
**Master Bioinformatique**

HAU805I - Stage

---

# Rapport de stage : développement, modification et utilisation d'outils pour l'analyse de données transcriptomiques d'espèces sauvages apparentées au blé.

---

*Auteur :*
Florent MARCHAL

*Encadrant pédagogique :*
Anthony BOUREUX

*Encadrants scientifiques :*
**Concetta BURGARELLA**
Nathalie CHANTRET
Vincent RANWEZ

[ 20 août 2024 | Version 1.0.0 ]

# 1 Introduction

Mon stage de master 1 en bioinformatique à la faculté des sciences de Montpellier, s'est déroulé au sein de l'UMR AGAP, équipe Ge2POP (INRAE / Institut Agro) du 2 mai 2024 au 26 juillet 2024. J'ai été encadré par Concetta Burgarella, Vincent Ranwez et Nathalie Chantret.

L'objectif de ce stage est d'évaluer la qualité des données de séquençage issues d'une étude antérieure (section 1.1) pour déterminer si celles-ci peuvent être utilisées pour rechercher des traces de sélection chez des espèces sauvages apparentées au blé. Ce rapport est rédigé à des fins descriptives et à des fins de traçabilité. De ce fait, en plus du contexte et des raisons qui ont motivé mes choix, je préciserai aussi les sources des données utilisées et comment faire pour reproduire les résultats.

## 1.1 Contexte

L'équipe dans laquelle je travaille a publié un article dans lequel a été étudié la diversité génétique des plantes en fonction de leur mode de reproduction [Bur+24]. Les données sur lesquelles ce stage est basé proviennent toutes de cette étude.

### 1.1.1 Modèles biologiques

Nous travaillons ici sur 13 espèces diploïdes sauvages apparentées au blé. Ces espèces appartiennent à la famille des *Poaceae* (Graminées). Le nom et les particularités de ces espèces sont récapitulées dans tableau 1. La phylogénie de ces espèces est, quant à elle, présentée dans la figure 1

Ce choix de modèles permet de travailler sur des espèces génétiquement proches (figure 1) ayant des modes de reproduction divers. En effet, certaines de ces espèces sont autogames, d'autres sont hétérogames et enfin, certaines ont un régime mixte (autogame et hétérogame).

Si, au cours de ce stage, j'ai travaillé sur des données issues de toutes les espèces mentionnées ci-dessus, certaines analyses se concentrent sur *Triticum urartu*. Ce choix est motivé par la présence d'un génome de référence publié (certaines des espèces n'en n'ont pas) et par son mode de reproduction fortement autogame. Ce dernier point permet de s'assurer que les mappings ne seront pas trop longs. En effet, un mode de reproduction autogame réduit la diversité génétique [Bur+24]. A titre indicatif, *Triticum urartu* à un génome diploïde de $4,8 Gpb$ qui correspond à la partie « A » du génome du blé tendre (Froment) qui est hexaploïde (3 génomes : « A », « B » et « C » [Wik24]).

### 1.1.2 Données

Les données sont issues du séquençage du transcriptome des inflorescences, des graines, et des feuilles de parents proches du blé (section 1.1.1). Les analyses bioinformatiques réalisées dans [Bur+24] suivent le pipeline de [Sar+17]. Le choix d'utiliser des données transcriptomiques est motivé par la nécessité d'avoir une meilleure représentation possible de la diversité fonctionnelle à l'échelle du génome. En effet, en 2013, il n'y avait aucun génome de référence pour les espèces considérées (même le blé cultivé n'en avait pas). Les données transcriptomiques ont aussi l'avantage de cibler les parties codantes du génome et donc les zones ayant le plus de chances d'être communes entre les espèces considérées. De fait, cela permet d'étudier les relations phylogénétiques entre les espèces ([Glé+19]) ainsi

| *Espèce* | Mode de reproduction | Nombre d'individus |
|---|---|---|
| *Aegilops bicornis* | Mixte | 3 |
| *Aegilops caudata* | Mixte | 3 |
| *Aegilops comosa* | Mixte | 2 |
| *Aegilops longissima* | Mixte | 4 |
| *Aegilops mutica* | Hétérogame stricte | 7 |
| *Aegilops searsii* | Mixte | 3 |
| *Aegilops sharonensis* | Mixte | 2 |
| *Aegilops speltoides* | Hétérogame stricte | 10 |
| *Aegilops tauschii* | Mixte – fortement autogame | 21 |
| *Aegilops umbellulata* | Mixte | 3 |
| *Aegilops uniaristata* | Mixte | 3 |
| *Triticum monococcum* | Mixte – fortement autogame | 10 |
| *Triticum urartu* | Mixte – fortement autogame | 20 |
| *Hordeum vulgare (outgroup)* | - | - |
| *Hordeum bulobosum (outgroup)* | - | - |
| *Eremopyrum Bonaepartis (outgroup)* | - | - |
| *Secale strictum (outgroup)* | - | - |
| *Secale vavilovi (outgroup)* | - | - |
| *Taeniatherum caputmedusae (outgroup)* | - | - |

TABLE 1 – Tableau énumérant les espèces utilisées, leur mode de reproduction et le nombre d'individus. Le nombre d'individus correspond au nombre de fichiers "*.bam*" (oldBAM) disponibles pour chacune des espèces. Les espèces marquées comme "outgroup" sont celles qui ont servi à ancrer la phylogénie de la figure 1. Les fichiers "*.bam*"de celles-ci n'ont pas été utilisés. **[ Si les modes de reproduction des espèces de l'outgroup sont connus, pouvez-vous me les communiquer pour que je les rajoute au tableau ? De même si le nombre d'individus utilisé est connu. ]**

que les traces de sélection au travers d'une phylogénie.

