README

Programme : Analyse_transcriptomique

Créé par Sandrine VRIGNON
Date de création 07/2024

Table des matières

Bonjour,
Ce ReadMe vous permettra de mieux comprendre les objectifs, les fonctions et la procédure d'utilisation de ce programme d'automatisation. Merci d'y prêter attention.

1. Objectif

L'objectif de ce programme est d'automatiser deux analyses :

- La création et l'analyse de la qualité d'un transcriptome *de novo*
- Analyse d'expression différentielle sur des données de transcriptomique *via* un transcriptome de référence

2. Fonctionnement du programme

2.1. Généralités du programme (Figure 1)

Ce programme se divise en 12 fonctions écrites en *Python* (version 3.8.10) interagissant avec 7 scripts écrits en *Bash* (version 5.0.17(1)). Le programme s'exécute sur un terminal de commandes *via* la commande **./execution.py**.

Afin d'effectuer les calculs requis (ayant une complexité d'analyse en temps et en mémoire très importante) sur des données brutes pouvant atteindre 100Go, l'analyse s'effectue sur deux environnements de travail distincts : (1) la machine de l'utilisateur et (2) celui situé sur le cluster d'analyse de la plateforme itrop de l'IRD de Montpellier (https://bioinfo.ird.fr/ - http://www.southgreen.fr).

Le programme se divise en 5 grandes étapes :

1. Récupération d'informations demandées à l'utilisateur sur la localisation des données brutes permettant l'analyse.
2. Vérification de la conformité des données pour réaliser l'analyse et choix de l'analyse en fonction du type de données sortie séquençage (pair-end ou single-end)
3. Modification des scripts Bash (permettant de lancer les analyses) en fonction des informations fournies par l'utilisateur
4. Transfert des scripts Bash modifiés sur le cluster d'analyse
5. Exécution sur le cluster de ces scripts permettant l'analyse

Seules les étapes 1 à 4 nécessitent à l'utilisateur d'être connecté et d'interagir sur sa machine (temps de connexion et d'interaction d'environ 5 minutes). L'étape 5 s'effectue sur le cluster grâce à différents jobs liés entre eux par une dépendance des scripts sur un nœud spécifique. Cette étape peut prendre plusieurs jours et dépend du nombre et de la taille des données à analyser. L'utilisateur n'est pas dans l'obligation de maintenir sa machine allumée, ni d'interagir avec le cluster lors de cette dernière étape.

L'ensemble des résultats obtenus sont automatiquement transférés dans le dossier contenant les données brutes de transcriptomique présent sur le cluster (chemin saisi par l'utilisateur lors de l'étape 1). Ils se répartissent dans différents dossiers en fonction des analyses effectuées.

➔ Afin d'apprécier plus en détail le programme, une description plus précise de chacun des scripts va vous être présentée.

2.2. Description des scripts python

12 scripts permettent d'automatiser l'analyse :

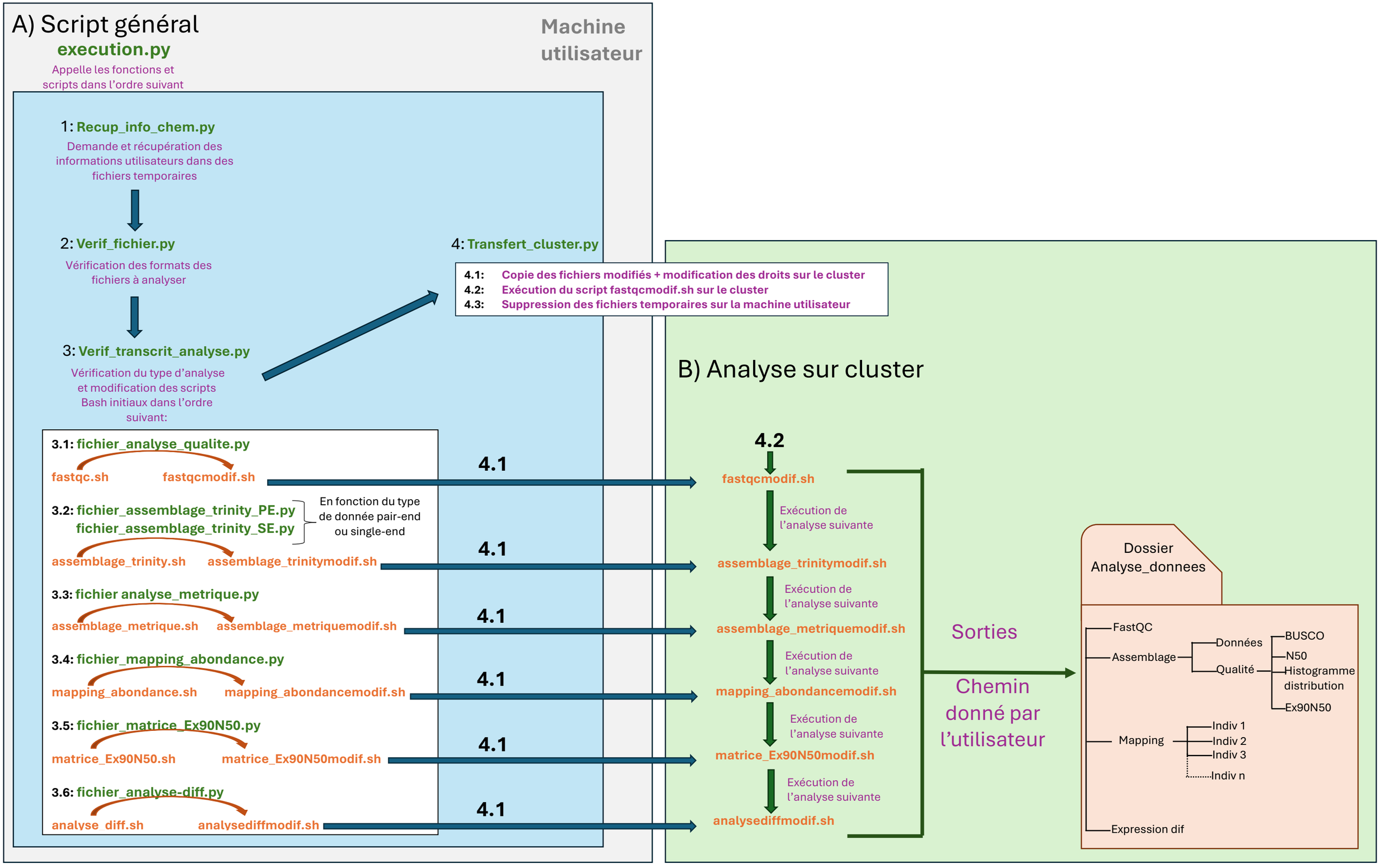


Figure 1 : Schéma du programme « Analyse_transcriptome » d'automatisation de l'analyse différentielle de données de transcriptomique.

2.2.1. Script execution.py (Figure 1 A)

Il contient la **fonction execution** qui permet le lancement de manière coordonnée, de l'ensemble des fonctions contenus dans les scripts décrits ci-dessous (sections 2.2.1 à 3.6).

2.2.1.1. Script recup_info_chem.py(Figure 1 A.1)

Il contient deux fonctions:

- **Fonction login** qui demande à l'utilisateur son login et le récupère dans une variable nommée "name" qui sera utilisée dans la fonction suivante.
- **Fonction recup_info_chem** qui prend en entrée la variable "name" (login de l'utilisateur, demandée dans la fonction précédente) et demande à l'utilisateur les informations de localisation des données (chemins), vérifie si ces chemins existent bien et stocke ces informations dans 6 fichiers textes temporairement stockés dans le dossier contenant le programme.

