Université de Bordeaux
Master Bio-informatique
Conception d'un projet de recherche et de développement

Rapport de projet : Analyse de données RNA-Seq (champignons *Fusarium*)

Auteurs :
Linda Khodja
Maroa Alani
Lucien Piat
Djemilatou Ouandaogo

Superviseur :
Marie Beurton-Aimar

Clients :
Nadia Ponts
Fabien Dumetz

Laboratoire :
L'Institut National de Recherche pour l'Agriculture, l'Alimentation et l'Environnement

Mai 2024

Table des matières

Introduction

Dans le cadre de la recherche menée à l'INRAE (Institut National de Recherche pour l'Agriculture, l'Alimentation et l'Environnement), plus précisément dans l'unité MycSA (Mycologie et Sécurité des Aliments), ce projet vise à comprendre les mécanismes moléculaires régissant les interactions entre différentes espèces de champignons du genre Fusarium. Ces champignons filamenteux pathogènes représentent un défi majeur pour l'agriculture en raison de leur capacité à contaminer diverses cultures, notamment les céréales telles que le blé, où ils produisent des mycotoxines nocives pour la santé humaine.

L'objectif de ce projet est d'analyser les séquences de petits ARNs (smARNs) produites dans chaque scénario de mono-culture ou de co-culture de différentes souches de Fusarium. Il s'agit d'identifier leurs caractéristiques distinctes, afin de mieux comprendre comment ces micro-organismes communiquent et comment cette communication peut influencer la production de toxines.

Pour ce faire, nous mettrons en place un pipeline bioinformatique intégrant différentes étapes d'analyse, du prétraitement des données à la visualisation des résultats.

Ce rapport détaille les objectifs, la méthodologie, et les résultats attendus de notre projet. Nous mettons en contexte l'importance biologique des espèces de Fusarium dans les domaines agricoles et alimentaires, ainsi que la communication au sein de la méta-communauté de Fusarium et le rôle des smARNs, notamment les miARNs, dans cette dynamique. Nous décrivons également l'état actuel de la recherche, le processus analytique, incluant le prétraitement des données, l'alignement, l'identification, la quantification, et l'analyse de l'expression différentielle des miARNs.

Notre étude vise à contribuer à la connaissance des Fusarium et à la prévention des mycotoxines dans les céréales, pour garantir la productivité agricole et la santé humaine.

Mots-clés : Fusarium, smARN, miARN RNA-Seq, pipeline bioinformatique, mycotoxines, co-culture, meta-Fusarium, communication.

Chapitre 1

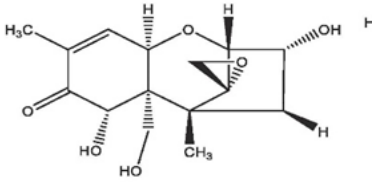
Analyse

1.1 Contexte

Il est essentiel d'examiner plusieurs aspects clés, allant de la biologie des champignons Fusarium à leur mode de communication interne. Ces éléments seront explorés en détail dans cette section.

1.1.1 Contexte biologique du genre *Fusarium*

Depuis un certain temps, les biologistes portent un vif intérêt aux champignons filamenteux du genre *Fusarium*. En effet, les *Fusarium* sont des phytopathogènes qui contaminent, entre autres, les céréales que consomme l'homme comme le blé. Chez ce dernier, ils entraînent la fusariose de l'épi qui détruit les cultures et entraîne des pertes économiques conséquentes. Ces mycètes, sont aussi à l'origine de la contamination des grains par des mycotoxines constituant un problème majeur de sécurité alimentaire. Ces toxines comme les B-trichothécènes sont très stables et se retrouvent dans les grains qui finiront dans l'alimentation. La figure 1.1 montre la formule semi-développée du Déoxynivalénol, une molécule de la famille des B-trichothécènes.



Deoxynivalenol = DON (MW=296)

FIGURE 1.1 – Formule du Désoxynivalénol, une molécule de la famille des B-trichothécènes.[1].

Jusqu'à présent, le processus de production des mycotoxines a été étudié en ne considérant "qu'un pathogène - une maladie". Cependant, des preuves irréfutables des interactions entre les espèces de *Fusarium* responsables de la fusariose, laisse suggérer que la communication entre ces champignons puisse moduler la régulation de production des toxines [2, 3].

Afin de mettre en exergue les mécanismes de production de ces molécules inter-individus, il est nécessaire de changer d'échelle d'analyse afin d'observer plus globalement le "Meta-Fusarium sp." qui comprend les principales espèces impliquées dans l'infection. La Figure 1.2 illustre les macroconidies de *Fusarium graminearum* (950X) et les microconidies de *Fusarium verticillioides* (avant *Fusarium moniliforme*) (1000X), qui sont des champignons faisant partie du Meta-*Fusarium sp* [4, 5].

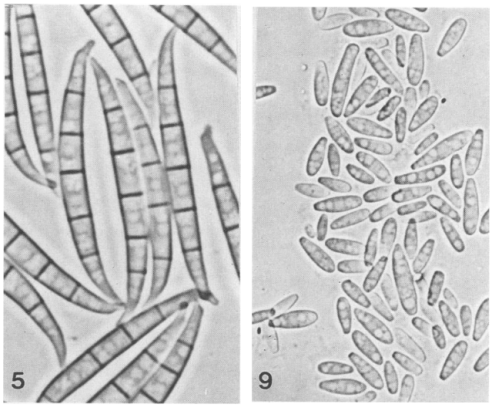


FIGURE 1.2 – (5) : Macroconidies de *F.graminearum* (950X), (9) : Microcondies de *F.verticillioides* (avant *F.moniliforme* [4]) (1000X).

1.1.2 Communication au sein du Meta- *Fusarium sp.*

Au sein du Meta-*Fusarium sp.* la communication s'effectue en partie par le biais de petits acides ribonucléiques comme les petits ARN (smRNA) ou les micro ARN (miRNA). Ces derniers sont des courtes séquences de bases non codantes qui peuvent être séquencées.

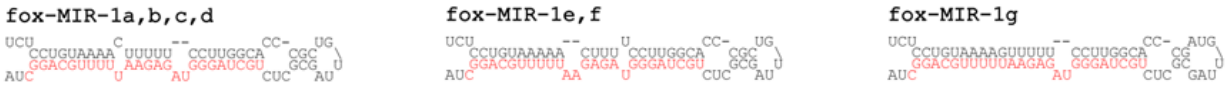


FIGURE 1.3 – Structure secondaire de quelques précurseurs miARNs déjà mis en évidence chez les champignons du genre *Fusarium* [6].

Ces petites molécules d'ARN non codantes d'environ 22 nucléotides, sont impliqués dans la régulation post-transcriptionnelle de l'expression génique. En identifiant les miARN spécifiquement produits lors de la communication induite par la rencontre entre deux Fusaria de la même souche, nous pourrons mieux comprendre les mécanismes sous-jacents aux interactions entre les espèces de *Fusarium* toxinogènes. Et ainsi, développer des stratégies de prévention contre la contamination des cultures par les champignons *Fusarium* [2, 3].

L'objectif du projet est d'analyser les données séquencées et d'identifier les microARN (miRNA) produits dans chaque scénario de culture, d'en repérer les spécificités et de les quantifier.

1.2 État de l'art

L'analyse des données issues du séquençage de l'ARN (RNA-seq) nécessite l'utilisation d'outils spécialisés, notamment pour le contrôle de qualité, le nettoyage, l'alignement des reads, ainsi que l'identification et la quantification des différents types d'ARN présents dans un échantillon biologique.

Les outils bioinformatiques utilisés pour l'analyse de RNA-seq sont variés. Afin de répondre au mieux aux problématiques du client, nous passerons en revue les recherches récentes et les méthodologies utilisées dans ce domaine. Cela nous permettra d'avoir un aperçu des meilleures pratiques et des approches les plus efficaces pour cette analyse dans le contexte des interactions fongiques.

1.2.1 Contrôle qualité et Prétraitement

Le contrôle qualité des lectures est une étape cruciale pour s'assurer de la fiabilité des analyses en aval. Parmi les outils les plus populaires, on peut citer FastQC et RSeQC pour évaluer la qualité des lectures, MultiQC pour agréger les résultats de contrôle qualité, Trimmomatic, Cutadapt et Fastp pour le nettoyage et le filtrage des lectures.

1.2.2 Alignement des reads

L'alignement des lectures sur un génome de référence permet d'identifier leur position et de détecter les événements d'épissage. Les outils les plus utilisés sont STAR, HISAT2, TopHat, Bowtie, Bowtie2, BWA-MEM, BWA-MEM2 et Subread. Ces outils offrent des algorithmes efficaces et sensibles pour aligner les lectures, et ils diffèrent en termes de rapidité, de sensibilité et de capacité à traiter des lectures longues ou courtes.

1.2.3 Identification des miARNs

Les miARN sont de petits ARN non codants impliqués dans la régulation de l'expression génique. Les outils les plus répandus pour leur identification à partir de données RNA-seq sont miRDeep2, miRge 2.0, miRNAkey, ShortStack et isomiR-SEA. Ces outils utilisent des approches probabilistes, des modèles de structures secondaires et des algorithmes d'alignement spécifiques aux miARN.

1.2.4 Quantification des miARNs

La quantification de l'expression des miARN est essentielle pour comprendre leurs rôles biologiques. Les outils comme miRDeep2, miRExpress, miRge 2.0, isomiR-SEA, ShortStack et seqcluster permettent d'estimer les niveaux d'expression des miARN à partir des données de comptage de reads.

1.2.5 Analyse de l'Expression Différentielle

L'analyse de l'expression différentielle identifie les gènes ou les miARN qui présentent des changements d'expression significatifs entre les conditions expérimentales ou les groupes d'échantillons. Les outils tels que DESeq2, edgeR, limma, DEGseq et les packages spécifiques aux miARN comme miRcomp permettent de réaliser cette analyse différentielle.

1.2.6 Autres outils essentiels

Picard Tools et SAMtools sont deux outils incontournables pour la manipulation et l'analyse des données de séquençage au format SAM/BAM. Picard Tools permet de réaliser diverses opérations comme le marquage des duplicats, la normalisation des données et le calcul de métriques. SAMtools offre des utilitaires pour trier, indexer et filtrer les lectures alignées. Bien qu'initialement conçus pour les données de séquençage de l'ADN, ces outils sont également utilisés pour le prétraitement et le contrôle qualité des données RNA-seq.

Le domaine de l'analyse des données RNA-Seq évolue rapidement avec le développement de nouvelles technologies et de nouveaux outils comme RNAdetector2, qui offre une interface graphique pratique pour analyser les ARN codants et non codants à partir de jeux de données RNA-Seq de n'importe quelle espèce biologique séquencée.

1.3 Présentation des données

Pour réaliser ce projet, nous avons travaillé avec un ensemble de données composé de 14 lots en co-culture et 8 lots en mono-culture, comme le montre la figure 1.4. Ces lots sont des données issues du séquençage Illumina à lecture courte (Short Read), fournies au format FASTQ compressé et désigné par l'extension de

fichier ".fastq.gz". Les co-cultures représentent des confrontations entre différentes souches de Fusarium. Les espèces spécifiques utilisées dans cette étude sont Fusarium graminearum, qui comprend les souches INRA349, INRA812 et INRA156, ainsi que Fusarium verticillioides avec la souche INRA63.