Dès le début de ce stage, nous avions à notre disposition :

**Les résultats du séquençage de chaque individu** sous la forme de "*.fastq*" déjà prêts à l'emploi. Les reads ont déjà été nettoyés et la longueur de ceux-ci uniformisés à 100 nucléotides par read. Les fichiers trop gros on été séparés en deux parties. La première partie contient toujours 25 000 000 de reads.

**Les transcriptomes de référence (TrEx)** utilisés dans [Glé+19] et construits selon la méthode de [Sar+17]. Ils proviennent de l'assemblage des reads de l'individu le plus couvert. Il en existe un par espèce.

**Les résultats des mappings (oldBAM)** de chaque individu ont été réalisés avec les "*.fastq*" et les TrEx précédemment mentionnés. Ils sont disponibles sur *NCBI SRA* avec l'identifiant : *PRJNA945064*. Les fichiers "*.bam*" qui en résultent seront nommés oldBAM. Vous trouverez quelques informations concernant le contenu de ces "*.bam*" ici : section 3.4.1

**Le génome de référence complet (GeMo)** de *Triticum urartu* provient de la release 59 de ensembl génomes : Triticum_urartu.IGDB.dna.toplevel.fa.gz

**Le transciptome de référence complet de *Triticum urartu*** provient lui aussi de ensembl biomart. Voici comment récupérer celui-ci :

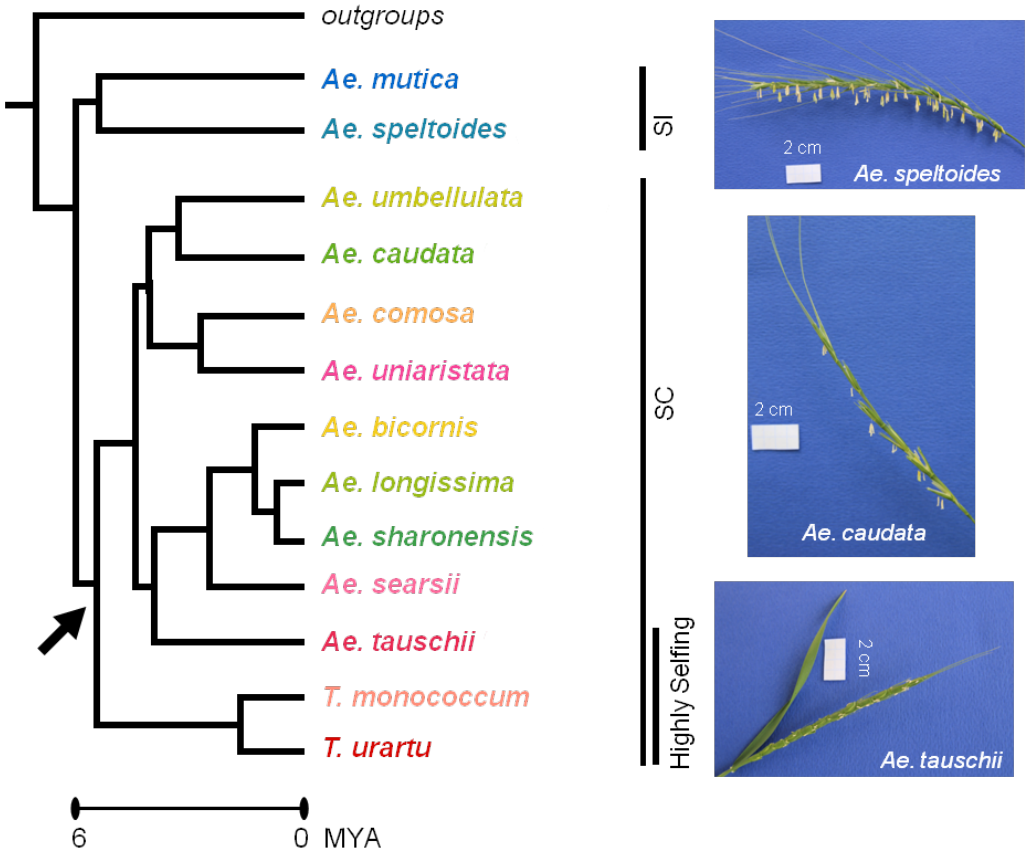



FIGURE 1 – Relation phylogénétique entre les 13 espèces diploïdes du genre *Aegilops* / *Triticum*. Les couleurs représentent un gradiant d'auto-fécondation. Les espèces hétérogames (*SI*) strictes sont bleues, les espèces avec un mode de reproduction mixte (*SC*) sont en vert / jaune et les espèces autogames (*Highly Selfing*) sont en rouge. Cette figure est issue de [Glé+19] et sa légende a été adaptée et traduite par l'auteur de ce rapport.