Les chemins demandés et les fichiers textes créés sont les suivants :

1. Chemin sur le cluster de la localisation des données qui permettront la création du transcriptome *de novo.* Puis création en parallèle des fichiers ***denovo.txt*** contenant le nom de chaque échantillon présent dans ce chemin et ***chemtranscript.txt*** contenant le chemin d'accès à ces données sur le cluster.
2. Chemin sur le cluster de la localisation des données qui permettront d'effectuer l'analyse de l'expression différentielle. Puis création en parallèle des fichiers ***donnees.txt*** contenant le nom de chaque échantillon présent dans ce chemin et ***chemmapp.txt*** contenant le chemin d'accès à ces données sur le cluster
3. Chemin sur le cluster de la localisation de la base de données proche de la nomenclature (genre,famille..) de l'espèce étudiée lors de l'évaluation de la qualité du transcriptome *de novo via* une analyse BUSCO (description des analyses effectuées en section 3.3.2). Puis création en parallèle du fichier ***cheminBUSCOBDD.txt*** contenant le chemin d'accès à cette base de données sur le cluster.
4. Chemin sur le cluster de la localisation du fichier ***sample.txt*** qui doit être préalablement créé avant le lancement de l'analyse (cf section 4.1). Puis création en parallèle du fichier ***chemsample.txt*** contenant le chemin d'accès à ce fichier sur le cluster.

2.2.1.2. Script verif_fichier.py (Figure 1 A.2)

Il contient la **fonction verif_fichier** qui vérifie et résume le nombre, les noms et formats des fichiers échantillons préalablement enregistrés dans ***denovo.txt*** et ***donnees.txt***. Si ces fichiers sont au format fastq l'analyse peut continuer. Si ce n'est pas le cas, l'utilisateur est averti du nombre ainsi que des noms des différents fichiers qui ne correspondent pas au format attendu. Dans ce cas, les fichiers temporaires sont supprimés et l'utilisateur devra effectuer des corrections concernant, soit le chemin de localisation des données, soit les types de fichiers présents au sein de ce chemin avant de relancer l'analyse.

2.2.1.3. Script verif_transcrit_analyse.py (Figure 1 A.3)

Il contient la **fonction verif_analyse** qui vérifie le type d'analyse à effectuer en fonction des données de séquençage (pair-end ou single-end). Cette fonction lance différents scripts permettant la création de scripts d'analyse modifiés en intégrant les informations fournies par l'utilisateur à partir des différents scripts *Bash* (cf. section 2.3). Il ouvre le fichier ***denovo.txt*** et vérifie grâce au numéro de répétition le type d'analyse. Dans le cas d'une analyse pair-end, une vérification des correspondances entre les répétitions est également effectuée.

2.2.1.4. Scripts Transfert_cluster.py (Figure 1 A.4)

Grâce à la **fonction transfert_cluster** et après demande du login de l'utilisateur, les fichiers *Bash* modifiés sont transférés sur le cluster. Pour l'ensemble de ces fichiers, leurs droits d'accès sont ensuite modifiés afin de permettre leur utilisation avant que le premier script (**fastqcmodif.sh**) soit exécuté. Les fichiers temporaires précédemment créés sont ensuite supprimés du dossier.

2.3. Description des scripts bash initiaux (Figure 1 B)

Afin de répondre aux objectifs, 6 scripts Bash ont été écrits. Ces scripts servent de base afin d'effectuer n'importe quelle analyse pour des études d'expression différentielle sur données transcriptomique sans transcriptome de référence dans le cas d'étude sur matériel végétal sur le cluster d'analyse de la plateforme itrop de l'IRD de Montpellier (https://bioinfo.ird.fr/ - http://www.southgreen.fr). Il se compose de :

2.3.1. fastqc.sh

Il transfert sur le nœud d'analyse, les données permettant la création du transcriptome *de novo* et permet un contrôle de la qualité des données transférées. Il crée ensuite un dossier **Fastqc** contenant ces données, puis transfert ce fichier dans le dossier résultat du chemin utilisateur avant d'exécuter le programme suivant.

2.3.2. assemblage_trinity.sh

Il permet la création du transcriptome *de novo* dans un dossier de sortie nommé **trinity_assemblage**, change les droits d'accès de ce dossier, ainsi que celui du transcriptome *de novo* créé (**Trinity.fasta**) afin qu'il soit utilisable pour la suite des analyses. Le fichier **Trinity.fasta** est ensuite écrit dans un dossier nommé **Assemblage** avant d'être transféré dans le dossier résultat du chemin utilisateur. Le programme suivant est ensuite exécuté.

2.3.3. analyse_metrique.sh

Il permet le lancement de deux analyses de la qualité du transcriptome *de novo* en créant pour chaque analyse un dossier spécifique contenant :

- l'analyse métrique de ce transcriptome au format texte (***trinityStats.txt***)
- les résultats de l'analyse BUSCO contenu dans le dossier **busco_resultat**

Ce script permet également de transférer les résultats de ces analyses dans un dossier préalablement créé et nommé **Qualite** avant d'exécuter le programme suivant.

2.3.4. mapping_abondance.sh

Il transfert sur le nœud d'analyse le fichier ***sample.txt***, ainsi que les données permettant l'analyse de l'expression différentielle. Il crée ensuite un index du transcriptome de référence (transcriptome *de novo* préalablement créé). Puis, il effectue la quantification et l'estimation de l'abondance des données à analyser contre le transcriptome *de novo* créé précédemment (section 2.3.2). Il crée ensuite un résumé du taux de mapping obtenu pour chaque échantillon dans un fichier au format texte (***resumemapping.txt***). La dernière étape de ce script consiste au transfert du dossier **Mapping** préalablement créé et contenant les résultats et résumé de l'abondance dans le dossier résultat du chemin utilisateur avant d'exécuter le programme suivant.

2.3.5. matrice Ex90N50.sh

Il permet la création d'une matrice d'expression, calcule l'expression Ex90N50, puis calcule le nombre de transcrits en fonction de l'expression Ex et ExN50 avant d'effectuer une visualisation graphique de ces résultats.
Ce script transfert ensuite les résultats obtenus dans le dossier **Qualite** qui est transféré dans le dossier résultat du chemin utilisateur avant d'exécuter le programme suivant.

2.3.6. analyse_diff.sh

Il génère les premières données d'analyse d'expression différentielle. Pour cela, il crée le fichier ***design.txt*** qui récupère une partie des informations contenues dans le fichier ***sample.txt*** (nom des échantillons ainsi que leur numéro de répétition). Ce fichier sera utilisé dans les analyses afin de faire correspondre les résultats obtenus avec l'individu analysé. Le script estime ensuite un seuil de comptage des données différentiellement exprimées, puis filtre ces données en retirant les plus rarement exprimées. Il compte le nombre de transcrits exprimés est créé un fichier contenant les résultats de cette analyse (***Trinity_trans.isoform.counts.matrix.Bat_F_vs_Bat_M.DESeq2.DE_results***) avant de créer une visualisation graphique (***Trinity_trans.isoform.counts.matrix.Bat_F_vs_Bat_M.DESeq2.DE_results.MA_n_Volcano.pdf***) des données différentiellement exprimées. Ces données contenues dans le dossier préfixé **DeSeq2** sont transférées dans un dossier nommé **Expressiondif** qui est transféré à son tour dans le dossier résultat du chemin utilisateur.

3. Description des analyses et réalisation de celles-ci (Figure 2)

3.1. Contrôle de la qualité des données

Un contrôle de la qualité de chaque échantillon sortie séquençage ou trimmé est effectué grâce au programme *Fastqc* (version 0.11.9), puis une compilation de ces analyses est réalisée grâce au programme *multiqc* (version 1.9).

3.2. Création du transcriptome *de novo*

Afin de créer le transcriptome *de novo*, le programme *Trinity* (version 2.5.1) est utilisé avec les options suivantes :
--seqType fq : spécifie le type de fichier à analyser (au format fastq)
--max_memory 50G : spécifie que la mémoire maximale à utiliser est de 50 Giga octet
--normalize_by_read_set : permet la normalisation des données. Cette normalisation se fait dans un premier temps individuellement pour chaque paire limitée à 200, puis en effectuant une combination finale des lectures normalisées individuelles.
-- left : suivi du nom des différents échantillons spécifiant les séquences forward ; --right : suivi du nom des différents échantillons spécifiant les séquences reverse. Ces deux option sont à utilisées pour des analyses effectuées sur un séquençage pair-end
--single : suivi du nom des différents échantillons spécifiant les séquences pour des analyses effectuées en single-end
--output trinity_assemblage : spécifie le nom du dossier contenant l'ensemble des fichiers issus de la création du transcriptome *de novo*

3.3. Analyse de la qualité du transcriptome *de novo*

Afin de visualiser la qualité du transcriptome *de novo*, trois analyses sont effectuées par le programme :

3.3.1. Analyse métrique (https://github.com/trinityrnaseq/trinityrnaseq/)

L'analyse métrique contient des informations sur le nombre total de transcrits et le nombre de 'gènes' (nombre de transcrits regroupés en cluster en fonction du nombre de séquences partagées), ainsi que la statistique Nx (statistique de longueur conventionnelle des contigs de sorte qu'au moins x% des nucléotiques de transcrits assemblés se trouvent dans des contigs qui sont au moins de la longueur Nx). Cette analyse est effectuée *via* le script **TrinityStats.pl** issus de la boîte à outils du programme *Trinity* (version 2.5.1).