Confrontation	Name	i7 RUDI	index_i7	i5 RUDI	index_i5
F. graminearum INRA349 / F. graminearum INRA349	C1	i7RUDI-481	GTTCGACAAT	i5RUDI-481	AAGAGGAGAT
F. graminearum INRA349 / F. graminearum INRA349	C2	i7RUDI-482	TGGTAGGTGG	i5RUDI-482	CCATGAGTCG
F. graminearum INRA349 / F. graminearum INRA812	C3	i7RUDI-483	GTAACCGATC	i5RUDI-483	TGCGATACGC
F. graminearum INRA349 / F. graminearum INRA812	C4	i7RUDI-484	CACCTCACCA	i5RUDI-484	GTTCTCCATA
F. graminearum INRA812 / F. graminearum INRA812	C5	i7RUDI-485	CCTGATTGTT	i5RUDI-485	CCTTGGAGCT
F. graminearum INRA812 / F. graminearum INRA812	C6	i7RUDI-486	TGCACACCAG	i5RUDI-486	AGACGGTTGG
F. graminearum INRA349 / F. verticillioides INRA63	C9	i7RUDI-489	AGCCTGTATT	i5RUDI-489	TAGCATCGAT
F. graminearum INRA349 / F. verticillioides INRA63	C10	i7RUDI-490	GAAGGCAACG	i5RUDI-490	CGTATCTGCG
F. verticillioides INRA63 / F. verticillioides INRA63	C17	i7RUDI-497	TTCAATCGCT	i5RUDI-497	AATTGCGCAT
F. verticillioides INRA63 / F. verticillioides INRA63	C18	i7RUDI-498	TTGGCCAATG	i5RUDI-498	TTAATCCTCG
F. graminearum INRA156 / F. graminearum INRA156	C23	i7RUDI-503	GCATAAGCGC	i5RUDI-503	TCCTGTCAAC
F. graminearum INRA156 / F. graminearum INRA156	C24	i7RUDI-504	AGGAGGCGTA	i5RUDI-504	CACGCTGTCA
F. graminearum INRA349 / F. graminearum INRA156	C25	i7RUDI-505	CGTACTCATT	i5RUDI-505	GTACCTTGTT
F. graminearum INRA349 / F. graminearum INRA156	C26	i7RUDI-506	TAAGCGCGCT	i5RUDI-506	TACCGGTGGT
F. graminearum INRA349	C27	i7RUDI-507	AGACTACTTG	i5RUDI-507	AGGTGTTACG
F. graminearum INRA349	C28	i7RUDI-508	TACGCAGTGC	i5RUDI-508	CTAGGTTGAC
F. graminearum INRA812	C29	i7RUDI-509	GCGCTTACAA	i5RUDI-509	GCCTAGATTA
F. graminearum INRA812	C30	i7RUDI-510	ATTCCGATCT	i5RUDI-510	TATCAACTGG
F. graminearum INRA156	C31	i7RUDI-511	CGTGTAGCCT	i5RUDI-511	TTGGAATGGT
F. graminearum INRA156	C32	i7RUDI-512	CCTCCTCTTG	i5RUDI-512	GACAATAACG
F. verticillioides INRA63	C39	i7RUDI-519	TCCTCCGTCA	i5RUDI-519	GCAGGCTTAA
F. verticillioides INRA63	C40	i7RUDI-520	GTATGTCGCT	i5RUDI-520	CGAGTACAGG

FIGURE 1.4 – Présentation du jeu de données fourni par le client

Chaque lot est identifiée par un nom unique et est associée à des séquences spécifiques, nommées i7 et i5. Ces séquences sont utilisées pour le multiplexage, une technique qui permet d'identifier chaque échantillon de manière unique. Par exemple, la confrontation entre deux souches de F. graminearum INRA349, désignée sous le nom de C1, est associée aux séquences i7 RUDI-481 et i5 RUDI-481.

1.4 Analyse des besoins

Avant de concevoir notre pipeline bioinformatique, nous devons d'abord comprendre précisément les besoins et les exigences de notre analyse.

1.4.1 Besoins fonctionnels

Pour le prétraitement des données, nos besoins comprennent l'acquisition des données issues du séquençage Illumina (Short Read) générées à partir de différentes conditions de culture de Fusarium au format FASTQ. Nous devons également prétraiter et nettoyer les données brutes afin d'éliminer les séquences de faible qualité et les erreurs techniques en réalisant un contrôle qualité des lectures. Enfin, la normalisation des données est un autre besoin fonctionnel à prendre en compte.

Pour l'analyse des données, les besoins englobent l'alignement des lectures prétraitées sur les génomes de référence des souches de Fusarium, l'identification des miARNs, la détermination de leur origine génomique ou de leur localisation, et l'évaluation de leur quantité. De plus, il est nécessaire de réaliser une analyse comparative des profils d'expression des miARN entre les échantillons de cultures pures et ceux des confrontations entre souches. La représentation visuelle des résultats à travers des tableaux et des graphiques codifiés présentant les profils d'expression des miARN dans les différentes conditions de culture est un besoin fonctionnel à considérer.

1.4.2 Besoins non fonctionnels

En ce qui concerne les besoins non fonctionnels, l'utilisation des langages de programmation adaptés à la bioinformatique, tels que Python, R et éventuellement Bash est recommandée. Un traitement efficace des données dans des délais raisonnables est également attendu, ainsi qu'une fiabilité et une robustesse pour minimiser les risques de perte de données ou d'erreurs dans l'analyse. L'intuitivité pour une utilisation aisée par les chercheurs non spécialisés en bioinformatique est un aspect important, tout comme une documentation claire et des interfaces rationnelles pour faciliter la navigation et l'utilisation des fonctionnalités.

1.4.3 Contraintes

Parmi les contraintes, il est primordial d'utiliser des outils open source afin d'assurer la transparence et la réutilisabilité du système. De plus, les tâches doivent être effectuées dans un cadre à accès pseudo-restreint. Enfin, un fichier BigWig compatible avec les plates-formes de navigateur de génome fonctionnelles existantes comme JBrowse doit être produit pour garantir une visualisation adéquate.

1.4.4 Ajouts optionnels

Parmi les ajouts optionnels, nous pouvons envisager la création d'une base de données pour stocker les résultats, la mise en place de fonctionnalités de sauvegarde et de reprise du processus d'analyse, ou l'identification des cibles potentielles des miARNs dans les génomes des souches, si le temps le permet.

Chapitre 2

Conception

Passant du cadre théorique à l'application pratique, cette section décrit la conception et le flux opérationnel de notre pipeline bioinformatique.

2.1 Flux opérationnel

Voici un diagramme en 2.1 illustrant les différents éléments envisagés dans la conception de notre travail. Comme nous le verrons par la suite dans la figure 3.1, il va évoluer au gré de notre développement.

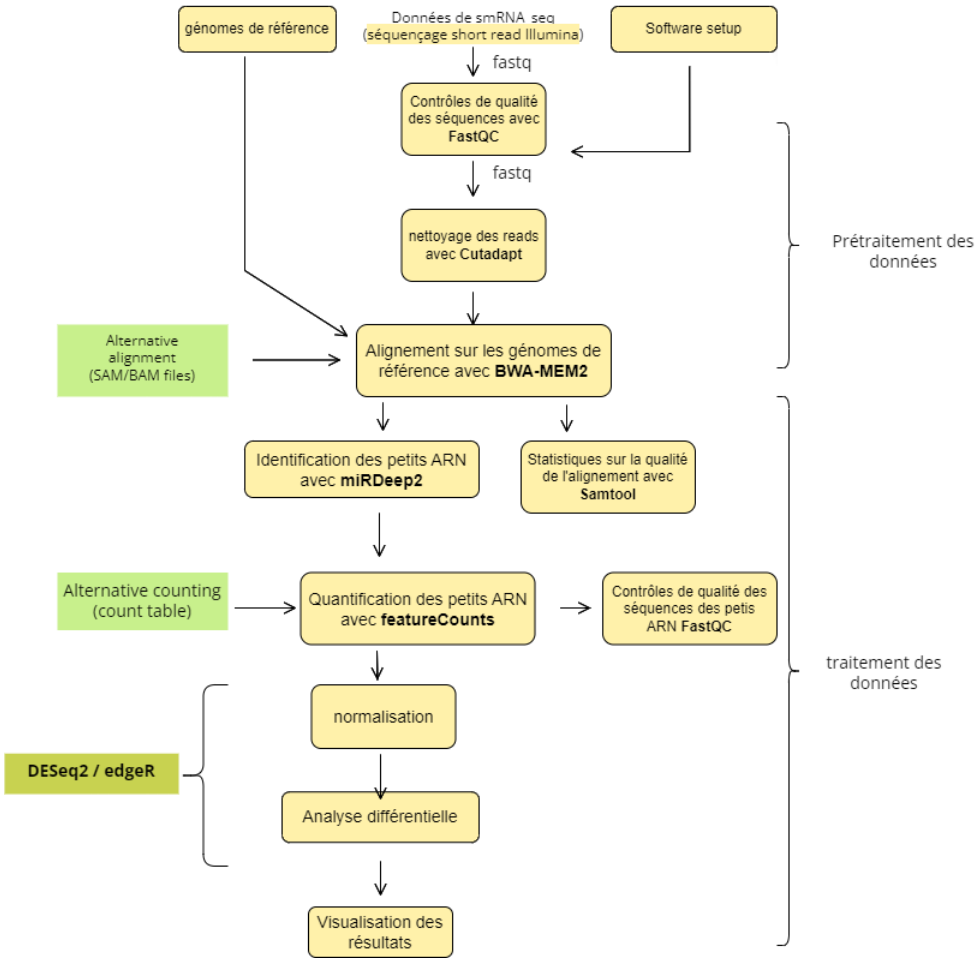


FIGURE 2.1 – Diagramme de Conception du Pipeline Bioinformatique.

2.2 Contrôle de qualité des reads

Les séquences que le client nous a fournis stockées dans des fichiers sont soumises à deux nombreuses erreurs potentielles. Avant d'arriver dans les fichiers, les séquences ont été extraites de matériel vivant, isolées, lavées, préparées, amplifiés et séquencées, etc. Afin de vérifier qu'au cours de ces étapes précises, le matériel biologique a correctement été transcrit dans les FASTQ un contrôle qualité de ces derniers est nécessaire.

Le contrôle qualité est une étape très classique appliqué sur des séquences de nucléotides. On en extraira de nombreuses métriques qui pourront nous indiquer d'une part si le séquençage s'est bien déroulé mais aussi nous donner des indications de l'anatomie générale des reads que nous avons obtenus.

Parmi les mesures que nous obtiendrons grâce à notre contrôle qualité, nous allons garder un œil sur plusieurs d'entre eux :

- La taille moyenne des reads devrait se rapprocher des tailles des miARN attendus.
- La qualité de lecture des bases devrait être forte car nous somme en short reads.
- De nombreuses séquences d'adaptateurs devrait être présentes dans nos échantillons car ils ont été utilisés pour la préparation des miARN. Leur présence devrait se traduire par un grand nombre de séquences dupliquées.

Nous avons choisi d'utiliser l'outil FastQC pour cette étape de l'analyse, car il est réputé comme étant le plus fiable pour cette tâche.

2.3 Nettoyage des reads (trimming)

En anticipant les résultats du contrôle qualité des séquences, il est prévu que plusieurs traitements seront nécessaires sur les fichiers bruts avant de pouvoir les exploiter. Le trimming représente une étape cruciale car il permet d'éliminer les séquences superflues en vue de nos analyses futures. De plus, certaines parties des reads peuvent avoir été utilisées pour leur indexation, rendant ainsi indispensable leur suppression afin de restaurer un contexte biologique cohérent.