— Dataset : Triticum urartu genes (Tu2.0)
— Gene type : protein_coding ; Ensembl Canonical : Only
— Attributs : Gene stable ID ; Transcript stable ID ; cDNA sequences ; Upstream flank [200] ; Downstream flank [200]

**Tableau SNP par contigs** contenant pour chaque contig lié à une espèce, le nombre de SNP présent sur les deux allèles. Ils ont été produits grâce à l'outil dNdSpNpS.

## 1.2 Principes de la recherche de traces de sélection

La recherche de traces de sélection, à l'echelle d'une phylogénie se base sur le concept d' « horloge moléculaire ». Cette idée stipule que la diversité accumulée par une espèce, en absence de sélection, est fonction du taux de mutation et du temps de divergence. Ainsi, si l'on assume que le taux de mutation est le même pour toutes les espèces et est constant dans le temps, plus le temps passe, plus les génomes accumulent de mutations. En conséquence, deux espèces « proches » accumulent un nombre similaire de substitutions depuis qu'elles ont divergé.

| Espéce | Individu | Séquence | | | | | |
|---|---|---|---|---|---|---|---|
| **espéce_1** | individu 1 | ATG<br>M | AGA<br>**R** | TGC<br>C | CGA<br>R | CGT<br>**R** | TTT<br>F |
| **espéce_1** | individu 2 | ATG<br>M | AGA<br>**R** | TGC<br>C | CGA<br>R | CGT<br>**R** | TTT<br>F |
| **espéce_1** | individu 3 | ATG<br>M | AGA<br>**R** | TGC<br>C | CGA<br>R | CGT<br>**R** | TTT<br>F |
| **espéce_2** | individu 1 | ATG<br>M | CGT<br>**R** | TGC<br>C | CGA<br>**P** | TGT<br>**C** | TTT<br>F |
| **espéce_2** | individu 2 | ATG<br>M | CGT<br>**R** | TGC<br>C | CCA<br>**R** | TGT<br>**C** | TTT<br>F |
| **espéce_2** | individu 3 | ATG<br>M | CGT<br>**R** | TGC<br>C | CGA<br>**P** | TGT<br>**C** | TTT<br>F |

TABLE 2 – Exemple de séquence portant du polymorphisme. Pour chaque individu, la première séquence correspond aux nucléotides et la seconde aux acides aminés. Le codon n°2 (en bleu) est un exemple de substitution synonyme ; le codon n°4 (en orange) est un exemple de polymorphisme ; le codon n°5 (en rouge) est un exemple de substitution non synonyme

Dans les parties codantes du génome, en l'absence de sélection, on s'attend à ce que les sites synonymes évoluent à la même vitesse que les sites non synonymes. Cependant, en présence de sélection deux cas se distinguent :

**la sélection purificatrice** Elle fonctionne "contre" les mutations non synonymes. De fait, elle tend à conserver la séquence d'acides aminés et, en conséquence, la séquence nucléotidique. Dans la majorité des cas, lorsqu'un site est soumis à une sélection, c'est à celle-ci. Ainsi, l'action de cette sélection réduit le nombre de sites non synonymes.

**la sélection positive** Elle favorise la fixation de mutations dans une espèce. Dans un gène sous sélection positive, on s'attend à trouver plus de substitutions non synonymes que ce à quoi l'on aurait pu s'attendre en regardant les substitution synonymes.

Ces deux types de sélections peuvent s'étudier dans les deux conditions suivantes :

**au sein d'une même espèce** Dans ce cas, les variations de séquences sont appelées sites polymorphe

**au sein d'un groupe d'espèce** Les différences entre séquence sont appelées "substitution" ou "divergence" lorsqu'elles sont fixées dans les espèces. Dans les fait, on considère que ces différences sont toujours fixées dans les espèces.

Des exemples sont consultable dans le tableau 2.

L'étude des sélections se fait en utilisant les indicateurs suivant :

$P_n$ nombre de sites polymorphes non synonymes au sein d'une population. C'est le nombre de mutations entraînant un changement dans la séquence d'acides aminés de la protéine

$P_s$ nombre de sites polymorphes synonymes au sein d'une population. C'est le nombre de mutations n'entraînant aucun changement dans la séquence d'acides aminés de la protéine

$D_n$ nombre de substitutions non synonymes entre deux populations. C'est le nombre de sites différents entraînant un changement dans la séquence d'acides aminés de la protéine

$D_s$ nombre de substitutions synonymes entre deux populations. C'est le nombre de sites différents n'entraînant pas de changements dans la séquence d'acides aminés de la protéine

Ces indicateurs permettent de calculer les deux ratios qui suivent :