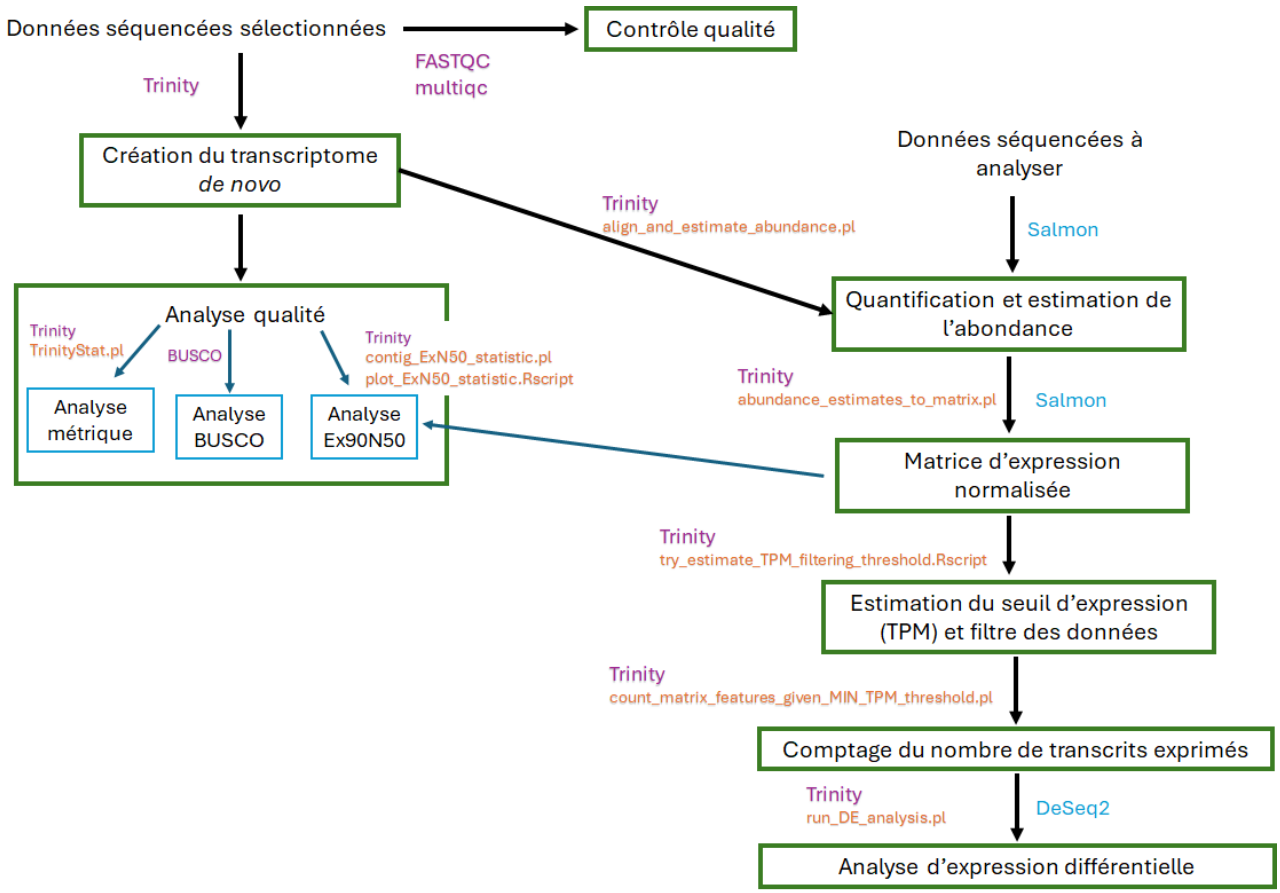


Figure 2 : Description et enchainement des analyses effectuées par le programme Analyse_transcriptomique. Les carrés verts représentent les différentes étapes de l'analyse ; Les carrés bleus représentent les analyses complémentaires ; Le texte en violet représente les programmes utilisés au sein de la pipeline d'analyse ; le texte en orange représente les scripts utilisés au sein du programme Trinity ; le texte en bleu représente les choix d'analyses effectués.

3.3.2. Analyse BUSCO (https://busco.ezlab.org/)

Cette analyse permet, à partir de mesures quantitatives, d'obtenir des informations sur la complétude du transcriptome nouvellement créé en le comparant à une base de données de référence répertoriée dans le programme. Elle est réalisé avec le programme *BUSCO* (version 5.5.0) en utilisant les options suivantes:
-i Trinity.fasta : spécifie le nom du transcriptome de référence à utiliser
-m transcriptome : spécifie que l'analyse est réalisée sur des données de transcriptomique
-c 2 : spécifie que le nombre de cœurs doit être de 2 pour réaliser l'analyse
-o busco/busco_resultat : spécifie la localisation et le nom du dossier contenant les résultats de l'analyse
-l : spécifie la localisation ainsi que le nom de la base de données de comparaison permettant l'analyse de la complétude

3.3.3. Analyse du Ex90N50 (https://github.com/trinityrnaseq/trinityrnaseq/wiki)

Les valeurs du Nx, et notamment du N50 donnent une première idée sur la longueur des contigs. Cependant, cette dernière peut souvent être surestimée dû à une génération plus importante du nombre d'isoformes. Il est donc important d'analyser ces données uniquement sur les 'gènes' les plus exprimés. Ainsi il est important de regarder l'expression à 90% du N50 (Ex90N50).
Cette analyse a été effectuée en utilisant les résultats issus de la construction de la matrice d'expression normalisée présentée en section 3.5, ainsi que les scripts **contig_ExN50_statistic.pl** (donnée statistique) et **plot_ExN50_statistic.Rscript** (visualisation graphique) issus de la boite à outils du programme *Trinity* (version 2.5.1). Pour ces analyses, aucune option spécifique n'a été utilisée.

Il est important de noter que suite à différents problèmes rencontrés (compatibilité entre programmes, absence ou oubli de maintien sur le cluster), l'utilisation du programme *Trinity* est effectuée en tant que module par l'intermédiaire du conteneur *Singularity* (version 4.0.1). L'utilisation de ce conteneur sera utilisée pour l'ensemble des analyses suivantes.

3.4. Quantification et estimation de l'abondance permettant l'analyse de l'expression différentielle (https://github.com/trinityrnaseq/trinityrnaseq/wiki)

Afin de permettre l'analyse de l'expression différentielle, il est nécessaire d'effectuer un alignement et une quantification des données à analyser contre un transcriptome de référence. Dans le cas présent, ce transcriptome de référence correspond au transcriptome *de novo* préalablement créé (cf section 3.2). Cet alignement, ainsi que l'estimation de l'abondance est effectué *via* le quasi-mapper *salmon* présent dans le script **align_and_estimate_abundance.pl** contenu dans le programme *Trinity* (version 2.5.1). Les options utilisées pour permettre cet alignement sont les suivantes :
-e : permet l'exécution du script contenu dans le programme *Trinity via* le conteneur *Singularity*
--transcripts : spécifie la localisation et le nom du transcriptome de référence à indexer et que l'analyse s'effectuera sur des transcrits
--seqType fq : spécifie que le type des données pour l'analyse sont des données fastq
--samples_file : spécifie la localisation et le nom du fichier contenant les conditions d'analyse (***sample.txt***)
--est_method salmon : spécifie que la méthode d'analyse de l'abondance se fait avec le quasi-mapper *salmon*
--trinity_mode : spécifie que le mode d'analyse pour l'indexation se fait *via* le fichier **Trinity.fasta**
--outputdir : spécifie la localisation de la sortie des analyses
--prep_reference : spécifie que le nom du fichier de sortie regroupant l'ensemble des échantillons