2.3.1 Trimming classique

Lors de l'utilisation standard d'un outil de filtrage des séquences, nous débutons par la discrimination des lectures de bases de mauvaise qualité. En effet, nos données de séquençage, stockées dans des fichiers FastQ, associent chaque base à un score de qualité Phred. Ce score nous renseigne sur la probabilité que la lecture de la base actuelle soit incorrecte. Ainsi, pour garantir la fiabilité des données, il est nécessaire d'éliminer toutes les bases présentant une forte probabilité d'erreur [7].

Les séquençages Illumina présentent une perte de qualité croissante selon la longueur du fragment, attribuable aux méthodes de production. Ainsi, les bases de faible qualité se retrouvent souvent du côté 3' du brin d'ARN. Pendant le trimming, nous éliminerons les bases en 3' une par une jusqu'à ce qu'il ne reste plus de bases de faible qualité. Ce processus réduit légèrement la taille moyenne des fragments, mais dans notre cas de séquençage de miARN, ces derniers sont si courts que la qualité n'a pratiquement pas le temps de diminuer.

Ensuite, le séquençage Illumina nécessite l'utilisation d'adaptateurs pour fixer les ARN sur la flowcell avant amplification. Bien que ces séquences soient nécessaires, elles doivent être retirées des lectures avant leur étude [8].

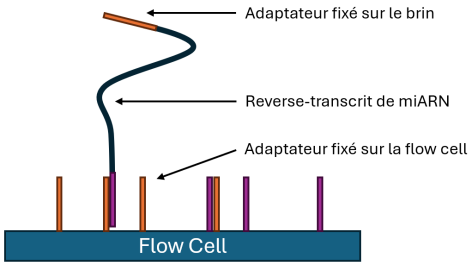


FIGURE 2.2 – Schéma d'une flowcell fixant un brin d'ADN reverse transcrit à partir d'une séquence de miARN.

2.3.2 Trimming additionnel

Au-delà des adaptateurs spécifiques à Illumina, nos données contiennent de nombreuses autres séquences qui doivent être supprimées. Voici une liste exhaustive des adaptateurs supplémentaires :

- Les **adaptateurs miRNA** : présents en 5' et 3', ils permettent d'améliorer considérablement les séquençages short reads des petits ARN.
- Les **séquences I5 et I7** : elles permettent de différencier plusieurs échantillons dans une même expérience, et sont uniques pour chaque condition expérimentale, étant présentes en 5' et 3'.
- Enfin, avant l'adaptateur Illumina, une **UDI (Unique Dual Index)** est présente en 3', permettant le multiplexage entre différentes expériences. Cela permet d'optimiser les ressources en introduisant une grande quantité de matériel génétique dans le séquenceur.

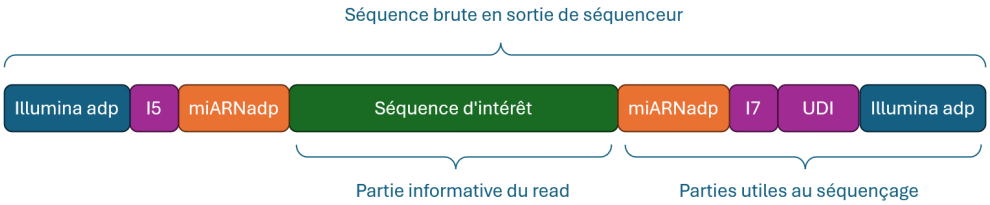


FIGURE 2.3 – Diagramme illustrant la composition des séquences brutes.

Notre code sera confronté à un défi majeur : les séquences I7 et I5, qui permettent de différencier les échantillons, sont uniques pour chaque condition expérimentale. Par conséquent, nous devons les retirer de manière dynamique. Pour gérer cette problématique, nous allons créé un fichier CSV (valeurs séparées par des virgules) qui servira de base de données pour sélectionner les différentes séquences à retirer. Les séquences I7 et I5 concernées seront intégrées à l'outil aux côtés des adaptateurs Illumina, des miARN et de l'UDI.

Nous avons opté pour Cutadapt en raison de sa spécialisation dans le retrait des adaptateurs. Cela en fait un choix optimal pour notre pipeline d'analyse de données de séquençage.

2.4 Alignement des séquences

Une fois nos séquences nettoyées et de meilleure qualité, nous allons les aligner sur leurs génomes de référence afin de caractériser leurs emplacements. Cette étape est cruciale car elle permettra d'avoir une visualisation plus globale de la distribution et de la localisation des différents ARN dans l'échantillon.

Deux complications nous font faces pour cet alignement. Tout d'abord la nature de nos séquences rend difficile leur localisation, les reads de miARN font environ 22 nucléotides de long ce qui laisse place à des erreurs d'alignement notamment sur les zones conservées de ces derniers. D'autre part, certains de nos échantillons proviennent de situations de co-culture, où plusieurs souches coexistent. Dans de tels cas, il est nécessaire

d'aligner les séquences sur plusieurs génomes de références.

Pour cette étape, nous envisageons d'utiliser BWA-MEM2, un outil réputé pour son efficacité et sa précision dans l'alignement de séquences à haut débit.

2.5 Identification des miARNs

Grâce a l'alignement des séquences, une identification des miARN sera possible. Cette dernière marquera le dernier traitement sur les données que nous effectuerons. Elle permettra d'apprécier les types de miARN produits et leurs quantités dans chaque zone du génome. Nous regarderons notamment la quantité d'ARN ribosomique ainsi que la position des reads au sein des régions non codantes.

Cette phase d'identification nous fournira des informations cruciales sur la diversité et la dynamique des populations de miARNs. Nous pourrons ainsi déterminer si cette diversité est élevée ou si certains groupes de miARNs dominent. De manière significative, nous pourrons également comparer quantitativement les profils d'expression des miARNs entre les scénarios de mono-culture et de co-culture. Cette comparaison nous permettra de détecter les miARNs potentiellement impliqués dans les interactions fongiques, notamment ceux qui sont surproduits lors de la confrontation entre deux champignons de genre *Fusarium*.

Nous envisageons d'utiliser l'outil miRDeep2, reconnu pour son efficacité dans l'identification précise des miARNs à partir des données de séquençage de petites ARN.

2.6 Quantification des miARNs

Une fois identifiés, quantifier les miARNs revêt une importance capitale pour évaluer leur expression relative dans nos échantillons. Nous avons décidé de continuer avec l'outil miRDeep2 pour estimer avec précision les niveaux d'expression des miARNs à partir des données de comptage de reads. Cette analyse quantitative nous permettra de comparer les niveaux d'expression entre différentes conditions expérimentales, offrant ainsi des insights cruciaux sur la régulation génique et les interactions fongiques.

Après quantification, nous normaliserons les données pour éliminer les biais techniques et biologiques, assurant ainsi des comparaisons précises entre les échantillons. Les miARNs présentant un fort ratio de différence d'expression entre les conditions expérimentales seront sélectionnés comme cibles prioritaires pour des études plus approfondies sur leurs fonctions spécifiques.

2.7 Pipeline bioinformatique

Pour organiser et lier tous ces outils, nous envisageons la création d'un pipeline. Ce dernier sera composé de plusieurs scripts interagissant entre eux pour manipuler les données. Chaque étape du pipeline sera conçue pour recevoir les résultats de la précédente, assurant ainsi une circulation fluide des données. Cette approche garantit une flexibilité maximale, car le programme pourra fonctionner tant que les formats de fichiers requis sont respectés. De plus, elle nous permettra d'exploiter différents langages de programmation, en choisissant celui le mieux adapté à chaque tâche.

Cette architecture modulaire présente de nombreux avantages, notamment une gestion simplifiée du code et la possibilité de réutiliser facilement des portions de code. De plus, elle offre une flexibilité accrue, permettant au client d'adapter le pipeline à de nouveaux besoins en modifiant simplement un script.

Nous envisageons d'utiliser des scripts Bash pour appeler et exécuter les outils, ainsi qu'un script Python pour offrir une interface rationnelle à l'utilisateur, lui permettant de sélectionner les tâches souhaitées. De plus, nous prévoyons de maintenir le programme flexible en permettant à l'utilisateur d'exécuter uniquement les étapes nécessaires sans être contraint de lancer l'ensemble du pipeline.

Chapitre 3

Réalisation

S'appuyant sur un pipeline bien conçu et minutieusement vérifié, la phase de réalisation implique le traitement effectif des données, passant de la théorie à la pratique.

3.1 Flux opérationnel

Voici un diagramme en 3.1 reprenant les différents points de la réalisation de notre travail.

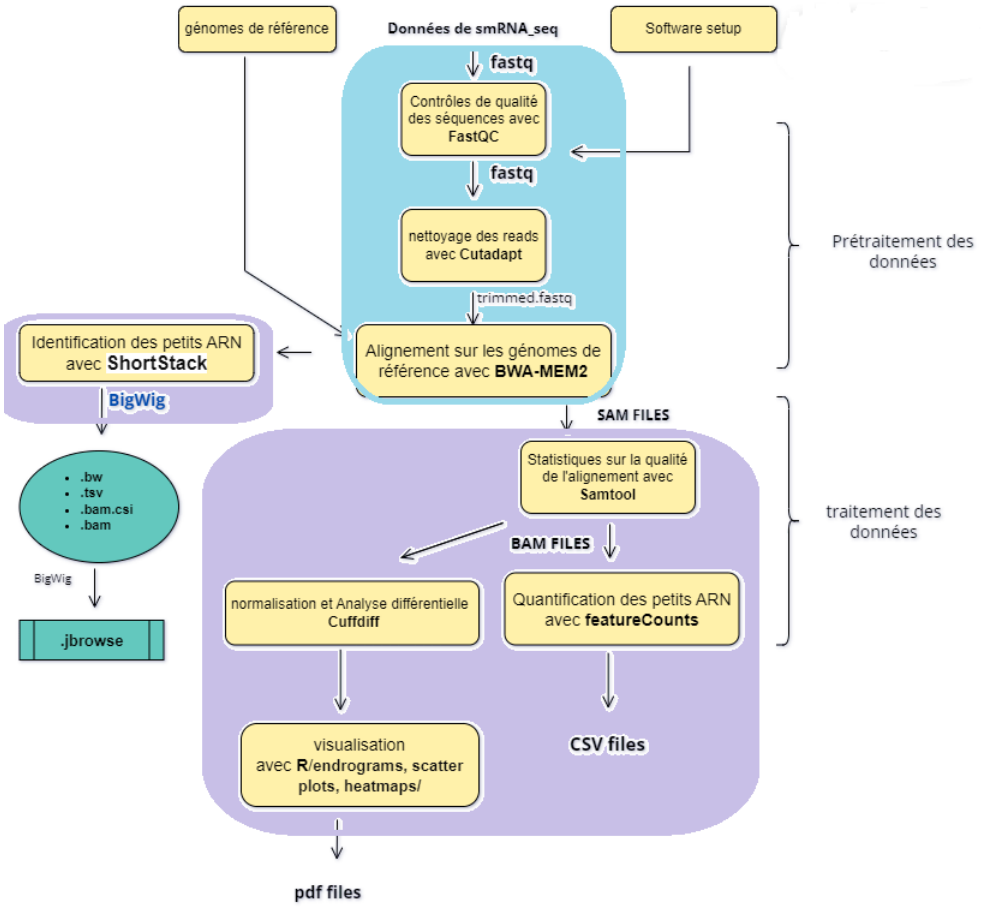


FIGURE 3.1 – Diagramme de Réalisation du Pipeline Bioinformatique.