$\frac{D_n}{D_s}$ aussi appelé $\omega$ ou $\frac{K_a}{K_s}$, ce ratio représente la balance entre les substitutions synonymes ($D_s$) et non synonymes ($D_n$) entre deux populations. Si $\frac{D_n}{D_s} > 1$ la substitution est probablement conservée par la sélection naturelle. A l'inverse, si $\frac{D_n}{D_s} < 1$ les substituions sont probablement éliminées par la sélection naturelle. La significativité du résultats se calcule en utilisant un test du khi carré

$\frac{P_n}{P_s}$ qui représente la balance entre les mutations synonymes ($P_s$) et non synonymes ($P_n$) dans une unique population

Tout ces éléments sont utilisés par des tests et des logiciels qui seront utilisés dans la section 4

# 2 Quantification des SNP dans les différentes espèces

Comme vu dans l'introduction (section 1.2), les SNP jouent un rôle crucial dans l'analyse des traces de sélection, ceux-ci étant la principale source d'information de nos modèles d'analyse. Ainsi, afin de déterminer si les données en notre possession sont utilisables pour ce type d'analyse, il nous faut trouver un moyen de quantifier la présence des SNP dans nos données. Cela a motivé la création d'un programme ayant pour objectif de permettre la visualisation de la distribution des SNP au sein d'un ensemble de contigs par espèce. Cet outil est en libre accès sur GitHub :

— sur le dépôt contenant les scripts que j'ai développés lors de ce stage : [Flo24a]

— sur le dépôt du projet [Flo24b]. Ma participation à ce dernier s'étend de la fondation du dépôt à la version 1.2.0.

L'objectif de cette partie est d'expliquer le fonctionnement de ce logiciel et de déterminer si le nombre de contigs sur lesquels on peut envisager de travailler est suffisant :

— à l'échelle de chaque espèce individuellement

— le long de la phylogénie (figure 1), auquel cas des contigs orthologues et polymorphes sont requis pour au moins certaines espèces.

## 2.1 Fonctionnement de l'outil

Le but de cet outil est de créer un petit nombre de graphiques pouvant présenter :

— le nombre de contigs présentant un certain nombre de SNP (GNSP=). Ex : environ 4000 contigs contiennent 1 SNP (figure 2).

— le nombre de contigs présentant au moins un certain nombre de SNP (GNSP≥). Ex : environ 12 300 ($= 4000 + 3000 + 2000 + 1000 + 800...$) contigs contiennent au moins 1 SNP (figure 2).

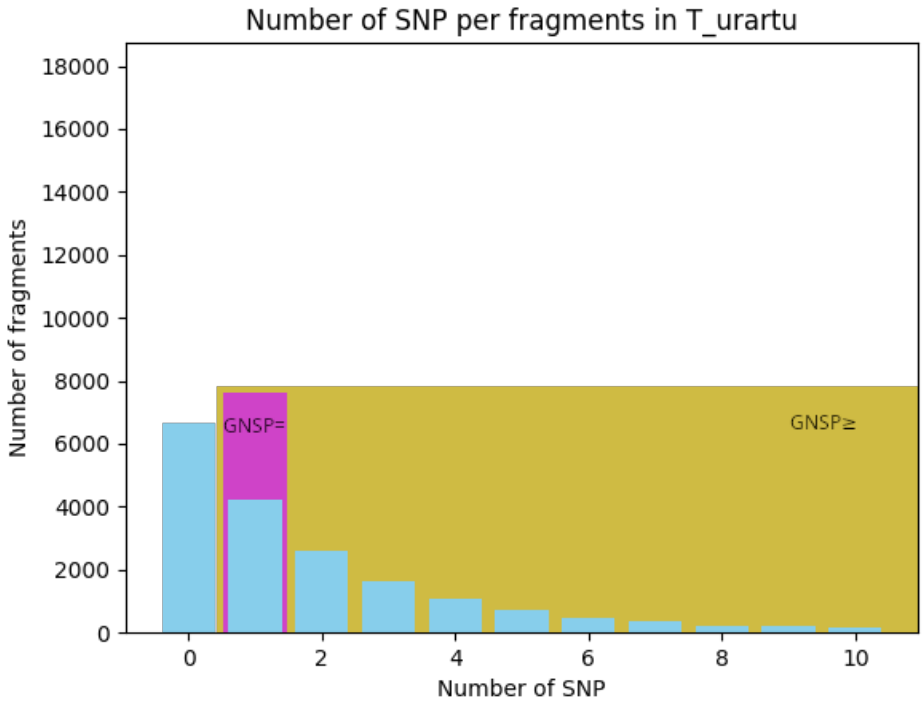



FIGURE 2 – Exemple de sortie de section 2. La zone violette représente la partie prise en compte par GNSP= tandis que la zone jaune représente le GNSP≥. Cette zone s'étend aussi en dehors des limites du graphique. L'abscisse représente le NbSNP et l'ordonnée représente le nombre de contigs présentant ce NbSNP. Cette image a été générée en même temps que la figure 3 et correspond au graphique quantitatif de *Triticum urartu*. La figure a été modifiée en utilisant Gimp.

