta de outras classes de RNA levou a uma classificação geral dos RNAs em codificadores e não codificadores. RNAs codificadores são exclusivamente os mRNAs que contêm a informação genética para a síntese de proteínas.

Os demais RNAs são denominados não codificadores (ncRNAs) e incluem o rRNA e o tRNA, além de outros, como *small nuclear* RNA (RNA nuclear pequeno - snRNA), *small nucleolar* RNA (RNA nucleolar pequeno - snoRNA), *micro* RNA (micro RNA - miRNA), *small interfering* RNA (RNA de interferência pequeno - siRNA) e *long noncoding* RNA (RNA não codificador longo - lncRNA).

2.1.2 Dogma Central da Biologia Molecular

O dogma central da biologia molecular (Figura 2.2) define como ocorre o fluxo da informação genética na célula. O DNA pode se replicar, dando origem a novas moléculas de DNA ou pode ser transcrito em um mRNA que, por sua vez, pode em seguida ser traduzido em proteína, ou ainda pode ser transcrito em um ncRNA, que tem sua função definida na célula.

Figura 2.2: Representação do Dogma Central da Biologia Molecular.

Replicação

A **replicação** é o processo no qual a molécula de DNA se duplica, originando duas moléculas idênticas à molécula inicial. Cada uma dessas moléculas recém formadas possui umas das fitas originais e outra proveniente dos novos nucleotídeos. Por essa razão a replicação do DNA é chamada de **semiconservativa**. A capacidade que cada célula possui de preservar seu material genético e transmití-lo para a próxima geração depende deste processo.

A replicação do DNA, mostrada de forma simplificada na Figura 2.3, inicia-se pela ação da enzima **helicase**, que realiza a abertura das fitas. Em seguida, proteínas denominadas **SSBP** (*Single Strand Binding Proteins*) ligam-se nas fitas recém desenroladas evitando que elas se liguem novamente. Em pontos adjacentes às regiões desenroladas há um superenrolamento que dificulta o processo de replicação. Sendo assim, as **topoisomerases** cortam uma das fitas de DNA, que se desenrola e diminui a tensão. A **DNA-polimerase** é a responsável pela síntese da nova fita, porém não é capaz de sintetizá-la a partir de nucleotídeos livres, por isso precisa de uma sequência curta de RNA denominada *primer* que é sintetizada por uma enzima chamada **primase**.

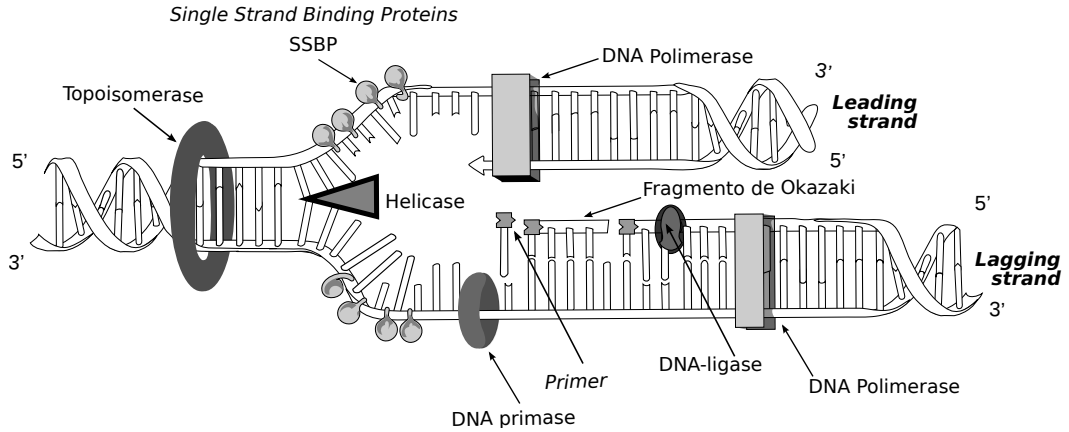


Figura 2.3: Esquema de replicação do DNA. Primeiro a helicase desenrola as fitas e proteínas denominadas SSBP se ligam a elas, mantendo-as abertas. As topoisomerases diminuem a tensão das fitas cortando uma delas. Como a DNA-polimerase forma a fita apenas no sentido 5'→ 3', a *leading strand* é elongada continuamente nesse sentido, enquanto a *lagging strand* é sintetizada como uma série de segmentos conhecidos como fragmentos de Okazaki, que posteriormente são juntados em uma única fita de DNA pela DNA-ligase. Figura adaptada de [20] e `https://en.wikipedia.org/wiki/DNA_replication`.

A DNA-polimerase forma a fita apenas no sentido 5'→ 3'. Uma das fitas, chamada ***leading strand***, é elongada continuamente no sentido 5'→ 3' enquanto a replicação ocorre. Mas a outra fita, denominada ***lagging strand***, é primeiro sintetizada como uma série de segmentos denominados **fragmentos de Okazaki**. Em seguida, uma enzima, a DNA-ligase, junta os fragmentos de Okazaki em uma única fita de DNA.

Transcrição

A **transcrição** é o processo de formação de uma fita de RNA denominada **RNA mensageiro** ou *messenger RNA* (mRNA), que carrega a informação genética do

DNA que será utilizada na síntese de proteínas da célula. O mRNA é uma cópia de uma das fitas de DNA, mas possui `U`s em vez de `T`s.

Uma enzima conhecida como **RNA polimerase** é responsável pela transcrição. As bactérias possuem um único tipo de RNA polimerase que sintetiza não apenas o mRNA mas também outros tipos de RNA que atuam na síntese de proteínas. Já os organismos eucariotos possuem três tipos de RNA polimerase em seu núcleo. Elas são numeradas como I, II e III, sendo a utilizada para a síntese do mRNA a RNA polimerase II.

A transcrição pode ser dividida em três estágios: iniciação, elongação e terminação. O processo inicia-se com a ligação da RNA polimerase a uma região do DNA denominada **promotor**. O promotor inclui o ponto de início da transcrição e tipicamente inicia-se algumas dezenas de nucleotídeos acima desse ponto de início. Além de determinar onde a transcrição começa, o promotor determina qual das fitas do DNA será usada como *template* ou molde.

Os promotores possuem regiões consenso localizadas a distâncias específicas e antes do ponto de início da transcrição, que são essenciais para a fixação da RNA polimerase. Essas regiões são conhecidas como **TATA *boxes***. Nos procariotos, a própria RNA polimerase reconhece e se liga ao promotor. Entretanto, nos eucariotos um conjunto de proteínas, chamadas **fatores de transcrição**, regulam a ligação da RNA polimerase e o início da transcrição. Apenas depois que certos fatores de transcrição estão ligados ao promotor é que a RNA polimerase se liga. O conjunto completo dos fatores de transcrição e a RNA polimerase ligada ao promotor é denominada **complexo de iniciação da transcrição**, conforme ilustrado na Figura 2.4.

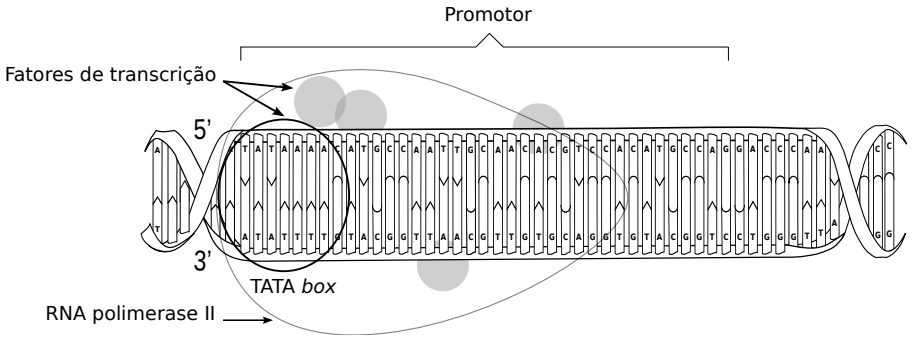


Figura 2.4: Iniciação da transcrição. Nos eucariotos, fatores de transcrição ligam-se ao promotor para que a RNA polimerase II reconheça e se ligue ao TATA *box*, formando o complexo de iniciação da transcrição. Figura adaptada de [20].

Depois que a polimerase está devidamente fixada ao promotor do DNA, as duas fitas se desenrolam, e a enzima começa a transcrever a fita molde.

Depois de iniciada a transcrição, a RNA polimerase move-se ao longo do DNA, desenrolando a dupla hélice e expondo cerca de 10 a 20 bases do DNA de cada vez para o pareamento com os nucleotídeos do RNA em formação. Enzimas adicionam nucleotídeos à extremidade 3' da molécula de RNA, alongando-a. O DNA já transcrito volta a se enrolar, recompondo sua estrutura em dupla hélice (Figura 2.5).

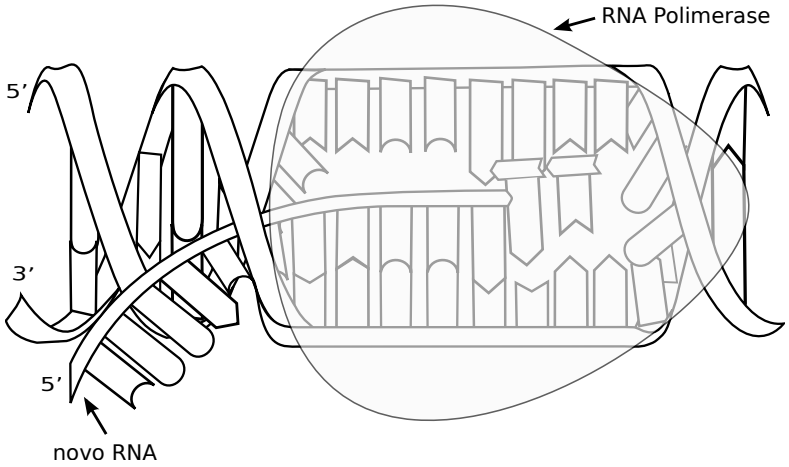


Figura 2.5: Fase de elongação da transcrição. Depois de estar devidamente ligada ao promotor, a RNA polimerase desenrola a dupla hélice do DNA e inicia a síntese do RNA. O DNA já transcrito recompõe sua dupla hélice. Figura adaptada de [20].

A transcrição continua até que a RNA polimerase transcreva uma sequência de DNA conhecida como **terminador**. Em seguida, o mRNA é liberado e a RNA polimerase dissocia-se do DNA.

Nos procariotos, o mRNA transcrito é imediatamente traduzido em uma sequência de aminoácidos de acordo com o código genético. Nos eucariotos, ao contrário, o mRNA recém liberado é conhecido como **pré-mRNA** e precisa sofrer várias modificações, como pode ser visto na Figura 2.6, antes de ser transportado para o citoplasma como mRNA, para ser traduzido.

A primeira modificação é o *capping* que ocorre na extremidade 5' do pré-mRNA, onde é adicionada uma forma modificada de guanina denominada **cap**. Depois, ocorre a poliadenilação na extremidade 3' do pré-mRNA, onde uma enzima adiciona uma cauda poli-A formada de 30 a 200 nucleotídeos de adenina. Essas duas

modificações protegem o mRNA de ser degradado e ajudam na fixação do ribossomo para a tradução. Outra etapa importante no processamento do pré-mRNA é um processo conhecido como **RNA *splicing***. Durante esse processo, regiões não codificadoras denominadas ***introns*** são removidas e as sequências codificadoras denominadas ***exons***, são conectadas. Embora a ordem dos *exons* sempre seja preservada, alguns *exons* podem ser removidos juntamente com os *introns*, originando diferentes RNAs. Este processo é chamado de ***splicing* alternativo** e permite a produção de diferentes proteínas a partir do mesmo gene. Cada *splicing* alternativo gera o que é chamado de **isoforma**.

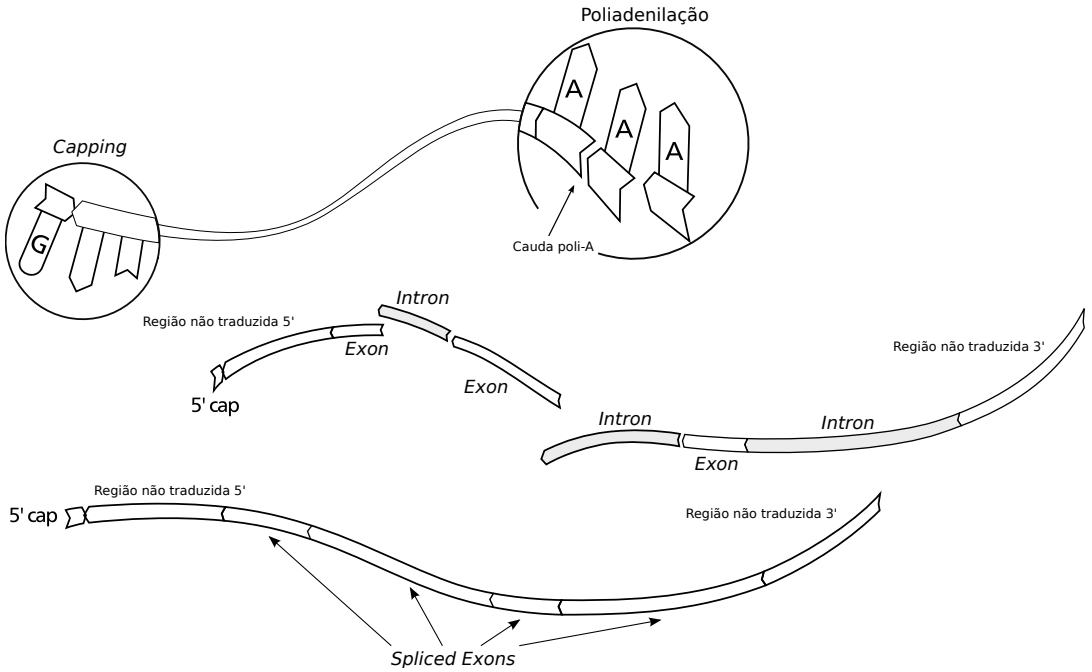


Figura 2.6: Processamento do pré-mRNA. Após ser transcrito, na extremidade 5' do pré-mRNA, ocorre o *capping* com a adição de uma guanina modificada. Em seguida, adiciona-se uma cauda poli-A na extremidade 3'. Por fim, ocorre o *splicing* com a remoção dos *introns* e junção dos *exons*. Figura adaptada de `http://en.wikipedia.org/wiki/Transcription_(genetics)`.

Tradução

Após a transcrição, o mRNA maduro deixa o núcleo através do poro nuclear, para ser traduzido no citoplasma. Durante a tradução ocorre uma mudança de linguagem. A célula deve traduzir trincas de nucleotídeos denominadas **códons** em aminoácidos que formarão a proteína. Essa tradução é sintetizada dentro de estruturas celulares chamadas **ribossomos** e com o auxílio dos tRNAs.

Os ribossomos são constituídos de proteínas e rRNAs e são divididos em duas subunidades independentes. Cada subunidade é chamada de maior e menor, devido a diferença na massa de cada uma. A subunidade maior possui três sítios denominados sítio E, sítio P e sítio A. O sítio A armazena o tRNA que carrega o próximo aminoácido que será adicionado a cadeia. No sítio P encontra-se o tRNA com a cadeia polipeptídica que está sendo construída e o sítio E é o local onde o tRNA sem o aminoácido deixa o ribossomo. O tRNA possui um anticódon que é complementar ao códon presente no mRNA. Preso ao final do tRNA encontra-se o aminoácido correspondente.

Na Figura 2.7 é possível ver uma versão simplificada da estrutura esquemática do ribossomo.

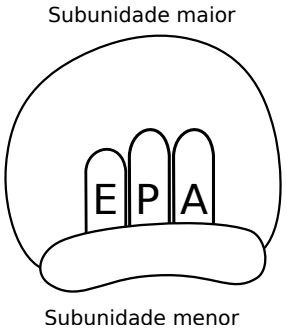


Figura 2.7: Estrutura de um ribossomo. O ribossomo é formado por duas subunidades independentes, maior e menor. A subunidade maior possui 3 sítios de ligação denominados E, P e A. Figura adaptada de [20].

A correspondência entre códons e aminoácidos é chamada de **código genético**, e pode ser vista na Tabela 2.1. Considerando que existem quatros bases distintas (`A`, `C`, `G` e `U`) e que cada códon é formado por três bases, espera-se um total de $4^3 = 64$ aminoácidos diferentes. Contudo, existem apenas vinte aminoácidos distintos, indicando que o código genético é degenerado, no sentido de que dois ou mais códons codificam o mesmo aminoácido. Além disso existem códons que não especificam aminoácido. Eles são denominados *stop* códons (`UAA`, `UAG`, e `UGA`) e determinam o fim da síntese de proteínas.

A tradução, por sua vez, é dividida em três etapas: iniciação, elongação e terminação. Na iniciação a subunidade menor do ribossomo liga-se ao mRNA e move-se até o códon `AUG` que indica o ponto de início da tradução. Em seguida, o tRNA liga-se a este códon carregando o aminoácido metionina. Por fim a subunidade maior une-se aos elementos anteriores formando o **complexo de iniciação** da tradução.

Tabela 2.1: Tabela de aminoácidos e seus códons correspondentes.

Aminoácidos	Código de 1 letra	Abreviatura	Códons correspondentes
Alanina	A	Ala	GCA, GCC, GCG, GCU
Arginina	R	Arg	AGA, AGG, CGA, CGC, CGG, CGU
Asparagina	N	Asn	AAC, AAU
Aspartato	D	Asp	GAC, GAU
Cisteína	C	Cys	UGC, UGU
Fenilalanina	F	Phe	UUC, UUU
Glicina	G	Gly	GGA, GGC, GGG, GGU
Glutamato	E	Glu	GAA, GAG
Glutamina	Q	Gln	CAA, CAG
Histidina	H	His	CAC, CAU
Isoleucina	I	Ile	AUA, AUC, AUU
Leucina	L	Leu	CUA, CUC, CUG, CUU, UUA, UUG
Lisina	K	Lys	AAA, AAG
Metionina	M	Met	AUG
Prolina	P	Pro	CCA, CCC, CCG, CCU
Serina	S	Ser	AGC, AGU, UCA, UCC, UCG, UCU
Tirosina	Y	Tyr	UAC, UAU
Treonina	T	Thr	ACA, ACC, ACG, ACU
Triptofano	W	Trp	UGG
Valina	V	Val	GUA, GUC, GUG, GUU

Com o complexo formado, inicia-se a etapa de elongação. Nesse momento, o primeiro tRNA ocupa o sítio P e o segundo tRNA entra no sítio A do ribossomo. Em seguida os rRNAs atuam como enzimas que ajudam a construir as ligações peptídicas entre os dois aminoácidos. O ribossomo então move-se um códon na direção 3', e neste momento o primeiro tRNA que se encontrava no sítio E é liberado. Enquanto o segundo tRNA encontra-se no sítio P, um novo tRNA pode entrar no sítio A, alinhando-se com o mRNA e construindo um nova ligação com o aminoácido do sítio P, alongando assim a cadeia de aminoácidos. O ribossomo repete esse processo até encontrar um *stop* códon no sítio A. Este códon não codifica aminoácido, mas age como um sinal para parar a tradução.

Quando o *stop* códon entra no sítio A, inicia-se a etapa de finalização. Nesse momento uma proteína chamada **fator de liberação** liga-se diretamente ao *stop* códon, adicionando uma molécula de água em vez de um aminoácido. Essa reação com o tRNA presente no sítio P libera o polipeptídio do ribossomo, terminando a tradução e dissociando as duas partes do ribossomo.

2.2 Sequenciadores

Desde a descoberta da estrutura helicoidal do DNA, determinar a sequência de nucleotídeos do DNA tem sido essencial para a compreensão dos diferentes organismos, o que levou ao desenvolvimento de diferentes técnicas de sequenciamento.

A técnica desenvolvida por Sanger e colaboradores [96] em 1977 foi o método de sequenciamento de DNA utilizado nos 30 anos seguintes, servindo de base para a área da genômica. Mas apesar da evolução obtida, a necessidade por maior eficiência e menores custos resultaram no desenvolvimento dos chamados sequenciadores de nova geração (*Next-Generation Sequencing* – NGS). Estes novos sequenciadores compartilham três grandes melhorias. Primeiro, não é necessário fazer a clonagem bacteriana dos fragmentos de DNA. Segundo, milhares de sequências são produzidas em paralelo. Terceiro, a detecção das sequências não utiliza eletroforese em gel; a determinação de uma sequência ocorre ciclicamente e em paralelo [109].

Dentre as plataformas NGS mais utilizadas na última década estão: Roche 454 [77], Illumina [45, 108] e Ion Torrent [94].

Embora cada plataforma NGS possua características que a torna diferente das demais, essas plataformas compartilham vários atributos. Primeiro, o preparo inicial das amostras, que consiste em fragmentar o DNA e ligar adaptadores (específicos para cada plataforma) às extremidades de cada fragmento que será sequenciado. Em seguida, todas as plataformas utilizam diferentes superfícies sólidas para realizar a amplificação, por meio de reações que produzem várias cópias de cada fragmento. Outro aspecto em comum é que o sequenciamento ocorre em ciclos, gerando fragmentos de diferentes tamanhos e em diferentes quantidades. Esses fragmentos são denominados ***reads***.

2.2.1 Roche 454

Em 2005 foi apresentada a primeira plataforma NGS a ser comercializada, o *Roche's 454 Life Sciences sequencing systems* [77]. Tal plataforma utiliza um método de pirosequenciamento [92, 93]. Nesse método, a ordem dos nucleotídeos é feita com base na detecção de uma luz produzida quando o nucleotídeo é incorporado.

O processo de sequenciamento inicia-se com o isolamento e fragmentação do DNA

genômico que será analisado. Em seguida, diferentes adaptadores são ligados às extremidades 5' e 3' de cada fragmento e são separados em fita simples. Os fragmentos são ligados a microesferas conhecidas como ***beads***. Essa ligação é feita sob condições que favorecem a junção de um fragmento por *bead*, as *beads* são capturadas por gotículas de uma emulsão de PCR contendo água e óleo. Dentro de cada gotícula ocorre a amplificação, resultando em *beads* que carregam milhares de cópias de um único DNA molde. A emulsão é quebrada e *beads* que carregam fitas simples de DNA são depositados em poços de uma placa, onde também são adicionados reagentes para o sequenciamento.

O sequenciamento é realizado em ciclos, e a cada ciclo um determinado nucleotídeo é adicionado à reação. Se o nucleotídeo adicionado for incorporado à sequência, há a liberação de um pirofosfato que em seguida, inicia uma série de reações químicas e cujo produto final é um sinal de luz que é capturado pela câmera CCD (*Charged-coupled device*) do equipamento. A intensidade do sinal determina o número de nucleotídeos que foram incorporados na sequência. A intensidade do sinal e a informação de cada nucleotídeo adicionado em cada ciclo determina a sequência molde.

Um sequenciador 454 pode gerar, em aproximadamente 23 h, cerca de 700 mil *reads* de até 500 bases cada. O maior comprimento desses *reads* facilita seu mapeamento em um genoma de referência. Além de ser uma vantagem para montagens *de novo* de genomas ou em aplicações de metagenômica [109].

A montagem *de novo* consiste em formar sequências maiores a partir da sobreposição dos *reads* produzidos, enquanto que o mapeamento de um *read* em um genoma de referência consiste em encontrar o local mais semelhante entre o *read* e o genoma. Tanto o mapeamento em um genoma de referência quanto a montagem *de novo* tentam, a partir dos *reads*, permitem reconstruir o genoma que os originaram. Essas duas técnicas serão discutidas na Seção 2.3.3.

Apesar de produzir *reads* considerados maiores, o sequenciamento feito na plataforma 454 possui algumas desvantagens, como o alto custo dos reagentes e uma alta taxa de erros de repetições de homopolímeros[1] [109].

[1] um mesmo nucleotídeo

2.2.2 Illumina

A plataforma Illumina surgiu em 2006 a partir do trabalho de Turcatti e colaboradores [45, 108] e a junção de quatro empresas: Solexa, Lynx Therapeutics, Manteia Predictive Medicine e Illumina. Essa tecnologia utiliza o sequenciamento por síntese, que emprega nucleotídeos com uma capacidade de terminação reversível e marcados com diferentes fluoróforos.

Na plataforma Illumina o DNA é fragmentado aleatoriamente e cada fragmento recebe dois tipos diferentes de adaptadores. Em seguida, os fragmentos são separados e fixados a uma superfície de vidro denominada *flow cell*. Esta superfície possui diversas *lanes*. Cada *lane* é revestida com dois tipos de oligonucleotídeos que são complementares aos adaptadores presentes em cada fragmento.

A amplificação das sequências ocorre por meio de uma técnica chamada *bridge* PCR [3, 45]. Nesse método, o fragmento fixado na placa dobra-se e liga-se ao segundo tipo de oligonucleotídeo fixo na placa. A polimerase gera uma fita complementar, formando uma "ponte" de fita dupla. A ponte é desnaturada, o que resulta em duas fitas simples. Esse processo é repetido diversas vezes, formando milhares de cópias em diversos *clusters*.

Depois que os *clusters* estão formados, o sequenciamento por síntese é iniciado. Nesse método, a cada ciclo, quatro nucleotídeos fluorescentes competem para serem adicionados à cadeia que está sendo formada, mas apenas um nucleotídeo é incorporado de acordo com a sequência do molde. Depois da adição de cada nucleotídeo, os *clusters* são excitados por uma fonte de luz e um sinal fluorescente é emitido e capturado por uma câmera CCD. O número de ciclos determina o comprimento do *read*. O comprimento da onda e a intensidade do sinal determinam a base que foi incorporada.

A Illumina atualmente é líder na indústria de NGS e a maioria dos protocolos de produção de bibliotecas são compatíveis com seu sistema. Além disso, oferece maior vazão e o menor custo por base. Um ponto negativo é o balanceamento da amostra. Devido à dispersão aleatória dos *clusters* no *flow cell*, a concentração da amostra deve ser rigorosamente controlada. Uma sobrecarga resulta na sobreposição de *clusters* e na baixa qualidade das sequências.

Outra característica do sequenciador Illumina é a possibilidade de escolher entre

sequenciamento *single-end* (SE) ou *paired-end* (PE). Para realizar o sequenciamento, o DNA é fragmentado e em seguida ligado a adaptadores. Se o sequenciamento for SE, significa que apenas uma extremidade do fragmento será sequenciado. Mas se for PE, as duas extremidade do mesmo fragmento são sequenciados, originando dois *reads*. Além disso, sabe-se a distância aproximada entre cada *read*. Essa distância é útil no mapeamento do *read* no genoma de referência, pois ajuda a determinar a posição correta do *read*, podendo revelar novas junções de *exons* ou ainda identificar a região repetitiva do genoma. Os dois *reads* obtidos pelo sequenciamento PE são referenciados como R1 e R2.

Na montagem *de novo* a distância entre os fragmentos também é uma informação útil, no entanto, os métodos de montagem que utilizam *reads paired-end* geralmente ignoram essa informação na primeira fase da montagem, onde os *contigs* são gerados, mas utilizam esse fato para construir *scaffolds* [100]. Os *scaffolds* são *contigs* em uma determinada ordem e orientação separados por buracos (*gaps*). Os *reads paired-end* também são usados para resolver pequenas repetições [116].

A Illumina possui diversos modelos de sequenciadores capazes de produzir uma quantidade variável de *reads*. O tamanho dos *reads* varia entre 50 bp (SE) até 2 x 250 bp (PE) e é possível produzir de 1 até 150 milhões de *reads*. Tudo isso em um tempo que varia de 6 horas até 12 dias.

2.2.3 Ion Torrent

Em 2010, a Ion Torrent Systems, que depois foi adquirida pela *Life Technologies*, lançou o *Personal Genome Machine* (PGM). Um sequenciador que utiliza uma tecnologia de semicondutores para implementar uma nova abordagem para a detecção de nucleotídeos. Nessa abordagem o sequenciador detecta os íons de hidrogênio liberados durante a incorporação do nucleotídeo, em vez de utilizar rótulos radioativos ou fluorescentes [94].

Esse método não utiliza nucleotídeos modificados, equipamentos ópticos nem enzimas especiais. Em vez disso utiliza um *Ion Sensitive Field Effect Transistor* (ISFET) para detectar a liberação do íon de hidrogênio. O ISFET funciona como um sensor de íons hipersensível, que mede a concentração de íons H+. Alterações na concentração de íons H+ altera proporcionalmente a corrente que passa através do ISFET.

Assim como as demais plataformas, para o sequenciamento com o Ion Torrent o DNA é fragmentado e cada fragmento recebe adaptadores em suas extremidades. De maneira similar ao que ocorre com a tecnologia 454, os fragmentos são ligados a *beads* ou *Ion Spheres Particles* (ISP) para serem amplificadas.

Cada fragmento é fixado em uma *bead* e copiados milhares de vezes. As *beads* são depositadas em um *chip* semicondutor, cuja superfície possui milhares de poços, cada um com capacidade para receber apenas uma *bead*.

No Ion Torrent o sequenciamento também é realizado em ciclos. Em cada ciclo apenas um tipo de nucleotídeo é adicionado à reação.

Quando um nucleotídeo é incorporado na molécula de DNA pela polimerase, um íon (H+) é liberado e induz uma alteração de pH no poço e o sensor ISFET reconhece que um nucleotídeo foi adicionado. A cada momento o *chip* é inundado com um nucleotídeo após o outro, se não for o nucleotídeo correto, não há alteração na voltagem e se dois nucleotídeos são incorporados, haverá a detecção do dobro da voltagem.

A vantagem dessa tecnologia é que ela não precisa de digitalização óptica, nem de nucleotídeos fluorescentes; as corridas são rápidas e podem gerar até 1 Gb de dados brutos. Mas assim como o sequenciador 454, possui uma taxa de erros elevada em repetições de nucleotídeos [109].

A Tabela 2.2 mostra os principais sequenciadores utilizados atualmente. Nela podemos observar o tempo que cada um dos equipamentos gasta para gerar uma determinada quantidade de *reads* de um tamanho específico.

Tabela 2.2: Comparação entre os principais sequenciadores utilizados atualmente. Para cada plataforma é mostrado o tamanho médio de cada fragmento, o número de fragmentos produzidos e tempo necessário para produzi-los.

Sequenciador	Tamanho	Quantidade	Tempo
Roche 454	500	700 mil	23 h
Illumina	50 - 250	1 - 150 milhões	6 h - 12 dias
Ion Torrent	35 - 400	1 G	3 - 8 h

Qualidade dos Fragmentos

O formato FASTQ [27] é utilizado para representar a qualidade dos *reads* gerados pelos diferentes sequenciadores. Nesse formato, cada *read* sequenciado é representado por quatro linhas e pode ser visto na Figura 2.8. A primeira linha inicia-se com o caractere "@" e é seguido pelo identificador da sequência e uma descrição opcional. Na segunda linha, encontra-se a sequência. A terceira linha inicia-se com o caractere "+" e opcionalmente seguido pelo mesmo identificador da linha 1. A linha 4 contém a qualidade da sequência, medida por meio do Phred *quality score* [42, 43].

```
@HISEQ:294:D2BLKACXX:2:1101:2804:1917 1:N:0:CGATGT
TATCCTTCTCTGAGACAGCAGTGAAGGTCATGGCGTGGGTCATAAGTGACTCACCAAAAGTCAGCCTCTCCGCTTTATTCATGTTCTTCAAGGAGACACC
+HISEQ:294:D2BLKACXX:2:1101:2804:1917 1:N:0:CGATGT
CCCFFFFFHHHHHJGIJJJJJJJIHIJGIJJJJGIIJJJFIIJJJJHIIIJJJJJJJJJIJ@GIGJJHHHHFFFDDBEDDEEEDDEDDDDEDDDDDDDCDD
@HISEQ:294:D2BLKACXX:2:1101:3050:1951 1:N:0:CGATGT
GCCAGATCCTTTGATAATAAATACTAAAGAATACCATTTGTCCATACAGGGGAGCTACCTTATAAACACCAAATATAGTCACATGTTACAAAAACAAAAA
+HISEQ:294:D2BLKACXX:2:1101:3050:1951 1:N:0:CGATGT
CCCFFFFDHHHHHIJJJJJJJJJJJJJJJIJIJJJIJJJJJJJJJJJJJJJJJGIIJJJJJJJJJJJJJJJIJHGHHHFFCFFFFEEEEFEEDDDDDDDDDD
```

Figura 2.8: Exemplo do formato FASTQ. Pedaço de um arquivo no formato FASTQ gerado por um sequenciador Illumina e composto por dois *reads* de 100 bases cada.

O Phred *quality score* está associado à probabilidade de erro na leitura de cada nucleotídeo. Uma base com qualidade 10 indica a probabilidade de ocorrer 1 erro a cada 100 bases, já em um Phred *score* de 40 a possibilidade é de ocorrer 1 erro a cada 10.000 bases. Desse modo, quanto maior o Phred *quality score*, maior a qualidade daquela base. A fórmula utilizada para calcular a qualidade é definida como

$$Q = -10 \log_{10} P,$$

onde Q é o Phred *quality score* e P é a probabilidade da base ter sido identificada incorretamente.