3.5. Construction de la matrice d'expression normalisée (https://github.com/trinityrnaseq/trinityrnaseq/wiki)

Afin de comparer les données d'expression il est important de normaliser les données et de créer une matrice d'expression. Cette analyse est effectuée *via* le script **abundance_estimates_to_matrix.pl** issu de la boite à outils du programme *Trinity* (version 2.5.1). Les options utilisées sont les suivantes :
-e : permet l'exécution du script contenu dans le programme *Trinity via* le conteneur *Singularity*
--est_method salmon : spécifie que la méthode d'analyse de l'abondance se fait avec le quasi-mapper *salmon*
--out-prefix : spécifie le rajout d'un préfixe dans le nom du fichier en sortie d'analyse (ici Trinity_trans)
--name_sample_by_basedir : spécifie que le nom de la colonne doit être identique au nom du répertoire
--gen_trans_map none : spécifie qu'aucun fichier de mapping gène-transcrit n'est utilisé

3.6. Analyse de l'expression différentielle (https://github.com/trinityrnaseq/trinityrnaseq/wiki)

L'analyse de l'expression différentielle passe par une étape préliminaire d'estimation du seuil d'expression (TPM) au-dessus duquel les transcrits sont conservées. Les transcrits trop faiblement exprimés et ayant une significativité biologique discutable sont alors retirés de l'analyse. Cette analyse se fait *via* le script **try_estimate_TPM_filtering_threshold.Rscript** issu de la boîte à outils du programme *Trinity* (version 2.5.1). Les options utilisées sont les suivantes :
-e : permet l'exécution du script contenu dans le programme *Trinity via* le conteneur *Singularity*
--E_inputs Trinity_trans.isoform.TMM.EXPR.matrix.by-transcript.E-inputs : spécifie le nom du fichier à utiliser pour l'analyse

Une fois cette estimation effectuée, un comptage du nombre de transcrits exprimés est effectué grâce au script **count_matrix_features_given_MIN_TPM_threshold.pl** issu de la boîte à outils du programme *Trinity* (version 2.5.1). Pour cette analyse, aucune option spécifique n'a été utilisée.

L'analyse de l'expression différentielle est ensuite effectuée grâce au package *DeSeq2* du programme R lancé grâce au script **run_DE_analysis.pl** (issu de la boîte à outils du programme *Trinity* (version 2.5.1)). Les options utilisées sont les suivantes:
-e : permet l'exécution du script contenu dans le programme *Trinity via* le conteneur *Singularity*
--matrix Trinity_trans.isoform.counts.matrix : spécifie la matrice utilisée pour réaliser l'analyse
--method DESeq2 : spécifie que l'analyse statistique permettant l'analyse d'expression différentielle se fait via le package *DESeq2* du logiciel *R*
--sample : spécifie le fichier correspondant aux échantillons en fonction des niveaux du facteur (ici ***design.txt***)

4. Mode opératoire

4.1. Prérequis

En premier lieu merci de vérifier que vous disposez d'une clé de connexion et d'un compte pour l'accès au cluster d'analyse.

Avant de lancer l'analyse, il est important de vérifier plusieurs points :

4.1.1. Nom des échantillons

Le nom des échantillons à analyser pour la création du transcriptome *de novo* ainsi que pour l'analyse d'expression différentielle doivent être au format suivant :

- Pour les données provenant d'un séquençage en single-end : echantillon_1.reste.fastq

- Pour les données provenant d'un séquençage en pair-end: echantillon1_1.reste.fastq et echantillon1_2.reste.fastq

Le « _1 » associé correspondra aux échantillons forward et « _2 » correspondra aux échantillons reverse.

4.1.2. Localisation des données

Les données doivent être situées dans un dossier présent sur le cluster est accessible depuis le programme *FileZilla* (version 3.67.0). Dans ce dossier, seules les données à analyser devront s'y trouver comme montré dans l'exemple image 1. Ainsi, pour l'utilisation du programme il y aura deux dossiers : un dossier contenant les données permettant la création et l'analyse du transcriptome *de novo* et un dossier contenant les données permettant l'analyse d'expression différentielle.

A) B)

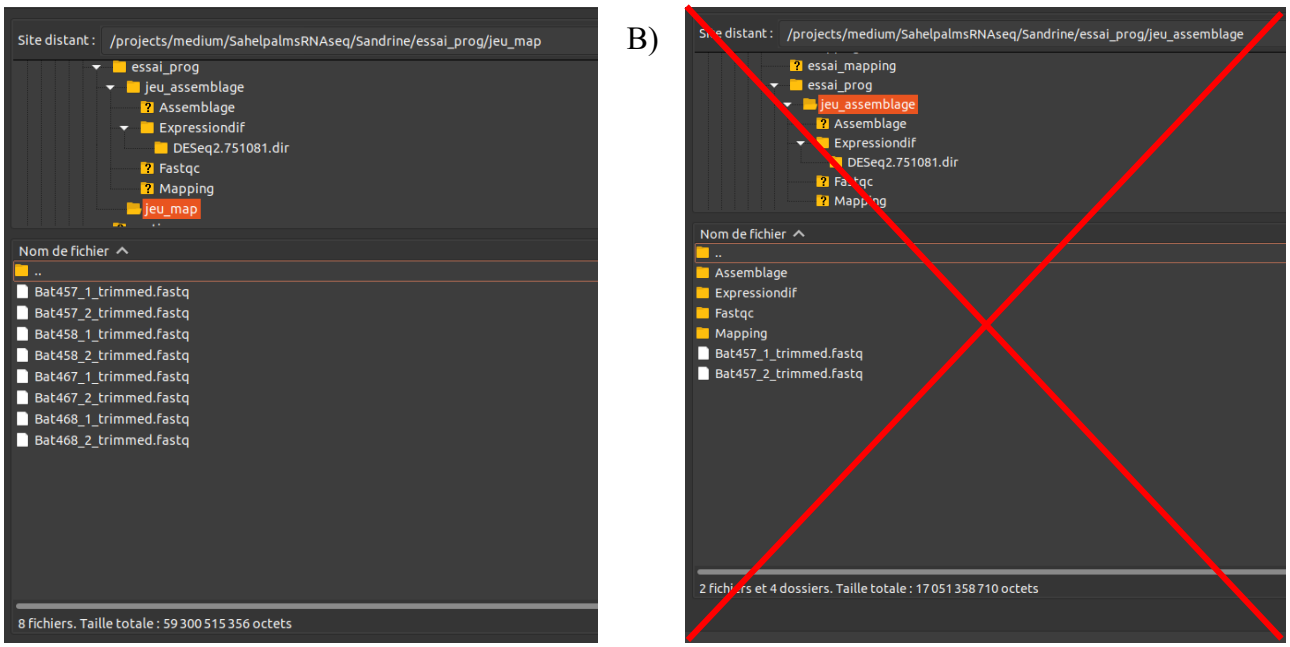


Image 1: Exemple de dossier contenant les données à analyser pour l'utilisation du programme visualisé grâce à *Fillezilla* (version 3.67.0). A) Exemple d'un dossier adapté à l'analyse ; B) Exemple d'un dossier non adapté à l'analyse.

4.1.3. Création du fichier ***sample.txt***

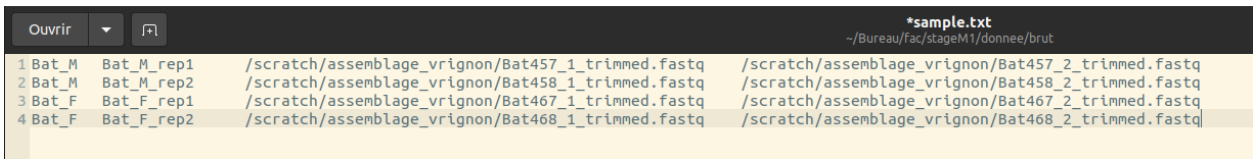
Ce fichier au format texte sera utilisé dans les analyses de quantification lors de l'analyse d'expression afin de faire correspondre les résultats obtenus avec l'individu analysé. Pour cela, sa création est nécessaire.