Si dans le cadre de ce stage, cet outil ne servira qu'à l'analyse des SNP et du nombre de contigs dans un fichier "*.bam*" (figure 5.7, figure 5.8), celui-ci peut être utilisé pour toutes données discrètes distribuées sur un axe discret.

Ce programme utilise plusieurs bibliothèques externes :

- Pytest
- MatPlotLib
- getopt
- sys
- os
- json

### 2.1.1 Données attendues

Les données doivent être contenues dans des fichiers tabulés (exemple : tableau 3). Chaque tableau doit contenir des données à propos d'une espèce. Les lignes du tableau doivent contenir, à minima, une clef primaire (ex : nom du contig) et une valeur numérique (ici le nombre de SNP (NbSNP)). Il est possible de séparer les données relatives à une espèce dans un ou plusieurs fichiers si l'option `--path` est utilisée. Cette option permet de préciser le chemin vers un "*.json*". Si plusieurs fichiers sont utilisés en même temps, les noms des colonnes considérées (colonne "clef primaire" et colonne "valeurs numériques") doivent être identiques. Des fichiers exemples sont présents dans le dossier *tests/data*.

| Contig_name | SNP | Autre |
|---|---|---|
| contig_1 | 1 | def |
| contig_2 | 2 | ghi |
| contig_3 | 3 | jkl |
| contig_4 | 4 | mno |
| contig_5 | 5 | pqr |
| contig_6 | 6 | stu |
| contig_7 | 2 | vwx |
| contig_8 | 3 | yz |
| contig_9 | 4 | abc |
| contig_10 | 5 | def |
| contig_11 | 6 | ghi |

TABLE 3 – Exemple de fichier tabulé pris en charge par [Flo24b]. La colonne "**Contig_name**" correspond à la clef primaire, la colonne "**SNP**" correspond au NbSNP, et la "**Autre**" correspond à d'autres informations présententes de le tableaux. Ces informations là ne sont pas utilisées par [Flo24b]. Ce tableau à été généré avec des données aléatoires et correspond aux premières lignes du fichier `tests/data/test1` de la version 1.2.0 de [Flo24b].

### 2.1.2 Sorties

Pour chaque ensemble d'espèces données (c'est à dire pour chaque groupe de tableaux), au moins un graphique est créé. Les graphiques obtenables sont décrits ci-dessous :

**L'histogramme quantitatif** montre simplement le nombre de gènes en ordonnée et le NbSNP en abscisse. Il est lié à une seule espèce. Il permet de connaître le GNSP=. La figure 2 est un exemple d'histogramme quantitatif. S'active avec l'option `-q`

**L'histogramme cumulatif** montre le nombre de gènes en ordonnée et le NbSNP en abscisse. Lui aussi est lié à une seule espèce. Il permet de connaître le GNSP≥. S'active avec l'option `-c`

**La heatmap cumulative** est une représentation différente des histogrammes cumulatifs. Le nombre de contigs est contenu dans les cellules de la heatmap au lieu d'être présent en ordonnée. Les heatmaps cumulatives présentent le GNSP≥ d'un seul groupe. S'active avec l'option `-u`

**La heatmap globale** est la concaténation de toutes les heatmaps cumulatives. De fait, elle présente l'ensemble des espèces sur une seule figure. La figure 3a et la figure 3b sont des exemples de d'heatmaps globale. S'active avec l'option `-g`

Ces graphiques ont tous, en abscisse ou dans les cellules, le nombre de SNP (NbSNP). Par défaut les légendes diffèrent entre chaque graphique, mais il est possible de les uniformiser via l'option `-y`

Tous les graphiques sont générés par la bibliothèque MatPlotLib et peuvent tous être exportés au format "*.png*", au format "*.svg*", au format "*.tsv*" ou simplement affichés au moment de leur création. Cela correspond respectivement aux options `-k`, `-v`, `-t` et `-d`.

### 2.1.3 Chargement du jeu de données

La fonction `utilities.extract_data_from_table` permet la lecture des fichiers tabulés. Elle se sert du nom des colonnes pour identifier quelles informations utiliser (cf. options `-s` et `-n`). Bien entendu elle ne charge pas tout le contenu du fichier en mémoire. A la place, elle crée un générateur. Ce générateur est utilisé par la fonction `snp_analyser.compile_gene_snp` pour créer un dictionnaire où chaque NbSNP est associé avec le nombre de contigs présentant ce NbSNP. Le nom du fichier d'origine (ou du groupe d'origine) est conservé.

### 2.1.4 Manipulation du jeu de données

Le programme utilise le résultat de l'étape précédente pour créer un dictionnaire intermédiaire via la fonction `snp_analyser.py.compile_gene_snp`. Cet intermédiaire permet de concaténer les dictionnaires créés précédemment tout en conservant l'espèce d'origine. Ce dictionnaire intermédiaire est utilisé par `snp_analyser.py.make_data_matrix` pour créer une matrice de données. Cette matrice est de taille $e * s$ avec $e$ l'espèce et $s$ le nombre de valeurs considérées, c'est-à-dire le nombre de NbSNP affiché. Ce dernier nombre peut être modifié via l'option `-m`. Par défaut, les valeurs considérées sont $[1 : 20]$. Chaque cellule contient le GNSP$\geq$ associé à la valeur.