3.2 Contrôle de qualité des reads

Avant de procéder à toute analyse, nous avons effectué un contrôle qualité des séquences à l'aide de l'outil FastQC. Cette étape est essentielle pour évaluer la qualité des données brutes et identifier d'éventuels problèmes ou biais qui pourraient affecter les résultats de nos analyses ultérieures.

3.2.1 L'outil choisi

FastQC est un outil facile à utiliser qui accepte les fichiers de données brutes en format FastQ. Il génère un rapport HTML interactif qui peut être visualisé dans n'importe quel navigateur web. Ce rapport contient une série de graphiques et de tableaux qui aident à comprendre la qualité des données de séquençage.

FastQC et à travers son rapport généré, nous permettent d'examiner différentes métriques de qualité, telles que la distribution des scores de qualité, la présence de séquences dupliquées, la présence d'adaptateurs, la qualité de base par position, etc. En examinant ces métriques, nous pouvons déterminer si les données répondent aux critères de qualité requis pour nos analyses et si des étapes supplémentaires, telles que le trimming, sont nécessaires.

L'interprétation des résultats de FastQC nécessite une certaine expertise. Chaque module du rapport fournit une "passe" ou un "échec" basé sur les seuils définis pour les données de séquençage de haute qualité. Cependant, un "échec" dans un module ne signifie pas nécessairement que les données sont de mauvaise qualité. Il est important de comprendre le contexte biologique et l'objectif de l'expérience pour interpréter correctement ces résultats [9, 10].

3.2.2 L'automatisation

Un script est écrit en Bash pour automatiser l'évaluation de la qualité des données de séquençage en utilisant FastQC. Il prend deux arguments en entrée : le répertoire d'entrée contenant les fichiers .fastq.gz et le répertoire de sortie où les résultats de l'analyse seront stockés. Si le répertoire de sortie n'existe pas, le script le crée. Ensuite, il utilise la commande find pour rechercher tous les fichiers .fastq.gz dans le répertoire d'entrée. Pour chaque fichier trouvé, il exécute l'outil FastQC, qui analyse la qualité des séquences. Un affichage en temps réel dans le terminal indiquera le pourcentage d'avancement des tâches. Les résultats de l'analyse FastQC sont ensuite enregistrés dans le répertoire de sortie.

3.3 Nettoyage des reads (trimming)

Après avoir analysé les résultats de FastQC, nous avons identifié plusieurs aspects nécessitant une attention particulière, notamment la présence d'adaptateurs, qui pourraient affecter la fiabilité des données. Ces séquences d'adaptateurs peuvent être éliminées à l'aide d'outils de trimming tels que Cutadapt. Le processus de nettoyage des données va permettre de préparer les données pour les analyses ultérieures.

3.3.1 L'outil choisi

Cutadapt représente un outil performant dédié au nettoyage des séquences, capable de filtrer les données selon de multiples paramètres. Fondé sur la transformée de Burrows-Wheeler, il permet une recherche hors ligne de motifs dans un texte. Plusieurs outils sont disponibles pour le trimming des séquences, et nous avons opté pour Cutadapt en raison de sa spécialisation dans le retrait des adaptateurs [11].

L'outil Cutadapt accepte en paramètre des longueurs maximales et minimales de lectures. Ces paramètres permettent d'exclure toutes les séquences qui ne répondent pas aux normes des miARN. Ainsi, notre longueur

maximale est fixée à 30 nucléotides et notre longueur minimale à 15 nucléotides. De cette manière, nous ne conservons que les séquences qui peuvent potentiellement nous être utiles.

3.3.2 L'automatisation

Le script qui gère l'exécution de cet outil est écrit en Bash. Tout d'abord, il appelle les différents fichiers d'entrée et de sortie des données, ainsi que le fichier CSV contenant les séquences I7 et I5. Pour permettre à l'utilisateur d'injecter ses propres séquences et éviter les éventuels problèmes et exécutions inutiles, une vérification préalable à l'exécution du trim est effectuée pour garantir que les séquences I7 etI5 sont conformes aux normes IUPAC. Ensuite, le fichier des données fastQ est temporairement décompressé avant d'être injecté dans Cutadapt. Pendant tout le traitement, les opérations réalisées par le script sont affichées dans le terminal pour fournir un retour d'information à l'opérateur.

3.4 Alignement des séquences

L'alignement des reads sur un génome est une opération a lourde charge computationnelle. Elle implique la recherche de motif (les reads) dans un texte très long (le génome).

3.4.1 L'outil choisi

BWA-MEM2 est un outil basé sur un algorithme qui associe toutes les techniques les plus modernes pour réduire au maximum le temps de calcul. Voici une liste non exhaustive des « astuces » utilisées ici :

- L'utilisations de la transformée de Burrow Wheeler, cité précédemment dans ce rapport, permet de fortement compresser le texte en regroupant les motifs de ce dernier.
- L'implémentation du parallélisme, il fait référence à la capacité d'exécuter simultanément plusieurs tâches ou instructions en utilisant plusieurs cœurs de processeur sur une même machine. En répartissant la charge de travail les opérations sont réalisées plus rapidement et efficacement.
- L'amélioration principale de BWA-MEM2 sur son prédécesseur BWA est l'implémentation d'une phase de « seeding ». Elle implique d'identifier de courtes correspondances exactes entre un read et le génome de référence. Ces correspondances exactes servent de points de départ pour le processus d'alignement. L'objectif de la phase de « seeding » est d'identifier rapidement des régions potentielles d'alignement, réduisant ainsi l'espace de recherche et la complexité computationnelle des étapes d'alignement suivantes [12, 13].

Grâce à ces optimisations, cet algorithme est environ deux fois plus rapide que son prédécesseur BWA-MEM [14, 15].

3.4.2 La séquence de commandes

Pour cet outil aussi, le script est écrit en bash, le shell permet une grande simplification du script. Ce script commence par charger les chemins de fichiers à partir d'un fichier JSON de configuration. Ensuite, il invite l'utilisateur à entrer une liste de noms de fichiers des génomes de référence et affiche cette liste. Le script vérifie et crée si nécessaire le répertoire de sortie.

L'indexation de l'alignement

Les index (dans le cadre de l'alignement) sont des prétraitements du texte qui permettent de localiser ultérieurement les reads en son sein. Ces index dans le cadre de BWA_MEM2 sont la transformée de Burrows-Wheeler, la liste des suffixes ou encore l'index de Ferragina-Manzini (compression de la Burrows-Wheeler).

Ce genre de prétraitement est important dans le cas des méthodes dites « offlines » d'alignement, ces méthodes « offlines » prétraitent le texte à l'inverse des méthodes dites « online » qui prétraitent le pattern.

Ici, nous avons des millions de patterns (reads) et seulement quelques textes (génomes de référence) nous pouvons ainsi capitaliser les méthodes offlines pour faciliter les calculs.

L'apposition des reads sur le génome

Il se trouve que la génération des indexes peut-être plutôt demandante en ressource. Ainsi dans notre script, elle sera appliquée si et seulement si le génome n'est pas déjà annoté.

Les fichiers de reads nettoyés sont listés à partir du répertoire spécifié. Pour chaque combinaison de génome et de reads, BWA-MEM2 est exécuté pour aligner les reads sur le génome, générant des fichiers SAM.

3.5 Traitement des alignements

Générer des index pour les fichiers BAM (Binary Alignment Map) est essentiel pour optimiser l'efficacité et la rapidité des analyses de séquençage. Les fichiers BAM contiennent les données d'alignement de séquences et sans index, accéder à des parties spécifiques de ces fichiers peut être extrêmement lent.

Les fichiers BAM sont produits à partir des fichiers SAM (Sequence Alignment Map). Les fichiers SAM contiennent les données d'alignement en format texte, tandis que les fichiers BAM sont des versions binaires compressées des fichiers SAM.

La suite SAMtools sera utilisée pour cette étape car elle offre une bonne polyvalente et des fonctionnalités complètes facilitant son utilisation. Avec SAMtools, nous pouvons trier, indexer et convertir ces formats, ainsi que filtrer les lectures et détecter les variants. C'est un pilier dans le traitement et l'analyse des données de séquençage de nouvelle génération.

3.5.1 La suite de commandes

Le script est écrit en Bash, langage qui permet une très simple manipulation des données d'entrée sorties. Voici les étapes de ce script :

- Tout d'abord, les différents fichiers SAM sont convertis grâce à SAMtools view. Cette conversion réduit la taille du fichier et permet un traitement plus rapide.
- Le fichier BAM est trié par position génomique avec SAMtools sort, ce qui est une condition préalable à l'indexation.
- SAMtools index est utilisé pour créer un index pour le fichier BAM trié. Cet index permet d'accéder rapidement aux alignements à des positions spécifiques sans avoir à lire l'ensemble du fichier.
- Avec SAMtools idxstats, des statistiques sur les alignements, telles que le nombre de lectures mappées et non mappées pour chaque référence, peuvent être générées.

3.6 Identification et quantification des miARNs

L'identification des miARN se fait à partir des reads alignés sur un génome de référence pour déterminer les régions du génome où les miARN en question sont exprimés.

Initialement, nous avions envisagé d'utiliser miRDeep2 pour cette tâche. Cependant, pour simplifier notre pipeline, nous avons opté pour un outil alternatif, ShortStack. Contrairement à miRDeep2, ShortStack produit directement des fichiers bigwig à partir des reads alignés, évitant ainsi une étape de conversion de fichier bam en bigwig avec l'outil bedtools. Cette approche rend notre pipeline plus efficace, rapide et économique en termes de ressources.

3.6.1 L'outil

ShortStack est un outil efficace pour l'identification des micro-ARN (miARN). Grâce à ses algorithmes sophistiqués, il permet aux chercheurs de détecter et d'analyser rapidement les miARN à partir de données de séquençage à haut débit. Ses fonctionnalités avancées permettent de filtrer les résultats et de visualiser les données de manière intuitive, facilitant ainsi l'identification des miARN régulateurs clés impliqués dans divers processus biologiques. Il supporte plusieurs formats accentuant sa flexibilité. De plus, il propose en sortie des rapports qui peuvent êtres important pour une étude plus en détail des spécificité de nos échantillons.

3.6.2 L'environnement de travail

Bien que ShortStack soit simple d'utilisation, il nécessite un environnement de travail pour fonctionner. Ainsi, Conda doit être présent et activé sur la machine de l'utilisateur pour permettre l'utilisation de cet élément du pipeline [16]. Conda est un gestionnaire de paquets open-source qui permet d'installer, de gérer et de déployer facilement des logiciels et leurs dépendances [17].

3.6.3 L'automatisation

Pour automatiser les commandes, un script Bash est créé. Lors de son exécution, le script charge d'abord les configurations à partir d'un fichier JSON, assurant une intégration transparente avec notre flux de travail établi. Il active ensuite l'environnement Conda spécifié sans que l'utilisateur n'ait à intervenir. À chaque itération, le script traite les fichiers FASTQ du répertoire de lecture désigné, en exploitant ShortStack. ShortStack est invoqué pour chaque génome spécifié dans la configuration, permettant un profilage complet des miARN. Tout au long de l'exécution, des messages informatifs sont affichés, assurant la transparence et facilitant la détection des erreurs. En sortie, nous obtenons un fichier qui comptabilise le nombre de reads alignés sur chaque partie du génome des individus.