2.3 Genômica

Durante a última década, muitas melhorias foram feitas em relação à velocidade, tamanho e quantidade de *reads*, além da redução do custo do sequenciamento [109]. Essas características resultaram em um aumento no volume de dados e a necessidade de diversas ferramentas computacionais especializadas para processar e analisar tais dados. Essas melhorias permitiram também o desenvolvimento de diversas aplicações. Entre as principais aplicações destacam-se o sequenciamento e ressequencia-

mento genômico de diferentes organismos, a análise de expressão gênica e os estudos metagenômicos.

As análises genômicas têm como objetivo estudar a estrutura, organização e evolução dos genomas e também as funções dos genes e regiões não codificadoras. O processo de análise envolve muitas etapas e o uso de muitos *softwares* desde o sequenciamento dos *reads* até as análises envolvendo as informações funcionais dos organismos estudados.

Na Figura 2.9 são mostradas as etapas necessárias até se obter um genoma anotado, que depois é utilizado nas mais diversas análises. Geralmente, na primeira etapa os dados são avaliados, e em seguida filtrados. Na etapa seguinte ocorre a montagem, seguida da obtenção dos *contigs* e por fim sua anotação. Mais detalhes acerca de cada etapa serão vistos a seguir.

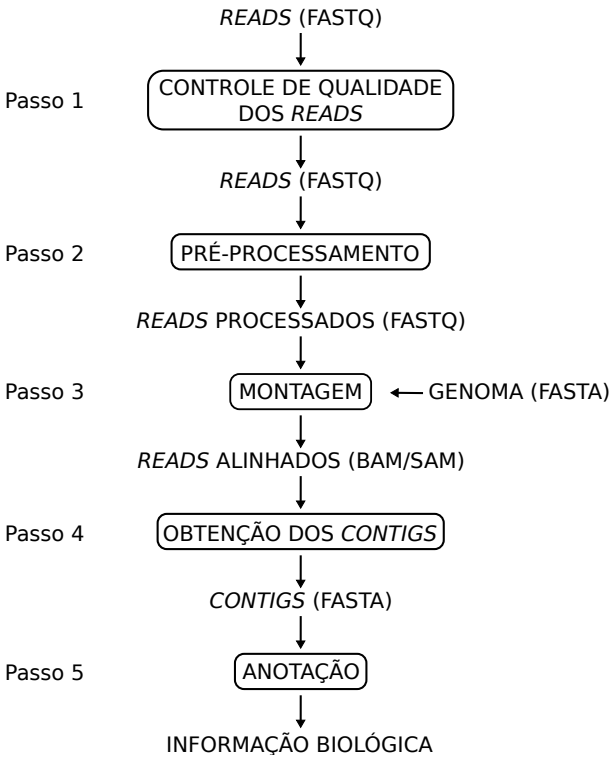


Figura 2.9: Visão geral de um *pipeline* para anotação de genomas. No Passo 1, a qualidade dos *reads* é avaliada. Conforme o resultado, os dados com baixa qualidade ou muito curtos são removidos na fase de pré-processamento. Em seguida, os *reads* são alinhados a um genoma de referência. A partir do resultado do alinhamento, no Passo 4, são obtidos os *contigs* que serão utilizados na etapa de anotação. Entre parênteses estão os formatos de arquivos produzidos em cada etapa.

2.3.1 Controle de Qualidade

Nessa primeira etapa, é verificada a qualidade de cada *read*, pela busca de bases de baixa qualidade, possíveis nucleotídeos incorretos, adaptadores, duplicações, entre outros. A saída desse passo é um relatório contendo o número de *reads* e informações de qualidade que auxiliam nas decisões da próxima etapa. *Softwares* como FastQC [8] e o PRINSEQ [97] são opções disponíveis para verificar a qualidade dos *reads* gerados.

2.3.2 Pré-processamento

Os dados produzidos pelos sequenciadores apresentam ruídos, como sequências com baixa qualidade, contaminações e adaptadores, e removê-los antes de realizar o mapeamento reduz o conjunto de *reads*, economizando recursos computacionais e aumentando a confiabilidade dos dados.

O processo de limpeza dos *reads* consiste em remover bases de baixa qualidade (*trimming*) e cortar adaptadores de sequeciamento e possíveis contaminantes (*clipping*).

A remoção das bases de baixa qualidade pode ocorrer nas duas extremidades dos *reads*. E às vezes o *read* inteiro é descartado. A remoção de contaminantes é necessária pois, em alguns casos, durante o protocolo de preparo da amostra, os *reads* permanecem com as sequências adaptadoras ou contaminação. Dependendo do tipo de sequenciamento ou da aplicação, essa etapa é ignorada ou é limitada apenas a um controle de qualidade básico, pois em todo processo de limpeza sempre existe a possibilidade de se perder informação.

FastQC [8], FastX-Toolkit [1], PRINSEQ [97], TagCleaner [98], Cutadapt [78], Trimmomatic [14] e SeqyClean [2] são alguns programas disponíveis para filtrar o conjunto de *reads* produzidos.

2.3.3 Montagem e Mapeamento

O sequenciamento de genomas é feito de maneira aleatória e geram sequências relativamente curtas, o que resulta na necessidade de um processo de montagem dos fragmentos para obter a sequência do DNA. O processo de montagem consiste na sobreposição dos *reads* a fim de formar trechos contínuos de DNA, denominados *contigs*.

Apesar da grande quantidade e qualidade dos *reads* produzidos pelas diferentes plataformas de sequenciamento, é preciso considerar os seguintes aspectos: os *reads* são pequenos e há milhões deles. Além disso, há erros de sequenciamento e há regiões repetitivas nos genomas. Todas essas características dificultam o processo de montagem e requerem o desenvolvimento de algoritmos e ferramentas computacionais mais eficientes para analisar esse grande volume de dados.

Se existir algum organismo muito semelhante ao genoma estudado, este pode ser usado como referência. Dessa maneira os *reads* podem ser mapeados nesse organismo. Neste caso a montagem é denominada montagem por referência, mapeamento ou alinhamento, ou ainda montagem guiada. O genoma de referência deve ser muito próximo filogeneticamente do genoma estudado.

Na ausência de um genoma de referência, a estratégia utilizada é conhecida como montagem *de novo*. Nesse caso, a montagem não toma qualquer informação acerca do genoma estudado e monta os *contigs* somente com base na sobreposição dos fragmentos.

MAQ [66], Stampy [75], BWA [64], Bowtie [60], MapSplice [111], SpliceMap [13], TopHat [105], GSNAP [114] e QPALMA [35] são alguns dos programas utilizados no mapeamento dos *reads* em genoma de referência. Esses programas utilizam estratégias como tabelas de dispersão e *Burrows-Wheeler Transform* (BWT) [19] para resolver o problema de modo eficiente levando em consideração os aspectos já citados.

Uma tabela de dispersão ou *hash table* [31] é uma estrutura de dados que cria índices para dados complexos associando chaves de pesquisa a valores, com o objetivo de realizar buscas de maneira rápida e eficiente. Os programas de mapeamento podem criar esses índices tanto no genoma de referência quanto nos *reads* de entrada, considerando todas as subsequências de um certo tamanho k, também chamado de k-*mers*, contidas na sequência. A chave da tabela de dispersão é um k-*mer*, enquanto o valor é uma lista de todas as posições onde o k-*mer* pode ser encontrado na referência.

O BWT é uma técnica de permutação reversível de sequências, inicialmente desenvolvida para a compressão de dados, que juntamente com estruturas de dados auxiliares pode criar índices de um genoma gastando pouco espaço. Esse índices são

conhecidos como *FM-index* [47]. O *FM-index* pode ser usado para buscar sequências num domínio reduzido de subsequências, sem a necessidade de procurar no genoma inteiro.

Geralmente, os montadores *de novo* utilizam dois algoritmos principais: *Overlap Layout Consensus* (OLC) e grafo de Bruijin [29], que são implementados em programas como ABySS [101], MIRA [26], SSAKE [113], VCAKE [54], Velvet [116] e Trinity [50].

A abordagem OLC possui três etapas. Primeiro, é realizada uma comparação par a par entre todos os *reads* para identificar os que se sobrepõem. Em seguida, com base na sobreposição encontrada no passo anterior, grafos de sobreposição são construídos. Por fim, os *contigs* são obtidos por caminhos no grafo de sobreposição.

Normalmente a montagem baseada no grafo de Bruijin possui duas etapas. Primeiro escolhe-se o tamanho do *k-mer* e divide-se os *reads* em *k-mers*. Depois com todos os *k-mers*, constrói-se um grafo de Bruijin, a partir do qual encontra-se a sequência genômica.

O grafo de Bruijin pode ser construído para qualquer sequência. O primeiro passo é escolher o tamanho do *k-mer* e separar a sequência original em *k-mers*. Depois é construído um grafo orientado conectando pares de *k-mers* que possuem uma sobreposição entre os últimos $k-1$ nucleotídeos de um *k-mer* e os primeiros $k-1$ nucleotídeos de outro *k-mer*. A sequência original é obtida calculando-se um caminho no grafo que inclui todos os *k-mers* uma única vez.

Independentemente do método utilizado, um dos critérios para avaliar a qualidade da montagem é uma medida conhecida como `N50`. O `N50` é calculado ordenando-se todos os *contigs* do maior para o menor. Em seguida, soma-se o tamanho de cada *contig* até que a soma atinja 50% do tamanho total da montagem. O `N50` é o tamanho do último *contig* que foi incluído na soma. Quanto maior o `N50`, melhor é a montagem. Note que o `N50` é calculado sobre o tamanho da montagem, não o tamanho do genoma de referência.

Outra medida de qualidade da montagem é a **cobertura genômica**, definida como o número de vezes que uma determinada região do genoma é coberta pelos *reads*. Uma cobertura maior aumenta a precisão da montagem.

A cobertura de um *contig* é definida pelo total de bases que o representa dividido pelo seu tamanho. Logo a cobertura de uma montagem é a soma da cobertura de todos os *contigs* dividido pelo número de *contigs* e representa quantas vezes, em média, cada base da montagem foi coberta.

Há também a cobertura de sequenciamento usada para estimar quantas vezes, em média, cada base do genoma foi sequenciada. A **cobertura de sequenciamento** é definida pelo total de bases sequenciadas dividido pelo tamanho do genoma de interesse, ou ainda, definida como o número de *reads* vezes o tamanho de cada *read* dividido pelo tamanho do genoma.

2.3.4 Anotação

A anotação do genoma consiste em localizar todos os genes dentro do genoma, incluindo genes que codificam proteínas, pseudogenes e uma variedade de genes não codificadores de proteínas, além de descobrir sua provável função e identificar as vias metabólicas e processos nos quais os genes estão relacionados.

A anotação envolve várias etapas, mas pode ser dividida em três categorias: anotação em nível de nucleotídeos, anotação em nível de proteínas e anotação em nível de processos biológicos [103].

A anotação em nível de nucleotídeos detecta genes codificadores e não codificadores de proteínas, regiões repetitivas e marcadores genéticos conhecidos.

Os genomas de bactérias e de arqueas possuem uma alta densidade de genes e pequenas regiões intergênicas adicionais. Normalmente, possuem genomas circulares e há cerca de um gene em cada 1.000 bases de DNA genômico. Essas características dos procariotos tornam a identificação dos genes uma tarefa mais simples e consiste em encontrar longas ORFs (*Open Reading Frames*). Uma **ORF** é uma subsequência de DNA cujo comprimento é múltiplo de 3, começa com um *start* códon (`ATG` por exemplo) e termina imediatamente antes de um *stop* códon (`TAA`, `TAG`, `TGA`). Programas como GLIMMER [95] e GenMark [74] eficientemente encontram genes em bactérias [87].

Em contraste com os genomas procariotos, os genomas eucariotos contêm uma porcentagem pequena de genes e enormes quantidades de regiões não codificadoras. Este material não codificante inclui regiões repetitivas, genes que têm funções re-

guladoras e *introns* que são removidos de transcritos de RNAs maduros. Por isso, identificar genes codificadores de proteínas em genomas eucariotos é muito mais complexo do que em procariotos [87].

A anotação em nível de proteínas cataloga as proteínas encontradas, nomeando-as e identificando suas prováveis funções. Apesar de parecer um processo simples, na prática isso não é verdade. Durante a evolução, as proteínas podem ser duplicadas uma ou várias vezes, e suas cópias são diferentes, formando uma família de proteínas relacionadas, conhecidas como parálogas. Entretanto, nem sempre proteínas da mesma família possuem funções similares [103].

Uma forma comum de fazer a anotação de proteínas é procurar por similaridade, usando ferramentas como BLASTP ou PSI-BLAST [5], em diferentes bancos de dados de proteínas. Uma estratégia complementar é buscar por domínios protéicos em banco de dados como o PFAM [48]. Isso é possível porque partimos do pressuposto que proteínas muito similares têm estruturas similares e, portanto, tendem a desempenhar a mesma função [99].

A anotação em nível de processos identifica quais vias metabólicas e processos os diferentes genes/proteínas participam. Essa etapa depende da existência de um banco de dados com a capacidade de relacionar genes que foram anotados por diferentes grupos de pesquisa. Essa base de dados é o *Gene Ontology* (GO), criada em 1991 [30]. O GO é um vocabulário padrão para descrever as funções de genes. Ele está dividido em três partes: função molecular, processo biológio e componente celular. A parte de função molecular descreve as tarefas realizadas por produtos gênicos individuais. Processos biológicos são utilizados para objetivos biológicos mais amplos, como meiose ou crescimento e manutenção celular, por exemplo. Componentes celulares descrevem os genes em termos da estrutura subcelular na qual estão localizadas, como as organelas, bem como o complexo macromolecular a que pertencem, como os ribossomos, por exemplo.

2.3.5 Análises

A partir da anotação de um genoma, seja ele completo, onde todos os cromossomos estão completamente montados, seja ele incompleto, onde se tem apenas os *contigs*, é possível a execução de diversas análises, que são muito importantes na determinação de funcionalidades do organismo estudado.

Essas análises resultam em inferências quanto à localização da espécie estudada no escopo de organismos filogeneticamente próximos, assim como permitem estudos detalhados de genes específicos, quanto à sua sintenia, entre outros.

Esta subseção descreve algumas análises consideradas neste trabalho, que podem ser executadas a partir do sequenciamento e da anotação genômica. O desenvolvimento dessas ferramentas de análise estão fora do escopo deste trabalho, mas seu uso só foi possível a partir da execução do *pipeline* aqui proposto para genômica.

Os resultados obtidos por essas análises para cepas de *M. bovis* são mostrados no Capítulo 5.

Orthologsorter

Um dos problemas da genômica é encontrar métodos que permitam identificar e classificar espécies e subespécies de bactérias de maneira homogênea. Um dos métodos utilizados é a determinação de famílias de proteínas comuns a vários genomas. A maioria das proteínas pode ser agrupada em famílias com base na similaridade entre suas sequências de aminoácidos por algum método de clusterização. Essas famílias dão pistas sobre funcionalidades comuns aos organismos, além de fornecerem direções para a construção da filogenia das espécies envolvidas na comparação.

O pacote *Orthologsorter* [44] tem dois objetivos. O primeiro é o de prover aos pesquisadores uma ferramenta de consulta via *web* de famílias de proteínas resultantes da comparação de vários genomas. Essas famílias são encontradas pelo programa OrthoMCL [69]. As consultas são feitas por escolhas *booleanas* onde o usuário escolhe famílias que contêm proteínas de determinados genomas, não contém de outros ou famílias para as quais não importa se contêm ou não proteínas de determinados genomas. Esse tipo de análise serve, por exemplo, para determinar ausência/presença de genes com funcionalidades importantes, especialmente quando relacionados com virulência e patogenicidade.

O segundo objetivo é a construção de uma árvore filogenética das espécies/cepas comparadas, de tal forma a levar em conta apenas famílias que contêm exatamente um gene de cada genoma. Para efeito dessa descrição, esses genes são ditos **ortólogos**.

Pan/Core-genome

O conceito de *pan/core-genome*, introduzido por Medini [80], permite identificar a quantidade de informação gênica presente em um conjunto de genomas. O *pan-genome* descreve o conjunto formado por todos os genes presentes numa lista de organismos, enquanto que o *core-genome* representa os genes presentes em todos os genomas do conjunto. Uma analogia imediata associa o *pan-genome* à união dos conjuntos de genes dos genomas e o *core-genome* à intersecção. Além disso, há também os chamados *dispensable genes*, que são genes que não estão presentes em todos os genomas do grupo, mas estão presentes em pelo menos dois deles. Dentre as ferramentas que fazem esse tipo de análise está o PanGP (*Pan-Genome Profile Analyze Tool*) [117].

Alinhamento Múltiplo de Ortólogos

Essa ferramenta, que está em fase de desenvolvimento pelo grupo de Bioinformática da FACOM-UFMS, permite a visualização de sintenia entre genes presentes em uma mesma família. A ideia é elucidar a ausência de genes importantes através das famílias de genes ortólogos [21].

Essa visualização é possível a partir de um genoma âncora, para o qual se tem as coordenadas de todos os genes que codificam proteínas; e é feita por meio de uma tabela, onde cada linha representa uma família de ortólogos, e cada coluna representa um genoma a ser comparado. A primeira coluna é a coluna do genoma âncora.

A ideia aqui é encontrar famílias que possuem genes de todos os genomas comparados, com exceção de algum genoma de interesse, e que estejam flanqueadas por famílias contendo genes de todos os genomas. Essa informação, que diz respeito à sintenia dos genes e das famílias correspondentes, pode ser obtida a partir das anotações dos genomas. A Figura 2.10 ilustra uma possível região específica de interesse.

Distância Genômica

Uma possível análise que pode ser feita a partir dos dados genômicos obtidos relaciona-se à distância esperada entre os cromossomos dos genomas estudados. A abordagem aqui é a sugerida por Deloger e colegas [38], na qual um índice de dis-

tância genômica é calculado para cada par de genomas. Esse índice recebe o nome de **índice Mumi** e leva em conta critérios de diversidade, que são baseados nos chamados MUMs (*Maximal Unique Matches*) [36], compartilhados entre os genomas.

A partir desses índices, é possível, por exemplo, construir uma matriz de distâncias e consequentemente uma árvore filogenética baseada em distância entre os genomas.

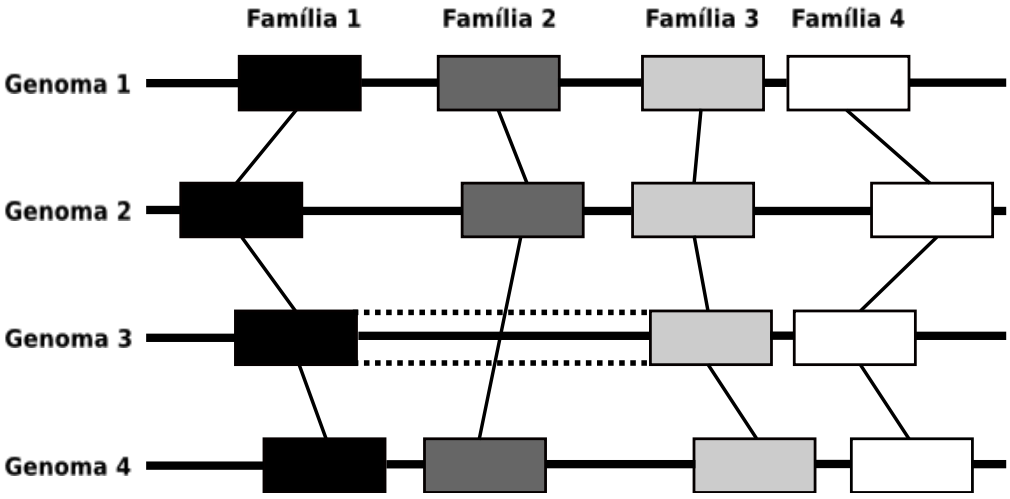


Figura 2.10: Famílias encontradas na comparação dos genomas 1, 2, 3 e 4. As famílias mostradas apresentam sintenia. A região marcada com um retângulo pontilhado consiste em uma possível região onde um gene do genoma 3 e da família 2 deveria estar. Essa região pode, por exemplo, ser considerada como entrada para a determinação de candidatos a *primer*.

Transferência Horizontal de Genes

Eventos de transferência de genes entre organismos que não tenham ocorrido por reprodução são conhecidos como eventos de **transferência horizontal de genes**. Este fenômeno é muito comum entre procariotos e é o principal fator na resistência a antibióticos e no espalhamento de fatores de virulência entre bactérias [58].

A abordagem aqui utilizada leva em conta principalmente o fato de que regiões genômicas recebidas por transferência horizontal recente muito provavelmente apresentarão frequências de dinucleotídeos, `GC`-*content*, ou *codon biases* distintos das outras regiões do genoma. Assim, a ideia é procurar regiões que tenham esse "comportamento diferenciado" do resto do genoma.

Um programa que pode ser utilizado nessa análise é o `Alien Hunter` (AH) [110]. O AH usa um modelo estatístico conhecido como *interpolated variable order motifs*

e encontra regiões com composição não usual, quando comparada com o resto do cromossomo, em termos de k-mers, para vários valores de k.

Uma região é considerada *alien* quando o *score* calculado pelo AH tem um valor maior do que um limite, também automaticamente calculado pelo programa. Esse limite leva em conta a composição do genoma como um todo.

2.4 Transcritômica

Wang e colaboradores [112] definem o transcritoma como sendo o conjunto completo de transcritos de uma célula, bem como sua quantidade, em um determinado estágio de desenvolvimento ou condição fisiológica. O transcritoma representa os genes expressos em um determinado momento, por isso sua análise é uma ferramenta essencial para a interpretação dos elementos funcionais do genoma, para a descoberta dos componentes moleculares de células e tecidos, e para o entendimento dos estágios de desenvolvimento e das doenças.

O estudo de transcritomas tem como objetivos [112]:

- catalogar todos os tipos de transcritos (mRNAs e ncRNAs);
- determinar a estrutura transcricional dos genes (sítios de iniciação, extremidades 5' e 3', padrões de *splicing* e outras modificações pós-transcricionais); e
- quantificar as mudanças nos níveis de expressão gênica de cada transcrito durante o desenvolvimento do organismo e sob diferentes condições.

Durante muitos anos a análise de microarranjos foi a abordagem utilizada na análise de transcritomas. Entretanto, o uso de dados RNA-Seq emergiu como uma alternativa para tal análise, permitindo não só o estudo de transcritos conhecidos, mas também a possibilidade de descobrir novos transcritos [112].

O RNA-Seq (*RNA Sequencing*) [83] é o método que utiliza plataformas NGS no estudo de RNAs, possibilitando identificar os RNAs que são expressos e estimar sua quantidade em um determinado momento.

Se comparada aos microarranjos, o RNA-seq possui diversas vantagens, como necessitar de uma amostra pequena de RNA e não precisar de uma lista pré-definida dos genes que se deseja detectar. Qualquer transcrito que estiver sendo expresso

pode ser detectado se o experimento tiver cobertura suficiente, ou seja, possuir um número suficiente de *reads* para cobrir uma determinada região do genoma de referência. Além disso, o RNA-seq permite detectar eventos de *splicing* alternativo e expressão de genes desconhecidos.

Dentre os diversos objetivos das análises de transcritomas, o estudo nas diferenças dos níveis de expressão dos genes é particularmente relevante e será discutida a seguir.

2.4.1 Análise de Expressão Diferencial

A análise de expressão diferencial consiste em identificar genes que são diferencialmente expressos em diferentes condições de um grupo de amostras, como por exemplo indivíduos saudáveis *versus* doentes, diferentes tecidos e diferentes estágios de desenvolvimento. Isso leva ao desenvolvimento de várias estratégias para melhor entender essas diferenças.

Apesar de existirem diferentes *pipelines* para a análise de expressão diferencial [33, 86], eles podem ser resumidos em cinco passos, mostrados na Figura 2.11. Depois de verificar a qualidade das sequências e limpá-las, elas são mapeadas em um genoma de referência. Em seguida, com o conjunto de *reads* mapeados e a anotação do genoma, é feita a contagem dos *reads* mapeados em cada gene. Por fim, métodos estatísticos são utilizados na análise de expressão diferencial.

Controle de Qualidade e Pré-processamento

Assim como ocorre na análise genômica, é necessário verificar a qualidade dos dados produzidos e filtrá-los. Os mesmos programas descritos nas Seções 2.3.1 e 2.3.2 podem ser utilizados nos dois tipos de análise.

Mapeamento de Transcritos

O alinhamento dos *reads* fornece várias informações sobre a amostra sequenciada. *Mismatches*, inserções e deleções no alinhamento podem identificar polimorfismo entre a amostra e o genoma de referência, ou até mesmo identificar eventos de fusão gênica em amostras de tumor. *Reads* que são alinhados fora de genes anotados são evidências de novos genes e ncRNAs. Os alinhamentos de transcritos também podem revelar novos eventos de *splicing* alternativo e isoformas. Além disso, os

alinhamentos podem ser usados para quantificar o nível de expressão de genes e transcritos, pois o número de *reads* produzidos por um transcrito é proporcional a sua abundância na célula [106].

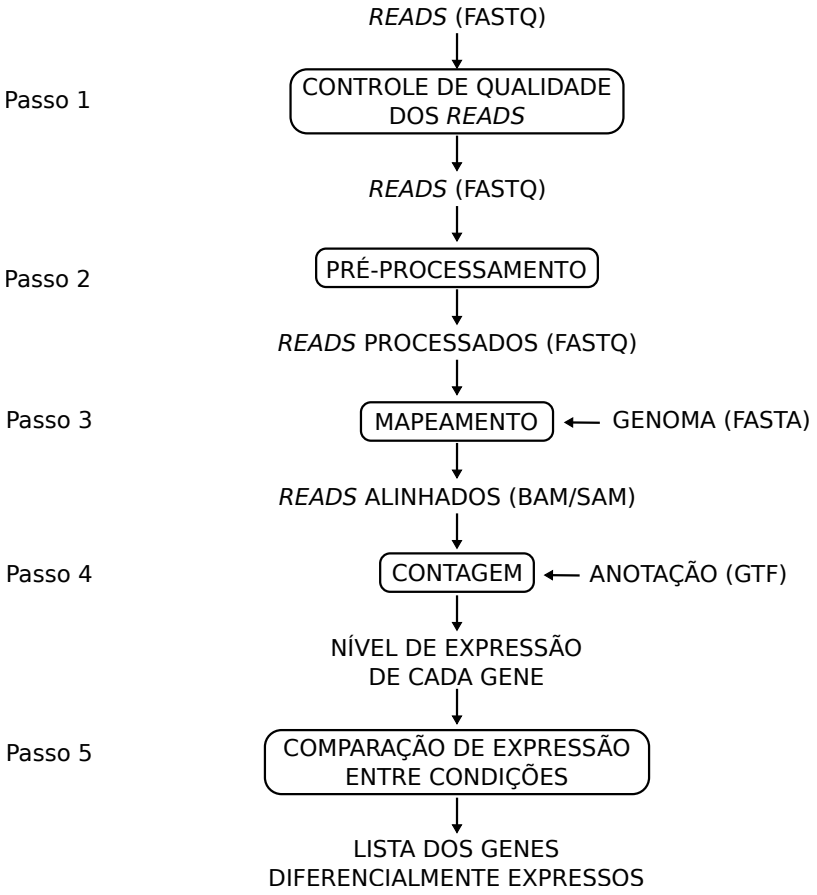


Figura 2.11: Visão geral de um *pipeline* para análise de expressão diferencial. Inicialmente os dados são avaliados e bases com baixa qualidade, adaptadores e contaminantes são removidos, além disso, sequências menores que um determinado tamanho são removidas. As sequências restantes são alinhadas e o seu resultado é utilizado para estimar a abundância de cada gene. Por fim, os resultados são avaliados e os genes diferencialmente expressos são identificados. Entre parênteses estão os formatos de arquivos produzidos em cada etapa.

Além dos aspectos já citados na subseção 2.3.3, no alinhamento dos dados de RNA-Seq os mapeadores devem lidar com os *spliced reads* gerados pelos eventos de *splicing*. Nesses casos os programas devem mapear esses *reads* de modo não contínuo no genoma, como pode ser visto na Figura 2.12. Por isso, o principal aspecto na escolha do tipo de alinhador usado nos estudo de RNA-Seq é se os alinhamentos precisam ser *spliced* ou não. Caso o organismo não possua *introns* pode-se utilizar programas

como Bowtie [60] ou BWA [64]. Caso contrário, alternativas são: TopHat [105], STAR [39] ou GSNAP [114].

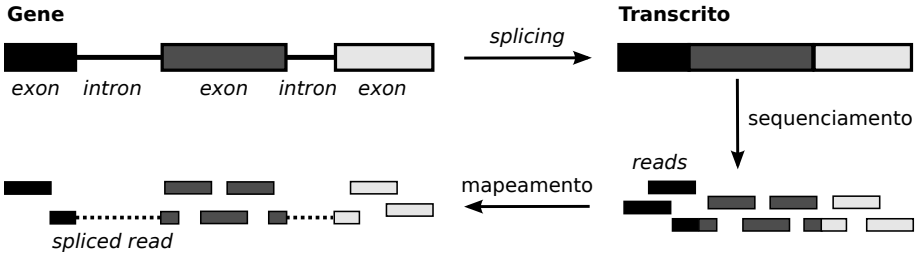


Figura 2.12: Mapeamento de *spliced read*. Devido aos eventos de *splicing*, os *reads* sequenciados originam-se de trechos não contínuos do genoma. Para serem corretamente mapeados, os alinhadores precisam considerar os *introns* que separam esses *exons*.

Os alinhadores do tipo *spliced* normalmente realizam o mapeamento em duas etapas. Primeiro, os *reads* são mapeados no genoma e usados para construir todas as possíveis junções *exon-exon*. Essa informação é utilizada em uma segunda etapa de mapeamento com os *reads* que não foram mapeados na primeira etapa.

Contagem

Depois de mapeados no genoma de referência, a localização dos *reads*, juntamente com a anotação genômica, permitem quantificar a expressão gênica. A maneira mais simples de estimar a expressão é contar o número de *reads* que foram mapeados em cada gene, transcrito ou *exon*. Programas como HTSeq [7], BEDTools [88], featureCounts [71] ou Cufflinks [107] desempenham essa tarefa. Existem também alguns pacotes do Bioconductor como o Rsubread [70] ou o GenomicRanges [61], que disponibilizam essa funcionalidade.

Teoricamente, contar o número de *reads* mapeados forneceria uma forma direta de estimar a abundância de um transcrito porém, na prática, outros fatores devem ser considerados, pois o número de *reads* gerados por transcrito depende da cobertura de sequenciamento e do tamanho do transcrito. Genes eucariotos, por exemplo, transcrevem várias isoformas que possuem *exons* em comum. Cada *software* utiliza uma estratégia diferente para lidar com *reads* mapeados em várias regiões genômicas, isoformas ou mapeamentos em regiões intrônicas ou não anotadas.

Expressão Diferencial

Após realizar a contagem das amostras, a fase para identificar os genes diferecialmente expressos envolve a normalização dos dados, a escolha de um modelo estatístico e o teste para expressão diferencial. Esses elementos são implementados por diferentes programas, dentre os quais podemos citar: baySeq [52], Cuffdiff [107], DESeq [6], DESeq2 [73], easyRNASeq [37], EBSeq [62], EdgeR [91], PoissonSeq [68] e SAMseq [67].

Os métodos de normalização tentam controlar as diferenças das amostras, e assim facilitar a precisão das comparações entre as diferentes condições estudadas. Existem diversos métodos desenvolvidos para a normalização dos dados, entretanto não há uma unanimidade sobre qual a melhor escolha.

Uma questão que deve ser considerada está relacionada ao tamanho do gene (Figura 2.13). Espera-se que o número de *reads* mapeados em um gene seja proporcional ao seu tamanho e a sua abundância de acordo com a isoforma transcrita. Genes maiores produzirão mais *reads* do que genes mais curtos, o que pode resultar na detecção de expressão diferencial, mesmo sendo igualmente expressos. Um método de normalização desenvolvido foi o *Reads Per Kilobase of transcript per Million mapped* (RPKM) [82]. A ideia do RPKM é dividir o número de *reads* mapeados no transcrito pelo número total de *reads* mapeados em cada amostra vezes o tamanho do transcrito. Então, o número de *reads* é normalizado pelo tamanho do gene, evitando o problema do grande número de *reads* mapeados em grandes genes; sendo também normalizado pelo total de *reads* na amostra. Desse modo, uma amostra com uma cobertura de sequenciamento maior não atrapalha a comparação entre amostras.