### 2.1.5 Automatisation

**pytest** Un workflow Pytest a été développé afin de s'assurer du bon fonctionnent de l'outil. Ce workflow correspond peu ou prou au workflow Pytest proposé par GitHub. Des tests unitaires ont été développés pour une grande partie des fonctions mais les sorties ne font l'objet d'aucun test.

**auto-doc** En plus du workflow précédent, j'ai décidé de faire un workflow supplémentaire ayant pour but de générer automatiquement la documentation Doxygen à chaque commit. Ce Workflow est relativement simple, il installe Doxygen et LaTeX dans un environnement Linux avant de se servir du $Doc/Doxyfile$ pour créer la documentation. Ce workflow présente tout de même trois problèmes :

- Il est relativement long (environ 3 à 4 minutes pour être exécuté)
- Il ajoute un commit supplémentaire dans la branche *main* (la branche principale).
- Le "*.pdf*" qui en résulte est relativement laid et difficilement consultable sans être téléchargé.

### 2.1.6 Points d'améliorations

Voici une liste d'améliorations et de critiques pouvant être adressées à l'égard de ce programme :

- Fonctionnement interne :
  - la nécessité d'une clef primaire est un reliquat des toutes premières versions de ce script. Elle pourrait complètement être effacée sans altérer le fonctionnement du logiciel.

  - le calcul de pourcentage se base toujours sur les valeurs présentes dans la matrice. Cela implique que la première colonne de la heatmap ou des histogrammes sera toujours 100%. Aussi, si l'on cherche à calculer le pourcentage de contigs ayant au moins $n$ SNP sans utiliser l'option `--start_at_0`, tous les contigs n'ayant aucun SNP seront exclus du calcul.
  - je me suis aperçu à la fin de mon stage que j'ai construit mon getopt à l'envers, ce qui rend impossible l'usage d'un "pipe" pour fournir des noms de fichiers au programme.
  - seuls les entiers supérieurs ou égaux à un sont acceptés. Le programme pourrait être modifié afin d'accepter des nombres réels.

- Visualisation des données :
  - il n'est pas possible de choisir une "tranche" de vision. En effet, la zone affichée dans les sorties est toujours $[0 : n]$ ou $[1 : n]$ avec une incrémentation de 1. Cela implique qu'il n'est pas possible d'afficher uniquement les données incluses dans un intervalle $[n : m]$ où est $n > 1$.
  - il n'est pas possible de définir une valeur maximale pour l'axe des ordonnées. Cela permettrait de faciliter la comparaison entre plusieurs groupes (exemple : figure 5.7 et figure 5.8) en uniformisant les légendes.
  - les valeurs sont toujours affichées avec un pas de 1, ce qui rend complexe la visualisation de certaine données (figure 5.7, figure 5.8).

- Confort de l'utilisateur :
  - les colonnes ne peuvent pas être identifiées en utilisant une valeur numérique.
  - il n'y a pas de possibilité pour l'utilisateur de changer le style des heatmaps ou des histogrammes. Un fichier de configuration pourrait être rajouté.