3.7 Analyse de l'Expression Différentielle

Avant de pouvoir comparer les résultats des étapes précédentes, une normalisation doit être appliquée sur les données.

L'analyse différentielle est une étape cruciale dans le traitement des données de séquençage elle permet de détecter les régions du génome qui présentent une expression différenciée entre différents échantillons. Cette analyse permet de répondre à la question que nous nous nous posons, quel sont les zones qui sont plus exprimés en cas de co-culture.

3.7.1 L'outil

Cuffdiff est un outil couramment utilisé pour effectuer une analyse différentielle à partir de données de séquençage RNA-Seq. Il compare les niveaux d'expression entre différents échantillons identifie les zones dont l'expression est significativement différente. Cuffdiff prend en entrée des fichiers BAM contenant les lectures alignées, ainsi que des annotations génomiques au format GTF ou GFF3. Il effectue ensuite une analyse statistique et génère des rapports détaillés sur les résultats.

3.7.2 L'automatisation de l'analyse

Notre script extrait les chemins et les paramètres nécessaires à partir du fichier de configuration JSON, prépare les données en convertissant les fichiers d'annotation de format GFF en format GTF, puis exécute Cuffdiff pour chaque condition expérimentale spécifiée dans le fichier de configuration.

Il produit de nombreux fichiers d'analyse, notamment le fichier gene_exp.diff qui contient les résultats des tests statistiques de l'analyse différentielle.

3.7.3 Post-processing/Visualisation des résultats

Finalement, afin de mettre en évidence les différences entre deux échantillons, le script result_processing.r peut être appelé sur l'un des fichiers de sortie pour générer des visualisations simples telles que des heatmaps. Ces visualisations seront exportées au format PDF dans le répertoire de travail.

3.8 Fonctionnalités du pipeline

3.8.1 Programme de management

Le programme `pipeline_manager.py`, écrit en python, est un gestionnaire de scripts conçu pour automatiser l'exécution de différents scripts bash en fonction d'une configuration JSON spécifiée `configuration.json`. Voici une description de ce que fait ce script et son importance dans notre pipeline :

Chargement et validation de la configuration : La fonction `load_config()` charge le fichier JSON de configuration et appelle `validate_config()` pour s'assurer que toutes les clés nécessaires sont présentes.

Validation des champs de configuration : La fonction `validate_config()` vérifie que chaque script dans la configuration contient les clés requises (nom, chemins et ordre des chemins). Cette validation permet d'éviter des erreurs d'exécution dues à des configurations incomplètes ou incorrectes.

Exécution des scripts : La fonction `run_script()` prend la configuration d'un script, construit la commande à exécuter, puis lance le script à l'aide de `subprocess.run()`. Les arguments sont construits en respectant l'ordre spécifié dans la configuration, et des arguments supplémentaires peuvent être ajoutés si nécessaire.

Affichage du menu et choix de l'utilisateur : Les fonctions `display_menu()` et `get_user_choice()` permettent à l'utilisateur de sélectionner le script à exécuter à partir d'un menu interactif. Cela offre une interface conviviale pour gérer l'exécution des différentes étapes du pipeline.

Gestion des erreurs et des journaux : Le script utilise le module logging pour enregistrer les informations, les avertissements et les erreurs.

Boucle principale : La fonction `main()` contient la boucle principale du script, affichant le menu, récupérant le choix de l'utilisateur, et exécutant le script sélectionné. Elle gère également la sortie du programme de manière propre et ordonnée.

En pratique, il commence par charger et valider le fichier de configuration JSON. Ensuite, il propose à l'utilisateur un menu interactif pour sélectionner le script à exécuter. Une fois le choix de l'utilisateur validé, le script exécute le script bash correspondant en fonction de la configuration fournie. Si nécessaire, des arguments supplémentaires peuvent être passés au script. Le flux d'exécution est géré de manière à permettre à l'utilisateur de choisir d'exécuter d'autres processus après chaque exécution.

L'importance de `pipeline_manager.py` dans notre pipeline réside dans sa capacité à centraliser et automatiser la gestion des scripts, réduisant ainsi le risque d'erreurs manuelles et augmentant l'efficacité du processus global. Grâce à cette gestion centralisée, il est plus facile de suivre et de contrôler chaque étape du pipeline, de valider les configurations, et d'assurer une modularité et exécution cohérente et reproductible des analyses RNA-seq.

3.8.2 Utilisation du fichier JSON

Dans notre programme, nous avons fait le choix d'utiliser un fichier JSON. Ce fichier constitue un élément clé de notre pipeline bioinformatique. Il sert de fichier de configuration centralisé, spécifiant les différents scripts à exécuter, leurs descriptions, les chemins d'entrée et de sortie, ainsi que les environnements nécessaires et les paramètres spécifiques. Voici une partie de ce fichier dans la figure 3.2 :

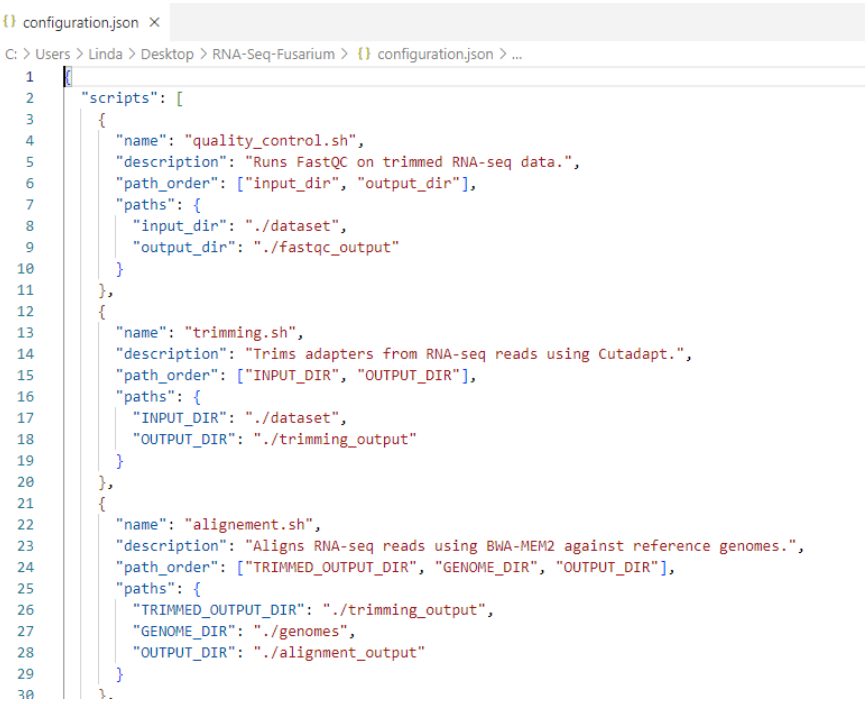


FIGURE 3.2 – Une partie de fichier JSON qui servira à la configuration des scripts.

L'importance de ce fichier JSON réside dans sa capacité à centraliser la configuration de tout le pipeline, permettant une exécution cohérente et reproductible des différentes étapes. Il assure que chaque script dispose des informations nécessaires pour fonctionner correctement. De plus, cette approche modulaire permet de modifier facilement le pipeline en ajoutant ou en modifiant des scripts sans affecter l'ensemble de la chaîne de traitement.

3.8.3 Utilisation de la journalisation (logging)

Le logging est une composante cruciale de tout programme informatique, fournissant une traçabilité de l'exécution du programme et facilitant la détection et la résolution des problèmes. Dans le contexte du développement d'une application Python pour l'exécution de commandes Bash, le logging a été intégré de manière à assurer la transparence de l'exécution du programme.

La configuration du logging est définie au début du script, spécifiant le niveau de logging sur INFO et le format des messages. Par exemple, voici une configuration de logging typique :

```
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s -
    %(message)s')
```

Dans notre programme, le logging est utilisé pour enregistrer des messages d'importance INFO et supérieure, fournissant ainsi une vue détaillée du déroulement du programme. Par exemple, des messages INFO sont utilisés pour signaler le démarrage et la fin de l'exécution des scripts, ainsi que pour afficher les arguments utilisés lors de l'exécution. Voici un exemple de message INFO enregistré pendant l'exécution du script :

```
2024-05-12 15:30:00 - INFO - Running script: /path/to/script.sh with arguments
    : [arg1, arg2, arg3]
```

En outre, le logging est également utilisé pour enregistrer les erreurs survenues pendant l'exécution du programme. Les messages d'erreur sont enregistrés avec un niveau de journalisation ERROR, permettant ainsi de les identifier facilement parmi les messages d'état normaux. Par exemple, voici un exemple de message d'erreur enregistré lorsqu'une erreur se produit pendant l'exécution d'un script :

```
2024-05-12 15:35:00 - ERROR - Failed to execute /path/to/script.sh: Return
    code 1
```

Enfin, le logging est utilisé pour signaler des situations potentiellement problématiques qui ne sont pas des erreurs graves. Ces messages sont enregistrés avec un niveau de journalisation WARNING. Par exemple, voici un exemple de message WARNING enregistré lorsque l'utilisateur saisit une entrée invalide lors du choix d'un script à exécuter :

```
2024-05-12 15:40:00 - WARNING - Invalid choice, please choose a number between
    1 and 4
```

L'ensemble de ces messages de logging fournit une traçabilité complète de l'exécution du programme, facilitant ainsi la détection et la résolution des problèmes éventuels. En enregistrant les messages dans un format structuré, il devient également plus facile d'analyser les journaux et d'extraire des informations importantes sur le fonctionnement du programme.

3.9 Fichier README

Un fichier README est essentiel pour tout repository de projet, car il fournit des informations cruciales sur l'objectif, l'installation, l'utilisation et la configuration du projet. Il sert de guide pour les utilisateurs et les développeurs, facilitant la compréhension et la mise en œuvre du projet. Il sera attaché à notre script de manière visible pour encourager sa lecture.

3.10 Utilisation de Git

Pour faciliter notre travail collaboratif et assurer une gestion efficace de notre code source, nous avons opté pour l'utilisation de GitHub comme plateforme principale. Les fonctionnalités de contrôle de version de Git ont permis aux membres de l'équipe de travailler ensemble et de suivre les modifications apportées au code tout au long du projet. L'ensemble des scripts, ainsi que le cahier des charges et le rapport en format TeX, ont été déposés dans le répertoire GitHub dédié au projet (https://github.com/Lucien-Piat/RNA-Seq-Fusarium). De plus, GitHub offre à l'ensemble de l'équipe et aux clients un accès pratique et sécurisé au code. En accordant aux clients l'accès au dépôt, ils peuvent non seulement exécuter le programme, mais également examiner le code source lui-même à tout moment.

Chapitre 4

Résultats et Discussion

Ce chapitre présente les résultats de notre analyse des données RNA-seq à travers l'exemple parlant d'un échantillon. Nous discutons également des défis rencontrés tout au long du pipeline et des perspectives d'amélioration pour optimiser ce programme. L'objectif est d'évaluer l'efficacité de notre approche tout en identifiant des pistes d'amélioration.