Ce fichier doit contenir 3 ou 4 colonnes (en fonction du type des échantillons) séparées par des tabulations, contenant, pour chacune d'entre elle les informations suivantes (exemple image 2 A et B):

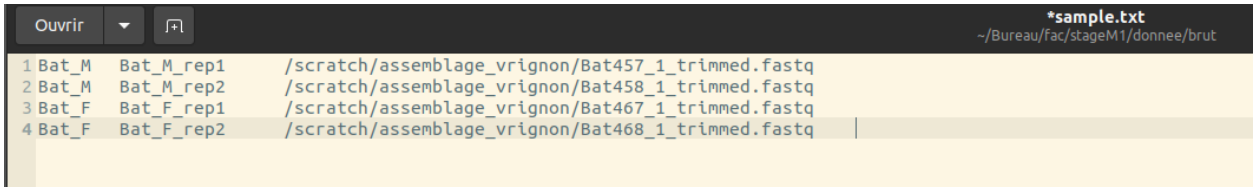
- Les traitements
- Les répétitions expérimentales
- La localisation des fichiers en copiant cet accès: **/scratch/assemblage_votre login/echantillon_1.reste.fastq** et en ne remplaçant que 'votre login' par votre login et echantillon_1.reste.fastq par le nom de votre échantillon.

Attention il est important que votre fichier ne contienne aucune ligne vide (exemple image 2 C)

A)



B)



C)

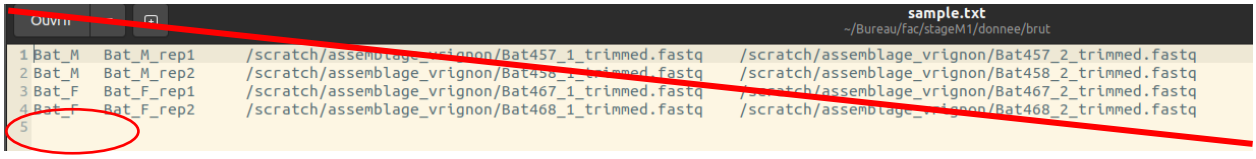


Image 2: Exemple de fichier sample.txt contenant les informations à analyser (M : représente les échantillons mâles et F : représente les échantillons femelles). A) Représentation du fichier pour des échantillons séquencés en pair-end ; B) Représentation du fichier pour des échantillons séquencés en single-end ; C) Représentation du fichier ne permettant pas l'analyse étant donnée la présence d'une ligne vide.

4.1.4. Récupération de la base de données pour les analyses BUSCO

Afin d'effectuer cette analyse, il est important de récupérer la base de données la plus proche de l'organisme étudié. En théorie, vous n'avez qu'à inscrire le nom de la base de données à utiliser suivant la liste que vous trouverez sur le site de la documentation du programme *BUSCO* (https://busco-data.ezlab.org/v5/data/lineages/). *BUSCO* fera automatiquement la recherche de cette base de données *via* le logiciel avant d'effectuer l'analyse.
Cependant, suite à des problèmes de compatibilité et de mise à jour sur le cluster, il est fortement conseillé de télécharger la base de données voulue et de la décompresser dans le dossier de votre choix du cluster. Les bases de données à télécharger sont accessibles sur le lien suivant : https://busco-data.ezlab.org/v5/data/lineages/

4.2. Procédure

Afin de réaliser l'analyse, merci de vous rendre sur le lien github suivant: https://github.com/sandrinevrignon/Analyse-transcriptomique et télécharger le dossier au format zip (**Image 3).**

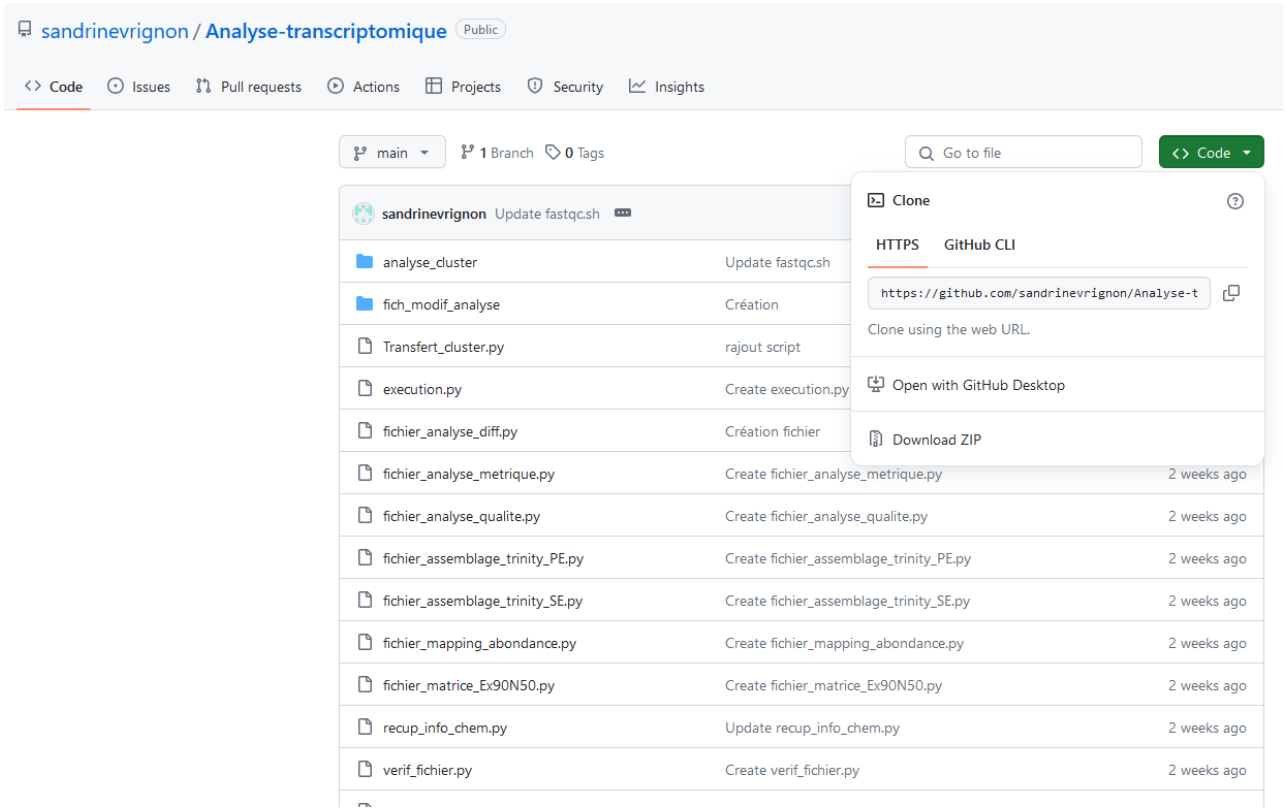


Image 3: Téléchargement du dossier Analyse-transcriptomique *via* le lien github

Placer le dossier à l'endroit souhaité sur la machine, puis dézippez-le (image 4)

A)