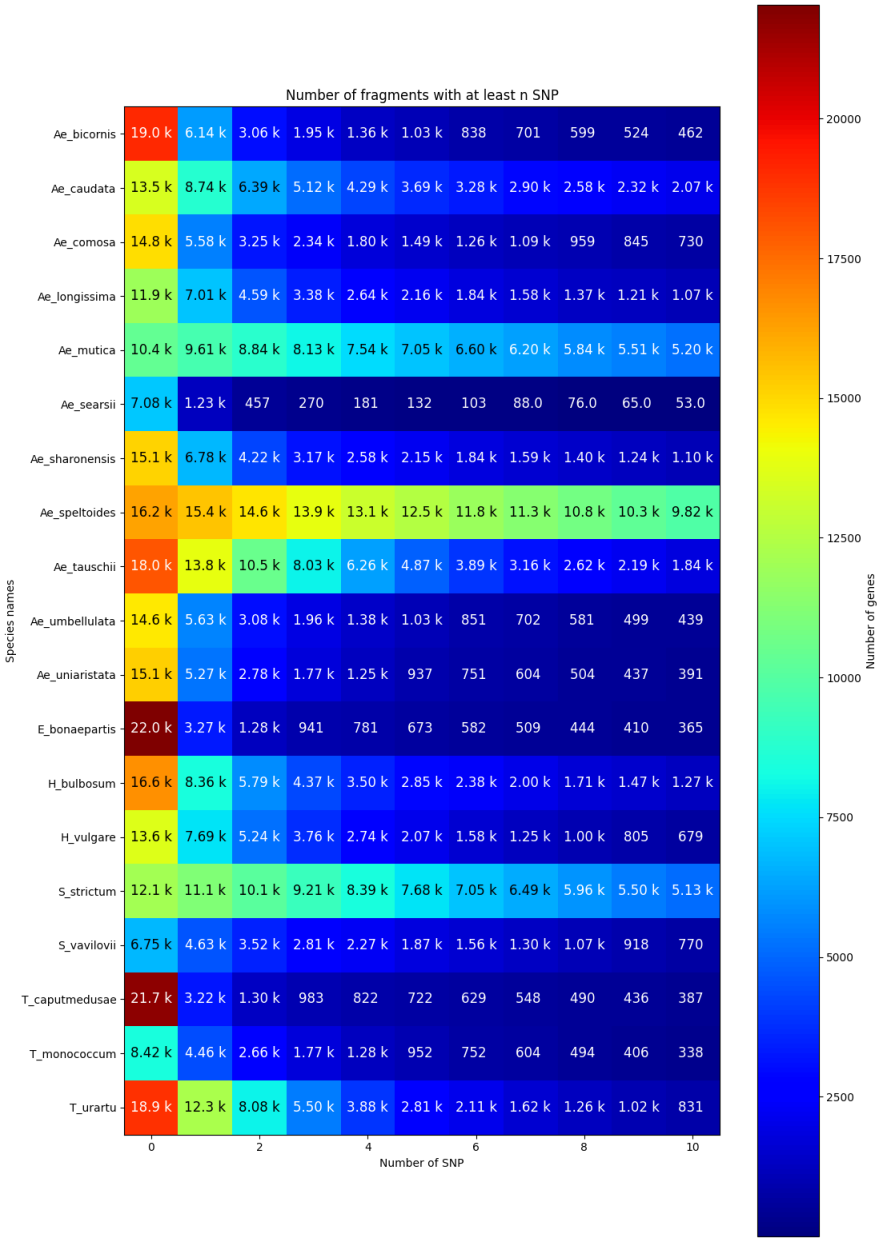
- Maintenance :
  - la documentation de certaines fonctions n'est peut être plus à jour. Aucune révision de la documentation n'a été faite depuis la version 1.0.0 du fait d'un manque de temps.
  - il n'y a aucun test automatisé des sorties du programme.
  - certaines fonctions n'ont pas de tests unitaires ou des tests unitaires incomplets.
  - les images d'exemple du *README* ne sont pas mises à jour automatiquement à chaque commit. Un workflow pourrait être ajouté.

- Reproductibilité et portabilité :
  - le *README* généré à la fin de chaque exécution ne contient pas la ligne de commande utilisé pour démarrer le programme mais le dictionnaire d'argument reçu par la fonction démarrant le programme.
  - il n'existe pas d'environnement Conda ou de fichier *pyproject.toml* pour faciliter l'installation et l'utilisation du logiciel. Les erreurs d'import donnent tout de même lieu à des erreurs personnalisées pour aider l'utilisateur à installer les bibliothèques manquantes.