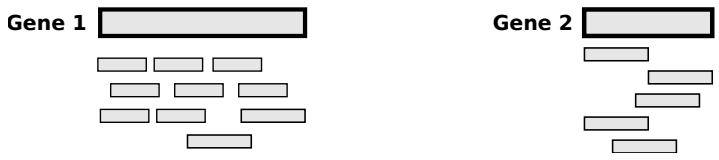


Figura 2.13: Genes maiores provalvemente produzirão mais *reads* que genes menores, mesmo com níveis de expressão similares.

De modo análogo, o *Fragments Per Kilobase of transcript per Million mapped* (FPKM) foi introduzida por Trapnell e colaboradores [107] para lidar com dados *paired-end*. O FPKM é uma unidade de expressão que considera cada fragmento mapeado e não cada *read*. O fragmento é definido como um par de *reads*.

Outro ponto que deve ser considerado diz respeito à contagem total de uma amostra, que pode ser altamente dependente de poucos genes altamente expressos (Tabela 2.3). Isso faz com que a expressão dos demais genes seja subestimada. Para lidar com esse problema, a estratégia é utilizar fatores de escala como o *Trimmed Means of M values* (TMM) [90] ou o *upper-quartile* [18].

Tabela 2.3: Diferença na contagem das amostras. Exemplo de contagem onde o número de *reads* nas duas amostras são iguais, mas com diferentes distribuições.

	Amostra A	**Amostra** B
gene 1	100	80
gene 2	100	80
⋮	⋮	⋮
gene 99	100	80
gene 100	20	2000
	$d_A = 9920$	$d_B = 9920$

O TMM é calculado removendo uma determinada porcentagem dos genes considerados os mais extremos entre as amostras comparadas. Este fator de normalização é então usado para corrigir as diferenças no tamanho das amostras.

A distribuição de probabilidades associa uma probabilidade a cada resultado numérico de um determinado experimento. A distribuição de Poisson está associada a variáveis aletórias discretas e normalmente é utilizada na contagem de dados. Essa distribuição considera que cada *read* sequenciado de uma determinada amostra foi gerado de forma independente e uniforme [85].

A distribuição de Poisson parece descrever bem a variação observada entre replicatas técnicas de uma mesma espécie, ou seja, amostras sequenciadas em diferentes *lanes* de uma *flow cell* [85]. Nesse caso, espera-se que a média e variância sejam as mesmas.

As replicatas biológicas acrescentam outro nível de variabilidade, a variabilidade biológica existente entre os diferentes indivíduos. Essa diferença resulta em uma distribuição de probabilidade diferente para cada *read* sequenciado. Assim, na contagem observada em replicatas biológicas, a variação é a soma da variabilidade técnica com a variabilidade biológica, o que resulta numa variação observada maior do que a esperada na distribuição de Poisson. Essa variação maior que a média é chamada de superdispersão (*overdispersion*). Nesse caso, a distribuição de Poisson não pode lidar com essa variabilidade adicional, então utiliza-se uma distribuição bi-

nomial negativa, onde a relação entre variância ν e a média μ é definida pela função $\nu = \mu + \phi\mu^2$, onde ϕ é um fator de dispersão.

A comparação do nível de expressão dos genes em duas diferentes condições invariavelmente possuirá diferentes contagens. Essa diferença pode ser devida a verdadeira diferença biológica entre as condições ou apenas ruídos do experimento. A combinação de modelos de distribuição e testes estatísticos permitem distinguir entre essas duas possibilidades.

Abaixo estão alguns programas utilizados em cada etapa das análises genômicas e de expressão diferencial.

- **Controle de Qualidade**:

  1. FastQC [8]
  2. PRINSEQ [97]

- **Pré-processamento**:

  1. FastQC [8]
  2. FastX-Toolkit [1]
  3. PRINSEQ [97]
  4. TagCleaner [98]
  5. Cutadapt [78]
  6. Trimmomatic [14]
  7. SeqyClean [2]

- **Mapeamento e Montagem**:

  1. MAQ [66]
  2. Stampy [75]
  3. BWA [64]
  4. Bowtie [60]
  5. MapSplice [111]
  6. SpliceMap [13]
  7. TopHat [105]
  8. GSNAP [114]
  9. QPALMA [35]
  10. ABySS [101]
  11. MIRA [26]
  12. SSAKE [113]
  13. VCAKE [54]
  14. Velvet [116]
  15. Trinity [50]

- **Contagem**:

  1. HTSeq [7]
  2. BEDTools [88]
  3. featureCounts [71]
  4. Cufflinks [107]
  5. Rsubread [70]
  6. GenomicRanges [61]

- **Expressão Diferencial**:

1. baySeq [52]
2. Cuffdiff [107]
3. DESeq [6]
4. DESeq2 [73]
5. easyRNASeq [37]
6. EBSeq [62]
7. EdgeR [91]
8. PoissonSeq [68]
9. SAMseq [67]

Capítulo 3

Pipeline Automatizado para Genômica

Neste capítulo descrevemos o *pipeline* genômico desenvolvido neste trabalho e as ferramentas computacionais utilizadas em cada etapa. Na Seção 3.1 é apresentado a visão geral do *pipeline*. O controle de qualidade é mostrado na Seção 3.2. Na Seção 3.3 apresentamos como é feita a limpeza dos dados. A Seção 3.4 trata do mapeamento dos *reads* e a Seção 3.5 detalha a obtenção dos *contigs* que serão utilizados na anotação do genoma, apresentada na Seção 3.6. Os resultados obtidos pela execução do *pipeline* para cepas de *M. bovis* são mostrados no Capítulo 5.

3.1 Visão Geral do *Pipeline*

O *pipeline* genômico é formado por uma série de *scripts* desenvolvidos em Perl e um conjunto de arquivos de configuração e obtém de forma automatizada os *contigs* que são utilizados na anotação de um genoma. A Figura 3.1 mostra, para cada etapa do *pipeline* os principais arquivos de entrada que podem ser utilizados, os diretórios que são criados, bem como os arquivos gerados em cada diretório após a execução do programa apropriado.

A entrada do *pipeline* é um conjunto de arquivos no formato FASTQ. Cada um dos arquivos é avaliado no primeiro passo. Em seguida, ocorre a filtragem, no segundo passo. No passo 3, os *reads* são alinhados no genoma de referência e o resultado do

mapeamento é convertido em *contigs*, que são anotados e servem de base para as análises genômicas.

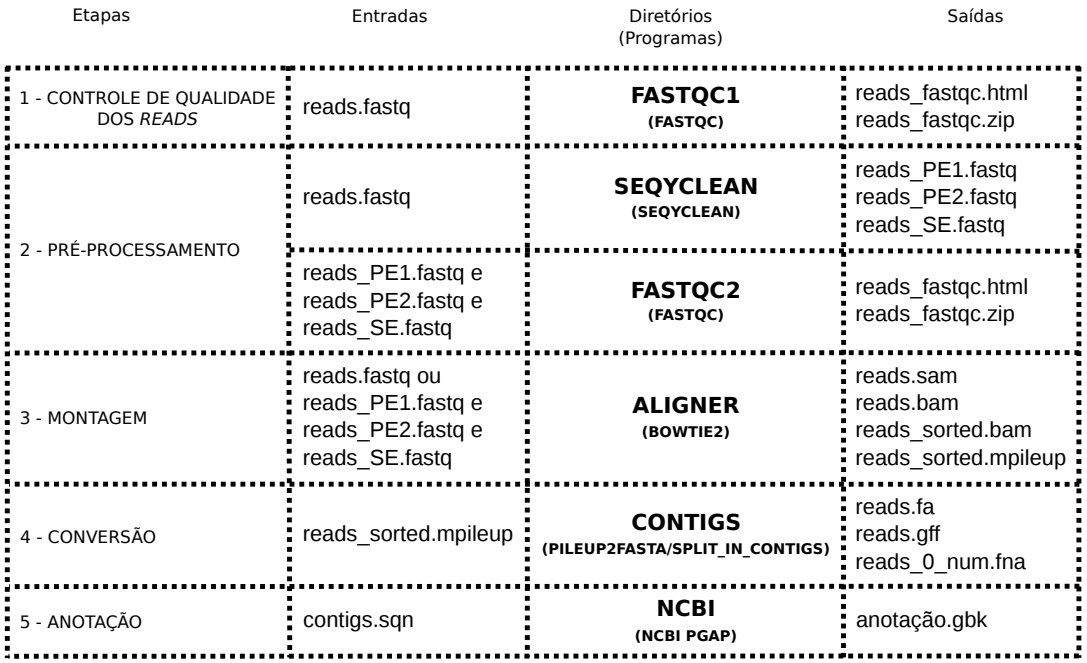


Figura 3.1: Visão geral das etapas do *pipeline* genômico, bem como os principais dados de entrada, diretórios criados, programas utilizados e arquivos produzidos ao final de cada etapa.

O *script* `run_genomic_pipeline.pl` executa o *pipeline* mostrado na Figura 3.1 de acordo com os parâmetros especificados no arquivo de configuração `genomic_pipeline.conf`.

O `genomic_pipeline.conf` é um arquivo tabular formado por duas colunas (Figura 3.2). A primeira com variáveis que são usadas no *pipeline* e a segunda com os respectivos valores. Além disso, linhas iniciadas com o caractere "#", em todos os arquivos de configuração, representam comentários e são ignoradas pelos *scripts*. O arquivo contém informações sobre localização dos *reads*, do genoma de referência e de sua anotação, os arquivos de configurações de cada *software* e os diretórios que são criados de acordo com a escolha dos programas que são usados na análise. Contém também o tamanho mínimo que um *contig* deve ter.

Além do `genomic_pipeline.conf`, há o `seqyclean.conf` e `bowtie.conf` contendo os parâmetros de cada programa utilizado durante a análise.

```
# Diretório onde os dados serão analisados e que contém o diretório de reads
project_dir     /home/myuser/MyProject
# Arquivos de configuração dos diferentes softwares
conf_seqyclean  /home/myuser/MyProject/conf_files/seqyclean.conf
conf_bowtie     /home/myuser/MyProject/conf_files/bowtie.conf
# Arquivo do genoma de referência
genome_fna      /home/myuser/MyProject/genomes/myGenome.fa
# Índice do genoma de referência - Usar o mesmo prefixo que o .fa e o .gtf
genome_idx      /home/myuser/MyProject/genomes/myGenome
# Criar o índice do genoma? 0 para não e 1 para sim
genome_index    1
# Tamanho mínimo do contig
contig_length   500
# Qualidade mínima das bases
base_cov   20
# Os reads são paired-end? 0 para não e 1 para sim
PAIRED     1
# Início do header dos reads
header     @HISEQ
tipo       fastq
# Diretórios que serão criados durante a análise
dirs       reads,fastqc1,seqyclean,fastqc2,bowtie,contigs
# Softwares que deverão ser executados - 1 para executar e 0 caso contrário
FASTQC1    1
SEQYCLEAN  1
FASTQC2    1
BOWTIE     1
CONTIGS    1
```

Figura 3.2: Arquivo de configuração `genomic_pipeline.conf` do *pipeline* genômico.

3.2 Controle de Qualidade

Ao iniciar o `run_genomic_pipeline.pl`, o programa FastQC [8][1] verifica a qualidade dos dados. O FastQC é um *software* desenvolvido em JAVA pelo Babraham Institute, e realiza diversos testes, fornecendo um relatório que permite avaliar se existe alguma anormalidade nos dados sequenciados. Cada teste é classificado como *pass*, *warning* ou *fail* dependendo de quão distante os dados estão do que se espera ser normal. Mesmo que o dado seja classificado como *warning* ou *fail* isso não significa que há um problema com o conjunto de dados, mas que não é usual. Isso pode ser resultado da característica biológica da amostra analisada. A Figura 3.3 mostra um dos gráficos gerados pelo FastQC.

Dentre as análises feitas pelo FastQC estão o *Phred quality score* de todas as bases, a qualidade geral das sequências, o conteúdo GC, o número de duplicações, sequências que estão muito representadas entre outras informações que ajudarão a verificar a qualidade dos dados e a definir os parâmetros de filtragem, além de mostrar informações como a quantidade e o tamanho do menor e do maior *read*.

[1] `http://www.bioinformatics.babraham.ac.uk/projects/fastqc/`

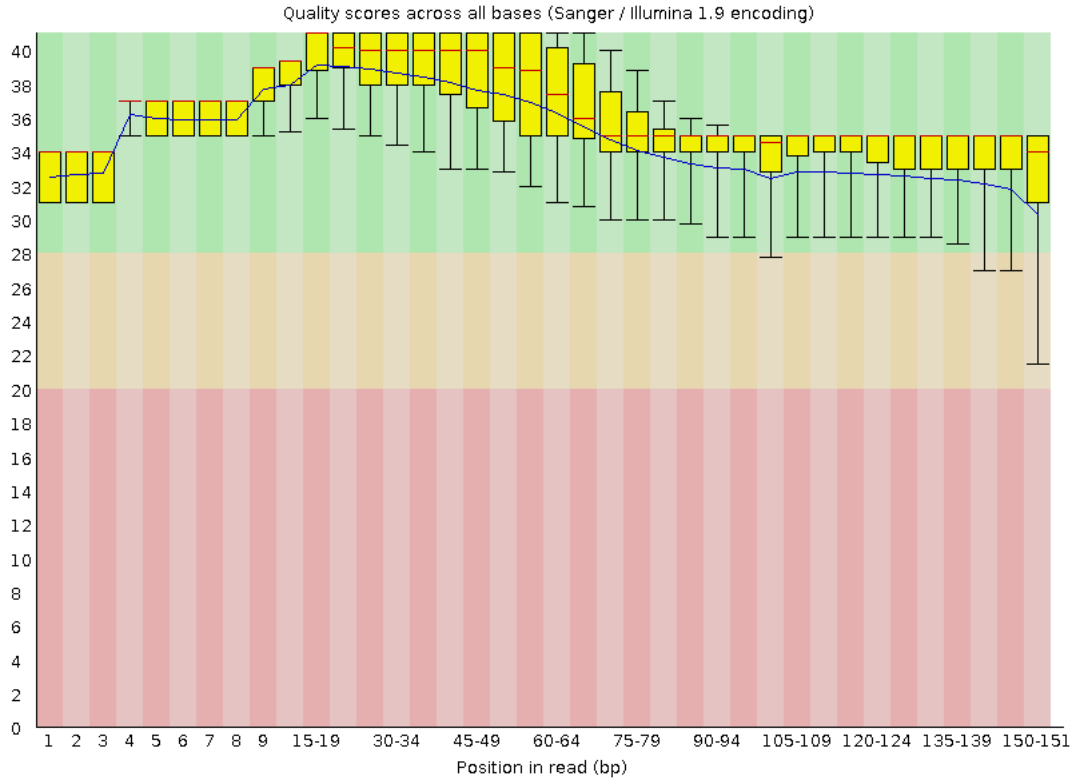


Figura 3.3: Exemplo de um gráfico de qualidade gerado pelo FastQC. Ele apresenta uma visão global dos valores de qualidade de todas as 151 bases de todos os *reads* de um arquivo fastq. O eixo X representa as posições de cada base dos *reads* enquanto o eixo Y mostra os valores de qualidade no Phred *quality score*. Quanto maior o valor, melhor é qualidade da base. A cor de fundo do gráfico divide o eixo Y de acordo com a qualidade das bases. A linha azul representa a qualidade média obtida. A linha vermelha é o valor da mediana. O retângulo amarelo representa o inter-quartil.

A versão 0.10.1 do FastQC gera seu relatório em um arquivo compactado (`.zip`) e um diretório com todas as análises que foram feitas no diretório de origem dos *reads*. Por meio do *script* `exec_fastqc.pl`, que executa o FastQC, a saída do programa é armazenada no diretório apropriado (`01_fastqc1`).

3.3 Pré-processamento

Com base no relatório gerado pelo FastQC, a filtragem dos *reads* usa o *software* SeqyClean [2]. O SeqyClean é uma ferramenta desenvolvida para limpar dados de Illumina e Roche 454, e que permite remover adaptadores, caudas poli A/T, contaminantes e vetores, além dos *reads* com baixa qualidade.

Dependendo dos parâmetros e estratégias de limpeza utilizados, os nomes dos arquivos de saída serão diferentes. Para *reads* de Roche 454 a saída padrão é no formato SFF (*Standard Flowgram Format*), ou FASTQ se essa for a opção escolhida. Há também dois arquivos de relatório: `prefix_SummaryStatistics.txt` com o número de *reads* processados, limpos e descartados e o `prefix_Report.txt` com detalhes estatísticos de cada *read*.

Para os dados de Illumina o SeqyClean cria dois ou três arquivos de saída no formato FASTQ, além dos arquivos de relatórios. Para dados *single-end* são criados `prefix.fastq` e `prefix_shuffled.fastq`, enquanto para os dados *paired-end* são criados `prefix_PE1.fastq`, `prefix_PE2.fastq` com os *reads* pareados e o `prefix_SE.fastq` com *reads single-end*.

O *script* `exec_seqyclean.pl` executa a filtragem e salva todos os resultados no diretório `02_seqyclean`. Para verificar a qualidade da filtragem, o *pipeline* executa o FastQC novamente e o resultado é armazenado no diretório `03_fastqc2`.

3.4 Mapeamento

Depois de filtrados, os *reads* são montados usando um genoma de referência. Por se tratar de um *pipeline* para genomas procariotos foi escolhido *software* Bowtie2 [59], nova versão do Bowtie [60].

O Bowtie2 possui um bom desempenho e necessita de pouca memória, pois cria índices para o genoma de referência. Esses índices podem ser encontrados no site do Bowtie2[2] ou da Illumina[3], mas podem ser criado utilizando o comando `bowtie2-build`. Esse comando cria seis arquivos com extensão `.bt2` e formam o índice que é usado no alinhamento.

O Bowtie2 utiliza apenas o índice do genoma e o arquivo de *reads* durante o mapeamento. Ele aceita arquivos no formato FASTA ou FASTQ e pode usar dados *single-end* ou *paired-end*. No caso de dados *paired-end* é necessário que os *reads* estejam na mesma ordem, dessa forma o Bowtie2 pode trabalhá-los como um par. Sua saída é um arquivo no formato SAM, mas para economizar espaço ele pode ser transformado em um formato binário e compactado, o BAM.

[2] `http://bowtie-bio.sourceforge.net/bowtie2/index.shtml`
[3] `http://support.illumina.com/sequencing/sequencing_software/igenome.html`

O formato SAM (*Sequence Alignment/Mapping*) foi criado para representar o alinhamento de *reads* em uma sequência de referência, mas que pode ser transformado em outros formatos com o SAMtools [65].

O SAMtools é um *software* implementado na linguagem C e utilizado para análise e manipulação de arquivos no formato SAM/BAM. O *software* também possui diversas funcionalidades que permitem gerar outros formatos, recuperar um subconjunto de alinhamentos, ordenar os alinhamentos pelas coordenadas dos cromossomos ou pelo nome dos *reads*, juntar múltiplos alinhamentos ordenados, gerar informações por base no formato pileup[4], encontrar SNPs (*Single Nucleotide Polymorphism*) e visualizar textualmente o alinhamento.

O *script* `exec_genomic_bowtie_paired.pl`, além de executar o Bowtie2, utiliza algumas funcionalidades do SAMtools para preparar os dados para a próxima etapa.

Depois do mapeamento, o resultado é compactado com o `samtools view`, ordenado com o `samtools sort` e com o `samtools mpileup` obtém-se um arquivo no formato pileup. Esse formato é uma síntese de cada base que foi alinhada no genoma de referência, o que facilita a visualização de SNP/indel. O formato é útil pois permite extrair quais bases existem em cada posição do genoma de referência, possibilitando a obtenção do *contig*, além disso informa quantas, quais e a qualidade de cada base mapeada naquela posição.

3.5 Obtenção dos *Contigs*

Utilizando o *script* `pileup2fasta.pl`, criado por John Nash[5], obtém-se um arquivo fasta. Esse *script* recebe como entrada um arquivo no formato pileup e gera um arquivo no formato FASTA contendo a sequência de nucleotídeos resultante do mapeamento e um arquivo no formato GFF (*General Feature Format*) contendo as informações sobre inserções, deleções e SNPs que são encontradas.

Esse *script* insere "*" na sequência obtida, por isso usamos o *script* `split_in_contigs.pl` para separar a sequência em vários *contigs*. Apenas aqueles com um tamanho mínimo definido pelo usuário são mantidos no resultado final.

[4] `http://samtools.sourceforge.net/pileup.shtml`
[5] `http://code.google.com/p/nash-bioinformatics-codelets/`

3.6 Preparação para a Anotação

O *pipeline* foi desenvolvido para obter de modo automatizado *contigs* a partir de um conjunto de *reads* e alguns arquivos de configuração. Tais *contigs* devem ser anotados para serem utilizados em diferentes análises.

Para anotar o genoma é usado o NCBI *Prokaryotic Genome Annotation Pipeline* (PGAP) [9], disponível em: `http://www.ncbi.nlm.nih.gov/genome/annotation_prok/`. Porém, para utilizá-lo é necessário transformar os *contigs* em um formato específico, o que envolve uma série de passos, feitos manualmente utilizando outro conjunto de programas.

A primeira tarefa para fazer a anotação é modificar o identificador de cada *contig* e salvar o arquivo com uma extensão `.fsa`. Cada *contig* possui um identificador único e algumas informações adicionais como organismo, localização, país de origem, entre outras características. O *script* `include_cfg.pl` recebe o arquivo de contigs e um arquivo `.cfg` contendo essas informações adicionais e cria o arquivo `.fsa`.

Depois, para submeter o genoma é necessário registrá-lo no *BioSample database*[6]. Além disso, cada genoma pertence a um determinado *BioProject*. Tanto o *BioSample* quanto o *BioProject* possuem identificadores que são necessários para a criação do arquivo `.sbt`. Esse arquivo é utilizado para formar o arquivo que é enviado para anotação e possui informações sobre os autores, publicação, o *BioSample* e o *BioProject*, em um formato próprio. Para criá-lo é necessário preencher um formulário disponível em: `http://www.ncbi.nlm.nih.gov/WebSub/template.cgi`.

A próxima tarefa é juntar os arquivos `.fsa` e `.sbt`. Isso é feito utilizando-se o programa `tbl2asn`[7]. O `tbl2asn` automatiza a criação do arquivo `.sqn` que é submetido ao GenBank. Antes de fazer a submissão é possível verificar se o arquivo `.sqn` foi corretamente criado em `http://www.ncbi.nlm.nih.gov/genomes/frameshifts/frameshifts.cgi`.

[6] `https://submit.ncbi.nlm.nih.gov/subs/biosample/`
[7] `http://www.ncbi.nlm.nih.gov/genbank/tbl2asn2/`

Capítulo 4

Pipeline Automatizado para Análise de Expressão Diferencial

Este capítulo descreve todas as etapas e ferramentas do *pipeline* para análise de expressão diferencial. Na Seção 4.1 é apresentado a visão geral do *pipeline*. O controle de qualidade é mostrado na Seção 4.2, e na Seção 4.3 apresentamos como é feita a limpeza dos dados. A Seção 4.4 trata do mapeamento dos *reads* e a Seção 4.5 determina o número de *reads* que estão mapeados em cada gene. Por fim, na Seção 4.6 é realizada a análise de expressão diferencial. Os resultados obtidos com o uso deste *pipeline* para três projetos de transcritômica são mostrados no Capítulo 6.

4.1 Visão Geral do *Pipeline*

O *pipeline* desenvolvido utiliza os mesmos passos apresentados na Subseção 2.4.1, mas toda a análise é feita de maneira automatizada por meio de uma série de *scripts* escritos em Perl que fazem as chamadas de diversos *softwares* e alguns arquivos de configuração que definimos. A Figura 4.1 mostra uma visão geral dos principais *softwares* utilizados em cada etapa, além dos diretórios que são criados, os arquivos de entrada e saída armazenados em cada diretório depois da execução do programa adequado. Nas subseções seguintes, cada passo é descrito de forma mais detalhada.

O *pipeline* é executado a partir do *script* `run_pipeline_de.pl` e do arquivo de configuração `pipeline_de.conf`. De acordo com os parâmetros especficados, o *script*

cria os diretórios necessários, verifica se os índices dos genomas foram criados e executa os *scripts* corretos para cada conjunto de *reads*.

Etapas	Entradas	Diretórios (Programas)	Saídas
1 - CONTROLE DE QUALIDADE DOS *READS*	reads.fastq	**FASTQC1** (FASTQC)	reads_fastqc.html reads_fastqc.zip
2 - PRÉ-PROCESSAMENTO	reads.fastq	**CUTADAPT** (CUTADAPT)	reads.clipped.fastq reads.short.fastq log_cutadapt_reads.txt
	reads.fastq ou reads.clipped.fastq	**PRINSEQ** (PRINSEQ)	reads.trimmed.fastq reads.trimmed.log
	reads.clipped.fastq ou reads.trimmed.fastq	**PAIRED**	reads.paired.fastq
	reads.clipped.fastq ou reads.trimmed.fastq ou reads.paired.fastq	**FASTQC2** (FASTQC)	reads_fastqc.html reads_fastqc.zip
3 - ALINHAMENTO	reads.fastq ou reads.clipped.fastq ou reads.trimmed.fastq ou reads.paired.fastq	**ALIGNER** (TOPHAT2/BOWTIE2)	logs/ accepted_hits.bam align_summary.txt deletions.bed prep_reads_info insertions.bed reads.sam junctions.bed unmapped.bam
4 - CONTAGEM	reads.sam	**HTSEQ_ALIGNER** (HTSEQ-COUNT)	reads.count
5 - COMPARAÇÃO DE EXPRESSÃO ENTRE CONDIÇÕES	reads.count	**DE_ALIGNER** (EDGER/DESEQ2)	counts.xls ec_MDS.pdf ec_Smearcomplex1.pdf ec_Smearcomplex2.pdf ec_deg1.xls ec_deg2.xls d2_deg1.xls d2_deg2.xls

Figura 4.1: Visão geral das etapas do *pipeline* para análise de expressão diferencial, bem como os principais dados de entrada, diretórios criados, programas utilizados e arquivos produzidos ao final de cada etapa.

O `pipeline_de.conf` (Figura 4.2) possui linhas de comentários que se iniciam com o caractere "#" e linhas de configuração formadas por duas colunas separadas por TAB. Na primeira coluna estão as variáveis e na segunda seus respectivos valores, que podem ser inteiros ou textuais. Esse arquivo contém informações sobre os arquivos de configurações de cada programa utilizado, localização do *reads*, do genoma de referência e de sua anotação, diretórios que serão criados de acordo com a escolha dos *softwares* que serão usados na análise e quais comparações serão testadas.

O arquivo de configuração define ainda quais etapas são realizadas, permitindo assim que as etapas 1 e 2 sejam ignoradas, mesmo que o *pipeline* tenha sido projetado para executar todas as etapas.

Para que o *pipeline* funcione corretamente assumimos que diretório com os *reads*

sempre é `00_reads` e os arquivos contendo os *reads* possuem a extensão `.fastq` e são rotulados usando sua condição, replicata e paridade, separados por "_". Os arquivos `condição_replicata_R1.fastq` e `condição_replicata_R2.fastq` representam *reads paired-end*.

```
# Diretório onde os dados serão analisados e que contém o diretório de reads
project_dir   /home/myuser/MyProject
# Arquivos de configuração dos diferentes softwares
conf_cutadapt /home/myuser/MyProject/conf_files/cutadapt.conf
conf_prinseq  /home/myuser/MyProject/conf_files/prinseq.conf
conf_tophat   /home/myuser/MyProject/conf_files/tophat.conf
conf_bowtie   /home/myuser/MyProject/conf_files/bowtie.conf
conf_htseq    /home/myuser/MyProject/conf_files/htseq.conf
conf_de   /home/myuser/MyProject/conf_files/de.conf
# Arquivo do genoma de referência
genome_fna    /home/myuser/MyProject/genomes/myGenome.fa
# Arquivo gtf com a anotação do genoma de referência
genome_gtf    /home/myuser/MyProject/genomes/myGenome.gtf
# Índice do genoma de referência - Usar o mesmo prefixo que o .fa e o .gtf
genome_idx    /home/myuser/MyProject/genomes/myGenome
# Criar o índice do genoma? 0 para não e 1 para sim
genome_index  1
# Os reads são paired-end? 0 para não e 1 para sim
PAIRED    1
# Os genes DE serão mostrados pelo ID (0) ou pelo nome (1)
gene_id_name  1
# Início do header dos reads
header    @HISEQ
#
tipo      fastq
# Diretórios que serão criados durante a análise
dirs      reads,fastqc1,cutadapt,prinseq,paired,fastqc2,tophat,htseq,de
# Softwares que deverão ser executados - 1 para executar e 0 caso contrário
FASTQC1   1
CUTADAPT  1
PRINSEQ   1
FASTQC2   1
TOPHAT    1
BOWTIE    0
HTSEQ     1
DE   1
# Número de comparações que serão realizadas
num_test  2
# Comparações que deverão ser feitas
test_1    A   B
test_2    A   C
```

Figura 4.2: Arquivo de configuração `pipeline_de.conf` do *pipeline* para análise de expressão diferencial.

Além do `pipeline_de.conf`, há um arquivo de configuração contendo os parâmetros para cada *software* (`cutadapt.conf`, `prinseq.conf`, `tophat.conf`, `bowtie.conf`, `htseq.conf` e `de.conf`) utilizado durante a análise.

Para facilitar a organização, o resultado de cada etapa é armazenado em um diretório diferente. Por exemplo, se o *pipeline* for executado com todos os passos para *reads paired-end* e usando o TopHat na etapa de mapeamento, serão criados

oito diretórios: `01_fastqc1`, `02_cutadapt`, `03_prinseq`, `04_paired`, `05_fastqc2`, `06_tophat`, `07_htseq_tophat` e `08_de_tophat`.

4.2 Controle de Qualidade

Assim como no *pipeline* genômico, utilizou-se o FastQC para analisar a qualidade de todos os dados sequenciados e armazenados no diretório de *reads* (`00_reads`), mas a versão utilizada nesse *pipeline* foi a 0.11.2 que gera seu relatório em um arquivo `.html` e um arquivo compactado `.zip` com todas as análises que foram feitas. Nessa versão os resultados também são produzidos no diretório de entrada dos dados, mas o *script* dessa fase organiza os dados no diretório correto.

4.3 Pré-processamento

Nessa etapa de filtragem foram utilizados dois programas diferentes: o Cutadapt [78] e o PRINSEQ [97].

Implementado em Python, o Cutadapt utiliza um alinhamento semi-global para encontrar os adaptadores em todos os *reads* e removê-los. A remoção é feita reescrevendo as sequências modificadas. Em alguns casos, dependendo dos parâmetros utilizados, a sequência toda é removida.

O *script* `exec_cutadapt.pl` lê o arquivo `cutadapt.conf` com os parâmetros para o *clipping* e o resultado são três arquivos armazenados no diretório `*_cutadapt`. O arquivo `*.clipped.fastq` guarda os *reads* que passaram pelo *clipping* e serão utilizados na fase de *trimming* pelo PRINSEQ, se necessário. O arquivo `*.short.fastq` contém os *reads* que estão abaixo do limite especificado no arquivo de configuração e o por fim o arquivo `log_cutadapt_*.fastq` armazena um relatório com as informações sobre a filtragem realizada.

O PRINSEQ foi desenvolvido em Perl e permite filtrar as sequências a partir de um tamanho específico, eliminar as extremidades 5' e 3', eliminar a cauda poli-A/T, remover sequências abaixo de um determinado valor. Pode-se também remover os identificadores das sequências ou renomeá-las.

O PRINSEQ possui uma versão *web* e outra versão para ser executada por linha de comando. Assim como FastQC, ele fornece diversos relatórios sobre a qualidade

dos dados sequenciados, mas também permite realizar a filtragem e reformatação dos *reads* utilizando diferentes parâmetros. Contudo, nesse projeto o PRINSEQ foi utilizado apenas para filtragem.

O *trimming* é executado pelo *script* `exec_prinseq.pl`, que lê os parâmetros que devem ser utilizados do arquivo `prinseq.conf` e gera dois arquivos de saída: o `*.trimmed.log` com informações sobre a filtragem e o `*.trimmed.fastq` com os *reads* filtrados. Dependendo dos parâmetros de configuração o *trimming* pode ser realizado a partir dos *reads* originais ou dos arquivos `*.clipped.fastq`.

Se os dados de entrada forem *paired-end* é necessário "parear" os dois arquivos (R1 e R2), pois ao realizar a filtragem a quantidade de *reads* em cada um dos arquivos pode ficar diferente. Para garantir que isso não ocorra, o *script* `write_paired_fastq_files.pl` reescreve os arquivos R1 e R2 apenas com os *reads* que estão presentes nos dois arquivos, salvando-os nos arquivos `*_R1.paired.fastq` e `*_R2.paired.fastq`.