4.1 Analyse des résultats

4.1.1 Phase de pré-traitement

L'analyse FastQC des données brutes de séquençage a permis d'évaluer leur qualité et d'identifier des aspects critiques. Le script bash dédié au contrôle de qualité "quality_control" a appelé l'outil FastQC et l'a exécuté efficacement sur les fichiers de données brutes stockés au format FASTQ.gz dans un répertoire d'entrée spécifié “dataset”. Les résultats de cette analyse, présentés sous forme de rapports HTML interactifs, ont été générés dans un dossier de sortie dédié “fastqc_output”, facilitant leur consultation ultérieure.

Prenons l'exemple de l'échantillon C1 pour illustrer le fonctionnement du pipeline. L'analyse effectuée par FastQC sur cet échantillon a révélé une bonne qualité globale des données, comme indiqué dans la figure 4.1. Cependant, un aspect important a été mis en évidence qui est la présence significative de séquences adaptatrices dans les reads.

Ces séquences d'adaptateurs, introduites lors du processus de séquençage, risquaient de biaiser les analyses en aval et de compromettre la fiabilité des résultats.

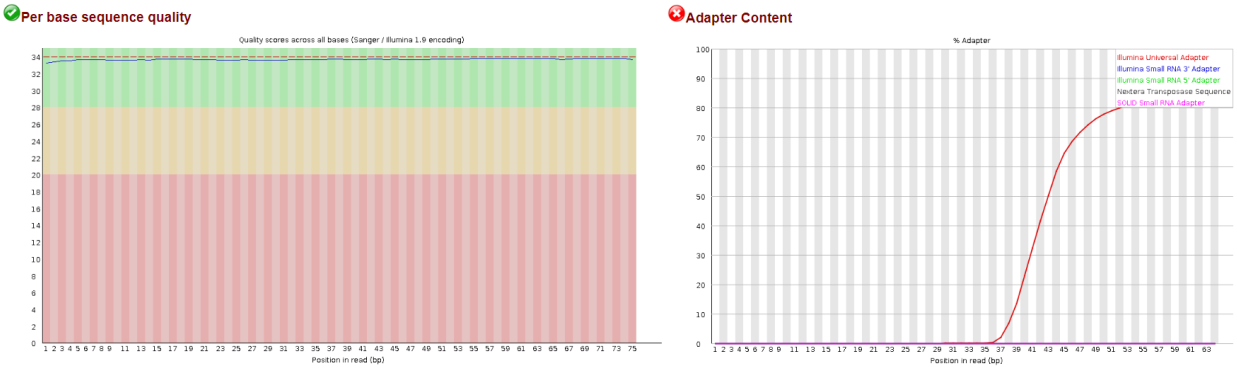


FIGURE 4.1 – Illustration d'une partie de contrôle qualité pour l'échantillon C1.

Le script de trimming "trimming.sh" a été exécuté en prenant trois arguments d'entrée : le répertoire contenant les fichiers FASTQ.gz bruts "dataset", le répertoire de sortie pour les fichiers traités "trimming_output", et un fichier CSV contenant les informations sur les échantillons et les séquences d'adaptateurs. Les séquences d'adaptateurs spécifiées comprenaient les adaptateurs Illumina, SOLID, miRNA 5' et 3', ainsi que d'autres séquences pertinentes fournies dans le fichier CSV.

Pour l'échantillon C1, le résultat final a été compressé et enregistré dans le fichier de sortie "C1_trimmed.fastq.gz". Un fichier récapitulatif "trimming_summary.txt" a été généré, fournissant des statistiques détaillées sur le processus de trimming pour chaque échantillon. Ces informations montrent que l'échantillon C1 contenait un nombre élevé de séquences d'adaptateurs, avec 100% des reads traités ayant des adaptateurs. Après le processus de trimming, seulement 8.4% des reads ont passé les filtres de qualité et de longueur définis, comme mentionné dans la figure 4.2. Ces résultats indiquent que la contamination par les adaptateurs était significative, justifiant ainsi la nécessité de cette étape de trimming pour améliorer la qualité des données pour les analyses en aval.

```
=== Summary ===

Total reads processed:              14,961,282
Reads with adapters:                14,955,970 (100.0%)

== Read fate breakdown ==
Reads that were too short:               6,194 (0.0%)
Reads that were too long:           13,696,538 (91.5%)
Reads written (passing filters):     1,258,550 (8.4%)

Total basepairs processed: 1,122,096,150 bp
Total written (filtered):     35,537,370 bp (3.2%)
```

FIGURE 4.2 – Une partie du résumé du processus de trimming pour l'échantillon C1.

Après le processus de trimming, les adaptateurs ont été efficacement éliminés, tel que démontré sur la figure 4.3, ne laissant que les séquences avec une longueur correspondant aux miARNs que nous cherchions à aligner et à identifier. Cela garantit que seules les données pertinentes et de haute qualité sont préservées pour les analyses ultérieures.

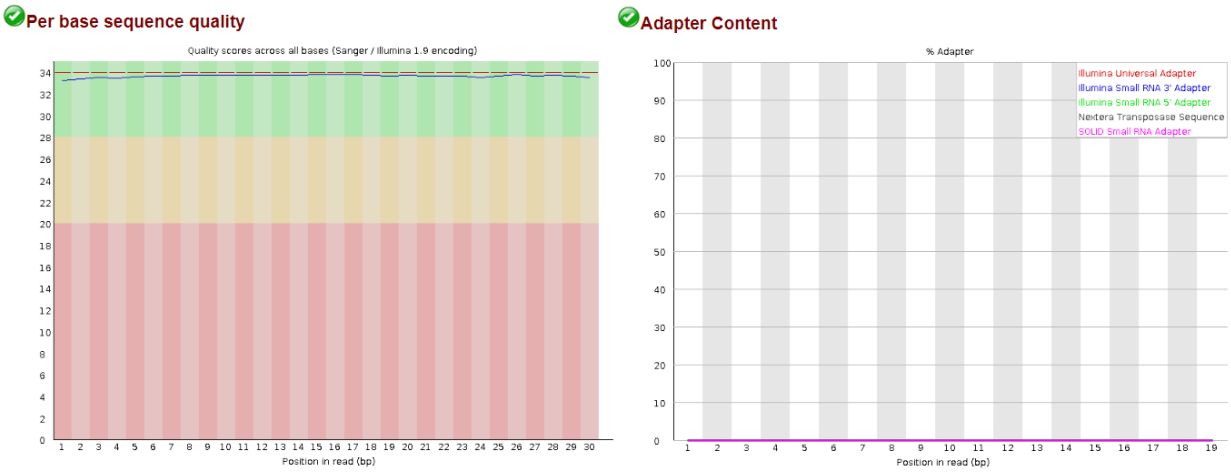


FIGURE 4.3 – Illustration d'une partie de contrôle qualité pour l'échantillon C1 après le trimming.

4.1.2 Phase de traitement

Alignement et traitement d'alignement

Un alignement des lectures (reads) de séquençage sur des génomes de référence à l'aide des scripts "alignement.sh" et "samtools.sh" a été effectue avec succes. Le script "alignement.sh", qui utilise l'outil BWA-MEM2, commence par extraire les chemins des répertoires d'entrée "trimming_output" et de sortie "alignment_output". Ensuite, il indexe les génomes de référence si cela n'a pas déjà été fait, puis aligne les lectures sur chaque génome de manière itérative. Une fois les lectures alignées, le script "samtools.sh" prend le relais. Il convertit les fichiers SAM (Sequence Alignment Map) en fichiers BAM (Binary Alignment Map), trie les fichiers BAM et génère des index pour faciliter l'accès aux données.

Poursuivant avec l'exemple de l'échantillon C1 : Les données extraites du fichier "idxstats.txt", illustrées dans la figure 4.4 et obtenues grâce au script "samtools.sh" permettent d'évaluer l'efficacité de l'alignement et d'identifier les régions du génome qui ont été le mieux couvertes par nos données de séquençage. Voici un aperçu de ces résultats.

```
Reference       Length  Mapped Reads    Unmapped Reads
HG970332        11760891        83343   0
HG970333        8997558 56096   0
HG970334        7792947 38217   0
HG970335        9395062 43437   0
HG970330        5846    0       0
HG970331        95638   7100    0
*       0       0       1030357
```

FIGURE 4.4 – Statistiques d'alignement des lectures de l'échantillon C1 sur le génome de référence.

— HG970332 : Sur une longueur de 11 760 891 bases, 83 343 lectures ont été alignées avec succès, indiquant une bonne couverture de ce génome.
— HG970333 : Avec une longueur de 8 997 558 bases, 56 096 lectures ont été alignées, reflétant également une couverture significative.
— HG970334 : Ce génome de 7 792 947 bases a été couvert par 38 217 lectures alignées.
— HG970335 : Sur 9 395 062 bases, 43 437 lectures ont été alignées.
— HG970330 et HG970331 : Bien que HG970330 et HG970331 soient des séquences de faible longueur (respectivement 5 846 et 95 638 bases), elles ont respectivement aligné 0 et 7 100 lectures, reflétant leur longueur réduite et leur probable faible représentation dans les données.

En outre, le fichier indique également la présence de lectures non alignées "Unmapped Reads". Ces lectures non alignées pourraient provenir de régions non couvertes par les génomes de référence utilisés ou de bruit de séquençage.

Les données obtenues pour l'échantillon C1 montrent une couverture significative des différents génomes de référence par les lectures alignées, ce qui indique que le processus d'alignement a été réalisé avec succès. L'analyse de ces résultats confirme l'efficacité du pipeline pour la partie d'alignement des lectures sur les génomes de référence et l'utilisation de SAMtools.

Identification et Quantification des miRNAs

Le script "miRNA_identification.sh" commence par charger une configuration à partir d'un fichier JSON, spécifiant notamment les chemins vers les fichiers génomiques "genomes" et les répertoires contenant les données de séquençage après nettoyage "trimming_output". Ensuite, il crée et active un environnement Conda spécifié dans la configuration, le cas échéant. Cet environnement est utilisé pour exécuter ShortStack4. Une fois l'environnement activé, le script parcourt chaque fichier FASTQ des reads, les aligne avec le génome

de référence correspondant en utilisant ShortStack (plus précisement l'algorithme bowtie 1.3.1), et produit des résultats détaillés au format GFF3, des fichiers BigWig permettant la visualisation sur JBrowse, ainsi qu'un résumé des alignements dans un fichier texte. Les résultats obtenus fournissent une vue d'ensemble des clusters de miARN identifiés, de leur emplacement dans le génome de référence, de leur expression relative et de leur distribution en fonction de la longueur des séquences.

Continuant avec les résultats de l'échantillon C1 comme exemple : Aprés un alignement sur le génome de *Fusarium graminearum*, des miARNs tels que ceux situés aux positions CM000578.1 :50417-50831 (Cluster_1) et CM000578.1 :72304-72718 (Cluster_2) ont été identifiés, avec des séquences majeures comme ACUCCCACUGAGGCG et UUUACAGCGCAGAUA respectivement. En ce qui concerne la quantification des miARNs, elle a été réalisée en comptant le nombre de reads alignés sur chaque cluster de miARNs.

Cluster ID	Chromosome	Début	Fin	Longueur	Reads Alignés	Séquence Majeure
Cluster_1	CM000578.1	50417	50831	415	1	ACUCCCACUGAGGCG
Cluster_2	CM000578.1	72304	72718	415	1	UUUACAGCGCAGAUA
Cluster_11	CM000578.1	473255	473879	625	3118	GAAUGGCUCAGUGAGGCGUC
Cluster_14	CM000578.1	548257	548731	475	23851	UCUUCCGUAGUAUAGUGGUC

FIGURE 4.5 – Clusters de miARNs identifiés dans les génomes de *Fusarium graminearum*.