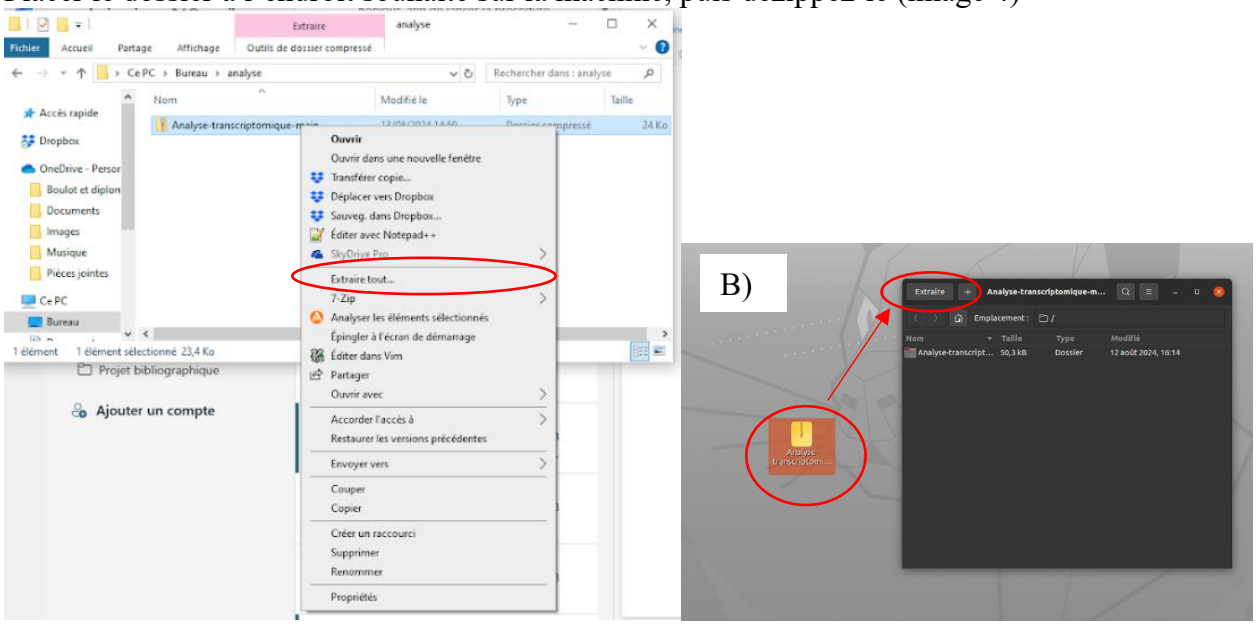
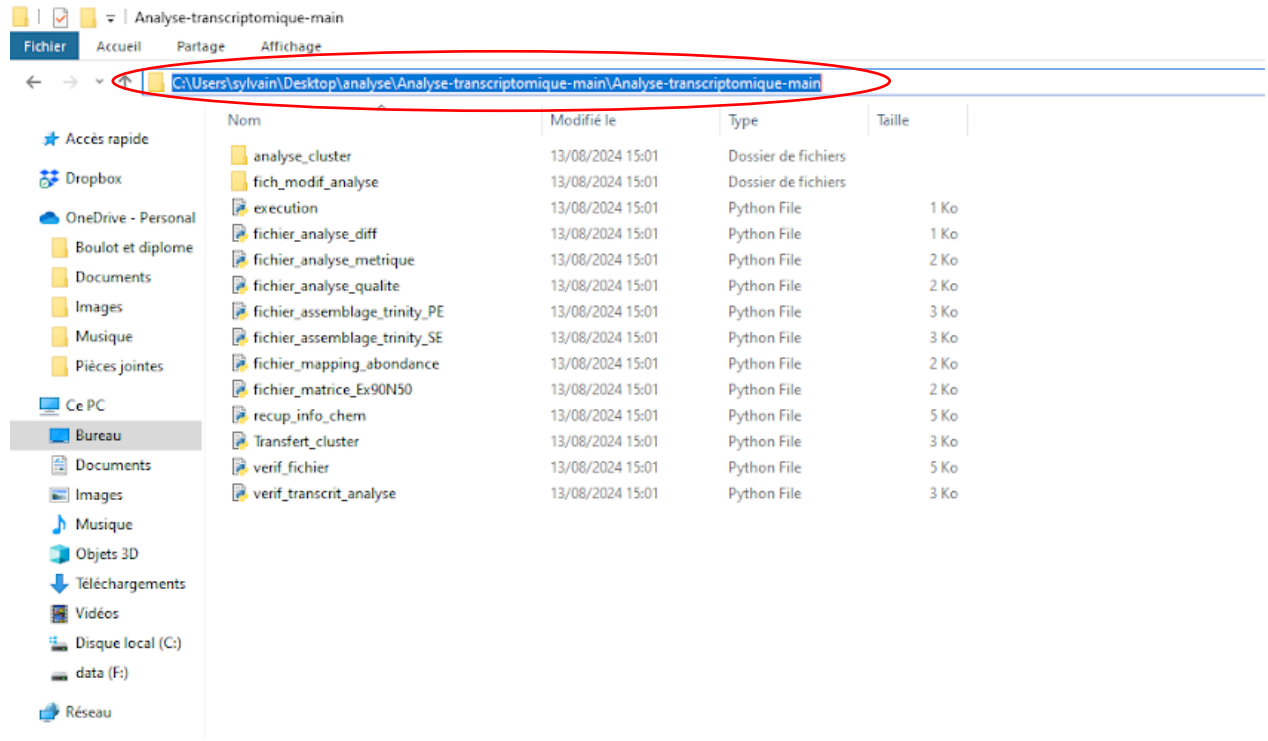


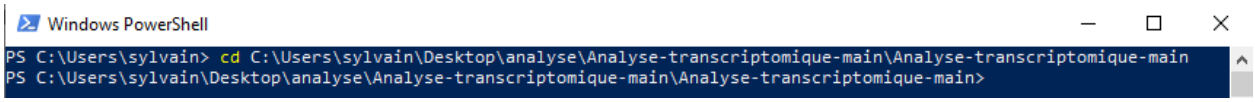
Image 4: Procédure d'extraction du fichier. A) Procédure sur machine Windows ; B) Procédure sur machine Linux Ubuntu.

Dans le cas de l'utilisation de ce programme sur une machine Windows, merci d'ouvrir votre terminal de commande et de vous placer dans le dossier Analyse-transcriptomique. Pour cela, copier la barre du chemin d'accès dans lequel se trouve votre dossier(image 5 A). Puis dans votre terminal, écrire « cd » suivi d'un espace puis coller le chemin d'accès (image 5B). Ensuite, exécutez le programme en tapant « .\execution.py » (image 5C)

A)



B)



C)

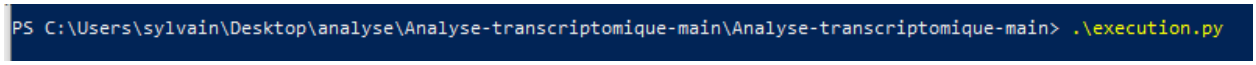
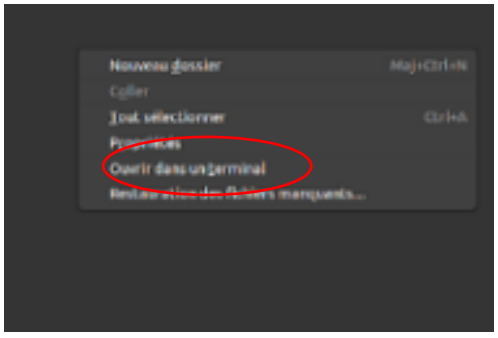


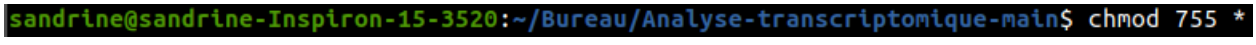
Image 5: Procédure d'exécution du programme sur machine Windows. A) Localisation de la barre du chemin d'accès ; B) Déplacement de la localisation du travail dans le dossier d'exécution ; C) exécution du programme.

Dans le cas de l'utilisation d'une machine Ubuntu, merci de vous placez à l'intérieur du dossier « Analyse-transcriptomique », avec votre souris effectuez un clic gauche et cliquez sur ouvrir dans un terminal (image 6 A). Il est impératif de changer les droits d'exécution de l'ensemble des fichiers avec la commande « chmod 755 * » (image 6B) puis exécuter le programme en inscrivant « ./execution.py » (image 6C)

A)



B)



C)

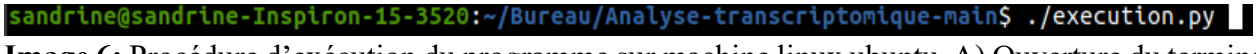


Image 6: Procédure d'exécution du programme sur machine linux ubuntu. A) Ouverture du terminal ; B) Changement des droits de l'ensemble des fichier ; C) exécution du programme.