4.4 Alinhamento

O *pipeline* permite fazer o mapeamento dos *reads* utilizando o Bowtie2 ou o TopHat2.

O TopHat2 [55] é uma versão com melhorias do TopHat [105], que utiliza o Bowtie2 para fazer o alinhamento e por isso usa o índice do genoma de referência, criado pelo programa Bowtie2, o arquivo de *reads* e opcionalmente, o arquivo de anotação do genoma no formato GTF (*General Transfer Format*), mas nesse *pipeline* seu uso é obrigatório.

Para executar o TopHat2 é necessário o índice do genoma criado pelo Bowtie2 descrito anteriormente, além do seu respectivo arquivo FASTA. Se ele não existir no mesmo diretório do índice, o TopHat2 irá criá-lo em toda execução, a partir dos arquivos de índice.

Normalmente o TopHat2 é utilizado para fazer alinhamentos *spliced*. Ele é otimizado para alinhar *reads* a partir de 75 bases e utiliza um alinhamento em várias etapas. Primeiro, se estiver disponível a anotação do genoma, o TopHat2 extrai os transcritos dos arquivos de índices e utiliza o Bowtie2 para alinhar os *reads* nesses transcritos. Os *reads* que não foram completamente alinhados ao transcritoma são alinhados no

genoma com o Bowtie2. Nesse ponto apenas os *reads* que se alinham continuamente em um *exon* são mapeados, enquanto os *reads* não mapeados são divididos em sequências menores e mapeados novamente contra o genoma.

O TopHat2 produz como resultado um diretório `logs` e vários arquivos, descritos a seguir, no diretório de saída:

- `accepted_hits.bam`: armazena o alinhamento ordenado pela coordenada do cromossomo, no formato BAM;
- `junctions.bed`: contém as junções de *exons* descobertas;
- `insertions.bed`: contém as inserções descobertas;
- `deletions.bed`: contém as eliminações descobertas;
- `align_summary.txt`: informa as taxas de alinhamento e quantos *reads* possuem múltiplos alinhamentos.

Para o *pipeline* utiliza-se apenas o arquivo `accepted_hits.bam`, que é ordenado pelo nome dos *reads*, com o `samtools sort` e em seguida é transformado no formato SAM com o `samtools view`.

Os *scripts* `exec_(bowtie|tophat)_(single|paired).pl` realizam o mapeamento e a preparação dos dados para a etapa de contagem.

4.5 Contagem

HTSeq [7] é uma biblioteca escrita em Python que permite processar dados NGS. Um de seus pacotes é o htseq-count, que a partir do resultado do mapeamento no formato SAM/BAM e da anotação do genoma no formato GTF/GFF conta quantos *reads* foram mapeados em cada gene. O resultado dessa contagem é um arquivo tabular onde cada linha representa um gene diferente e o número de *reads* mapeados. No final do arquivo há cinco linhas que guardam o número de *reads* que não foram atribuídos a nenhum gene devido a essas razões:

1. O alinhamento não se sobrepõe com qualquer gene (`no_feature`);
2. O alinhamento se sobrepõe com mais de um gene e não é atribuído a nenhum deles (`ambiguous`);

3. A qualidade do alinhamento está abaixo do valor especificado pelo usuário (`too_low_aQual`);

4. Eles não alinharam (`not_aligned`);

5. Tomando como base a *tag* `NH`, eles alinharam em mais de um lugar na referência (`alignment_not_unique`).

Por padrão o htseq-count espera receber dados *paired-end* ordenados pelo nome do *read*. Assim, os pares estão em sequência, diminuindo a quantidade de memória necessária.

O htseq-count possui três modos de execução (`union`, `intersection-strict` e `intersection-nonempty`) para tratar os *reads* que se sobrepõem em mais de um gene. No *pipeline* os três modos são executados automaticamente, salvando cada resultado em um diretório apropriado (`union`, `strict` e `nonempty`). Além disso, no *pipeline* é possível salvar esse resultado pelo código do gene ou por seu nome.

4.6 Análise de Expressão Diferencial

Inicialmente o *pipeline* foi projetado para encontrar os genes diferencialmente expressos utilizando apenas o edgeR [91], contudo em sua versão final foi incluído também DESeq2 [73], como método complementar.

Esses programas, feitos na linguagem R [89], encontram os genes diferencialmente expressos a partir do arquivo de contagem de cada amostra. Esses arquivos são carregados no R, processados e os resultados da análise são salvos em arquivos tabulares. Além disso, são gerados alguns gráficos que ajudam na avaliação dos resultados. Todo esse trabalho é feito pelo *script* `create_execute_R_file.pl` que cria um arquivo com todos os comandos em R que devem ser executados para encontrar os genes diferencialmente expressos.

No geral, a análise de expressão diferencial inicia-se com uma matriz M com uma linha para cada gene e uma coluna para cada amostra. A posição M_{ij} armazena o número de *reads* mapeados no gene i na amostra j. Como no *pipeline* a contagem das amostras estão armazenadas em diferentes arquivos, o primeiro passo é agrupá-los em uma única matriz de contagem usando a função `readDGE`, do edgeR.

Depois de criar a matriz, genes com zeros em todas as amostras ou com uma contagem muito baixa são removidos devido a dois motivos. A razão biológica é que um gene deve ser expresso em um nível mínimo antes de ser traduzido em uma proteína ou ser biologicamente importante. A razão estatística é que contagens baixas fornecem pouca informação estatística para diferenciar entre hipótese nula ou alternativa. Em seguida, usando a função `DGEList`, a matriz é convertida em um objeto do edgeR para que as análises sejam realizadas. Esse objeto contém uma matriz `counts` que armazena a contagem e um data.frame `samples` com informações sobre as amostras. O data.frame `samples` contém uma coluna `lib.sizes` que guarda o tamanho de cada amostra, uma coluna `group` que identifica a qual grupo cada amostra pertence e o `norm.factors`

Em seguida, a função `calcNormFactors` calcula os fatores de normalização que minimizam o *log-fold change* entre as amostras para a maioria dos genes. Essa função possui quatro métodos de normalização implementados, mas o método padrão é o TMM. Fatores abaixo de 1 indicam que um grande número de fragmentos foram atribuídos a um número pequeno de genes altamente expressos na amostra, o que significa que uma cobertura menor está disponível para o restante dos genes.

A Figura 4.3 é o exemplo de um gráfico MDS (*Multiple Dimensional Scaling*) usado para verificar as diferenças entre amostras analisadas. No edgeR o gráfico MDS é gerado por meio da função `plotMDS`. Essa função avalia a similaridade entre diferentes amostras e projeta seu resultado em duas dimensões. A dimensão 1 é a que melhor separa as amostras, sem levar em consideração se são condições diferentes ou replicatas. A dimensão 2 não possui nenhuma relação com a primeira, apesar de algumas vezes parecer que uma dimensão separa replicatas e outra separa tratamentos.

Após a normalização, o edgeR precisa estimar a dispersão dos dados para só então determinar quais genes são diferencialmente expressos. Um modo de fazer isso é usar as funções `estimateCommonDisp` e `estimateTagwiseDisp` para determinar a dispersão e usar o `exactTest` para escolher os genes. A segunda opção é usar as funções `estimateGLMCommonDisp`, `estimateGLMTrendedDisp` e `estimateGLMTagwiseDisp` para dispersão e as funções `glmFit` e `glmLRT` para realizar o teste. No *pipeline* foi utilizada a segunda opção.

O resultado produzido pelos testes são acessados pela função `topTags`. Ela agrupa o resultado dos genes mais significativos, ordenando-os pelo *p-value* ou pelo *fold-*

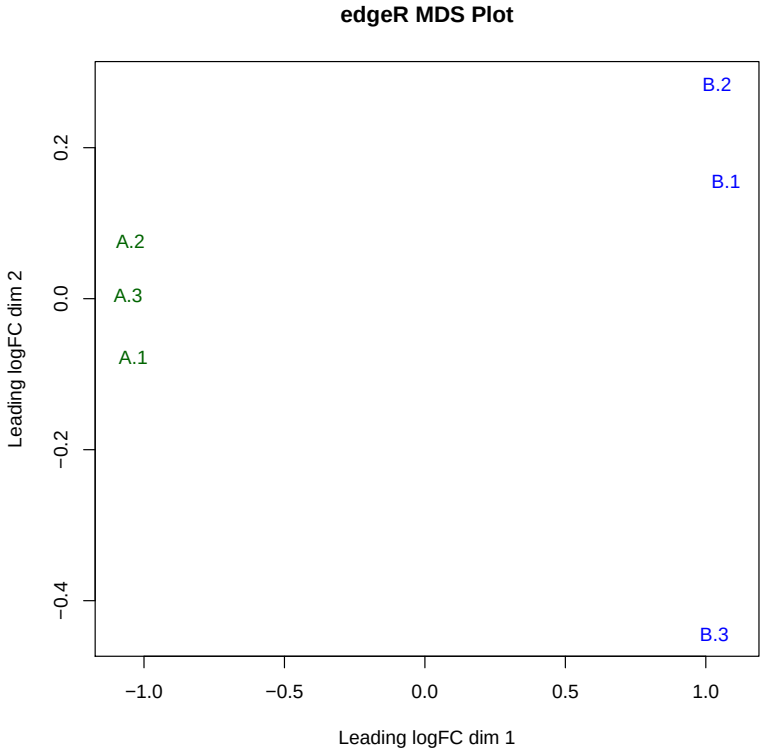


Figura 4.3: Exemplo de um gráfico MDS gerado pela função `plotMDS`. Ele mostra as similaridades e diferenças entre triplicatas de diferentes condições A e B.

change. Por padrão a função mostra os 10 genes considerados mais significativos ordenados pelo *p-value*, mas isso pode ser alterado. No *pipeline* o resultado está ordenado pelo *fold-change* e mostra todos os genes diferencialmente expressos que possuem *false discovery rate* (FDR) < 0.05 e PValue < 0.05 (esses valores são determinados no arquivo `de.conf`). Nesse caso, a lista de genes é salva em um arquivo `ec_*_deg1_*.xls` contendo os valores de logFC, logCPM, LR, PValue e FDR. O arquivo `ec_*_deg2_*.xls` contém os genes com os mesmos valores que os `ec_*_deg1_*.xls` mas que também possuem logFC > 0.5 ou logFC $<$ -0.5.

A Figura 4.4 é um gráfico MA gerado pela função `plotSmear`. Esse gráfico fornece uma visão geral de um experimento com dois grupos sendo comparados. Ele mostra a relação entre *log-fold change* (o logaritmo da relação entre os níveis de expressão para cada marcador entre dois grupos experimentais) e *log concentration* (o nível de expressão média para cada gene entre os dois grupos). Cada ponto no gráfico representa um gene. O eixo x é a expressão média sobre todas as amostras, o eixo y é o *log-fold change* entre as amostras A e as amostras B. Os genes com níveis de

expressão similares aparecem horizontalmente em torno de $y = 0$ e são representados por pontos pretos, enquanto os pontos vermelhos mostram os genes diferencialmente expressos.

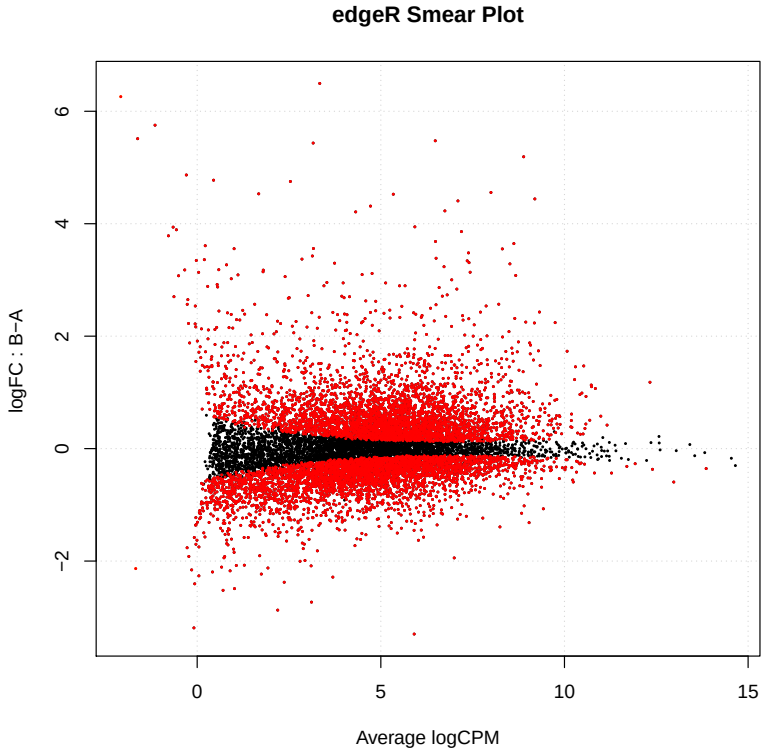


Figura 4.4: Exemplo de um gráfico MA gerado pela função `plotSmear`. Cada ponto vermelho representa graficamente um gene diferencialmente expresso e pontos pretos mostram os genes com níveis de expressão muito similares entre amostras A e B.

No *pipeline* o arquivo `ec_*_Smearcomplex1_*.pdf` representa os genes do arquivo `ec_*_deg1_*.xls`. Enquanto o arquivo `ec_*_Smearcomplex2_*.pdf` representa os genes do arquivo `ec_*_deg2_*.xls`.

Como os dados já estão armazenados em uma matriz, no DESeq2, é utilizado a função `DESeqDataSetFromMatrix` para construir o objeto `DESeqDataSet` utilizado durante a análise. No DESeq2, toda a análise é feita pela função `DESeq`. Ela normaliza, calcula a dispersão e realiza o teste de expressão diferencial, retornando um objeto `DESeqDataSet` contendo o resultado da análise.

Para acessar os genes diferencialmente expressos utiliza-se a função `results` que cria um DataFrame onde, para cada gene, estão armazenados a contagem média normalizada (`baseMean`), o *log-fold change* na base 2 que diz quanto a expressão do gene mudou de uma condição para outra (`log2FoldChange`), a estimativa de erro

associada ao `log2FoldChange` (`lfcSE`), e a incerteza de uma estimativa associada ao teste estatístico realizado (`stat`), (`pvalue`) e (`padj`).

Os genes que possuem `pvalue` < 0.05 e `padj` < 0.05 são salvos no arquivo `d2_*_deg1_*.xls`. Os genes que se encaixarem nesses valores de `pvalue` e `padj` e ainda possuirem `log2FoldChange` > 0.5 ou `log2FoldChange` $<$ -0.5 são armazenados no arquivo `d2_*_deg2_*.xls`.

Como a contagem é feita para os três modos de execução do htseq-count, as análises de expressão diferencial também são feitas para os três métodos de contagem.

Capítulo 5

Resultados Obtidos para Genômica

Neste capítulo apresentamos os resultados obtidos com a execução do *pipeline* de genômica na anotação de diferentes cepas de *M. bovis* (Seção 5.1). Além disso, apresentamos algumas análises que ajudam na determinação de funcionalidades do organismo estudado (Seção 5.2).

5.1 Genômica de *Mycobacterium bovis*

A tuberculose é uma doença causada por bactérias do gênero *Mycobacterium* que acomete animais silvestres, aves, ruminantes, suínos e humanos. A tuberculose bovina, em especial, é uma importante enfermidade infecto-contagiosa, responsável por consideráveis perdas econômicas. Estima-se que os prejuízos anuais com a tuberculose bovina sejam da ordem de U$ 3 bilhões em todo mundo [49]. Além deste aspecto, a tuberculose bovina é um problema de saúde pública [34]. Embora a maioria dos casos humanos de tuberculose sejam causados por *Mycobacterium tuberculosis*, preocupações relacionadas a *M. bovis*, que causa a tuberculose bovina, têm sido expressas devido a ocorrência de transmissões por essa bactéria entre pessoas [41].

Por ser uma doença de evolução crônica, na maioria das vezes, os animais não demonstram alterações perceptíveis ao produtor, sendo o diagnóstico feito por meio de técnicas específicas, sob a responsabilidade de um médico veterinário.

No Brasil, o diagnóstico *ante-mortem* da tuberculose bovina é feito por meio do uso de provas intradérmicas, que consistem na injeção de um composto de proteínas

da bactéria denominado Derivado Proteico Purificado ou PPD (*Purified Protein Derivative*). Animais infectados manifestam um aumento localizado da espessura da dobra da pele 72 horas após a injeção do PPD e devem ser abatidos. Já o diagnóstico *post-mortem* de *M. bovis* envolve a inspeção de carcaças com lesões macroscópicas nos frigoríficos e análise microbiológica, cujo processo pode demorar de dois a três meses, além de duas ou três semanas extras para a identificação bioquímica dos isolados.

Visando o desenvolvimento de métodos mais rápidos de diagnósticos, pesquisadores do Brasil e Argentina têm trabalhado no sequenciamento e anotação de genomas de alguns isolados de *M. bovis*, considerados de importância epidemiológica nos dois países.

Baseado na análise genômica e na identificação de SNPs em famílias de ortólogos, comuns a todos os genomas investigados é possível desenvolver um método de genotipagem de isolados de *M. bovis* da América do Sul que permita a discriminação dos isolados, uma metodologia automatizada por PCR (*Polymerase Chain Reaction*) em tempo real de alto rendimento [10].

O principal benefício esperado com o sequenciamento e a análise comparativa de diferentes cepas é o aumento da precisão do diagnóstico *post-mortem* da tuberculose bovina, em lesões encontradas em abatedouro, por meio de teste rápido com alta especificidade e sensibilidade. A partir da determinação das diferenças encontradas entre as cepas de *M. bovis*, por exemplo, a PCR em tempo real poderá fornecer resultados em 24 horas após a chegada do material ao laboratório, contrastando com o cultivo e caracterização bioquímica, que pode demandar até mais de três meses para o diagnóstico definitivo.

Para entender as diferentes cepas de *M. bovis*, foram analisadas 28 amostras, sequenciadas no Brasil, na Argentina e nos Estados Unidos. Dessas cepas, 17 isolados (Brasil e Argentina) passaram por toda análise de bioinformática desde o sequenciamento, passando pela filtragem, montagem e anotação até a fase de análise comparativa. Nosso *pipeline* para genômica, então, foi utilizado para essas 17 cepas do Brasil e Argentina.

5.1.1 Sequenciamento e Filtragem

Foram sequenciados 17 isolados de *M. bovis* utilizando dois sequenciadores. Um Ion Torrent PGM com *chip* 318 e um Illumina Mi-Seq. No total foram três rodadas de sequenciamento, sendo uma com o Ion Torrent e duas com o Illumina. O Ion Torrent produziu *reads* de até 260 bases de comprimento enquanto o Illumina gerou *reads* com comprimento de 2 x 150 nucleotídeos na primeira rodada e de 2 x 250 na segunda (*paired-end*). A Tabela 5.1 mostra o número de sequências obtidas em cada rodada de sequenciamento.

Tabela 5.1: Número de *reads* obtidos em cada rodada de sequenciamento.

Cepa	Ion Torrent	Illumina - 1	Illumina - 2
Mbovis_09-1191	1.021.507	452.273	-
Mbovis_05-567	822.140	1.264.562	-
Mbovis_05-566	681.579	236.901	-
Mbovis_04-303	651.617	260.567	-
Mbovis_534	639.979	816.128	-
Mbovis_09-1193	655.235	346.178	-
Mbovis_09-1192	752.560	1.893.282	-
Mbovis_45-08B	-	-	1.626.101
Mbovis_18-08C	-	-	1.973.913
Mbovis_61-09	-	-	2.389.730
Mbovis_32-08	-	-	1.562.882
Mbovis_49-09	-	-	2.192.943
Mbovis_08-08BF2	-	-	769.648
Mbovis_50	-	-	1.299.442
Mbovis_35	-	-	688.251
Mbovis_AN5	-	-	3.240.633
Mbovis_0822-11	-	-	1.164.997

Após avaliar a qualidade dos *reads* com o FastQC, os dados de Illumina foram filtrados com o SeqyClean, mantendo-se apenas *reads* com pelo menos 70 bases e com Phred *score* mínimo de 24.

5.1.2 Montagem e Obtenção dos *Contigs*

O mapeamento dos *reads* foi feito usando-se como referência o genoma do *M. bovis* AF2122/97. O genoma do *M. bovis* AF2122/97 possui um cromossomo com 4.345.492 bases. Possui 4.001 genes, dois quais 3.918 são CDS (*Coding DNA Sequence*), 33 pseudogenes e 50 RNAs.

O alinhamento ocorreu em duas situações diferentes. Na primeira, foram combinados dados de Illumina e Ion Torrent. Na segunda montagem foram utilizados apenas *reads* de Illumina. Mas nos dois casos, foi utilizada a opção `-very-sensitive-local` do Bowtie2. Para transformar o mapeamento em sequências FASTA, o *script* `pileup2fasta.pl` foi usado com cobertura mínima de base igual a 10, e apenas *contigs* com pelo menos 500 bases foram mantidos no resultado final, para serem anotados. Na Tabela 5.2 é possível observar o número de *contigs*, o N50 e a cobertura média obtida em cada montagem.

Tabela 5.2: Número de *contigs* obtidos em cada montagem.

Cepa	**# contigs**	**N50**	**Cobertura**
Mbovis_09-1191	86	122.051	46,4×
Mbovis_05-567	69	161.636	72,9×
Mbovis_05-566	89	149.993	28,2×
Mbovis_04-303	169	85.558	27,3×
Mbovis_534	72	178.004	50,5×
Mbovis_09-1193	89	161.475	32,0×
Mbovis_09-1192	56	163.375	96,7×
Mbovis_45-08B	72	118.080	108,7×
Mbovis_18-08C	63	156.655	119,7×
Mbovis_61-09	68	178.909	137,7×
Mbovis_32-08	83	130.260	104,5×
Mbovis_49-09	61	156.521	142,3×
Mbovis_08-08BF2	108	97.475	38,4×
Mbovis_50	84	117.592	86,7×
Mbovis_35	94	91.897	39,0×
Mbovis_AN5	70	150.219	144,0×
Mbovis_0822-11	85	96.640	76,7×

5.1.3 Anotação

A partir dos *contigs*, cada cepa foi preparada e foi enviada para anotação pelo PGAP e depositada no NCBI (`http://www.ncbi.nlm.nih.gov/bioproject/PRJNA214551`). A Tabela 5.3 mostra o número de genes, CDS, pseudogenes, rRNAs e tRNAs encontrados.

Dentre os isolados sequenciados na América do Sul, destacamos dois. O primeiro é o *M. bovis* 04-303, isolado de um javali selvagem na Argentina e altamente virulento. O segundo é o *M. bovis* AN5, usado no Brasil para a produção de PPD, um

extrato utilizado em testes de diagnóstico para a presença de tuberculose. Ambos os genomas foram publicados no *Genome Announcements* [22, 84].

Tabela 5.3: Número de elementos obtidos com a anotação feita pelo PGAP.

Cepa	**Genes**	**CDS**	**Pseudo Genes**	**rRNAs**	**tRNAs**	**ncRNAs**
Mbovis_09-1191	4.013	3.900	58	3	46	6
Mbovis_05-567	4.003	3.892	57	3	45	6
Mbovis_05-566	4.009	3.895	59	3	46	6
Mbovis_04-303	4.129	3.988	92	3	46	-
Mbovis_534	4.091	3.991	46	3	45	6
Mbovis_09-1193	4.099	3.998	47	3	45	6
Mbovis_09-1192	4.007	3.974	49	3	45	6
Mbovis_45-08B	4.086	3.988	44	3	45	6
Mbovis_18-08C	4.087	3.987	46	3	45	6
Mbovis_61-09	4.085	3.985	46	3	45	6
Mbovis_32-08	4.076	3.975	47	3	45	6
Mbovis_49-09	4.086	3.984	48	3	45	6
Mbovis_08-08BF2	4.105	4.006	45	3	45	6
Mbovis_50	4.095	3.997	44	3	45	6
Mbovis_35	4.059	3.959	45	4	45	6
Mbovis_AN5	4.084	3.938	98	3	45	-
Mbovis_0822-11	4.079	3.978	47	3	45	6

O sequenciamento e a análise desses genomas têm permitido o desenvolvimento de novas técnicas de diagnóstico. Uma dessas técnicas foi publicada em [10].

5.2 Análises

Além das informações obtidas com a anotação do genoma, foi possível realizar outras análises que permitem entender a relação entre diferentes espécies ou diferentes cepas de uma mesma espécie. Nessa seção apresentamos algumas dessas análises que foram realizadas nas diversas cepas de *M. bovis*.

Orthologsorter

A Figura 5.1 mostra a tela inicial do Orthologsorter [44], ferramenta descrita na Seção 2.3.5, onde o usuário pode escolher quais famílias de proteínas deseja ver, a partir da escolha daquelas que contém ou não genes de determinados genomas. Além desse tipo de escolha, é possível ainda escolher famílias por palavra-chave. A

última coluna da tabela mostrada na figura retorna os chamados *singletons*, que são proteínas que não fazem parte de famílias. O arquivo FASTA das proteínas do genoma pode ser obtido através do *link* com o nome de cada genoma.

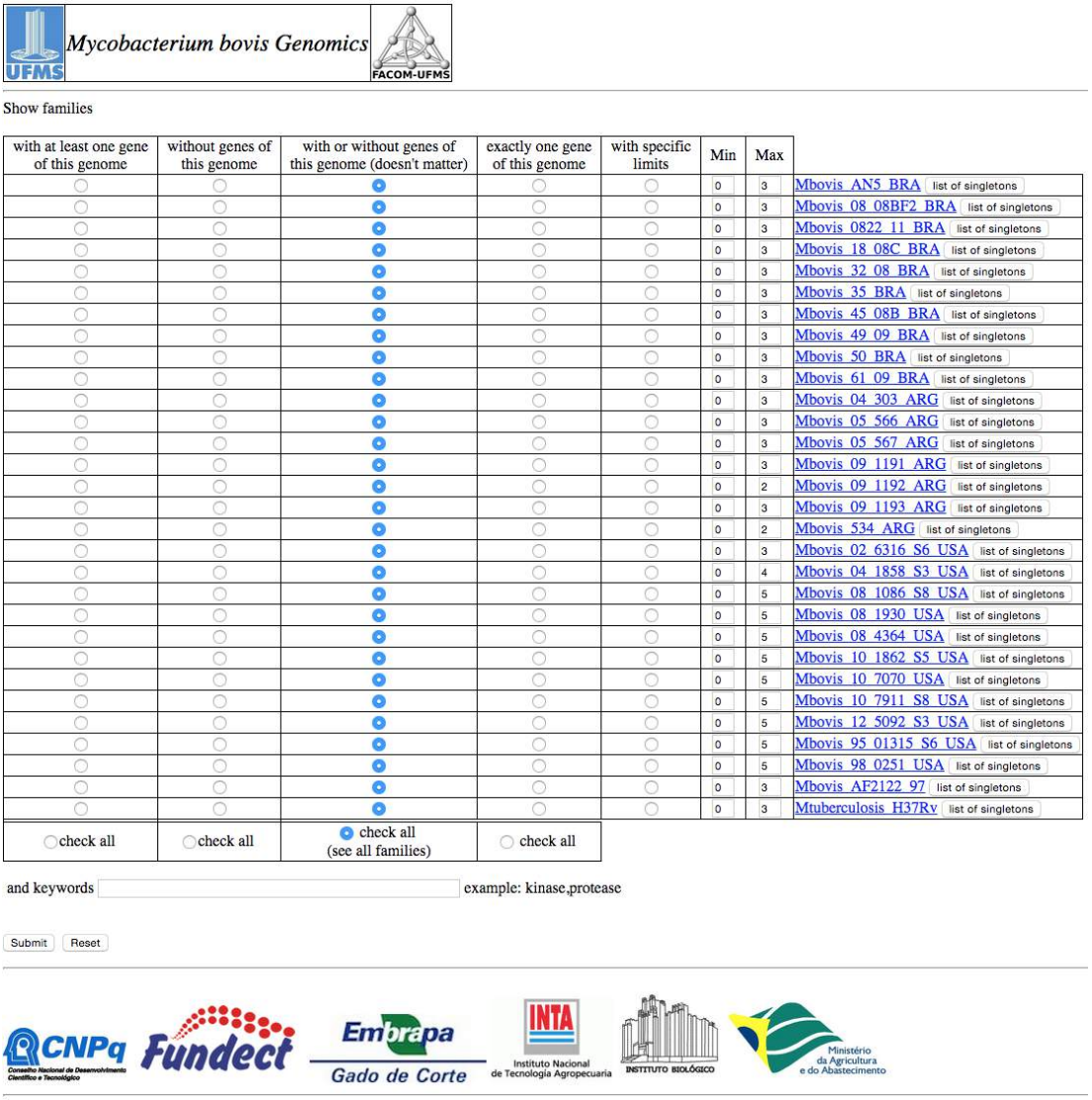


Figura 5.1: Tela inicial do Orthologsorter [44] para os dados de 30 cepas de *M. bovis*. A partir da escolha do usuário, as famílias são mostradas.

A partir da escolha do usuário, as famílias são mostradas. Como exemplo, a família identificada como 3.657 mostrada na Figura 5.2 tem 19 proteínas. As Figuras 5.3 e 5.4 mostram, respectivamente, as telas resultantes dos *links* com o número da família e `Gblocks`. A primeira mostra o arquivo no formato FASTA das proteínas da família, enquanto que a segunda mostra o alinhamento dessa família, construído pelo programa MUSCLE [40] e limpo pelo Gblocks [24], que remove colunas não-informativas.

```
========================================
>Family 3657  -- Gblocks 68/68 (100%)
91774670              nr   ESK74670.1 sulfur carrier protein ThiS [Mycobacterium bovis AN5]
90424898              nr   KFW24898.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 08-08BF2]
90559655              nr   KFW59655.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 0822-11]
90933421              nr   KFW33421.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 18-08C]
91036363              nr   KFW36363.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 32-08]
91159111              nr   KFW59111.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 35]
91248529              nr   KFW48529.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 45-08B]
91336506              nr   KFW36506.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 49-09]
91422693              nr   KFW22693.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 50]
91659254              nr   KFW59254.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 61-09]
90177575              nr   ESK77575.1 sulfur carrier protein ThiS [Mycobacterium bovis 04-303]
90217122              nr   KEY17122.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 05-566]
90315652              nr   KEY15652.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 05-567]
90616451              nr   KEY16451.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 09-1191]
90746670              nr   KFW46670.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 09-1192]
90846766              nr   KFW46766.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 09-1193]
91522885              nr   KFW22885.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 534]
31791594              nr   NP_854087.1 sulfur carrier protein ThiS [Mycobacterium bovis AF2122/97]
15607557              nr   NP_214930.1 sulfur carrier protein ThiS [Mycobacterium tuberculosis H37Rv]
------------------------
number of proteins = 19
```

Figura 5.2: Uma das famílias encontradas na comparação das 30 cepas de *M. bovis* pelo OrthoMCL [69].

```
>15607557 NP_214930.1 sulfur carrier protein ThiS [Mycobacterium tuberculosis H37Rv]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>31791594 NP_854087.1 sulfur carrier protein ThiS [Mycobacterium bovis AF2122/97]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>90177575 ESK77575.1 sulfur carrier protein ThiS [Mycobacterium bovis 04-303]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>90217122 KEY17122.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 05-566]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>90315652 KEY15652.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 05-567]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>90424898 KFW24898.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 08-08BF2]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>90559655 KFW59655.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 0822-11]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>90616451 KEY16451.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 09-1191]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>90746670 KFW46670.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 09-1192]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>90846766 KFW46766.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 09-1193]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>90933421 KFW33421.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 18-08C]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>91036363 KFW36363.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 32-08]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>91159111 KFW59111.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 35]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>91248529 KFW48529.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 45-08B]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>91336506 KFW36506.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 49-09]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>91422693 KFW22693.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 50]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>91522885 KFW22885.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 534]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>91659254 KFW59254.1 thiamine biosynthesis protein ThiS [Mycobacterium bovis 61-09]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
>91774670 ESK74670.1 sulfur carrier protein ThiS [Mycobacterium bovis AN5]
MIVVVNEQQVEVDEQTTIAALLDSLGFGDRGIAVALNFSVLPRSDWATKICELRKPVRLE
VVTAVQGG
```

Figura 5.3: Arquivo FASTA com as proteínas da família identificada como 3.657 e mostrada na Figura 5.2.