Les résultats montrent une grande variabilité dans l'abondance des miARNs, certains étant fortement exprimés tandis que d'autres sont présents à des niveaux beaucoup plus faibles. Par exemple, le cluster CM000578.1 :473255-473879 (Cluster_11) présente un nombre élevé de reads alignés (3118 reads), indiquant une expression significative de ce miARN.

Visualisation des fichiers BigWig

Les fichiers BigWig sont largement utilisés pour représenter graphiquement les données de séquençage, telles que les niveaux d'expression génique ou les régions enrichies en séquences spécifiques, sur un génome de référence. Dans cette étude, nous avons utilisé les fichiers BigWig, conformément aux exigences de nos clients, pour visualiser les données d'expression génique et identifier les régions d'intérêt sur les génomes de Fusarium graminearum et Fusarium verticillioides.

Les fichiers BigWig ont été générés à l'aide de ShortStack, puis visualisés à l'aide de JBrowse, un navigateur parmi plusieurs de génomes interactif. Les visualisations BigWig permettent d'observer les niveaux d'expression le long des chromosomes. Par exemple, dans le fichier "C9_Fverticillioides.jbrowse" que nous pouvons voir sur la figure 4.6, nous pouvons voir des pics d'expression génique à des positions spécifiques, indiquant des régions de forte activité transcriptionnelle. Ces pics sont représentés par des valeurs élevées sur l'axe des ordonnées, correspondant à une couverture élevée des reads dans ces régions.

Un exemple parlant est l'observation d'un pic d'expression élevé sur le chromosome CM000578.1 à la position 500000-510000. Ce pic indique une région d'intérêt où plusieurs gènes sont fortement exprimés. En comparant les niveaux d'expression entre différentes conditions expérimentales, nous pouvons identifier les gènes qui sont régulés de manière différentielle. Par exemple, un gène spécifique peut montrer une augmentation significative de l'expression en condition de co-culture par rapport à la mono-culture.

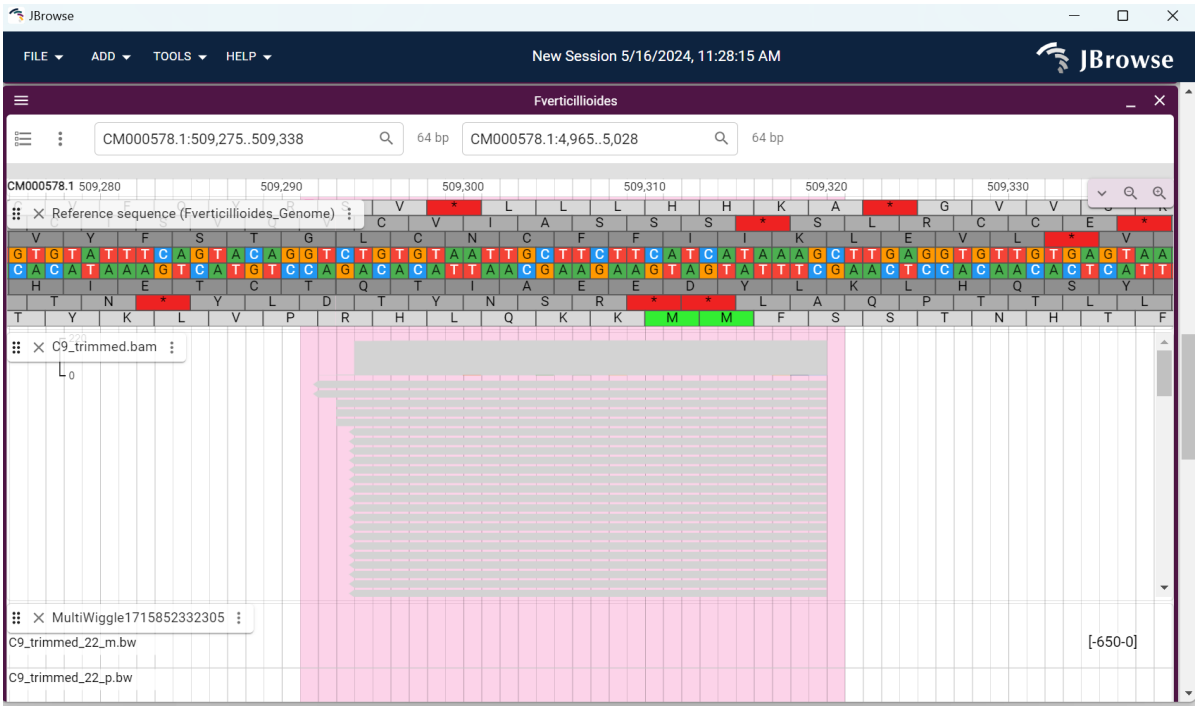


FIGURE 4.6 – Visualisation des fichiers BigWig avec JBrowse.

Analyse Différentielle

Poursuivant avec les résultats des échantillons C25 et C1, l'analyse de l'expression différentielle met en évidence plusieurs gènes et isoformes présentant des variations significatives entre les conditions de mono-culture et de coculture des souches de Fusarium. De nombreux gènes montrent une régulation à la hausse ou à la baisse en réponse à la co-culture par rapport à la mono-culture. Par exemple, le gène FVEG_15025 présente une valeur d'expression différente de zéro dans les deux conditions, avec un P-value et un Q-value de 1, indiquant que le changement de pli log2 est de 0 et n'est pas considéré comme significatif.

Les isoformes des gènes, tels qu'ABC1-iso1 et DEF2-iso2, présentent également des variations. L'isoforme ABC1-iso1 présente une diminution marquée de l'expression en co-culture, avec un changement logarithmique (log2 fold change) de -2,8 (p-value $< 0{,}05$). Cependant, les niveaux d'expression de l'isoforme FVEG_15025-t26_1 sont identiques dans les deux conditions (1, 296, 590), avec un changement de facteur log2 de 0. Cela indique qu'il n'y a aucun changement d'expression entre les deux conditions. Les valeurs p et q étant toutes deux égales à 1.

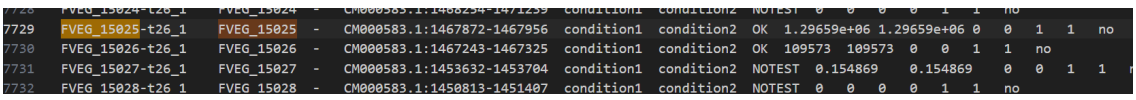


FIGURE 4.7 – Données spécifiques des gènes du fichier gene_exp.diff pour l'échantillon.

Le fichier "gene_exp.diff" se concentre sur les changements d'expression au niveau des gènes, tandis que le fichier isoform_exp.diff fournit des informations similaires au niveau des isoformes. Cette structure permet une analyse détaillée des différences d'expression sous différentes conditions expérimentales, facilitant ainsi la compréhension de la régulation à la fois au niveau des gènes et des isoformes.

Pour une discussion plus complète, il serait utile de comparer ces résultats avec d'autres isoformes ou gènes au sein du même groupe fonctionnel. Malheureusement, cela n'est pas possible en raison du manque de ressources.

Post-processing/Visualisation des résultats

Le script "result_postprocessing.r" a généré des visualisations, dont des heatmaps, pour mettre en évidence les différences d'expression entre échantillons, facilitant ainsi une comparaison visuelle des niveaux d'expression des gènes ou des isoformes. Les visualisations exportées au format PDF sont aisément consultables et intégrables dans des rapports, offrant ainsi un moyen puissant de synthétiser les données différentielles.

La figure 4.8 présente une heatmap et un dendrogramme de clustering hiérarchique pour l'échantillon C25, illustrant les différences d'expression génétique entre deux conditions expérimentales. La heatmap utilise une échelle de couleurs allant du bleu (faible expression) au rouge (haute expression) pour représenter les niveaux d'expression des gènes, facilitant la visualisation des variations d'expression. Les gènes sont listés sur l'axe des ordonnées, tandis que les échantillons sont représentés sur l'axe des abscisses. Comme exemple, les gènes FVEG_14753 et FVEG_15071 montrent des niveaux d'expression élevés dans certains échantillons, suggérant leur régulation spécifique dans ces conditions. De tels patterns peuvent indiquer des rôles fonctionnels importants et des réponses adaptatives aux interactions entre espèces.

Le dendrogramme, quant à lui, regroupe les gènes en fonction de leurs similarités d'expression, permettant d'identifier des clusters de gènes avec des profils d'expression similaires. Par exemple, les gènes FVEG_15025 et FVEG_16930 forment un cluster distinct, indiquant une forte similitude dans leurs profils d'expression. Cette proximité suggère qu'ils pourraient être co-régulés ou impliqués dans des voies biologiques similaires. L'axe des ordonnées représente la hauteur des clusters, qui est une mesure de la dissimilarité entre les groupes de gènes. Des branches plus courtes indiquent des similarités plus élevées entre les gènes au sein d'un même cluster. Par exemple, les gènes FVEG_15365 et FVEG_16095 présentent une faible dissimilarité, ce qui pourrait indiquer une co-expression étroite dans les conditions expérimentales étudiées.

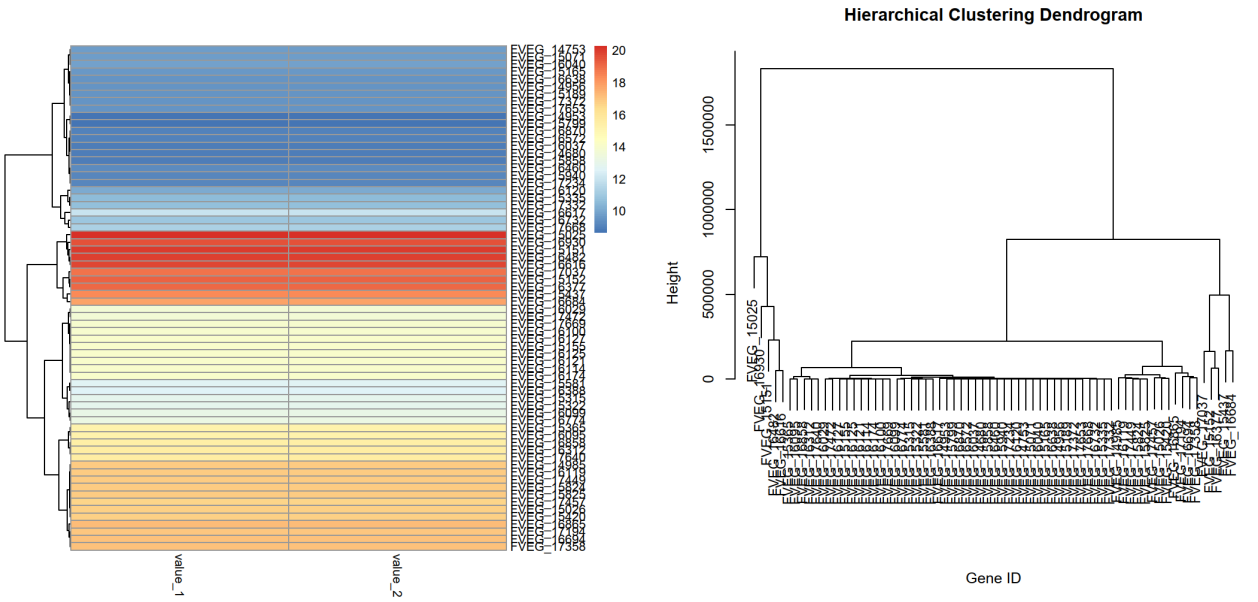


FIGURE 4.8 – Dendrogramme de clustering hiérarchique et Heatmap des niveaux d'expression génique dans l'échantillon

4.2 Difficultés rencontrées

Au cours de notre projet, nous avons rencontré plusieurs défis qui ont nécessité des ajustements et des adaptations en cours de route.