Vous lancez ainsi le programme Analyse_transcriptomique. Il vous suffit ensuite de suivre les indications du programme. Vous trouverez ci-dessous deux exemples de l'interface utilisateur : (1) exécution du programme présentant une erreur lors de la copie du chemin (image 7) et (2) exécution totale du programme sans erreur (image 8).
Vous pouvez voir l'avancée de l'analyse sur le cluster en vous y connectant et en tapant « squeue -u » + votre login (image 9). Le rapport d'exécution de chacune des analyses est disponible sur le cluster, il est possible de le consulter directement (exemple image 10). Afin de visualiser les rapport d'exécution des analyses, veuillez écrire:

- cat fastqc.txt → rapport du contrôle qualité des données sortie séquençage ou trimmées (script fastqcmodif.sh)
- cat assemblage_trinity.txt → rapport de l'analyse permettant la création du transcriptome *de novo* (script assemblage_trinitymodif.sh)
- cat analyse_metrique.txt → rapport de l'analyse qualité du transcriptome *de novo* (analyse_metriquemodif.sh)
- cat mapping_abondance.txt → rapport de l'analyse de quantification des données pour l'analyse d'expression différentielle (script mapping_abondance.sh)
- cat matrice_Ex90N50.txt → rapport de l'analyse Ex90N50 (script matice_Ex90N50modif.sh)
- cat analyse_diff.txt → rapport de l'analyse d'expression différentielle (script analyse_diffmodif.sh)

L'image 11 vous présente le dossier final contenant les résultats d'analyse.

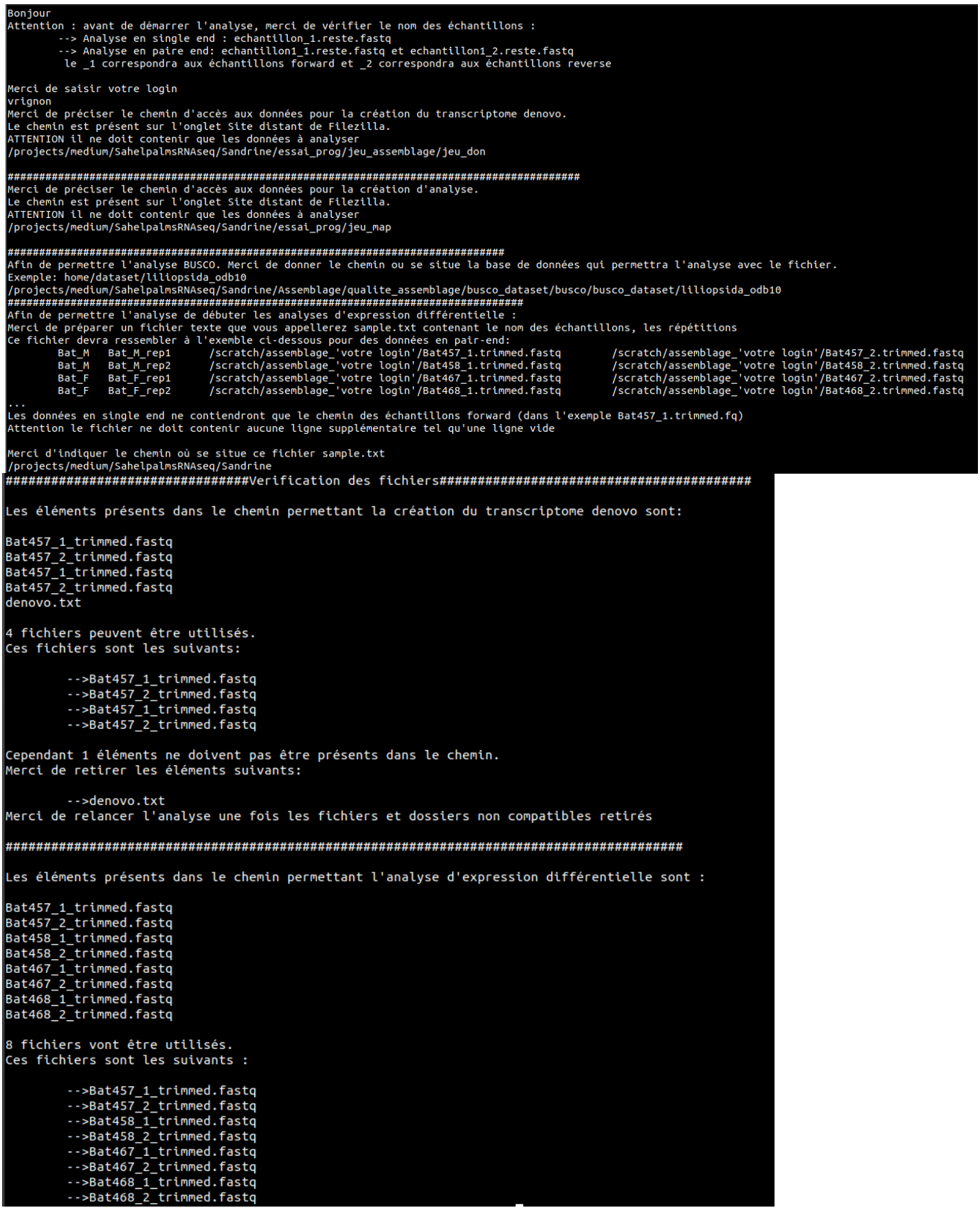


Image 7: Interface utilisateur suite à l'exécution du programme avec erreur de chemin saisi par l'utilisateur

```
Bonjour
Attention : avant de démarrer l'analyse, merci de vérifier le nom des échantillons :
        --> Analyse en single end : echantillon_1.reste.fastq
        --> Analyse en paire end: echantillon1_1.reste.fastq et echantillon1_2.reste.fastq
         le _1 correspondra aux échantillons forward et _2 correspondra aux échantillons reverse

Merci de saisir votre login
vrignon
Merci de préciser le chemin d'accès aux données pour la création du transcriptome denovo.
Le chemin est présent sur l'onglet Site distant de Filezilla.
ATTENTION il ne doit contenir que les données à analyser
/projects/medium/SahelpalmsRNAseq/Sandrine/essai_prog/jeu_assemblage/jeu_don

################################################################################
Merci de préciser le chemin d'accès aux données pour la création d'analyse.
Le chemin est présent sur l'onglet Site distant de Filezilla.
ATTENTION il ne doit contenir que les données à analyser
/projects/medium/SahelpalmsRNAseq/Sandrine/essai_prog/jeu_map

#########################################################################
Afin de permettre l'analyse BUSCO. Merci de donner le chemin ou se situe la base de données qui permettra l'analyse avec le fichier.
Exemple: home/dataset/liliopsida_odb10
/projects/medium/SahelpalmsRNAseq/Sandrine/Assemblage/qualite_assemblage/busco_dataset/busco/busco_dataset/liliopsida_odb10
#########################################################################
Afin de permettre l'analyse de débuter les analyses d'expression différentielle :
Merci de préparer un fichier texte que vous appellerez sample.txt contenant le nom des échantillons, les répétitions
Ce fichier devra ressembler à l'exemble ci-dessous pour des données en pair-end:
        Bat_M   Bat_M_rep1      /scratch/assemblage_'votre login'/Bat457_1.trimmed.fastq        /scratch/assemblage_'votre login'/Bat457_2.trimmed.fastq
        Bat_M   Bat_M_rep2      /scratch/assemblage_'votre login'/Bat458_1.trimmed.fastq        /scratch/assemblage_'votre login'/Bat458_2.trimmed.fastq
        Bat_F   Bat_F_rep1      /scratch/assemblage_'votre login'/Bat467_1.trimmed.fastq        /scratch/assemblage_'votre login'/Bat467_2.trimmed.fastq
        Bat_F   Bat_F_rep2      /scratch/assemblage_'votre login'/Bat468_1.trimmed.fastq        /scratch/assemblage_'votre login'/Bat468_2.trimmed.fastq
...
Les données en single end ne contiendront que le chemin des échantillons forward (dans l'exemple Bat457_1.trimmed.fq)
Attention le fichier ne doit contenir aucune ligne supplémentaire tel qu'une ligne vide

Merci d'indiquer le chemin où se situe ce fichier sample.txt
/projects/medium/SahelpalmsRNAseq/Sandrine
################################Verification des fichiers#########################################

Les éléments présents dans le chemin permettant la création du transcriptome denovo sont:

Bat457_1_trimmed.fastq
Bat457_2_trimmed.fastq

2 fichiers vont être utilisés.
Ces fichiers sont les suivants:

        -->Bat457_1_trimmed.fastq
        -->Bat457_2_trimmed.fastq
```

```
#########################################################################

Les éléments présents dans le chemin permettant l'analyse d'expression différentielle sont :

Bat457_1_trimmed.fastq
Bat457_2_trimmed.fastq
Bat458_1_trimmed.fastq
Bat458_2_trimmed.fastq
Bat467_1_trimmed.fastq
Bat467_2_trimmed.fastq
Bat468_1_trimmed.fastq
Bat468_2_trimmed.fastq

8 fichiers vont être utilisés.
Ces fichiers sont les suivants :

        -->Bat457_1_trimmed.fastq
        -->Bat457_2_trimmed.fastq
        -->Bat458_1_trimmed.fastq
        -->Bat458_2_trimmed.fastq
        -->Bat467_1_trimmed.fastq
        -->Bat467_2_trimmed.fastq
        -->Bat468_1_trimmed.fastq
        -->Bat468_2_trimmed.fastq
Les scripts permettant l'analyse vont être transferer sur le cluster

Merci de re-saisir votre login
vrignon
fastqcmodif.sh                                                                                100% 1043     9.9KB/s   00:00
assemblage_trinitymodif.sh                                                                    100% 1717    13.8KB/s   00:00
analyse_metriquemodif.sh                                                                      100% 1854    10.7KB/s   00:00
mapping_abondancemodif.sh                                                                     100% 2481    14.2KB/s   00:00
matrice_Ex90N50modif.sh                                                                       100% 2342    13.4KB/s   00:00
analyse_diffmodif.sh                                                                          100% 2866    54.0KB/s   00:00
Submitted batch job 210451
Les analyses sont maintenant en cours sur le cluster vous pouvez éteindre votre ordinateur et attendre que les analyses se terminent.
```