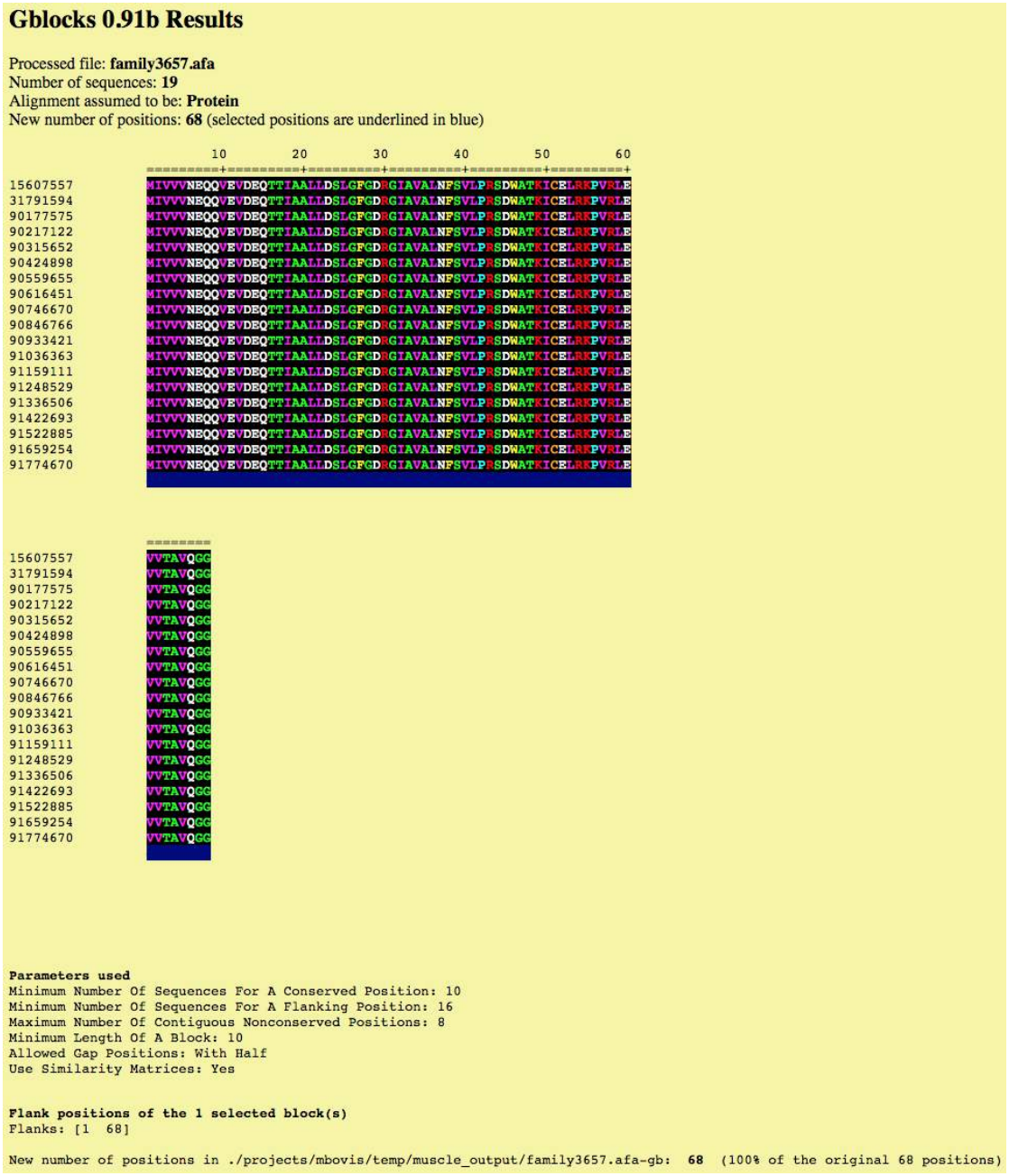


Figura 5.4: Alinhamento da família identificada como 3.657 e mostrada na Figura 5.2, feito pelo programa MUSCLE [40] e filtrado pelo programa Gblocks [24].

Além disso, cada proteína da família possui um *link* para seu registro no Genbank, quando está disponível, e ainda um *link* (`nr`), que executa o Blastp *on-line* para essa proteína contra o banco atualizado no `nr`.

Em outra análise, dessa vez envolvendo 31 cepas de *M. bovis* da América Latina e dos Estados Unidos, um total de 4.275 famílias foi obtido pelo Orthomcl [69], das

quais 3.076 foram escolhidas por conterem exatamente uma proteína de cada uma das cepas, exceto *M. bovis* AF2122/97, que foi usado como *outgroup*. Essas famílias foram então alinhadas e os seus respectivos alinhamentos foram concatenados. O alinhamento resultante, com 1.025.751 colunas, foi utilizado para a construção da árvore filogenética dessas cepas, mostrada na Figura 5.5. A árvore, como já foi citado, é construída usando o programa RAxML [102].

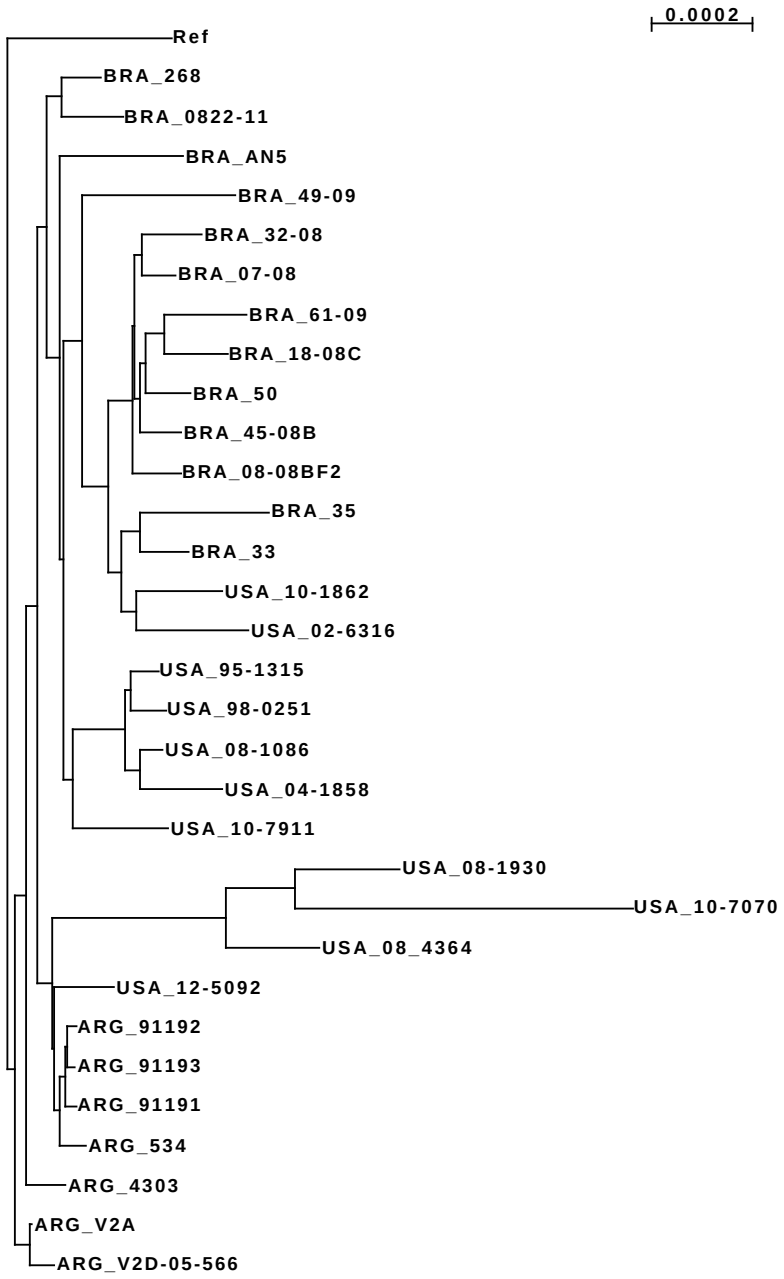


Figura 5.5: Árvore resultante do alinhamento obtido pelo Orthologsorter. A cepa usada como *outgroup* é a de *M. bovis* AF2122/97, mostrada como `Ref` na figura.

Pan/Core-genome

Usando o programa PanGP (*Pan-Genome Profile Analyze Tool*) [117], foi possível construir os gráficos *pan/core-genome* para outra análise, desta vez feita com 29 cepas de *M.bovis*. A Figura 5.6 mostra as curvas de *pan-genome* e *core-genome* para as 29 cepas analisadas.

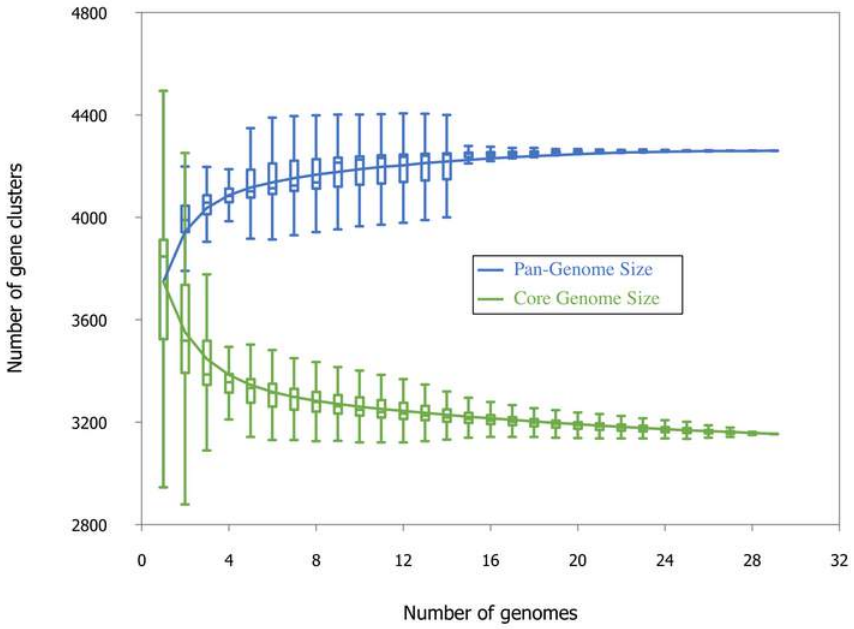


Figura 5.6: Curvas de *pan/core-genome* obtida com 29 cepas de *M. bovis*.

O principal resultado a ser analisado aqui é a verificação de quão *fechada* está a curva do *pan-genome*. A curva muito fechada, tendendo à horizontal, significa que a inclusão de novas cepas na análise não mais acrescentarão novas informações, em termos de conteúdo gênico e, dessa forma, fornece uma pista de quão completo está o conjunto de cepas analisadas. No caso desta análise em particular, o conjunto foi estabilizado com aproximadamente 4.300 genes.

O *core-genome* é o conjunto de genes comuns a todas as cepas. Quanto à curva do *core-genome*, vista na Figura 5.6, pode-se notar que esse mesmo conjunto ainda não se estabilizou, significando que ainda é possível que o número, em torno de 3.200, pode ainda diminuir com a inclusão de novas cepas na análise.

A Figura 5.7 mostra, para cada k, quantos novos genes foram incluídos, quando comparados às $k-1$ cepas. Para esta análise, fica claro que dificilmente novos genes aparecerão com o acréscimo de novas cepas na análise.

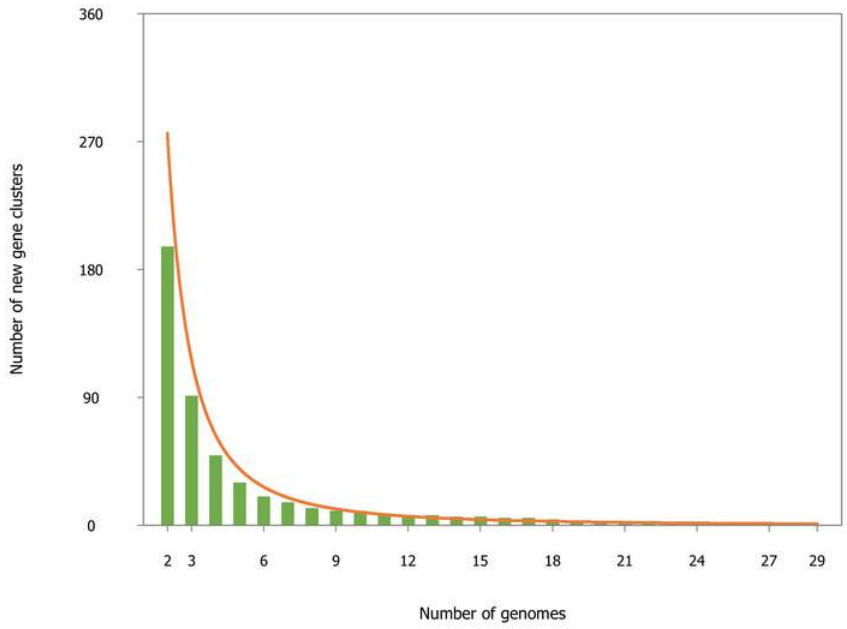


Figura 5.7: Curva de novos genes para a análise de 29 cepas de *M. bovis*.

Alinhamento Múltiplo de Ortólogos

Essa ferramenta, como já foi dito, permite a visualização de regiões com sintenia entre genes presentes em uma mesma família. A ideia é elucidar a ausência de genes importantes através das famílias de genes ortólogos.

A partir de um genoma âncora, para o qual se tem as coordenadas de todos os genes que codificam proteínas, é possível obter uma tabela, onde cada linha representa uma família de ortólogos, e cada coluna representa um genoma a ser comparado. A primeira coluna é a coluna do genoma âncora. A escolha dos genomas é feita pelo programa, cuja tela inicial pode ser vista na Figura 5.8.

A Figura 5.9 mostra um trecho do alinhamento das famílias de cinco genomas, tendo o genoma MtH37Rv (*M. tuberculosis* H37Rv) como âncora. O alinhamento mostra um grande trecho do MtH3Rv contendo genes relacionados a proteínas de membrana Mce, localizadas contiguamente nesse genoma e ao mesmo tempo não presentes nos outros quatro genomas e ainda flanqueadas por regiões sintênicas. Essa região é passível de investigação, objetivando a determinação de *primers* específicos para o genoma âncora, como no trabalho que atualmente está sendo desenvolvido [21].

Figura 5.8: Tela inicial para a escolha do genoma âncora e os outros genomas a serem comparados.

Ortholog alignments based on OrthoMCL

The anchor genome is in the first column, where genes are in order.
Each row is a family. Gene format: [+ or -] [locus_tag] ([gene order in the replicon] | [replicon])

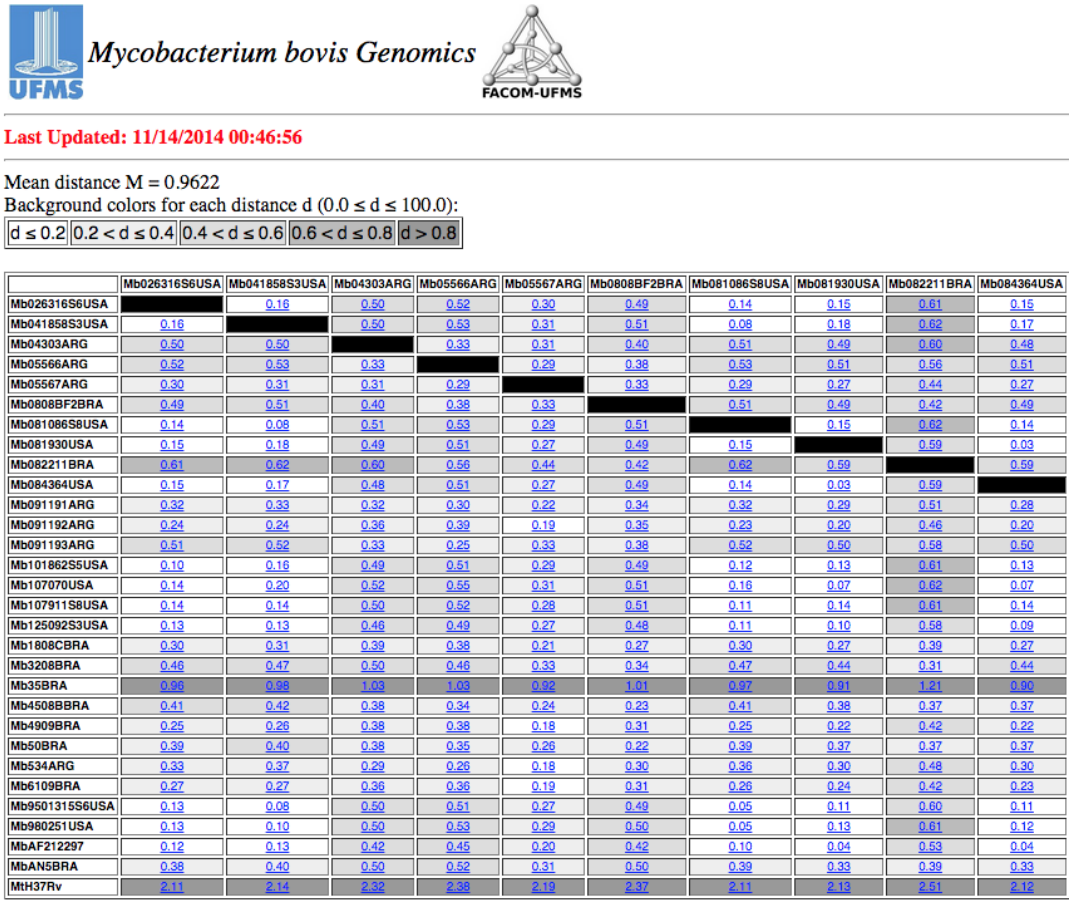
MtH37Rv	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	Product
-Rv1960c(1949|c1)	+91772150(14379|c1)	+91655249(14378|c1)	+90420460(14376|c1)	+90554634(14377|c1)	[3574] antitoxin ParD1
+Rv1961(1950|c1)	+91772023(7278|c1)	+91655203(7277|c1)	+90420414(7275|c1)	+90554622(7276|c1)	[1756] hypothetical protein
-Rv1962c(1951|c1)	+91772024(7274|c1)	+91655204(7273|c1)	+90420415(7271|c1)	+90554623(7272|c1)	[1755] ribonuclease VapC35
-Rv1962A(1952|c1)	+91772128(14879|c1)	+91655205(14878|c1)	+90420416(14876|c1)	+90554624(14877|c1)	[3699] antitoxin VapB35
-Rv1963c(1953|c1)	+91772025(7270|c1)	+91655206(7269|c1)	+90420417(7267|c1)	+90554625(7268|c1)	[1754] transcriptional repressor Mce3R
+Rv1964(1954|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] integral membrane protein
+Rv1965(1955|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] integral membrane protein
+Rv1966(1956|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] Mce family protein Mce3A
+Rv1967(1957|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] Mce family protein Mce3B
+Rv1968(1958|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] Mce family protein Mce3C
+Rv1969(1959|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] Mce family protein Mce3D
+Rv1970(1960|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] Mce family lipoprotein LprM
+Rv1971(1961|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] Mce family protein Mce3F
+Rv1972(1962|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] Mce associated membrane protein
+Rv1973(1963|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] Mce associated membrane protein
+Rv1974(1964|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] membrane protein
+Rv1975(1965|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] hypothetical protein
-Rv1976c(1966|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] hypothetical protein
+Rv1977(1967|c1)	MbAN5BRA	Mb6109BRA	Mb0808BF2BRA	Mb082211BRA	[-1] hypothetical protein
+Rv1978(1968|c1)	+91772026(7266|c1)	+91655208(7265|c1)	+90420419(7263|c1)	+90554627(7264|c1)	[1753] hypothetical protein
-Rv1979c(1969|c1)	+91772027(1234|c1)	+91655209(1233|c1)	+90420420(1231|c1)	+90554628(1232|c1)	[245] permease

Figura 5.9: Região do genoma MtH37Rv contendo proteínas de membrana Mce e não presentes nos outros 4 genomas.

Distância Genômica

O resultado da análise de distância genômica, descrita na Seção 2.3, para os mesmos 30 genomas de *Mycobacterium* é mostrado nas Figuras 5.10 e 5.11. As distâncias são calculadas pelo programa Mumi [38], a partir dos chamados MUMs (*Maximal*

Unique Matches), que são determinados pelo programa MUMmer [36].



Mycobacterium bovis Genomics

FACOM-UFMS

Last Updated: 11/14/2014 00:46:56

Mean distance M = 0.9622
Background colors for each distance d (0.0 ≤ d ≤ 100.0):

d ≤ 0.2	0.2 < d ≤ 0.4	0.4 < d ≤ 0.6	0.6 < d ≤ 0.8	d > 0.8

	Mb026316S6USA	Mb041858S3USA	Mb04303ARG	Mb05566ARG	Mb05567ARG	Mb0808BF2BRA	Mb081086S8USA	Mb081930USA	Mb082211BRA	Mb084364USA
Mb026316S6USA		0.16	0.50	0.52	0.30	0.49	0.14	0.15	0.61	0.15
Mb041858S3USA	0.16		0.50	0.53	0.31	0.51	0.08	0.18	0.62	0.17
Mb04303ARG	0.50	0.50		0.33	0.31	0.40	0.51	0.49	0.60	0.48
Mb05566ARG	0.52	0.53	0.33		0.29	0.38	0.53	0.51	0.56	0.51
Mb05567ARG	0.30	0.31	0.31	0.29		0.33	0.29	0.27	0.44	0.27
Mb0808BF2BRA	0.49	0.51	0.40	0.38	0.33		0.51	0.49	0.42	0.49
Mb081086S8USA	0.14	0.08	0.51	0.53	0.29	0.51		0.15	0.62	0.14
Mb081930USA	0.15	0.18	0.49	0.51	0.27	0.49	0.15		0.59	0.03
Mb082211BRA	0.61	0.62	0.60	0.56	0.44	0.42	0.62	0.59		0.59
Mb084364USA	0.15	0.17	0.48	0.51	0.27	0.49	0.14	0.03	0.59	
Mb091191ARG	0.32	0.33	0.32	0.30	0.22	0.34	0.32	0.29	0.51	0.28
Mb091192ARG	0.24	0.24	0.36	0.39	0.19	0.35	0.23	0.20	0.46	0.20
Mb091193ARG	0.51	0.52	0.33	0.25	0.33	0.38	0.52	0.50	0.58	0.50
Mb101862S5USA	0.10	0.16	0.49	0.51	0.29	0.49	0.12	0.13	0.61	0.13
Mb107070USA	0.14	0.20	0.52	0.55	0.31	0.51	0.16	0.07	0.62	0.07
Mb107911S8USA	0.14	0.14	0.50	0.52	0.28	0.51	0.11	0.14	0.61	0.14
Mb125092S3USA	0.13	0.13	0.46	0.49	0.27	0.48	0.11	0.10	0.58	0.09
Mb1808CBRA	0.30	0.31	0.39	0.38	0.21	0.27	0.30	0.27	0.39	0.27
Mb3208BRA	0.46	0.47	0.50	0.46	0.33	0.34	0.47	0.44	0.31	0.44
Mb35BRA	0.96	0.98	1.03	1.03	0.92	1.01	0.97	0.91	1.21	0.90
Mb4508BBRA	0.41	0.42	0.38	0.34	0.24	0.23	0.41	0.38	0.37	0.37
Mb4909BRA	0.25	0.26	0.38	0.38	0.18	0.31	0.25	0.22	0.42	0.22
Mb50BRA	0.39	0.40	0.38	0.35	0.26	0.22	0.39	0.37	0.37	0.37
Mb534ARG	0.33	0.37	0.29	0.26	0.18	0.30	0.36	0.30	0.48	0.30
Mb6109BRA	0.27	0.27	0.36	0.36	0.19	0.31	0.26	0.24	0.42	0.23
Mb9501315S6USA	0.13	0.08	0.50	0.51	0.27	0.49	0.05	0.11	0.60	0.11
Mb980251USA	0.13	0.10	0.50	0.53	0.29	0.50	0.05	0.13	0.61	0.12
MbAF212297	0.12	0.13	0.42	0.45	0.20	0.42	0.10	0.04	0.53	0.04
MbAN5BRA	0.38	0.40	0.50	0.52	0.31	0.50	0.39	0.33	0.39	0.33
MtH37Rv	2.11	2.14	2.32	2.38	2.19	2.37	2.11	2.13	2.51	2.12

Figura 5.10: Trecho inicial da matriz de distância genômica, calculada entre cada par dos 30 genomas de *M. bovis*. Cada posição da matriz representa a distância genômica calculada pelo programa Mumi [38]. O *link* em cada célula retorna o gráfico resultante da comparação feita pelo programa Mummer [36] para os genomas correspondentes. Um exemplo é mostrado na Figura 5.11.

A árvore filogenética apresentada na Figura 5.12 foi construída usando o pacote Phylip [46] a partir das distâncias determinadas por Mumi, usando o genoma MtH37Rv como *outgroup*.

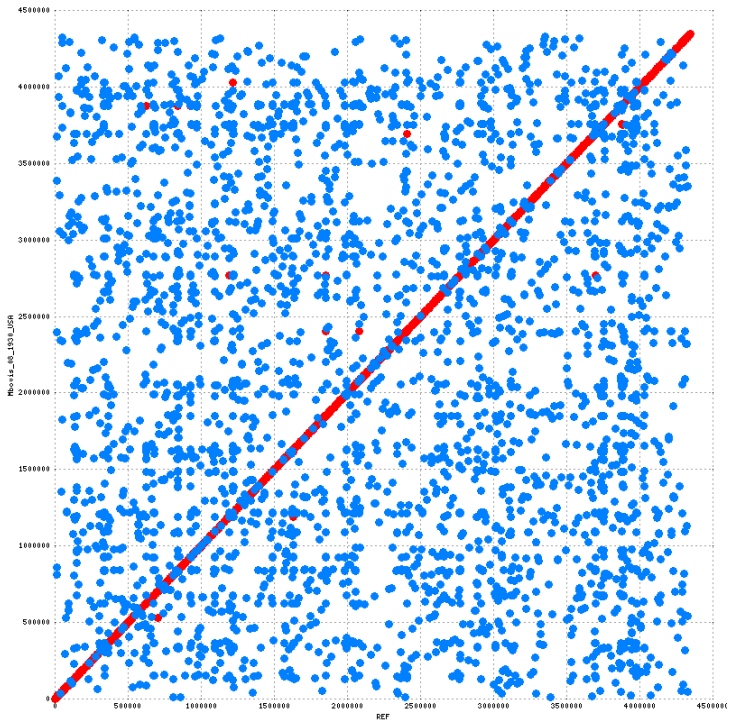


Figura 5.11: Gráfico resultante da comparação genômica dos genomas Mb026316S6USA e Mb081930USA, feita pelo programa Mummer [36].

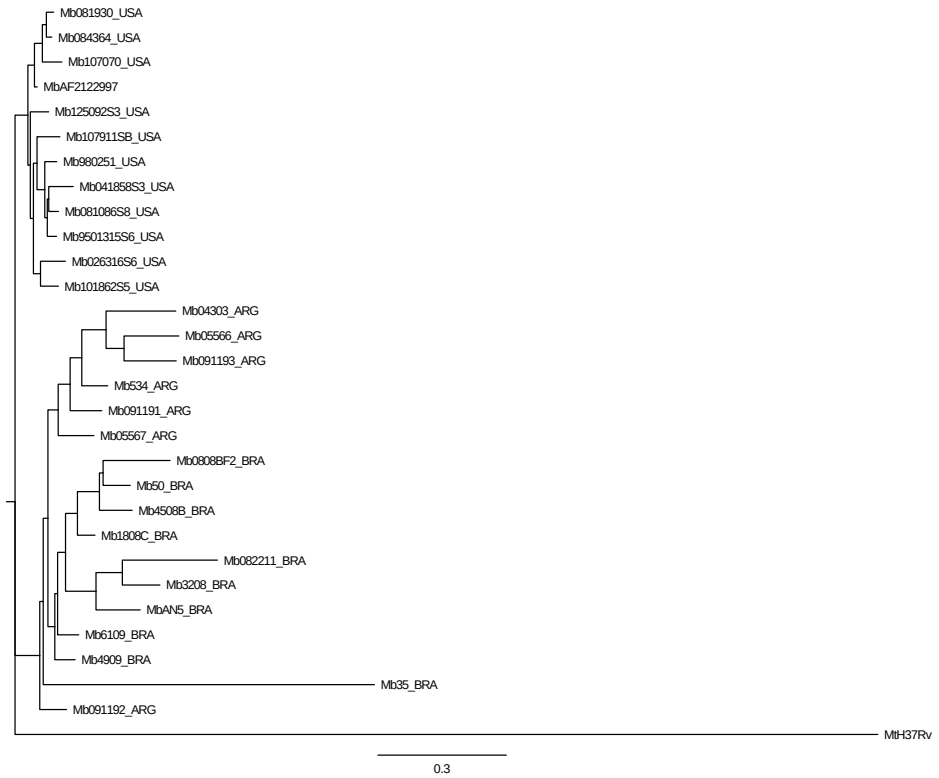


Figura 5.12: Árvore gerada com o uso do pacote Phylip [46] a partir das distâncias calculadas pelo programa Mumi, para os 30 genomas envolvidos na análise.

Transferência Horizontal de Genes

A Figura 5.13 mostra o resultado da análise de transferência horizontal de genes, usando o programa Alien Hunter, conforme descrito na Seção 2.3.5. Cada pico representa uma região de 500 bases que apresenta composição diferenciada de dinucleotídeos, quando comparada com o restante do genoma. Essas regiões, que podem ser causadas por modificações ocorridas ao longo do tempo (*in vitro* no caso de AN5), podem também apresentar genes candidatos a serem rotulados como genes de transferência horizontal.

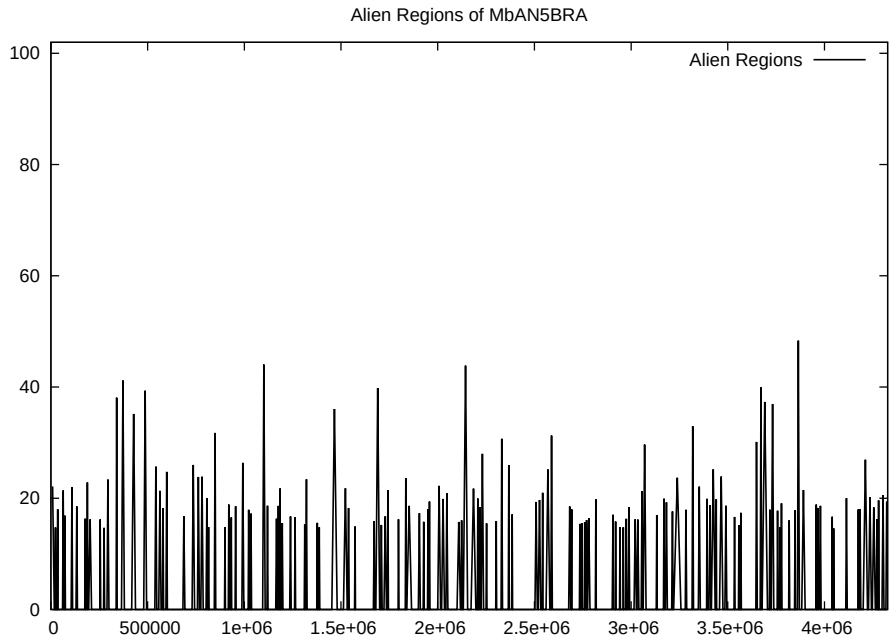


Figura 5.13: Regiões anômalas encontradas pelo programa Alien Hunter [110] no genoma MbAN5BRA.

Para cada uma das chamadas *regiões anômalas*, os genes candidatos a serem rotulados como vindos de transferência horizontal são também mostrados, como pode ser visto no exemplo mostrado na Figura 5.14.

```
gi|91774474 ESK74474.1 hypothetical protein
present in 5 genomes:
- Mbovis_04_303_ARG
- Mbovis_05_566_ARG
- Mbovis_05_567_ARG
- Mbovis_09_1191_ARG
- Mbovis_AN5_BRA
```

Figura 5.14: Gene do genoma MbAN5BRA possivelmente adquirido por transferência horizontal.

Capítulo 6

Resultados Obtidos para Transcritômica

Neste capítulo apresentamos os resultados obtidos com a execução do *pipeline* de expressão diferencial em três projetos, na análise de três transcritomas diferentes. Na primeira análise (Seção 6.1) são estudados transcritos de linfócitos T humanos de indivíduos saudáveis e indivíduos transplantados com diferentes níveis de tolerância. Já no segundo (Seção 6.2) estudo também são avaliados linfócitos T humanos, mas nesse caso são analisados linfócitos T sem tratamento e linfócitos tratados com diferentes anticorpos. No último caso (Seção 6.3), são avaliados células de camundongo sem infecção e células infectadas com diferentes formas infectivas do fungo *Fonsecaea pedrosoi*.

Os três projetos aqui descritos estão sendo realizados no Instituto de Biologia Molecular da UnB e encontram-se em fase final de execução. Publicações com os resultados alcançados estão sendo escritas.