4.2.1 Changement d'outils

Bien que nous ayons effectué une recherche approfondie sur l'état de l'art, certains outils se sont avérés plus complexes à utiliser qu'initialement prévu. Nous avons dû à plusieurs reprises revoir notre jugement et en intégrer d'autres pour maintenir l'atomicité de notre pipeline.

Initialement, nous avions prévu d'utiliser miRDeep2 pour l'identification des miARN, mais nous avons trouvé cet outil difficile à utiliser en raison de sa complexité et des problèmes de compatibilité avec nos données. En conséquence, nous avons décidé de le remplacer par ShortStack. De même, pour l'analyse des données d'expression génique, nous avions prévu d'utiliser DESeq2 et edgeR, mais des difficultés similaires nous ont conduits à adopter Cuffdiff à la place. Ce changement d'outils en cours de projet a impliqué un réapprentissage et une adaptation rapide, ce qui a consommé du temps précieux et a impacté notre calendrier initial.

4.2.2 Temps de calcul

Le temps de calcul nécessaire pour traiter l'intégralité des données s'est révélé très long. En raison de ces limitations, nous avons dû travailler avec des sous-ensembles de données, ce qui signifie que certaines analyses ont été réalisées sur des échantillons partiels. Ainsi, pour les analyses différentielles, il à fallut travailler quasiment à l'aveugle.

4.2.3 Manque d'implémentation du script R dans le pipeline

Enfin, en raison des divers défis techniques et des limitations de temps, nous n'avons pas été en mesure d'implémenter complètement le script de post-traitement R dans le pipeline. Ce script est crucial pour traiter les fichiers finaux et produire les visualisations comme prévu. Les visualisations sont essentielles pour interpréter les résultats et communiquer nos conclusions de manière claire et intuitive. Le projet s'est révélé très ambitieux et cette facette du travail a souffert du manque de temps.

4.3 Perspectives

Bien que notre projet ait rencontré plusieurs défis, nous avons également réalisé des progrès significatifs dans le développement d'un pipeline robuste pour l'analyse des données RNA-seq. En envisageant l'avenir, plusieurs pistes d'amélioration se dessinent, visant à optimiser notre approche et à maximiser l'impact de nos analyses.

4.3.1 Optimisation des outils et des workflows

L'un des principaux défis rencontrés a été le choix et l'utilisation des outils appropriés pour chaque étape de l'analyse. Alors que certains outils se sont révélés complexes ou peu adaptés à nos données, d'autres ont offert des performances supérieures. Pour améliorer notre pipeline, nous envisageons d'explorer davantage d'outils et de workflows, en mettant l'accent sur la simplicité d'utilisation, la compatibilité avec nos données et l'efficacité computationnelle. Cela pourrait impliquer des tests approfondis sur des ensembles de données simulées ou réelles pour évaluer la performance de chaque outil dans des conditions variées.

4.3.2 Optimisation des ressources computationnelles

Le temps de calcul a été un autre défi majeur, limitant la taille des ensembles de données que nous pouvions traiter en une seule fois. Pour surmonter cette limitation, nous envisageons d'optimiser l'utilisation des ressources computationnelles, en explorant des solutions telles que le parallélisme, la distribution des tâches sur des clusters de calcul, ou l'utilisation de services cloud. L'objectif est de réduire considérablement

les temps de traitement et d'analyse, permettant ainsi de travailler avec des ensembles de données plus vastes et plus représentatifs.

4.3.3 Amélioration des visualisations

Les visualisations sont essentielles pour interpréter les résultats de manière claire et intuitive. Bien que nous ayons prévu des visualisations dans notre pipeline, le manque de temps a entravé leur développement complet. Pour remédier à cela, nous prévoyons de consacrer plus de ressources à la création de visualisations dynamiques et informatives, qui mettent en évidence les tendances, les relations et les différences significatives dans les données. Cela permettra non seulement de mieux comprendre nos résultats, mais aussi de communiquer nos conclusions de manière plus convaincante à un large public.

Conclusion

En conclusion, ce projet a permis de mettre en place un pipeline d'analyse de données RNA-Seq pour étudier l'expression des gènes dans le contexte de cocultures de céréales et de champignons du genre Fusarium. À travers les différentes étapes du pipeline, depuis le prétraitement des données jusqu'à l'analyse en aval.

La première étape du pipeline, l'évaluation de la qualité des données, est réalisée avec précision grâce à des outils tels que FastQC et MultiQC, fournissant ainsi des statistiques détaillées sur la qualité des lectures. Ensuite, le nettoyage des lectures est effectué avec Cutadapt, spécialisé dans l'élimination des adaptateurs, afin de préparer les lectures pour l'alignement sur le génome.

L'alignement des lectures sur le génome, réalisé avec BWA-MEM2, constitue une étape cruciale pour obtenir des résultats précis. L'utilisation de ShortStack permet ensuite l'identification exhaustive des petits ARN produits, y compris ceux déjà connus et les nouveaux, facilitant ainsi une analyse approfondie des données. De plus, l'indexation des ARN avec la suite Samtools assure une gestion efficace des données génomiques.

Malgré les défis rencontrés, notamment l'incapacité de tester tous les échantillons pour valider pleinement notre pipeline, ce projet nous a offert une expérience précieuse en matière de travail de groupe. Nous avons appris à gérer les demandes des clients et à collaborer efficacement pour surmonter les obstacles.

Enfin, le pipeline est livré sous forme de scripts Python puis d'un manuel d'utilisation, offrant ainsi une flexibilité et une facilité d'utilisation pour l'utilisateur final, tout en assurant une manipulation efficace des données.

En somme, ce pipeline sur mesure répond aux exigences spécifiques de l'analyse de données de séquençage de petits ARN dans le contexte des cocultures de céréales et de champignons du genre Fusarium, offrant ainsi une solution complète et fiable pour les besoins du client.

Bibliographie

[1] Mohamed A Gab-Allah, Yeshitila Getachew Lijalem, Hyeonji Yu, Seulgi Lee, Se-Yeong Baek, Jaejoon Han, et al. Development of a certified reference material for the accurate determination of type b trichothecenes in corn. *Food Chemistry*, 404, 2023.

[2] Nadia Ponts, Leslie Couedelo, Laetitia Pinson-Gadais, Marie-Noelle Verdal-Bonnin, Christian Barreau, and Florence Richard-Forget. Fusarium response to oxidative stress by h2o2 is trichothecene chemotype-dependent. *FEMS Microbiology Letters*, 293 :255–262, 2009.

[3] Communication MycSA. Le projet anr 2022-2026 teamtox. 2023.

[4] Keith A Seifert, Takayuki Aoki, R P Baayen, D Brayford, L W Burgess, S Chulze, W Gams, D Geiser, J de Gruyter, J F Leslie, A Logrieco, W F O Marasas, H I Nirenberg, K O'Donnell, J Rheeder, G J Samuels, B A Summerell, U Thrane, and C Waalwijk. The name fusarium moniliforme should no longer be used. *Mycological Research*, 107(6) :643–644, 2003.

[5] Peter E Nelson, Maria C Dignani, and Elias J Anaissie. Taxonomy, biology, and clinical aspects of fusarium species. *Clinical microbiology reviews*, 1994.

[6] Rui Chen, Nanyu Jiang, Qing Jiang, Xia Sun, Yulin Wang, Hong Zhang, et al. Exploring microrna-like small rnas in the filamentous fungus fusarium oxysporum. *PLoS ONE*, 9, 2014.

[7] Peter J. A. Cock, Christopher J. Fields, Naohisa Goto, Michael L. Heuer, and Peter 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, December 2009.

[8] Rosario Michael Piro. *Sequencing technologies for epigenetics : From basics to applications*, page 161–183. Elsevier, 2020.

[9] Simon Andrews. Fastqc : A quality control tool for high throughput sequence data. `https://www.bioinformatics.babraham.ac.uk/projects/fastqc/`, 2010.

[10] Simon Andrews, Felix Krueger, Anne Segonds-Pichon, Laura Biggins, Christel Krueger, and Steven Wingett. FastQC. Babraham Institute, January 2012.

[11] Marcel Martin. Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet. journal*, 17(1) :10–12, 2011.

[12] Darryl Ho, Jialin Ding, Sanchit Misra, Nesime Tatbul, Vikram Nathan, Vasimuddin Md, and Tim Kraska. Lisa : Towards learned dna sequence search, 2019.

[13] bwa mem2. bwa-mem2 github page/readme, 2022.

[14] Heng Li. Bwa-mem2 : Faster and more accurate read alignment to large reference genomes. Accès en ligne, Year. Disponible sur : https ://github.com/bwa-mem2/bwa-mem2.

[15] Md. Vasimuddin, Sanchit Misra, Heng Li, and Srinivas Aluru. Efficient architecture-aware acceleration of bwa-mem for multicore systems. In *2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, May 2019.

[16] Michael J. Axtell. Shortstack : Comprehensive annotation and quantification of small rna genes. *RNA*, 19(6) :740–751, April 2013.

[17] conda. conda user guide page, 2024.

Annexes

Diagramme de Gantt

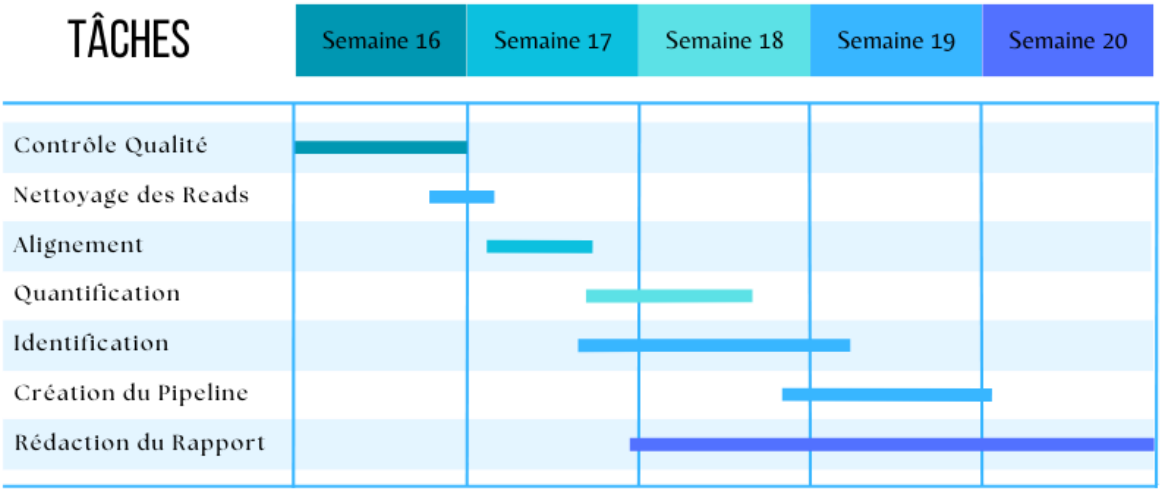


FIGURE 9 – Diagramme de Gantt qui nous servira de base pour l'organisation de notre travail