Image 8: Interface utilisateur suite à l'exécution totale du programme

```
[vrignon@master1 ~]$ squeue -u vrignon
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
            210452   highmem  trinity  vrignon PD       0:00      1 (Dependency)
            210451   highmem  trinity  vrignon  R       3:28      1 node27
```

Image 9: Contrôle de l'avancée de l'analyse

```
[vrignon@master1 ~]$ cat analyse_metrique.txt
Submitted batch job 209587
Loading samtools/1.9
  Loading requirement: singularity/3.8.0
Loading busco/5.5.0
  Loading requirement: singularity/4.0.1
mkdir: impossible de créer le répertoire « busco »: Le fichier existe
2024-08-03 15:15:39 INFO:       ***** Start a BUSCO v5.5.0 analysis, current time: 08/03/2024 15:15:39 *****
2024-08-03 15:15:39 INFO:       Configuring BUSCO with /opt/config.ini
2024-08-03 15:15:39 INFO:       Mode is transcriptome
2024-08-03 15:15:39 INFO:       Downloading information on latest versions of BUSCO data...
2024-08-03 15:15:41 INFO:       Input file is /scratch/assemblage_vrignon/trinity_assemblage/Trinity.fasta
2024-08-03 15:15:41 INFO:       Using local lineages directory busco/liliopsida_odb10
2024-08-03 15:15:44 INFO:       Running BUSCO using lineage dataset liliopsida_odb10 (eukaryota, 2020-09-10)
2024-08-03 15:15:44 INFO:       Running 1 job(s) on metaeuk, starting at 08/03/2024 15:15:44
2024-08-03 15:46:48 INFO:       [metaeuk]       1 of 1 task(s) completed
2024-08-03 15:48:02 INFO:       ***** Run HMMER on gene sequences *****
2024-08-03 15:48:02 INFO:       Running 3236 job(s) on hmmsearch, starting at 08/03/2024 15:48:02
2024-08-03 15:52:44 INFO:       [hmmsearch]     324 of 3236 task(s) completed
2024-08-03 15:56:33 INFO:       [hmmsearch]     648 of 3236 task(s) completed
2024-08-03 16:01:37 INFO:       [hmmsearch]     971 of 3236 task(s) completed
2024-08-03 16:03:56 INFO:       [hmmsearch]     1295 of 3236 task(s) completed
2024-08-03 16:06:03 INFO:       [hmmsearch]     1619 of 3236 task(s) completed
2024-08-03 16:07:56 INFO:       [hmmsearch]     1942 of 3236 task(s) completed
2024-08-03 16:09:40 INFO:       [hmmsearch]     2266 of 3236 task(s) completed
2024-08-03 16:11:41 INFO:       [hmmsearch]     2589 of 3236 task(s) completed
2024-08-03 16:16:34 INFO:       [hmmsearch]     2913 of 3236 task(s) completed
2024-08-03 16:26:54 INFO:       [hmmsearch]     3236 of 3236 task(s) completed
2024-08-03 16:30:06 INFO:       0 candidate overlapping regions found
2024-08-03 16:30:06 INFO:       21719 exons in total
2024-08-03 16:30:06 INFO:       Extracting missing and fragmented buscos from the file refseq_db.faa...
2024-08-03 16:30:14 INFO:       Running 1 job(s) on metaeuk, starting at 08/03/2024 16:30:14
2024-08-03 17:26:06 INFO:       [metaeuk]       1 of 1 task(s) completed
2024-08-03 17:26:39 INFO:       ***** Run HMMER on gene sequences *****
2024-08-03 17:26:39 INFO:       Running 717 job(s) on hmmsearch, starting at 08/03/2024 17:26:39
2024-08-03 17:28:38 INFO:       [hmmsearch]     72 of 717 task(s) completed
2024-08-03 17:30:13 INFO:       [hmmsearch]     144 of 717 task(s) completed
2024-08-03 17:31:28 INFO:       [hmmsearch]     216 of 717 task(s) completed
2024-08-03 17:32:52 INFO:       [hmmsearch]     287 of 717 task(s) completed
2024-08-03 17:34:12 INFO:       [hmmsearch]     359 of 717 task(s) completed
2024-08-03 17:35:04 INFO:       [hmmsearch]     431 of 717 task(s) completed
2024-08-03 17:35:32 INFO:       [hmmsearch]     502 of 717 task(s) completed
2024-08-03 17:35:53 INFO:       [hmmsearch]     574 of 717 task(s) completed
2024-08-03 17:36:48 INFO:       [hmmsearch]     646 of 717 task(s) completed
2024-08-03 17:39:23 INFO:       [hmmsearch]     717 of 717 task(s) completed
2024-08-03 17:41:28 INFO:       41 candidate overlapping regions found
2024-08-03 17:41:28 INFO:       13359 exons in total
2024-08-03 17:43:07 INFO:       Results:        C:78.2%[S:26.1%,D:52.1%],F:13.4%,M:8.4%,n:3236

2024-08-03 17:43:12 INFO:
```

Image 10: Exemple de consultation du rapport analyse_metrique.txt

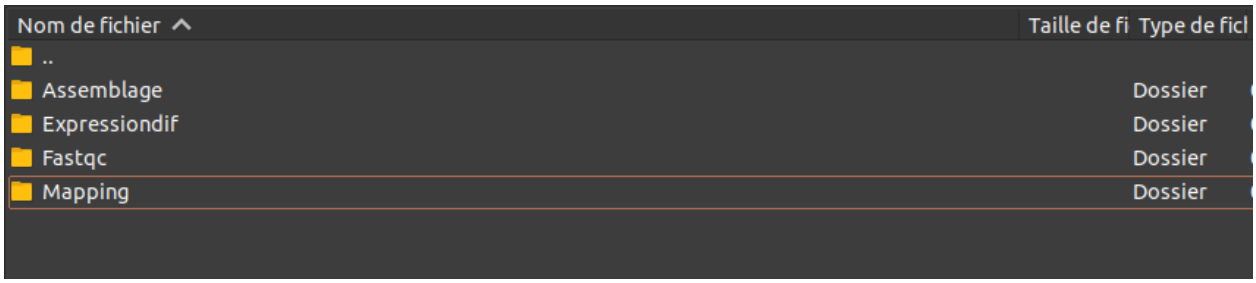


Image 11: Dossier généré dans le cluster suite à l'exécution du programme Analyse-transcriptomique (visualisation effectuée *via* le programme *FileZilla* (version 3.67.0))