6.1 Transcritoma de Pacientes Transplantados

O transplante de órgãos surgiu como uma solução para diversas patologias e vem se consolidando como uma abordagem de sucesso no tratamento da falência permanente de órgãos ou tecidos. No entanto, o transplante encontra na rejeição um de seus maiores desafios. Para evitar a rejeição, drogas imunossupressoras potentes têm sido utilizadas, mas que também podem causar diversos problemas para

o indivíduo transplantado. Dentre eles, podemos citar a maior suscetibilidade ao desenvolvimento de infecções, de neoplasias, assim como nefro e angiotoxicidade. Os imunossupressores agem principalmente bloqueando a produção de citocinas e a proliferação celular, importantes principalmente na rejeição mediada por células T [72].

O sistema imune é composto por uma gama de circuitos efetores e reguladores interconectados e que montam um sistema complexo capaz de reparar, manter e defender o indivíduo. De uma forma resumida, o sistema imune reage, constantemente, à diferentes experiências imunológicas, sejam elas de manutenção ou de perturbação, que resultam na ativação de um conjunto de células e moléculas, entre as quais os linfócitos T, que são células que auxiliam o organismo a se defender de bactérias, fungos, vírus e outras células estranhas ao organismo da pessoa, tornando-se um problema no transplante de órgãos [28].

Apesar do sucesso de diversos protocolos para indução de tolerância ao aloenxerto em modelos experimentais animais, a sua tradução para a clínica permanece um desafio. As diferenças entre os resultados experimentais em roedores e aqueles observados em animais de maior porte e nos ensaios clínicos podem ser decorrentes de múltiplos fatores como: (i) diferenças biológicas entre as espécies; (ii) maior heterogeneidade e diversidade das respostas imunes em humanos e animais maiores; (iii) efeito da imunidade heteróloga induzida por constantes exposições a microorganismos, que pode influenciar a resposta ao enxerto, por exemplo, desencadeando rejeição.

O desenvolvimento de novas técnicas cirúrgicas associadas a drogas imunossupressoras tornou rotineiro o transplante de órgãos. No entanto, a rejeição pós-transplantes ou falência do enxerto, devido à resposta do paciente, ainda é um fator limitante.

Sabe-se ainda que uma parcela dos pacientes apresenta um estado de estabilidade funcional do enxerto após a retirada de imunossupressores. Esses pacientes são chamados de tolerantes operacionais e seu estado de tolerância operacional é uma certificação de que a tolerância ao aloenxerto é possível em humanos [16].

O objetivo desse primeiro projeto é o de analisar o perfil transcritômico de linfócitos T de indivíduos saudáveis e de indivíduos transplantados com 3 níveis de tolerância (tolerante operacional, estável e rejeição crônica), que constituirá numa ferramenta importante para o estudo da expressão gênica e sua regulação.

O conhecimento do perfil transcritômico de linfócitos T forma uma excelente ferramenta, não só para aumentar a compreensão dos mecanismos envolvidos no estado de tolerância operacional, mas também para o seu diagnóstico futuro, permitindo a identificação de pacientes que podem beneficiar-se da diminuição ou retirada de drogas imunossupressoras. Esse conhecimento pode ainda subsidiar o desenvolvimento de novas terapias.

Nesse projeto foram sequenciadas células de linfócitos T de quatro grupos clínicos diferentes: saudável (SA), estável (ES), tolerante operacional (TO) e rejeição crônica (RC). Cada grupo possui duplicatas totalizando oito amostras sequenciadas. O sequenciamento foi feito em um aparelho Illumina Hi-Seq que gerou *reads paired-end* com 100 bases de comprimento.

O Cutadapt foi executado permitindo uma taxa máxima de erro de 15% (`-e` = 0.15), tamanho mínimo do *read* igual a 15 (`-m` = 15) e se a sobreposição entre o *read* e o adaptador for menor que 5 (`-O` = 5), o *read* não é modificado. O PRINSEQ removeu sequências com qualidade abaixo de 20 da extremidade 3' (`-trim_qual_right`), com qualidade mínima de 20 (`-min_qual_mean`) e tamanho mínimo 70 (`-min_len`). Além disso, o tamanho da janela deslizante usado para calcular o índice de qualidade foi 20 (`-trim_qual_window`), o tamanho do deslocamento usado para mover a janela foi 10 (`-trim_qual_step`) e o cálculo de qualidade usado foi a média (`-trim_qual_type`).

Na Tabela 6.1 são mostradas a quantidade inicial de *reads* de cada amostra e entre parênteses o total resultante depois da filtragem e do pareamento.

Tabela 6.1: Total de *reads* sequenciados em cada um dos quatros grupos clínicos analisados. Entre parênteses o total resultante depois da filtragem e do pareamento.

Condições	**Replicata 1**	**Replicata 2**
Paciente saudável (SA)	14.881.007 (13.422.917)	21.534.872 (19.534.670)
Paciente tolerante operacional (TO)	21.026.931 (19.158.964)	17.211.708 (15.388.788)
Paciente estável (ES)	17.851.640 (16.250.852)	17.576.125 (16.023.403)
Paciente com rejeição crônica (RC)	17.366.087 (15.936.981)	19.062.929 (17.128.849)

Para todos os conjuntos de dados, o TopHat2 foi usado para alinhar os *reads* no genoma humano (Homo_sapiens.GRCh37.75.fa), disponível no site do Ensembl, sem alterar os valores padrões de execução do programa. Em seguida, o htseq-count foi executado considerando que os dados não são `strand-specific` (`-s`) e apenas *reads* com qualidade de alinhamento mínima de 10 (`-a`) fossem considerados.

Depois que todas as contagens foram realizadas, o grupo saudável foi comparado contra todos os demais grupos. Na Figura 6.1, podemos verificar que não há uma divisão muito clara entre a amostra controle (SA) e os demais grupos testados usando o modo `union` de contagem do htseq-count.

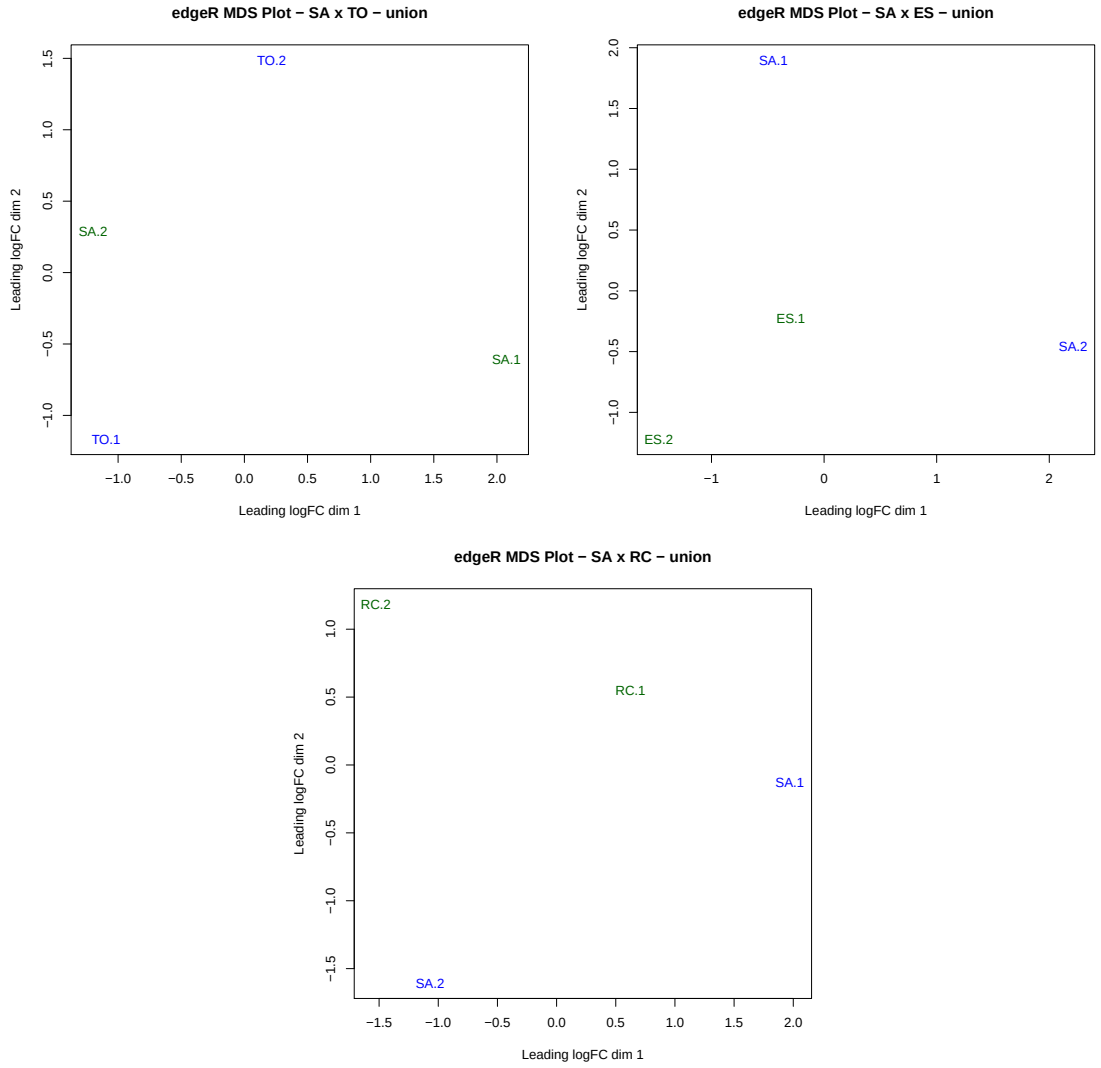


Figura 6.1: Gráficos MDS para as comparações entre os grupos clínicos analisados. Cada gráfico mostra a similaridade entre duas amostras testadas a partir do modo `union` de contagem do htseq-count.

A Tabela 6.2 mostra o número de genes diferencialmente expressos em cada um dos modos do htseq-count e para cada programa utilizado. Mas a Figura 6.2 mostra apenas a representação gráfica dos genes diferencialmente expressos encontrados pelo edgeR, a partir do modo de contagem `union` do htseq-count.

Tabela 6.2: Quantidade de genes diferencialmente expressos entre os grupos clínicos analisados. O total apresentado pode variar dependendo do programa, do modo de contagem e das diferentes restrições especificadas.

		SA X TO		SA X ES		SA X RC	
Modo	***Software***	**deg1**	**deg2**	**deg1**	**deg2**	**deg1**	**deg2**
`union`	edgeR	2	2	1	1	36	36
	DESeq2	2	2	0	0	57	57
`strict`	edgeR	3	3	3	3	115	115
	DESeq2	2	2	2	2	130	130
`nonempty`	edgeR	2	2	2	2	31	31
	DESeq2	2	2	0	0	57	57

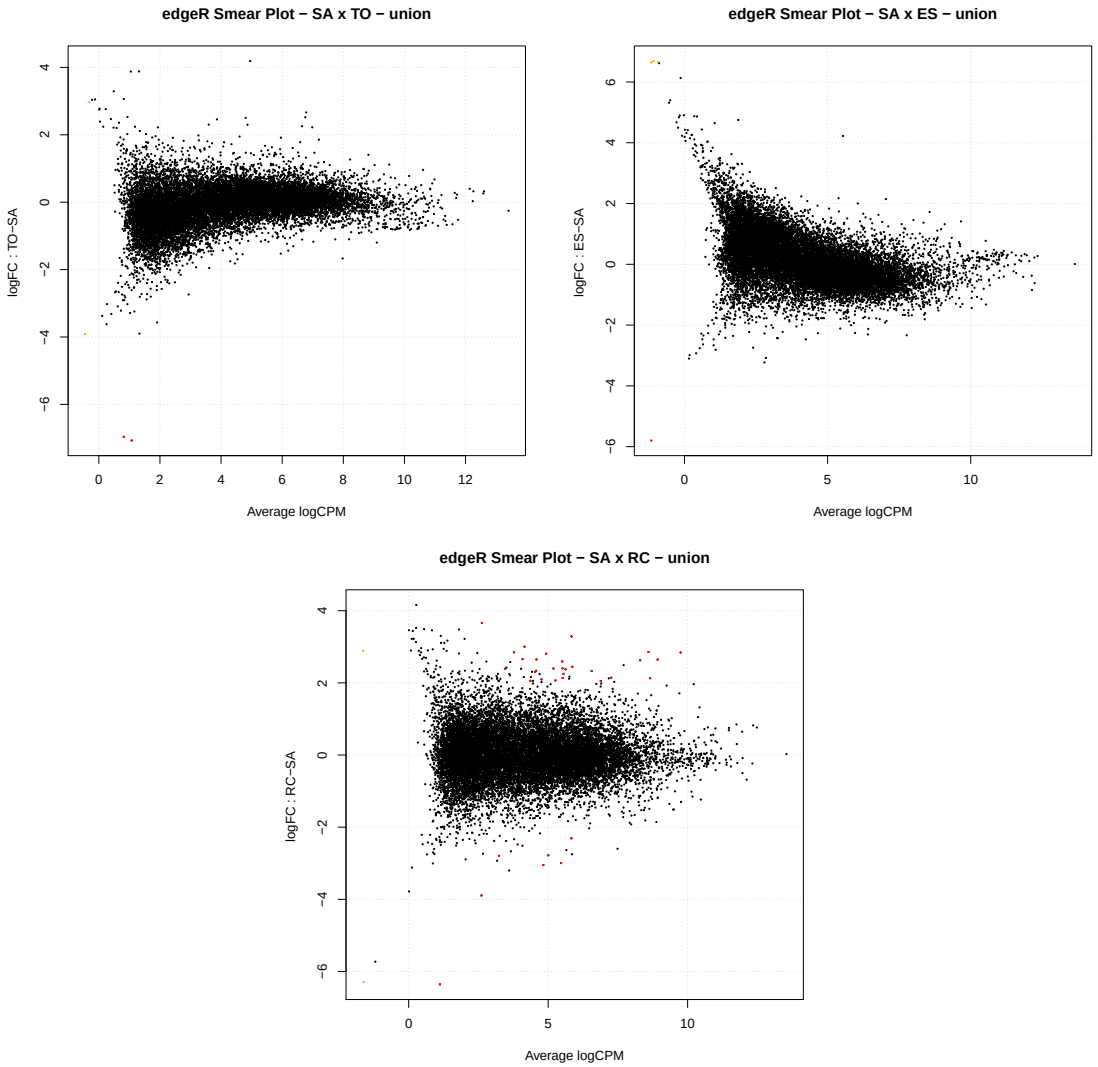


Figura 6.2: Gráficos MA gerados para a análise dos diferentes grupos clínicos. Cada gráfico mostra os genes diferencialmente expressos com FDR < 0.05 e PValue < 0.05 para o modo de contagem `union` do htseq-count.

6.2 Transcritoma de Linfócitos T tratados com Diferentes Anticorpos

Anticorpos são biomoléculas com atividade imunológica e constituem o braço humoral[1] do sistema imune, servindo de ligantes para os mais variados antígenos, por meio de seus determinantes antigênicos, os epítopos. Os anticorpos são produzidos a partir dos linfócitos B e, assim como os linfócitos T, são de origem clonal, oriundos de um processo único de recombinação gênica, de forma que cada clone de linfócito B produz um único anticorpo.

O desenvolvimento da tecnologia de produção de anticorpos monoclonais (mAb) em 1975 por Kolher & Milstein [57] revolucionou o estudo da imunologia, permitindo o uso de uma ferramenta específica para um epítopo. Utilizando esse método é possível obter anticorpos direcionados para um determinado antígeno ou um determinado tipo celular. Na clínica, os mAbs foram recebidos como uma nova geração de fármacos, com grande especificidade e poder terapêutico.

Em 1986, o primeiro anticorpo monoclonal murino[2] foi aprovado para uso clínico pelo FDA (órgão regulador americano) para uso em eventos de rejeição aguda de enxertos [32, 51]. Esse anticorpo era específico para o antígeno CD3, desenvolvido inicialmente como um marcador de linfócitos T (OKT3), e mostrou-se eficiente na depleção desses linfócitos em pacientes e, assim, passou a ser comercializado para uso em humanos.

Apesar do grande potencial, o uso de anticorpos monoclonais apresentaram alta toxicidade e imunogenicidade. Esses anticorpos possuem moléculas murinas que desencadeiam uma resposta imunológica conhecida como HAMA (*Human Anti-Mouse Antibody*) [56]. As primeiras tentativas para minimizar o potencial imunogênico desses anticorpos foram feitas por meio de engenharia genética, em que se construiu um anticorpo quimérico contendo regiões murinas e regiões humanas [81]. Embora os anticorpos quiméricos fossem menos imunogênicos que os murinos, eles geravam uma reação imunológica conhecida como HACA (*Human Antichimeric Antibody*).

No final da década de 90, uma nova revolução foi iniciada. A associação da técnica

[1] Braço humoral é um termo comumente utilizado em imunologia para se referir à imunidade mediada por anticorpos

[2] Anticorpos murinos são aqueles que possuem sequências originadas de camundongos

de geração de mAbs com técnicas de DNA recombinante permitiu o desenvolvimento de uma segunda geração de biofármacos, os anticorpos recombinantes.

Tentando reduzir a resposta imune do hospedeiro, foram desenvolvidas técnicas de humanização de anticorpos. Nessa técnica, cria-se um anticorpo formado em sua maioria por cadeias humanas.

Atualmente, os anticorpos anti-CD3 são representantes de uma nova categoria de agentes imunoterapêuticos, podendo promover a cura de auto-imunidades estabelecidas ou permitir uma sobrevida duradoura de órgãos transplantados provavelmente pela sua capacidade de induzir células T reguladoras [25].

O projeto visa analisar o perfil transcritômico de linfócitos T humanos para a compreensão dos mecanismos envolvidos na atividade imunossupressora dos anticorpos anti-CD3. Esse conhecimento poderá ser usado na geração de novos biofármacos.

Para realizar essa análise, as bibliotecas foram sequenciadas no Illumina Hi-Seq, que gerou *reads paired-end* com 151 nucleotídeos cada. As amostras pertencem a grupos diferentes: uma amostra controle de linfócitos T sem tratamento com qualquer anticorpo (NT) e quatro amostras de linfócitos T tratados com anticorpos diferentes (MUR, OKT3, RVL, TVL). Cada condição possui duplicatas totalizando dez amostras sequenciadas.

Depois de analisados com o FastQC, os dados foram filtrados apenas com o PRINSEQ. Ele foi executado com os mesmos parâmetros do projeto anterior: `-trim_qual_right` = 20, `-min_qual_mean` = 20, `-min_len` = 70, `-trim_qual_window` = 20, `-trim_qual_step` = 10 e `-trim_qual_type` mean. Na Tabela 6.3, é possível verificar a quantidade inicial de *reads* obtidos pelo sequenciamento bem como o total resultante da filtragem e que foi usado pelo TopHat2 para fazer o alinhamento contra genoma humano completo.

Tabela 6.3: Quantidade de *reads* inicial e entre parênteses o total depois de filtrados.

Condições	**Replicata 1**	**Replicata 2**
NT	29.811.880 (27.921.019)	30.722.484 (28.823.076)
MUR	31.137.182 (29.323.615)	32.571.139 (30.615.686)
OKT3	30.362.118 (28.037.699)	29.717.962 (27.421.662)
RVL	28.856.199 (27.242.187)	27.316.547 (25.679.646)
TVL	29.184.493 (27.197.007)	30.134.417 (28.095.389)

Após realizar o alinhamento e a contagem de cada amostra usando o htseq-count, a análise de expressão diferencial foi realizada par a par entre o controle (NT) e as demais condições (MUR, OKT3, RVL e TVL). Na Figura 6.3 é possível verificar todas as similaridades entre as diferentes condições avaliadas. A análise entre NT e OKT3 é a que apresenta a melhor divisão entre as condições, pois há uma clara separação entre os dois grupos.

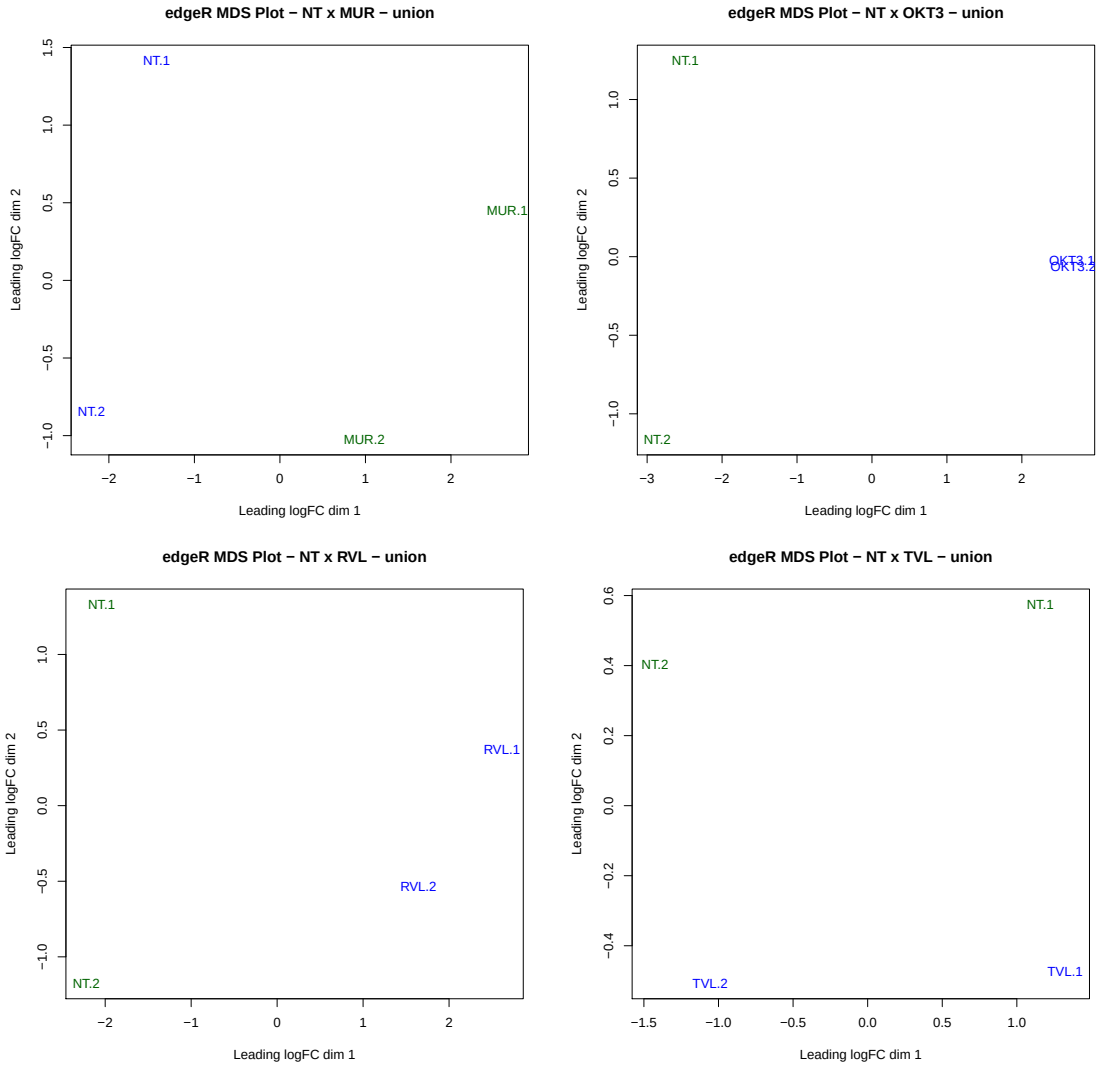


Figura 6.3: Gráficos MDS para a análise do perfil transcritômico de linfócitos T.

O número de genes diferencialmente expressos encontrados por cada aplicativo a partir dos diferentes modos de contagem do htseq-count podem ser vistos na Tabela 6.4. Mas na Figura 6.4 é possível verificar os genes DE com FDR < 0.05 e PValue < 0.05 usando o modo de contagem `union`.

Tabela 6.4: Quantidade de genes diferencialmente expressos resultantes da análise do perfil transcritômico de linfócitos T a partir dos três modos de contagem o htseq-count.

		NT x MUR		**NT x OKT3**		**NT x RVL**		**NT x TVL**	
Modo	***Software***	**deg1**	**deg2**	**deg1**	**deg2**	**deg1**	**deg2**	**deg1**	**deg2**
`union`	edgeR	1.059	1.059	5.824	5.824	1.724	1.724	4	4
	DESeq2	1.061	1.061	5.408	5.408	1.732	1.732	1	1
`strict`	edgeR	1.079	1.079	5.810	5.810	1.741	1.741	4	4
	DESeq2	1.075	1.075	5.393	5.393	1.765	1.765	1	1
`nonempty`	edgeR	1.073	1.073	5.973	5.973	1.733	1.733	4	4
	DESeq2	1.072	1.072	5.518	5.518	1.783	1.783	3	3

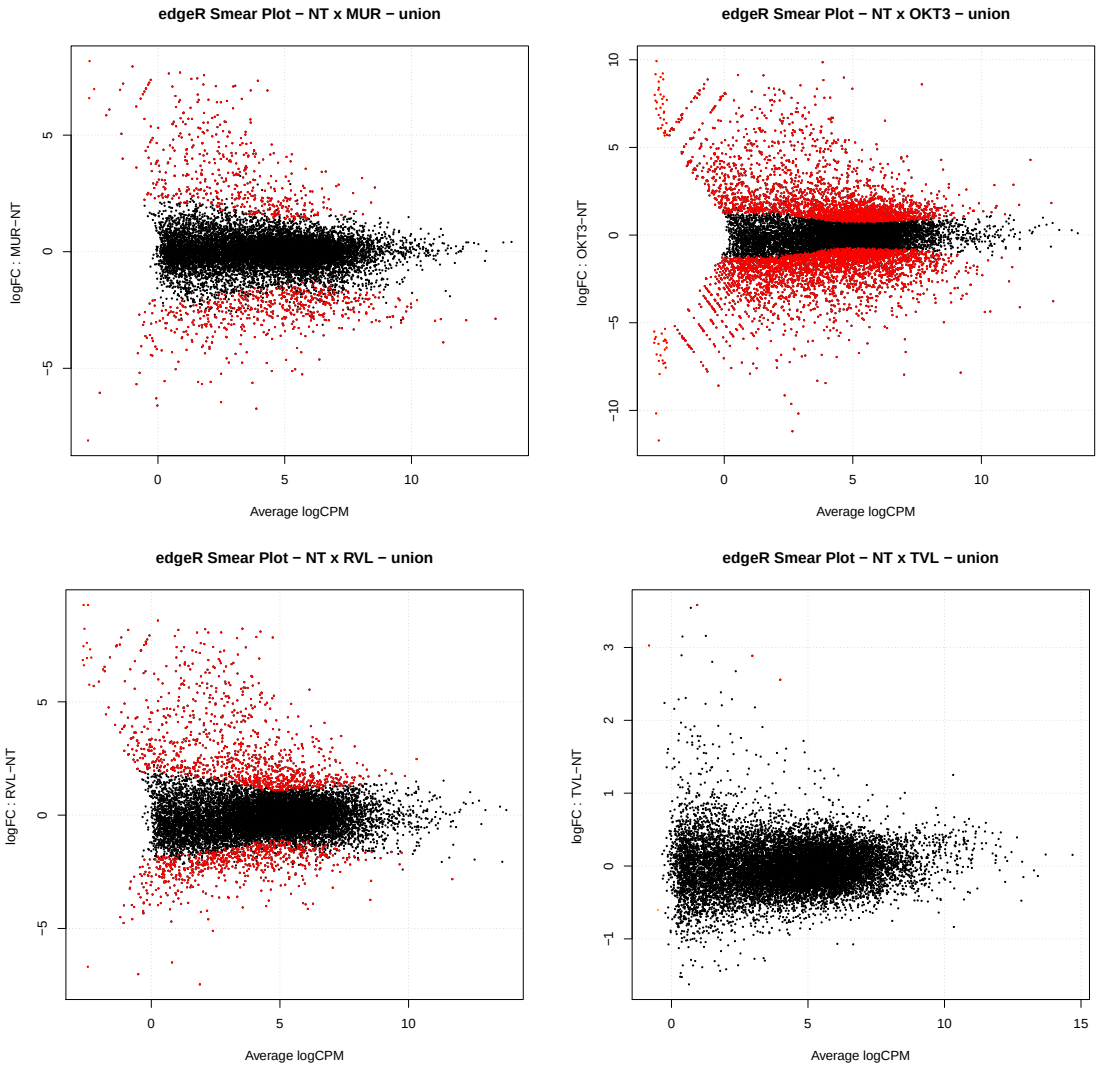


Figura 6.4: Gráficos MA para a análise do perfil transcritômico de linfócitos T. Cada gráfico mostra os genes DE com FDR < 0.05 e PValue < 0.05 para o modo de contagem `union` do htseq-count.

6.3 Interação de Macrófagos Murinos e *Fonsecaea pedrosoi*

A cromoblastomicose (CBM) é uma micose subcutânea crônica, causada por fungos filamentosos, dimórficos, pertencentes à família dos Dematiaceous. A infecção geralmente ocorre devido a lesões causadas por espinhos ou lascas de madeiras contaminados. Os agentes fúngicos desenvolvem-se como pequenos aglomerados de células conhecidos como corpos muriformes. Vários meses após a lesão, nódulos indolores aparecem na região afetada progredindo para placas escamosas e verrugosas [79].

Pacientes com CBM ainda são tidos como desafio terapêutico, principalmente devido à natureza resistente da doença, sobretudo nas formas clínicas mais severas. Para o tratamento, são utilizadas várias possibilidades de terapia, as quais envolvem tratamentos físicos, químicos e a combinação de ambos. O tratamento deve ser continuado até que haja resolução clínica, que ocorre geralmente após vários meses de terapia [79].

Os fungos comumente associados à doença são: *Fonsecaea pedrosoi*, *Phialophora compacta*, *Phialophora verrucosa*, *Rhinocladiella aquaspersa* e *Cladosporium carrionii*, os quais são encontrados como saprófitos[3] no solo e em plantas [79].

A patogenia de infecções fúngicas envolve vários fatores de virulência que permitem a sobrevivência e a persistência do parasita no interior do hospedeiro. Tais fatores envolvem geralmente: a produção de fosfolipases, proteases e elastases; a habilidade de alternar vias metabólicas indispensáveis à sobrevivência intracelular; termotolerância; e, por fim, a habilidade de existir em diferentes formas, alternando de uma para outra durante a infecção [4, 53].

A parede celular dos fungos varia sua composição, dependendo de sua morfologia, estágio e ambiente de crescimento. A melanina promove a sobrevivência do fungo em diferentes microambientes, aumentando sua resistência aos mecanismos efetores da resposta imune do hospedeiro bem como reduzindo a sua suscetibilidade aos antifúngicos [104]. Tais componentes podem se apresentar em diferentes concentrações dependendo da espécie e, por vezes, dependendo da forma fúngica em questão, acarretando em profundas mudanças na interação parasito-hospedeiro [76, 104].

[3]Fungos saprófitos são aqueles que se alimentam de matéria orgânica animal ou vegetal morta.

A maioria dos estudos relacionados à CBM é realizada em pacientes com a doença já estabelecida, carecendo a literatura de maiores informações acerca da patogenia da doença. Este trabalho visa à avaliação da interação parasito-hospedeiro das formas infectivas do fungo *F. pedrosoi* e seus respectivos impactos na patogenia da doença.

O objetivo desse projeto é o estudo de duas formas do fungo polimórfico *F. pedrosoi*: conídios e células muriformes. Elas representam os extremos do dimorfismo do fungo.

A análise de expressão diferencial é o ponto inicial para entender de que forma o fungo interage com as células do hospedeiro e dita o caminho da resposta imunológica. Assim, a forma infectiva do fungo, bem como a sua concentração no momento da infecção, tendem a ser fatores determinantes para o desenvolvimento de uma micose.

Foram sequenciadas uma amostra controle (MO), duas amostras de conídios (MOFp e MOFpTc) e uma amostra de células muriformes (MOCm). Todas as condições possuem triplicatas biológicas. As amostras foram sequenciadas pela abordagem *paired-end* com 100 bases por fragmento em um aparelho Illumina Hi-Seq. Na Tabela 6.5 são mostrados o número de *reads* de cada amostra.

Tabela 6.5: Total de *reads* sequenciados das diferentes formas do fungo *F. pedrosoi*.

Condições	**Replicata 1**	**Replicata 2**	**Replicata 3**
Controle (MO)	27.463.693	33.871.657	27.007.912
Células muriformes (MOCm)	33.728.448	29.260.699	23.575.952
Conídios (MOFp)	31.278.365	33.927.974	26.526.969
Conídios tratados com triciclazol (MOFpTc)	23.241.573	25.456.980	30.818.883

Assim como nos projetos já descritos, essa análise utilizou os mesmos parâmetros, apesar de se tratar de outro organismo estudado. Na Tabela 6.6 é possível ver o total de *reads* que foram submetidos ao TopHat2. Durante o alinhamento usou-se como referência o genoma do *Mus musculus* (Mus_musculus.GRCm38.75), disponível no site do Ensembl.

Depois de realizarmos o mapeamento e a contagem de todas as amostras, fizemos três análises de expressão diferencial. Primeiro, o controle (MO) foi comparado com conídios (MOFp). Depois, comparou-se o controle contra células muriformes (MOCm). Por fim, testou-se o controle com conídios tratados com triciclazol (MOFpTc). Na Figura 6.5 são mostradas as distâncias entre os grupos testados no modo `union` de

execução do htseq-count. O conjunto MO × MOCm apresenta uma melhor separação enquanto o MO × MOFp apresenta as menores distâncias.

Tabela 6.6: Total de *reads* após filtragem das diferentes formas do fungo *F. pedrosoi*.

Condições	Replicata 1	Replicata 2	Replicata 3
Controle (MO)	22.569.122	28.312.087	21.584.134
Células muriformes (MOCm)	27.812.950	24.060.007	18,879,911
Conídios (MOFp)	25.721.496	27.924.341	20.782.139
Conídios tratados com triciclazol (MOFpTc)	18.234.364	20.497.309	24.555.501

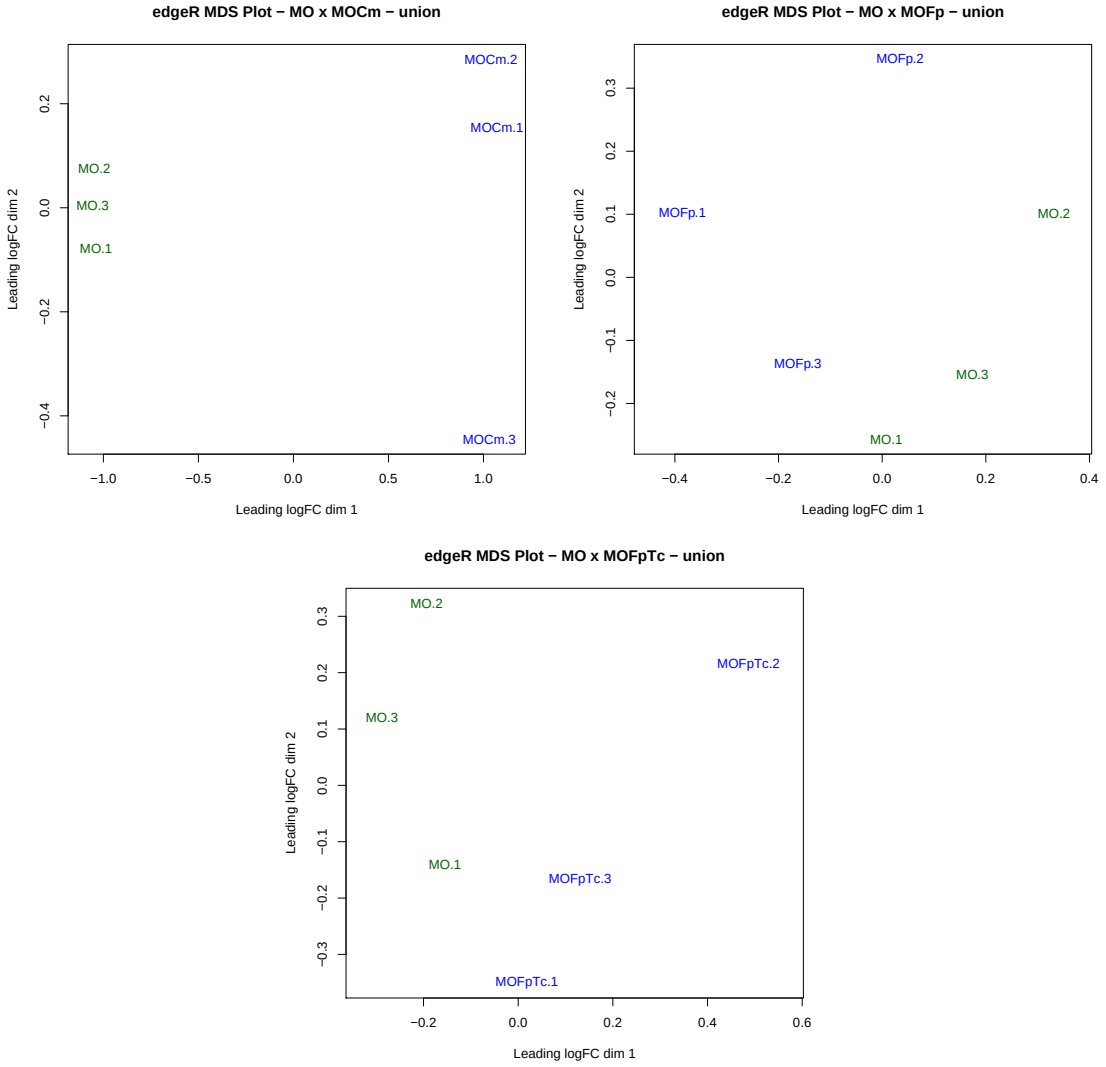


Figura 6.5: Gráficos MDS para a análise de diferentes formas do fungo *F. pedrosoi* utilizando o modo `union` de contagem do htseq-count.

O número de genes DE encontrado em cada comparação está disponível na Tabela 6.7. Na Figura 6.6 é possível visualizar os genes DE com FDR < 0.05 e PValue < 0.05 calculados a partir do modo de contagem `union`.

Tabela 6.7: Quantidade de genes diferencialmente expressos resultantes da comparação de diferentes formas do fungo *F. pedrosoi* encontrados por diferentes programas, utilizando os diferentes modos de contagem e com diferentes restrições.

		MO x MOCm		**MO x MOFp**		**MO x MOFpTc**	
Modo	***Software***	**deg1**	**deg2**	**deg1**	**deg2**	**deg1**	**deg2**
`union`	edgeR	7.608	3.625	395	51	1.085	101
	DESeq2	6.965	3.410	287	0	830	4
`strict`	edgeR	7.589	3.555	397	50	1.095	95
	DESeq2	6.957	3.360	281	0	815	3
`nonempty`	edgeR	7.781	3.712	401	56	1.078	104
	DESeq2	7.149	3.498	296	0	863	3

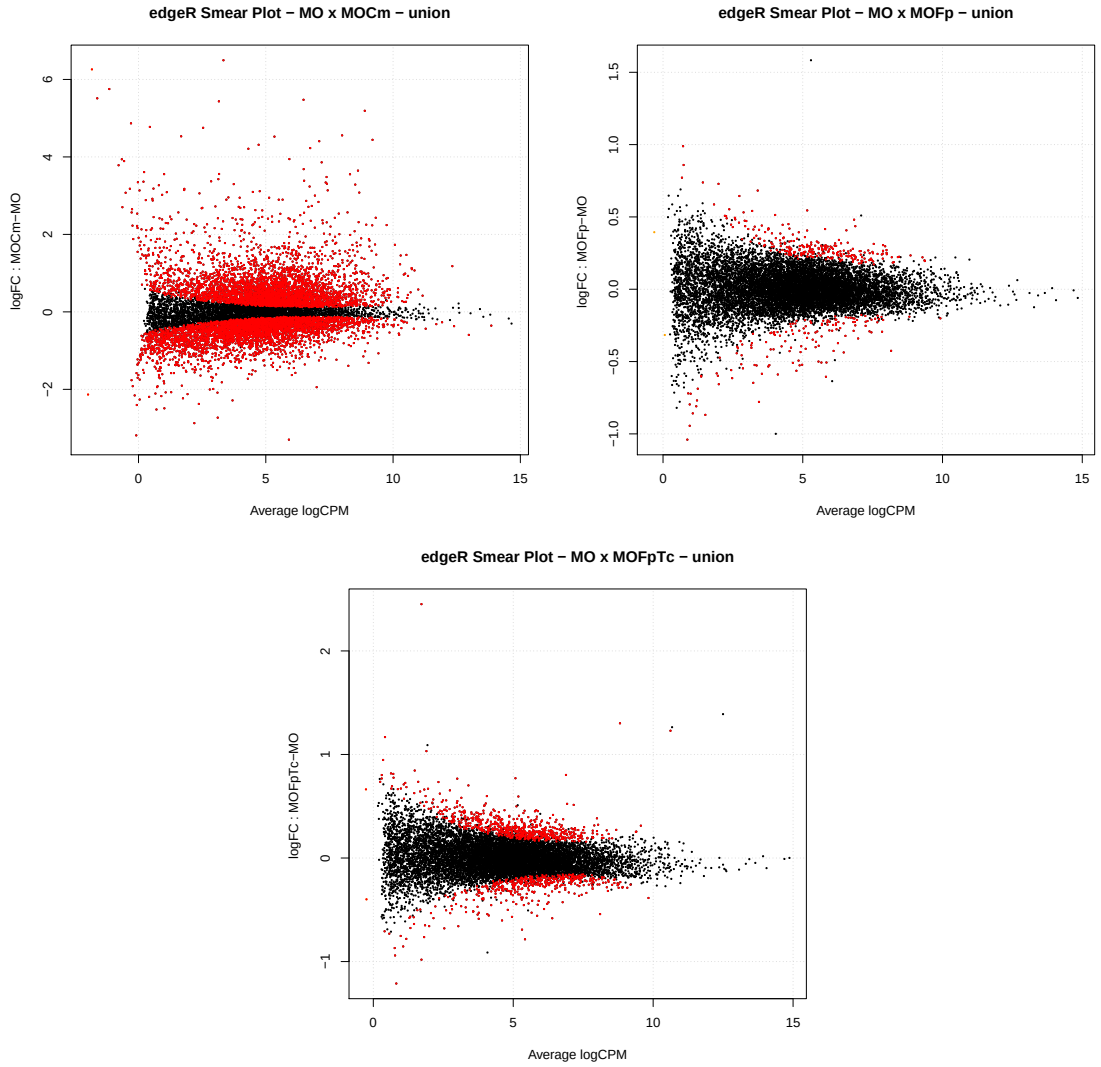


Figura 6.6: Gráficos MA para a análise de diferentes formas do fungo *F. pedrosoi*. Os gráficos mostram os genes DE com FDR < 0.05 e PValue < 0.05.

Capítulo 7

Conclusão

O baixo custo de sequenciamento tem possibilitado o estudo mais aprofundado e mais eficiente de um número cada vez maior de organismos. Dentre as várias possibilidades de estudo de dados biológicos oriundos de sequenciamento, pesquisas envolvendo genômica e transcritômica têm se destacado: a genômica por permitir a identificação e a análise funcional de genes e proteínas em escala genômica; a transcritômica por prover dados quantitativos de expressão gênica, permitindo identificação de genes diretamente relacionados a diferentes situações impostas a organismos, tecidos ou mesmo células.

A consequência direta desses avanços gera uma quantidade enorme de dados, o que demanda o desenvolvimento de métodos automáticos para agilizar e facilitar as análises que são realizadas. Essas análises geralmente são realizadas na forma de *pipelines*, caracterizando-se pela dificuldade no trato de arquivos de entrada e saída, além da especificidade das ferramentas utilizadas em cada etapa.

Contribuições

O escopo deste trabalho insere-se exatamente na tentativa de facilitar estudos envolvendo sequenciamento em larga escala, tendo como resultado o desenvolvimento de dois *pipelines* automatizados: um para genômica de bactérias e outro para transcritômica e análise de expressão diferencial. Por *pipeline* automatizado entende-se uma ferramenta computacional capaz de receber uma entrada bem definida e algumas escolhas de parâmetros, providas pelo pesquisador, e devolver os resultados das

análises esperadas em cada caso, minimizando a participação do usuário nos passos intermediários e minimizando também a exigência de experiência na utilização dos pacotes e programas utilizados.

No caso da genômica de bactérias, o trabalho teve como principal motivação o sequenciamento e análise de cepas sul-americanas de *M. bovis*, além da comparação dessas cepas com outras norte-americanas. Este estudo, feito em colaboração com a Embrapa Gado de Corte, visa, além da erradicação da Tuberculose Bovina, doença causada por *M. bovis*, o desenvolvimento de técnicas mais eficientes de diagnóstico da doença. O estudo já gerou quatro publicações [10, 11, 22, 84]. Outras duas publicações estão sendo escritas, uma delas direcionada a aspectos relacionados a genes de virulência [23] e outra mais geral, com a comparação de todas as cepas sequenciadas [12]. Essas duas publicações devem ser submetidas já no início do segundo semestre de 2015.

Para estudos de transcritômica e análise de expressão diferencial, três projetos oriundos de pesquisadores do Instituto de Biologia Molecular da UnB foram motivadores para este trabalho. O primeiro deles deseja-se determinar os genes diferencialmente expressos em linfócitos T humanos de indivíduos saudáveis e indivíduos transplantados, com diferentes níveis de tolerância. O segundo projeto busca-se a expressão diferencial em linfócitos T tratados com diferentes anticorpos. O terceiro projeto visa encontrar genes diferencialmente expressos em células de sem infecção e células infectadas com diferentes formas do fungo *Fonsecaea pedrosoi*. Todos esses três projetos estão em fase final de análise e devem gerar publicações a serem submetidas em 2015.

Trabalhos futuros

Várias são as possibilidades de dar continuidade a este trabalho. Essas possibilidades vêm desde a melhoria de partes específicas de cada programa utilizado até a forma de utilização de cada *pipeline* como um todo:

- **Novos programas:** Incluir outros programas em cada etapa do *pipeline*, aumentando as possibilidades de análises;

- **Novos módulos:** Criar módulos para instalação automática de todos os *softwares* necessários;

- **Acrescentar informação sobre os genes:** Adicionar aos resultados obtidos pelo *pipeline* de expressão diferencial, informações que descrevem e categorizam os genes e seus produtos gênicos, por meio de suas funções moleculares, sua localização celular e seu envolvimento em processos biológicos e vias metabólicas;

- **Nova filtragem:** Incluir um módulo de filtragem após o alinhamento dos *reads*, removendo do resultado, por exemplo, *reads* que se alinharam em rRNAs e que não pertencem ao escopo da análise.

Referências Bibliográficas

[1] Fastx-toolkit. `http://hannonlab.cshl.edu/fastx_toolkit/index.html`. Acessado em 19 de junho de 2015.

[2] Seqyclean. `https://github.com/ibest/seqyclean`. Acessado em 19 de junho de 2015.

[3] C. Adessi, G. Matton, G. Ayala, G. Turcatti, J. J. Mermod, P. Mayer, e E. Kawashima. Solid phase DNA amplification: characterisation of primer attachment and amplification mechanisms. *Nucleic Acids Research*, 28(20):e87, 2000.

[4] J. Ahrens, J. R. Graybill, A. Abishawl, F. O. Tio, e M. G. Rinaldi. Experimental Murine Chromomycosis Mimicking Chronic Progressive Human Disease. *The American Journal of Tropical Medicine and Hygiene*, 40(6):651–658, 1989.

[5] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, e D. J. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402, 1997.

[6] S. Anders e W. Huber. Differential expression analysis for sequence count data. *Genome Biology*, 11(10):R106, 2010.

[7] S. Anders, P. T. Pyl, e W. Huber. HTSeq - A Python framework to work with high-throughput sequencing data. *bioRxiv*, 2014.

[8] S. Andrews. FastQC, A Quality Control tool for High Throughput Sequence Data. `http://www.bioinformatics.babraham.ac.uk/projects/fastqc/`. Acessado em 19 de junho de 2015.

[9] S. V. Angiuoli, A. Gussman, W. Klimke, G. Cochrane, D. Field, G. M. Garrity, C. D. Kodira, N. Kyrpides, R. Madupu, V. Markowitz, T. Tatusova, N. Thomson, e O. White. Toward an Online Repository of Standard Operating Procedures (SOPs) for (Meta) genomic Annotation. *OMICS*, 12(2):137–141, 2008.

[10] C. P. Araújo, A. L. A. R. Osório, K. S. G. Jorge, C. A. N. Ramos, A. F. S. Filho, C. E. S. Vidal, E. Roxo, C. Nishibe, N. F. Almeida, A. A. Fonseca-Júnior, M. R. Silva, J. D. B. Neto, V. D. Cerqueira, M. J. Zumárraga, e F. R. Araújo. Detection of *Mycobacterium bovis* in Bovine and Bubaline Tissues Using Nested-PCR for TbD1. *PLoS One*, 9(3):e91023, 2014.

[11] F. R. Araújo, A. B. C. Castelão, A. A. Fonseca-Jr., M. A. Hodon, M. A. Issa, A. K. Ramalho, G. M. T. Mendes, C. A. N. Ramos, E. B. Sales, P. M. Soares-Filho, N. C. Farias, C. Nishibe, e N. F. Almeida. Typing of a Brazilian *Mycobacterium* isolate by whole-genome sequencing. In *BSB2012 Digital Proceedings (ISSN: 2316-1248)*. 2012. Campo Grande, Brazil, August 15-17, 2012.

[12] F. R. Araújo, A. B. C. Castelão, C. Nishibe, N. F. Almeida, J. C. Setubal, e et al. Comprative genomics of *Mycobacterium bovis* strains, 2015. In preparation.

[13] K. F. Au, H. Jiang, L. Lin, Y. Xing, e W. H. Wong. Detection of splice junctions from paired-end RNA-seq data by SpliceMap. *Nucleic Acids Research*, 2010.

[14] A. M. Bolger, M. Lohse, e B. Usadel. Trimmomatic: a flexible trimmer for Illumina sequence data. *Bioinformatics*, 30(15):2114–2120, 2014.

[15] M. R. Borges-Osório e W. M. Robinson. *Genética Humana*. Artmed, 2013.

[16] S. Brouard, E. Mansfield, C. Braud, L. Li, M. Giral, S. Hsieh, D. Baeten, M. Zhang, J. Ashton-Chess, C. Braudeau, F. Hsieh, A. Dupont, A. Pallier, A. Moreau, S. Louis, C. Ruiz, O. Salvatierra, J. Soulillou, e M. Sarwal. Identification of a peripheral blood transcriptional biomarker panel associated with operational renal allograft tolerance. *Proceedings of the National Academy of Sciences*, 104(39):15448–15453, 2007.

[17] T. A. Brown. *Genomes*. Wiley-Liss, 1999.

[18] J. Bullard, E. Purdom, K. Hansen, e S. Dudoit. Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments. *BMC Bioinformatics*, 11(1):94, 2010.

[19] M. Burrows e D. J. Wheeler. A block-sorting lossless data compression algorithm. Relatório técnico, Digital SRC, 1994.

[20] N. A. Campbell e J. B. Reece. *Biology*. Pearson Education, 2005.

[21] R. A. Cardoso. *Seleção de primers específicos usando famílias de proteínas e sequências características*. Dissertação de Mestrado, Faculdade de Computação, UFMS, 2015. Trabalho em desenvolvimento.

[22] A. B. C. Castelão, C. Nishibe, A. Moura, A. P. Alencar, M. A. de Issa, M. A. Hodon, P. M. P. C. Mota, E. B. Sales, A. A. Fonseca-Júnior, N. F. Almeida, e F. R. Araújo. Draft Genome Sequence of *Mycobacterium bovis* Strain AN5, Used for Production of Purified Protein Derivative. *Genome Announcements*, 2(2), 2014.

[23] A. B. C. Castelão, C. Nishibe, M. Zumárraga, A. Cataldi, F. Bigi, C. A. R. do Nascimento, N. F. Almeida, e F. R. Araújo. Comparison of virulence genes of two *Mycobacterium bovis* strains with contrasting pathogenic profiles, 2015. In preparation.

[24] J. Castresana. Selection of Conserved Blocks from Multiple Alignments for Their Use in Phylogenetic Analysis. *Molecular Biology and Evolution*, 17(4):540–552, 2000.

[25] L. Chatenoud. CD3-specific antibody-induced active tolerance: from bench to bedside. *Nature Reviews Immunology*, 3(2):123–132, 2003.

[26] B. Chevreux, T. Pfisterer, B. Drescher, A. J. Driesel, W. E. G. Müller, T. Wetter, e S. Suhai. Using the miraEST Assembler for Reliable and Automated mRNA Transcript Assembly and SNP Detection in Sequenced ESTs. *Genome Research*, 14(6):1147–1159, 2004.

[27] P. J. A. Cock, C. J. Fields, N. Goto, M. L. Heuer, e P. M. Rice. The sanger fastq file format for sequences with quality scores, and the solexa/illumina fastq variants. *Nucleic Acids Research*, 38(6):1767–1771, 2010.

[28] I. R. Cohen. Chapter 5 - On Autoimmunity. In *Tending Adam's Garden*, pp. 197–239. Academic Press, 2000. ISBN 978-0-12-178355-6.

[29] P. E. C. Compeau, P. A. Pevzner, e G. Tesler. How to apply de Bruijn graphs to genome assembly. *Nature Biotechnology*, 29(11):987–991, 2011.

[30] The Gene Ontology Consortium. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.

[31] T. H. Cormen, C. Stein, R. L. Rivest, e C. E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edição, 2001. ISBN 0070131511.

[32] A. B. Cosimi, R. C. Burton, R. B. Colvin, G. Goldstein, F. L. Delmonico, M. P. LaQuaglia, N. Tolkoff-Rubin, R. H. Rubin, J. T. Herrin, e P. S. Russell. Treatment of acute renal allograft rejection with OKT3 monoclonal antibody. *Transplantation*, 32:535–539, 1981.

[33] V. Costa, C. Angelini, I. De Feis, e A. Ciccodicola. Uncovering the complexity of transcriptomes with RNA-Seq. *Journal of Biomedicine and Biotechnology*, 2010:1–20, 2010.

[34] P. D. Davies. Tuberculosis in humans and animals: are we a threat to each other? *Journal of the Royal Society of Medicine*, 99(10):539–540, 2006.

[35] F. De Bona, S. Ossowski, K. Schneeberger, e G. Rätsch. Optimal spliced alignments of short sequence reads. *Bioinformatics*, 24(16):i174–i180, 2008.

[36] A. Delcher, A. Phillippy, J. Carlton, e S. L. Salzberg. Fast algorithms for large-scale genome alignment and comparison. *Nucleic Acids Research*, 30(11):2478–2483, 2002.

[37] N. Delhomme, I. Padioleau, E. E. Furlong, e L. M. Steinmetz. easyRNA-Seq: a bioconductor package for processing RNA-Seq data. *Bioinformatics*, 28(19):2532–2533, 2012.

[38] M. Deloger, M. El Karoui, e M-A. Petit. A genomic distance based on mum indicates discontinuity between most bacterial species and genera. *Journal of Bacteriology*, 191(1):91–99, 2009.

[39] A. Dobin, C. A. Davis, F. Schlesinger, J. Drenkow, C. Zaleski, S. Jha, P. Batut, M. Chaisson, e T. R. Gingeras. STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, 29(1):15–21, 2013.

[40] R. C. Edgar. Muscle: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32:1792–1797, 2004.

[41] J. T. Evans, E. G. Smith, A. Banerjee, R. M. M. Smith, J. Dale, J. A. Innes, D. Hunt, A. Tweddell, A. Wood, C. Anderson, R. G. Hewinson, N. H. Smith, P. M. Hawkey, e P. Sonnenberg. Cluster of human tuberculosis caused by *Mycobacterium bovis*: evidence for person-to-person transmission in the UK. *The Lancet*, 369:1270–1276, 2007.

[42] B. Ewing e P. Green. Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities. *Genome Research*, 8(3):186–194, 1998.

[43] B. Ewing, L. Hillier, M. C. Wendl, e P. Green. Base-Calling of Automated Sequencer Traces Using Phred. I. Accuracy Assessment. *Genome Research*, 8(3):175–185, 1998.

[44] N. C. Farias. *Orthologsorter: inferindo genotipagem e funcionalidade a partir de famílias de proteínas ortólogas*. Dissertação de Mestrado, UFMS (Universidade Federal de Mato Grosso do Sul), Campo Grande, Brazil, 2013.

[45] M. Fedurco, A. Romieu, S. Williams, I. Lawrence, e G. Turcatti. BTA, a novel reagent for DNA attachment on glass and efficient generation of solid-phase amplified DNA colonies. *Nucleic Acids Research*, 34(3):e22, 2006.

[46] J. Felsenstein. PHYLIP (Phylogeny Inference Package) version 3.6, 2005. Distributed by the author. Department of Genome Sciences, University of Washington, Seattle.

[47] P. Ferragina e G. Manzini. Opportunistic data structures with applications. In *Foundations of Computer Science, Annual IEEE Symposium on*, pp. 390–398. IEEE Computer Society, 2000.

[48] R. D. Finn, A. Bateman, J. Clements, P. Coggill, R. Y. Eberhardt, S. R. Eddy, A. Heger, K. Hetherington, L. Holm, J. Mistry, E. L. L. Sonnhammer, J. Tate, e M. Punta. Pfam: the protein families database. *Nucleic Acids Research*, 42(D1):D222–D230, 2014.

[49] T. Garnier, K. Eiglmeier, J. C. Camus, N. Medina, H. Mansoor, M. Pryor, S. Duthoy, S. Grondin, C. Lacroix, C. Monsempe, S. Simon, B. Harris, R. Atkin, J. Doggett, R. Mayes, L. Keating, P. R. Wheeler, J. Parkhill, B. G. Barrell,

S. T. Cole, S. V. Gordon, e R. G. Hewinson. The complete genome sequence of *Mycobacterium bovis*. *Proceedings of the National Academy of Sciences*, 100(13):7787–7882, 2003.

[50] M. G. Grabherr, B. J. Haas, M. Yassour, J. Z. Levin, D. A. Thompson, I. Amit, X. Adiconis, L. Fan, R. Raychowdhury, Q. Zeng, Z. Chen, E. Mauceli, N. Hacohen, A. Gnirke, N. Rhind, F. di Palma, B. W. Birren, C. Nusbaum, K. Lindblad-Toh, N. Friedman, e A. Regev. Full-length transcriptome assembly from RNA-Seq data without a reference genome. *Nature Biotechnology*, 29(7):644–652, 2011.

[51] Ortho Multicenter Transplant Study Group. A Randomized Clinical Trial of OKT3 Monoclonal Antibody for Acute Rejection of Cadaveric Renal Transplants. *New England Journal of Medicine*, 313(6):337–342, 1985.

[52] T. J. Hardcastle e K. A. Kelly. baySeq: Empirical Bayesian methods for identifying differential expression in sequence count data. *BMC Bioinformatics*, 11(1):1–14, 2010.

[53] L. H. Hogan, B. S. Klein, e S. M. Levitz. Virulence factors of medically important fungi. *Clinical Microbiology Reviews*, 9(4):469–88, 1996.

[54] W. R. Jeck, J. A. Reinhardt, D. A. Baltrus, M. T. Hickenbotham, V. Magrini, E. R. Mardis, J. L. Dangl, e C. D. Jones. Extending assembly of short DNA sequences to handle error. *Bioinformatics*, 23(21):2942–2944, 2007.

[55] D. Kim, G. Pertea, C. Trapnell, H. Pimentel, R. Kelley, e S. Salzberg. TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. *Genome Biology*, 14(4):R36, 2013.

[56] J. A. Kimball, D. J. Norman, C. F. Shield, T. J. Schroeder, P. Lisi, M Garovoy, J. B. O'Connell, F. Stuart, S. V. McDiarmid, e W. Wall. The OKT3 antibody response study: a multicentre study of human anti-mouse antibody (HAMA) production following OKT3 use in solid organ tranplantation. *Transplant Immunology*, 3(3):212–221, 1995.

[57] G. Kohler e C. Milstein. Continuous cultures of fused cells secreting antibody of predefined specificity. *Nature*, 256(5517):495–497, 1975.

[58] E. V. Koonin, K. S. Makarova, e L. Aravind. Horizontal gene transfer in prokaryotes: quantification and classification. *Annual Review of Microbiology*, 55(1):709–742, 2001.

[59] B. Langmead e S. L. Salzberg. Fast gapped-read alignment with Bowtie 2. *Nature Methods*, 9(4):357–359, 2012.

[60] B. Langmead, C. Trapnell, M. Pop, e S. Salzberg. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biology*, 10(3):R25–10, 2009.

[61] M. Lawrence, W. Huber, H. Pagès, P. Aboyoun, M. Carlson, R. Gentleman, M. T. Morgan, e V. J. Carey. Software for Computing and Annotating Genomic Ranges. *PLoS Computational Biology*, 9(8):e1003118, 2013.

[62] N. Leng, J. A. Dawson, J. A. Thomson, V. Ruotti, A. I. Rissman, B. M. G. Smits, J. D. Haag, M. N. Gould, R. M. Stewart, e C. Kendziorski. EBSeq: an empirical Bayes hierarchical model for inference in RNA-seq experiments. *Bioinformatics*, 29(8):1035–1043, 2013.

[63] B. Lewin, J. E. Krebs, E. S. Goldstein, e S. T. Kilpatrick. *Lewin's Genes XI*. Jones and Bartlett Learning, 2014.

[64] H. Li e R. Durbin. Fast and accurate short read alignment with Burrows–Wheeler transform. *Bioinformatics*, 25(14):1754–1760, 2009.

[65] H. Li, B. Handsaker, A. Wysoker, T. Fennell, J. Ruan, N. Homer, G. Marth, G. Abecasis, R. Durbin, e 1000 Genome Project Data Processing Subgroup. The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, 25(16):2078–2079, 2009.

[66] H. Li, J. Ruan, e R. Durbin. Mapping short DNA sequencing reads and calling variants using mapping quality scores. *Genome Research*, 18(11):1851–1858, 2008.

[67] J. Li e R. Tibshirani. Finding consistent patterns: A nonparametric approach for identifying differential expression in RNA-Seq data. *Statistical Methods in Medical Research*, 22(5):519–536, 2013.

[68] J. Li, D. M. Witten, I. M. Johnstone, e R. Tibshirani. Normalization, testing, and false discovery rate estimation for RNA-sequencing data. *Biostatistics*, 13(3):523–538, 2012.

[69] L. Li, C. J. Stoeckert, e D. S. Roos. OrthoMCL: identification of ortholog groups for eukaryotic genomes. *Genome Research*, 13(9):2178–89, 2003.

[70] Y. Liao, G. K. Smyth, e W. Shi. The Subread aligner: fast, accurate and scalable read mapping by seed-and-vote. *Nucleic Acids Research*, 41(10):e108, 2013.

[71] Y. Liao, G. K. Smyth, e W. Shi. featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics*, 30(7):923–930, 2014.

[72] M. C. Londoño, R. Danger, M. Giral, J. P. Soulillou, A. Sánchez-Fueyo, e S. Brouard. A Need for Biomarkers of Operational Tolerance in Liver and Kidney Transplantation. *American Journal of Transplantation*, 12(6):1370–1377, 2012.

[73] M. Love, W. Huber, e S. Anders. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biology*, 15(12):550, 2014.

[74] A. V. Lukashin e M. Borodovsky. GeneMark.hmm: New solutions for gene finding. *Nucleic Acids Research*, 26(4):1107–1115, 1998.

[75] G. Lunter e M. Goodson. Stampy: A statistical algorithm for sensitive and fast mapping of Illumina sequence reads. *Genome Research*, 21(6):936–939, 2011.

[76] G. C. Machado, D. V. Moris, T. D. Arantes, L. R. F. Silva, R. C. Theodoro, R. P. Mendes, A. P. Vicentini, e E. Bagagli. Cryptic species of Paracoccidioides brasiliensis: impact on paracoccidioidomycosis immunodiagnosis. *Memórias do Instituto Oswaldo Cruz*, 108:637–643, 2013.

[77] M. Margulies, M. Egholm, W. E. Altman, S. Attiya, J. S. Bader, L. A. Bemben, J. Berka, M. S. Braverman, Y. Chen, Z. Chen, S. B. Dewell, L. Du, J. M. Fierro, X. V. Gomes, B. C. Godwin, W. He, S. Helgesen, C. H. Ho, G. P. Irzyk, S. C. Jando, M. L. I. Alenquer, T. P. Jarvie, K. B. Jirage, J. Kim, J. R. Knight, J. R. Lanza, J. H. Leamon, S. M. Lefkowitz, M. Lei, J. Li, K. L. Lohman, H. Lu, V. B. Makhijani, K. E. McDade, M. P. McKenna, E. W. Myers, E. Nickerson, J. R. Nobile, R. Plant, B. P. Puc, M. T. Ronan, G. T. Roth, G. J. Sarkis, J. F. Simons, J. W. Simpson, M. Srinivasan, K. R. Tartaro, A. Tomasz, K. A. Vogt, G. A. Volkmer, S. H. Wang, Y. Wang, M. P. Weiner,

P. Yu, R. F. Begley, e J. M. Rothberg. Genome sequencing in microfabricated high-density picolitre reactors. *Nature*, 437(7057):376–380, 2005.

[78] M. Martin. Cutadapt removes adapter sequences from high-throughput sequencing reads. *Bioinformatics in Action*, 17(1):10–12, 2011.

[79] R. L. Martínez e L. J. M. Tovar. Chromoblastomycosis. *Clinics in Dermatology*, 25(2):188–194, 2007.

[80] D. Medini, C. Donati, H. Tettelin, V. Masignani, e R. Rappuoli. The microbial pan-genome. *Current Opinion in Genetics and Development*, 15(6):589–594, 2005.

[81] S. L. Morrison, M. J. Johnson, L. A. Herzenberg, e V. T. Oi. Chimeric human antibody molecules: mouse antigen-binding domains with human constant region domains. *Proceedings of the National Academy of Sciences*, 81(21):6851–6855, 1984.

[82] A. Mortazavi, B. A. Williams, K. McCue, L. Schaeffer, e B. Wold. Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nature Methods*, 5(7):621–628, 2008.

[83] U. Nagalakshmi, Z. Wang, K. Waern, C. Shou, D. Raha, M. Gerstein, e M. Snyder. The Transcriptional Landscape of the Yeast Genome Defined by RNA Sequencing. *Science*, 320(5881):1344–1349, 2008.

[84] C. Nishibe, A. B. C. Castelão, R. D. Costa, B. J. Pinto, L. Varuzza, A. A. Cataldi, A. Bernardelli, F. Bigi, F. C. Blanco, M. J. Zumárraga, N. F. Almeida, e F. R. Araújo. Draft Genome Sequence of *Mycobacterium bovis* 04-303, a Highly Virulent Strain from Argentina. *Genome Announcements*, 1(6), 2013.

[85] A. Oberg, B. Bot, D. Grill, G. Poland, e T. Therneau. Technical and biological variance structure in mRNA-Seq data: life in the real world. *BMC Genomics*, 13(1):304, 2012.

[86] A. Oshlack, M. Robinson, e M. Young. From RNA-seq reads to differential expression results. *Genome Biology*, 11(12):220, 2010.

[87] J. Pevsner. *Bioinformatics and Functional Genomics*. Wiley, 2009.

[88] A. R. Quinlan e I. M. Hall. BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, 26(6):841–842, 2010.

[89] R Development Core Team. *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, 2009.

[90] M. Robinson e A. Oshlack. A scaling normalization method for differential expression analysis of RNA-seq data. *Genome Biology*, 11(3):R25, 2010.

[91] M. D. Robinson, D. J. McCarthy, e G. K. Smyth. edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics*, 26(1):139–140, 2010.

[92] M. Ronaghi, S. Karamohamed, B. Pettersson, M. Uhlén, e P. Nyrén. Real-Time DNA Sequencing Using Detection of Pyrophosphate Release. *Analytical Biochemistry*, 242(1):84–89, 1996.

[93] M. Ronaghi, M. Uhlén, e P. Nyrén. A Sequencing Method Based on Real-Time Pyrophosphate. *Science*, 281(5375):363–365, 1998.

[94] J. M. Rothberg, W. Hinz, T. M. Rearick, J. Schultz, W. Mileski, M. Davey, J. H. Leamon, K. Johnson, M. J. Milgrew, M. Edwards, J. Hoon, J. F. Simons, D. Marran, J. W. Myers, J. F. Davidson, A. Branting, J. R. Nobile, B. P. Puc, D. Light, T. A. Clark, M. Huber, J. T. Branciforte, I. B. Stoner, S. E. Cawley, M. Lyons, Y. Fu, N. Homer, M. Sedova, X. Miao, B. Reed, J. Sabina, E. Feierstein, M. Schorn, M. Alanjary, E. Dimalanta, D. Dressman, R. Kasinskas, T. Sokolsky, J. A. Fidanza, E. Namsaraev, K. J. McKernan, A. Williams, G. T. Roth, e J. Bustillo. An integrated semiconductor device enabling non-optical genome sequencing. *Nature*, 475(7356):348–352, 2011.

[95] S. L. Salzberg, A. L. Delcher, S. Kasif, e O. White. Microbial gene identification using interpolated Markov models. *Nucleic Acids Research*, 26(2):544–548, 1998.

[96] F. Sanger, S. Nicklen, e A. R. Coulson. DNA sequencing with chain-terminating inhibitors. *Proceedings of the National Academy of Sciences*, 74(12):5463–5467, 1977.

[97] R. Schmieder e R. Edwards. Quality control and preprocessing of metagenomic datasets. *Bioinformatics (Oxford, England)*, 27(6):863–864, 2011.

[98] R. Schmieder, Y. Lim, F. Rohwer, e R. Edwards. TagCleaner: Identification and removal of tag sequences from genomic and metagenomic datasets. *BMC Bioinformatics*, 11(1):341, 2010.

[99] J. Setubal e J. Meidanis. *Introduction to Computational Molecular Biology*. Brooks/Cole, 1997.

[100] D. H. Silver, S. Ben-Elazar, A. Bogoslavsky, e I. Yanai. ELOPER: elongation of paired-end reads as a pre-processing tool for improved de novo genome assembly. *Bioinformatics*, 29(11):1455–1457, 2013.

[101] J. T. Simpson, K. Wong, S. D. Jackman, J. E. Schein, S. J. M. Jones, e İ. Birol. ABySS: A parallel assembler for short read sequence data. *Genome Research*, 19(6):1117–1123, 2009.

[102] A. Stamatakis. RAxML Version 8: A tool for Phylogenetic Analysis and Post-Analysis of Large Phylogenies. *Bioinformatics*, 30(9):2688–2690, 2014.

[103] L. Stein. Genome annotation: from sequence to biology. *Nature Reviews Genetics*, 2(7):493–503, 2001.

[104] C. P. Taborda, M. B. Silva, J. D. Nosanchuk, e L. R. Travassos. Melanin as a virulence factor of *Paracoccidioides brasiliensis* and other dimorphic pathogenic fungi: a minireview. *Mycopathologia*, 165(4-5):331–339, 2008.

[105] C. Trapnell, L. Pachter, e S. L. Salzberg. TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics*, 25(9):1105–1111, 2009.

[106] C. Trapnell, A. Roberts, L. Goff, G. Pertea, D. Kim, D. R. Kelley, H. Pimentel, S. L. Salzberg, J. L. Rinn, e L. Pachter. Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks. *Nature Protocols*, 7(3):562–578, 2012.

[107] C. Trapnell, B. A. Williams, G. Pertea, A. Mortazavi, G. Kwan, M. J. van Baren, S. L. Salzberg, B. J. Wold, e L. Pachter. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nature Biotechnology*, 28(5):511–515, 2010.

[108] G. Turcatti, A. Romieu, M. Fedurco, e A. P. Tairi. A new class of cleavable fluorescent nucleotides: synthesis and optimization as reversible terminators for DNA sequencing by synthesis. *Nucleic Acids Research*, 36(4):e25, 2008.

[109] E. L. van Dijk, H. Auger, Y. Jaszczyszyn, e C. Thermes. Ten years of next-generation sequencing technology. *Trends in Genetics*, 30(9):418–426, 2014.

[110] G. S. Vernikos e J. Parkhill. Interpolated variable order motifs for identification of horizontally acquired DNA: revisiting the Salmonella pathogenicity islands. *Bioinformatics*, 22:2196–2203, 2006.

[111] K. Wang, D. Singh, Z. Zeng, S. J. Coleman, Y. Huang, G. L. Savich, X. He, P. Mieczkowski, S. A. Grimm, C. M. Perou, J. N. MacLeod, D. Y. Chiang, J. F. Prins, e J. Liu. MapSplice: Accurate mapping of RNA-seq reads for splice junction discovery. *Nucleic Acids Research*, 38(18):e178, 2010.

[112] Z. Wang, M. Gerstein, e M. Snyder. RNA-Seq: A Revolutionary Tool for Transcriptomics. *Nature Review Genetics*, 10(1):57–63, 2009.

[113] R. L. Warren, G. G. Sutton, S. J. M. Jones, e R. A. Holt. Assembling millions of short DNA sequences using SSAKE. *Bioinformatics*, 23(4):500–501, 2007.

[114] T. D. Wu e S. Nacu. Fast and SNP-tolerant detection of complex variants and splicing in short reads. *Bioinformatics*, 26(7):873–881, 2010.

[115] A. Zaha, H. B. Ferreira, e L. M. P. Passaglia. *Biologia Molecular Básica*. Artmed Editora, 5 edição, 2014.

[116] D. R. Zerbino e E. Birney. Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. *Genome Research*, 18(5):821–829, 2008.

[117] Y. Zhao, X. Jia, J. Yang, Y. Ling, Z. Zhang, J. Yu, J. Wu, e J. Xiao. PanGP: A tool for quickly analyzing bacterial pan-genome profile. *Bioinformatics*, 30(9):1297–1299, 2014.

[118] M. J. Zvelebil e J. O. Baum. *Understanding Bioinformatics*. Taylor and Francis Group, 2008.

Apêndice A

Pipeline User's Guide - Genomics

Overview of the Pipeline

This document describes an automatic pipeline for bacterial genomic analysis. The codes are available at: `http://projetos.facom.ufms.br/bioinfo/pipe-ge/tree/master`.

The input of the pipeline is the sample of reads and a configuration file defined by the user and the output is a multi-fasta file with contigs.

There are five stages in the pipeline:

1. **Quality control of reads**: The analysis starts with a general quality control. In this step reads are scanned for low confidence bases, adapters, duplicates, etc. The output of this step is statistics such as the number of reads and quality information which guides the next step.

2. **Preprocessing of reads**: This step removes low-quality bases, adapters and contaminants. The output is a new file without some reads.

3. **Assembly reads using a reference genome**: Using a reference genome, reads are mapped to find the location of every read. The output is an alignment file which shows the mapped reads and their mapping positions in the genome.

4. **Transformation**: The alignment file is processed to obtain contigs file.

5. **Annotation**: The contig file is used for the annotation of the genome.

Figure A.1 shows the main softwares, inputs and outputs used in each step of the pipeline.

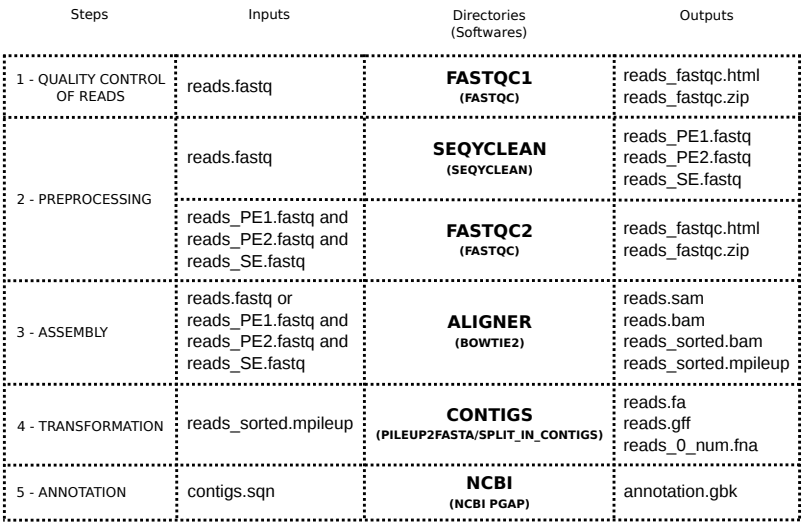


Figura A.1: Overview of the pipeline for genomic analysis. This includes the main inputs, outputs and directories created during the execution of the pipeline.

Requirements

To use the pipeline the list above must be installed. We used this versions of each software.

1. Linux
2. Perl - 5.14.2
3. FastQC - 0.10.1
4. SeqyClean - 1.4.13
5. Bowtie - 2.1.0
6. SAMtools - 0.1.19.0

Installation

The pipeline has several scripts and configuration files. To install it just unpack the `genomic_pipeline.tar.gz` using the command: `$ tar -zxvf genomic_pipeline.tar.gz`. The results are two directories:

- `scripts`: this directory keeps all scripts of the pipeline.

  - `exec_fastqc.pl`
  - `exec_seqyclean_paired.pl`
  - `exec_bowtie_paired.pl`
  - `pileup2fasta.pl`
  - `split_in_contigs.pl`
  - `run_genomic_pipeline.pl`
  - `include_cfg.pl`

- `conf_files`: this directory keeps all configuration files used by the pipeline and can be copied to each directory project.
  - `genomic_pipeline.conf`
  - `bowtie.conf`
  - `seqyclean.conf`

Include in your profile the path for the directory of scripts and be sure that all Perl files be executable (`chmod 777 *.pl`).

Using the Pipeline

The first step to use pipeline is to create a proper directory to store the input reads. This directory must be named as `00_reads` and the files also must have a specific name. The name of each file use the name and the type of read separated by "_".

- *paired-end reads*:
  - `name_R1.fastq`
  - `name_R2.fastq`

The next step is fill in the configuration files. For each software there is a configuration file with the parameters that must be used during the analysis. Besides, the `genomic_pipeline.conf` stores informations about the genome used as reference, location of input data and other configuration files, the directories must be created and which software must be executed.

The configuration files are text files with one or two columns that uses TAB to separate informations.

After specifying the parameters to be used in the analysis, just use the next command to execute the pipeline:

```
$ nohup run_genomic_pipeline.pl genomic_pipeline.conf > log.txt
```

This command will execute the entire pipeline automatically, but it's possible execute manually each step of pipeline or redo some step just changing the values in configuration files.

Steps of the Pipeline

This section describes each step of the pipeline.

Each script was implemented to receive a specific type of file.

Quality Control of Reads

The first step is verify the quality of the reads using the script `exec_fastqc.pl`. This script receives a fastq file and a directory where the results must be saved, then runs the FastQC program. In the pipeline all the input files are in `00_reads` and the result always be stored in the `fastqc1` directory. The `fastqc_report.html` file reports the analysis.

Preprocessing of Reads

SeqyClean is the software used to filtering the data. Script `exec_seqyclean.pl` receives the paired files, the output directory and the configuration file with the parameters which must be used. This script saves the output in the correct directory (`seqyclean`). SeqyClean saves the filtered reads in three files: `prefix_PE1.fastq`, `prefix_PE2.fastq` and `prefix_SE.fastq`. It also saves reports in text files.

Assembly Reads Using a Reference Genome

Using a reference genome, reads are mapped by Bowtie2. The software is executed by the script `exec_genomic_bowtie_paired.pl`. It also uses the SAMtools to prepare the alignment result to the next step. The input of the script is the reads file, the genome index, the output directory and the configuration file with other parameters used by Bowtie2.

The read file can be the original reads or the reads of `seqyclean` directory. The genome index is created in `run_genomic_pipeline.pl` by the command `bowtie2-build` that receives the genome file, with the extension `.fa`.

The output is an alignment file in SAM format which shows the mapped reads and their mapping positions in the genome and the unmapped reads in `unmapped.fastq` file. The alignment file is sorted by SAMtools to create a mpileup file which describes the base-pair information at each chromosomal position.

Observation: The Bowtie2 parameters `-no-head` and `-no-seq` options, must not be used, because they do not work with the next steps executed by SAMtools.

Transformation

The mpileup file is used by `pileup2fasta.pl` script to get the contigs that will be annotated. This script reads the mpileup file and creates a fasta file (`.fa`) with the assembled sequence guided by the reference genome and a GFF file (`.gff`) containing the SNP, insertion and deletion information.

The next step is split the fasta file (`.fa`) in a multi-fasta file (`.fna`) with the contigs with the length specified in configuration file. All this files are stored in `contigs` directory.

Annotation

The contig file is used to the annotation of the genome. But this is not covered by the pipeline.

This step must be executed manually using, for example, NCBI Prokaryotic Genome Annotation Pipeline (PGAP) [9], available at: `http://www.ncbi.nlm.nih.gov/genome/annotation_prok/`.

Example

Suppose that you have one sample to assembly. And the reads are paired-end. So your input data in `00_reads` is: `sampleA_1_R1.fastq` and `sampleA_1_R2.fastq`. Your genome `myGenome.fa` is the `genomes` folder and all configuration used in this analysis are in `conf_files` folder and all data is in `MyProject` folder which is located in `/home/myuser`.

If you want to filter the reads with at least 75 base pair and minimum quality equal to 24, in `seqyclean.conf` you must set `-minimum_read_length` with 75 and `-qual` with 24 24.

```
-qual 24 24
-minimum_read_length 75
```

In the pipeline by default the number of alignment threads to launch is 8 and we used the option `-very-sensitive-local`.

```
-p 8
--very-sensitive-local
```

In `genomic_pipeline.conf` you set the path for the data, the configuration files and to genome. You specify the minimum length of the contig and the minimum number of reads mapped in the genome, the prefix of the reads, the directories and softwares used. In the example below, the execution builds the genome index, creates the directories and executes the FastQC.

```
project_dir /home/myuser/MyProject
conf_seqyclean /home/myuser/MyProject/conf_files/seqyclean.conf
conf_bowtie /home/myuser/MyProject/conf_files/bowtie.conf
genome_fna /home/myuser/MyProject/genomes/myGenome.fa
genome_idx /home/myuser/MyProject/genomes/myGenome
genome_inde 1
contig_length 500
base_cov 10
PAIRED 1
header @HISEQ:276:
tipo fastq
dirs reads,fastqc1,seqyclean,fastqc2,bowtie,contigs
FASTQC1 1
SEQYCLEAN 0
FASTQC2 0
BOWTIE 0
CONTIGS 0
```

To execute the full pipeline just set `genome_index` with 0, because the index need to be created just once, and `SEQYCLEAN, FASTQC2, BOWTIE` and `CONTIGS` to 1, while change `FASTQC1` to 0. Then reexecute the `run_genomic_pipeline.pl`. At the end of the analysis, in `/home/myuser/MyProject` there are six directories: `00_reads`, `01_fastqc1`, `02_seqyclean`, `03_fastqc2`, `04_bowtie` and `05_contigs` .

Another example of execution is from original reads, just align it and get the contigs. In this case, the values must be:

```
dirs reads,bowtie,contigs
FASTQC1 1
SEQYCLEAN 0
FASTQC2 0
BOWTIE 1
CONTIGS 1
```

Apêndice B

Pipeline User's Guide - Differential Expression in Transcriptomics

Overview of the Pipeline

This document describes an automatic pipeline for differential expression analysis in transcriptome projects. The codes are available at: `http://projetos.facom.ufms.br/bioinfo/pipe-de/tree/master`.

The input of the pipeline is the set of reads with replicates in different conditions and a configuration file defined by the user.

The output is the list of differential expression genes and graphics that shows these genes.

There are five stages in the pipeline:

1. **Quality control of reads**: The analysis starts with a general quality control. In this step reads are scanned for low confidence bases, adapters, duplicates, etc. The output of this step is statistics such as the number of reads and quality information which guides the next step.

2. **Preprocessing of reads**: This step removes low-quality bases, adapters and contaminants. The output is a new file without some reads.

3. **Aligning reads to a reference genome**: Using a reference genome, reads are mapped to find the point of origin for every read. The output is an alignment file showing the mapped reads and their mapping positions in the genome.

4. **Calculating gene expression levels**: Using the genome annotation the goal is count the number of reads mapping to genes. The output is a table of counts for each gene.

5. **Differential expression analysis**: Using the table counts the goal is the idenfication of genes that are expressed in sigficantly different quantities in distinct groups.

Figure B.1 shows the main softwares, inputs and outputs used in each step of the pipeline.

Steps	Inputs	Directories (Softwares)	Outputs
1 - QUALITY CONTROL OF READS	reads.fastq	**FASTQC1** (FASTQC)	reads_fastqc.html reads_fastqc.zip
2 - PREPROCESSING OF READS	reads.fastq	**CUTADAPT** (CUTADAPT)	reads.clipped.fastq reads.short.fastq log_cutadapt_reads.txt
	reads.fastq or reads.clipped.fastq	**PRINSEQ** (PRINSEQ)	reads.trimmed.fastq reads.trimmed.log
	reads.clipped.fastq or reads.trimmed.fastq	**PAIRED**	reads.paired.fastq
	reads.clipped.fastq or reads.trimmed.fastq or reads.paired.fastq	**FASTQC2** (FASTQC)	reads_fastqc.html reads_fastqc.zip
3 - ALIGNING	reads.fastq or reads.clipped.fastq or reads.trimmed.fastq or reads.paired.fastq	**ALIGNER** (TOPHAT2/BOWTIE2)	logs/ accepted_hits.bam align_summary.txt deletions.bed prep_reads_info insertions.bed reads.sam junctions.bed unmapped.bam
4 - GENE EXPRESSION LEVEL	reads.sam	**HTSEQ_ALIGNER** (HTSEQ-COUNT)	reads.count
5 - DIFFERENTIAL EXPRESSION	reads.count	**DE_ALIGNER** (EDGER/DESEQ2)	counts.xls ec_MDS.pdf ec_Smearcomplex1.pdf ec_Smearcomplex2.pdf ec_deg1.xls ec_deg2.xls d2_deg1.xls d2_deg2.xls

Figura B.1: Overview of the pipeline for differential expression analysis. This includes the main inputs, outputs and directories created during the execution of the pipeline.

Requirements

To use the pipeline, the list above must be installed. We used the following versions of each software.

1. Linux
2. Perl - 5.14.2
3. FastQC - 0.11.2
4. Cutadapt - 1.7.1
5. Prinseq - 0.20.4
6. Bowtie - 2.0.5.0
7. Tophat - 2.0.11
8. SAMtools - 0.1.19.0
9. Htseq - 0.5.0
10. R - 3.0.3 - 3.1.3
11. edgeR - 3.4.2 - 3.8.6
12. DESeq - 1.14.0 - 1.18.0
13. DESeq2 - 1.2.10 - 1.6.3

Installation

The pipeline has several scripts and configuration files. To install it, just unpack the `de_pipeline.tar.gz` using the command: `$ tar -zxvf de_pipeline.tar.gz`. The results are stored in two directories:

- `scripts`: this directory keeps all the scripts of the pipeline.
  - `check_DE_genes.pl`
  - `count2counta.pl`
  - `create_execute_R_file.pl`
  - `exec_bowtie_paired.pl`
  - `exec_bowtie_single.pl`
  - `exec_cutadapt.pl`
  - `exec_fastqc.pl`
  - `exec_htseq.pl`
  - `exec_prinseq.pl`
  - `exec_tophat_paired.pl`
  - `exec_tophat_single.pl`
  - `gtf2gtfa.pl`
  - `run_pipeline_de.pl`
  - `write_paired_fastq_files.pl`
- `conf_files`: this directory keeps all the configuration files used by the pipeline.

- `pipeline_de.conf`
- `cutadapt.conf`
- `prinseq.conf`
- `tophat.conf`
- `bowtie.conf`
- `htseq.conf`
- `de.conf`

Include in your profile the path for the directory of scripts and be sure that all the Perl files are executable (`chmod 777 *.pl`).

Using the Pipeline

The first step to use pipeline is to create a proper directory to store the input reads. This directory must be named as `00_reads` and the files also must have a specific name. The name of each file use the condition, the number of the replicate and the type of read separated by "_".

- *paired-end reads*:
  - `condition_replicate_R1.fastq`
  - `condition_replicate_R2.fastq`
- *single-end reads*:
  - `condition_replicate_R1.fastq`

The replicates must be numbered consecutively from 1 to n, where n is the number of replicates.

It is important to mantain the name of files because these informations are used to generates the outputs during the pipeline.

The next step is to fill the configuration files. For each software there is a configuration file with the parameters that must be used during the analysis. Besides, the `pipeline_de.conf` stores informations about the genome, input data, all configuration files, directories must be created, comparison tested and which software must be executed.

After specifying the parameters to be used in the analysis, just use the next command to execute the pipeline:

```
$ nohup run_pipeline_de.pl pipeline_de.conf > log.txt
```

This command will execute the entire pipeline automatically, but it is possible to execute manually each step of pipeline, or redo some steps just changing the values in the configuration file.

Steps of the Pipeline

This section describes each step of the pipeline.

Each script was implemented to receive a specific type of file.

Quality Control of Reads

The first step is verify the quality of the reads using the script `exec_fastqc.pl`. This script receives a fastq file and a directory where the results must be saved and execute de FastQC program. In the pipeline the input data is in `00_reads` and the result always be stored in the `fastqc1` directory. It has a html file with the reports and a compact file (.zip).

Preprocessing of Reads

In this step, Cutadapt and PRINSEQ are used to clip and trim reads.

Cutadapt is executed by the script `exec_cutadapt.pl`. It receives the fastq file, the output directory name (`cutadapt`) and the configuration file with the parameters that must be used. The output is the clipped file `*.clipped.fastq`, the `*.short.fastq` file which stores reads below the parameters and the log file `log_cutadapt_*.txt`

PRINSEQ is executed by `exec_prinseq.pl` receives the fastq file, the output directory name (`prinseq`) and the configuration file with the parameters that must be used. At this point, the reads file can be the original reads or the data from the clipping directory (`cutadapt`). The output of this script is a log file with the informations about the trimming (`*.trimmed.log`) and the trimmed file (`*.trimmed.fastq`).

If the input data is paired-end, we use `write_paired_fastq_files.pl` to ensure that both file have the same reads. The files are saved with the extension `.paired.fastq` and stored in `paired` directory.

Aligning Reads to a Reference Genome

To align the reads it is possible to use Bowtie2 or TopHat2. Each software is executed by the scripts `exec_bowtie_paired.pl`, `exec_bowtie_single.pl`, `exec_tophat_paired.pl` or `exec_tophat_single.pl` according to the read type, single or paired-end. The results from Bowtie2 are stored in `bowtie` directory while TopHat2 saves the output in `tophat` directory.

Both aligners can receive data from different sources. For example, if reads are paired-end, the input of this step can be `*.fastq` or `*.paired.fastq` but if the reads are single, the input could be `*.fastq`, `*.clipped.fastq` or `*.trimmed.fastq`.

The output from Bowtie2 is the alignments in SAM format and the unmapped reads in BAM format. The TopHat2 produces a `log` directory and several result files:

- `accepted_hits.bam`: contains the alignments in BAM format, sorted according to chromosomal coordinates.
- `junctions.bed`: contains the discovered exon junctions in BED format.
- `insertions.bed`: contains the discovered insertions in BED format.
- `deletions.bed`: contains the discovered deletions in BED format.
- `align_summary.txt`: reports the alignment rate and how many reads and pairs had multiple alingments.

After the alignment, the `accepted_hits.bam` file is prepared with SAMtools to the next step.

Calculating Gene Expression Levels

Htseq-count is used to calculate the gene expression level. The script `exec_htseq.pl` needs the alignments sorted by the name of the read, the genome annotation in GTF format, the mode of htseq-count, the configuration file, the output directory

(`htseq_aligner`) and a flag to choose if the count table shows the genes by name or by identification.

In the pipeline, the three modes (`union`, `intersection-strict` and `intersection-nonempty`) available are used, so in the output directory there are three other directories (`union`, `strict` and `nonempty`) to store each count file (`*.count`) in the appropriate place.

If the user chooses the results must to be shown by name, the script `gtf2gtfa.pl` associates the identification and the name and save it in a file `.gtfa`, and `count2counta.pl` uses this file and the count file to modify the identification and save it with the extension `.counta`.

Differential Expression Analysis

To identify genes expressed in significantly different quantities in distinct groups of samples we use edgeR and DESeq2.

Using the script `create_execute_R_file.pl` we create a R file that contains all commands that must be executed to produce the results.

The input are the conditions, the directory where the count files are stored, the configuration file and the flag indicating how the results will be displayed.

The output has several files:

- `ec_*_count_*.xls`: contains filtered count table from all groups of samples.
- `ec_*_MDS_*.pdf`: a MDS plot showing the relations between samples.
- `ec_*_deg1_*.xls`: the list of differentially expressed genes sorted by log fold change produced by edgeR.
- `ec_*_Smearcomplex1_*.pdf`: the smearplot of the differentially expressed genes.
- `d2_*_deg1_*.xls`: the list of differentially expressed genes sorted by log fold change find out by DESeq2.

If the user set the values for the log fold change with 0.05 in `de.conf` file, the results include files `ec_*_deg2_*.xls`, `ec_*_Smearcomplex2_*.pdf` and `d2_*_deg2_*.xls` with this restriction: `log2FoldChange` $>$ 0.05 or `log2FoldChange` $<$ -0.05.

Example

Suppose that all analysis will be executed in the folder `/home/myuser/MyProject`. You have three conditions (A, B and C) and each of one has duplicates. The reads are paired-end, so your input data in `00_reads` is:

- `A_1_R1.fastq` and `A_1_R2.fastq`
- `A_2_R1.fastq` and `A_2_R2.fastq`
- `B_1_R1.fastq` and `B_1_R2.fastq`
- `B_2_R1.fastq` and `B_2_R2.fastq`
- `C_1_R1.fastq` and `C_1_R2.fastq`
- `C_2_R1.fastq` and `C_2_R2.fastq`

The genome used to map the reads is `myGenome.fa` and it is stored in `genomes` folder and all configuration files used in this analysis are in `conf_files` folder. In this example we will use all steps of the pipeline, so we need to set several files.

To remove the adapters with Cutadapt, they must be stored at file as below, where each line start with `-b` followed by a space and the adapter.

```
-b GATCGGAAGAGCTCGTATGCCGTCTTCTGCTTG
-b CAAGCAGAAGACGGCATACGAGCTCTTCCGATCT
-b CAAGCAGAAGACGGCATACGAGCTCTTCCGATCT
-b ACACTCTTTCCCTACACGACGCTCTTCCGATCT
```

Then include the path to this file in `-b` option of `cutadapt.conf`. Use `-e` to define the maximum allowed error rate, `-O` to define the minimum overlap length between the read and the adapter. The option `-m` is used to discard trimmed reads that are shorter than the defined length.

```
-b /home/myuser/MyProject/conf_files/adapters.txt
-e 0.15
-O 5
-m 75
```

The output will be in `02_cutadapt` folder.

The next step is to filter the reads with PRINSEQ. In `prinseq.conf` file, filter sequence shorter than 75 (`-min_len`) and with quality score mean below 20 (`min_qual_mean`), trim sequence by quality score from the 3'-end with this threshold score (`-trim_qual_right`), use the mean of quality score calculation (`-trim_qual_type`), use 20 to the sliding window size used to calculate quality score by type (`-trim_qual_window`) and 10 to step size used to move the sliding window (`-trim_qual_step`). These results will be saved in `03_prinseq` folder.

```
-out_format 3
-trim_qual_right 20
-trim_qual_window 20
-trim_qual_step 10
-trim_qual_type mean
-min_qual_mean 20
-min_len 75
```

After the filtering, the reads are processed and stored in `04_paired` folder.

In `tophat.conf`, we just set the number of threads (-p 8) that will be used during the alignment.

```
-p 8
```

In `htseq.conf` file, it is necessary to set whether the data is from a strand-specific assay (`-s`). By default, in htseq-count, the value is "yes", but here we set with "no". We also use the option `-a` to skip all reads with alignment quality lower than the given minimum value.

```
-s no
-a 20
```

In `de.conf` you can set the values for FDR (`fdr`), p-value (`pvalue`) and log fold change (`logfc`). By default this values are 0.05, 0.05 and 0.5, respectively. If you don't want to use the `logfc` option just includes the character "#" in the beginning of the line.

```
fdr 0.05
pvalue 0.05
logfc 0.5
```

In `pipeline_de.conf` you set the path for the data, the configuration files and to genome. You specify the prefix of the reads, the directories and softwares used, the number of tests and the tests that must be executed. In the example below, the execution builds the genome index, creates the directories and executes the entire analysis.

```
project_dir /home/myuser/MyProject
conf_cutadapt /home/myuser/MyProject/conf_files/cutadapt.conf
conf_prinseq /home/myuser/MyProject/conf_files/prinseq.conf
conf_tophat /home/myuser/MyProject/conf_files/tophat.conf
conf_bowtie /home/myuser/MyProject/conf_files/bowtie.conf
conf_htseq /home/myuser/MyProject/conf_files/htseq.conf
conf_de /home/myuser/MyProject/conf_files/de.conf
genome_fna /home/myuser/MyProject/genomes/myGenome.fa
genome_gtf /home/myuser/MyProject/genomes/myGenome.gtf
genome_idx /home/myuser/MyProject/genomes/myGenome
genome_index 1
PAIRED 1
gene_id_name 1
header @HISEQ
tipo fastq
dirs reads,fastqc1,cutadapt,prinseq,paired,fastqc2,tophat,htseq,de
FASTQC1 1
CUTADAPT 1
PRINSEQ 1
FASTQC2 1
TOPHAT 1
BOWTIE 0
SEGEMEHL 0
HTSEQ 1
DE 1
num_test 2
test_1 A B
```

```
test_2 A C
```

If you do not know how is the quality data, you can first execute just the FastQC. In this case, change the values from 1 to 0 of the softwares that must not be executed at this time. And change the `genome_index` to 0 because the index needs be build just once.

```
genome_index 0
FASTQC1 1
CUTADAPT 0
PRINSEQ 0
FASTQC2 0
TOPHAT 0
BOWTIE 0
HTSEQ 0
DE 0
```