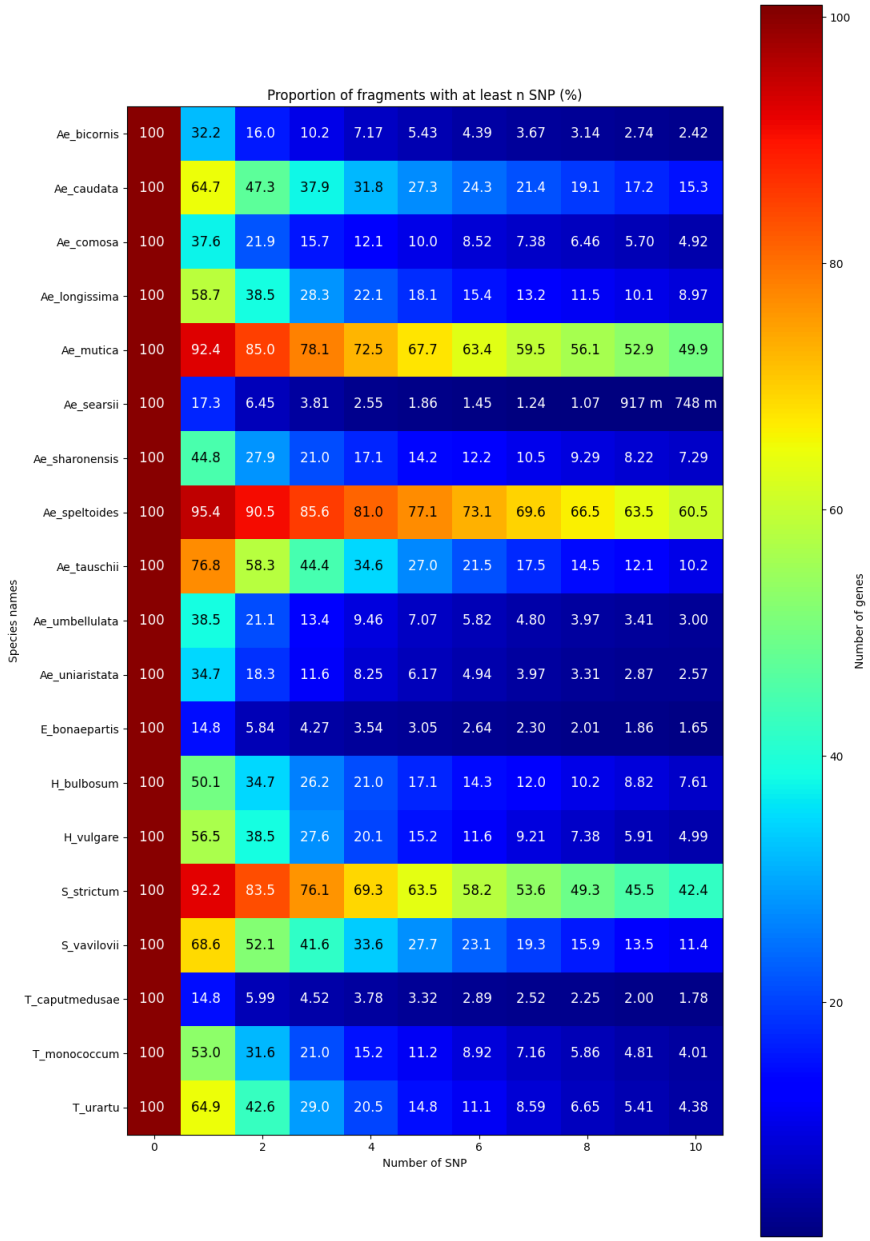
## 2.2 Résultats et conclusion

Pour cette analyse nous avons utilisé les tableaux récapitulant le nombre de SNP par contig mentionné dans 1.1.2 et le logiciel décrit dans cette partie (section 2). Les résultats décrits dans cette partie correspondent à la figure 3.

Dans notre cas, pour pouvoir procéder à une analyse de traces de sélection, on souhaiterait avoir au grand minimum 5 SNP par contigs. On observe sur la figure 3a que le nombre de contigs est très variable entre les différentes espèces étudiées (1). Celui-ci varie de 19 000 (*Ae. bicornis*) à 7080 (*Ae. searsii*). On remarque aussi que le nombre de SNP s'écroule très rapidement. Cette impression est confirmée par la figure 3b. Sur cette seconde figure, on observe que seules 4 des 13 espèces ont au moins 20% de leurs contigs avec un NbSNP supérieur à 5. **[ Quel aurait été le seuil acceptable pour procéder aux analyses (Quel pourcentage de contigs avec un NbSNP>5 permettrait de procéder à des analyses ?) ]**




(a) Nombre de contigs avec au moins $n$ SNP. L'ordonnée indique le nom de l'espèce considérée (1.1.1) et l'abscisse le NbSNP considéré. L'échelle de couleur indique le nombre de contigs contenant ce NbSNP.
Commande : `Contig_name BiAllelic_SNP ./TargetedFiles.json -m 11 -kgqcuwt -j Classic --show_values -1 -y --transparent --legends legends.json --start_at_0`




(b) Pourcentage de contigs ayant au moins $n$ SNP. L'ordonnée indique le nom de l'espèce considérée (1.1.1) et l'abscisse le NbSNP considéré. L'échelle de couleur indique le pourcentage de contigs contenant ce NbSNP.
Commande : `Contig_name BiAllelic_SNP ./TargetedFiles.json -m 11 -kgqcuwt -j Percent --show_values -1 -y --transparent --legends legends.json --start_at_0 --percent`

FIGURE 3 – Résultat de l'analyse des SNP(2). Fait avec le logiciel mentionné dans la 2. La version utilisée est la 1.2.0.

# 3 Mapping

L'analyse du NbSNP réalisée dans la partie précédente a montré que les contigs considérés possèdent peu de SNP. Une explication possible de ce manque de SNP pourrait être que la mapping utilisé pour analyser les contigs ait entraîné trop de perte. En effet, le mapping ayant été réalisé sur des transcriptomes ex-nihilo (TrEx), il est fort possible que ces derniers soient moins complets que le génome de référence « moderne » (GeMo) ou qu'un transcriptome de référence (TrMo) issu de ce dernier. Afin d'évaluer l'effet de la référence sur le mapping, nous avons effectué de nouveaux mappings et comparé leurs résultats respectifs.

Les nouveaux mappings ont été réalisés sur :

— le génome de référence (GeMo)

— le transcriptome de référence (TrMo)

— le transcriptome de référence créé par l'équipe (TrEx)

Ce travail a été effectué avec le support de la Plateforme ISDM-MESO de l'Université de Montpellier financée dans le cadre du CPER par l'État, la Région Occitanie, la Métropole de Montpellier et l'Université de Montpellier.

## 3.1 Le pipeline

GeCKO est le pipeline qui sera utilisé pour les étapes de mapping. Celui-ci a été développée dans l'équipe qui m'a accueilli. Elle est en libre accès sur GitHub [GE224], [Ard+24]. Ce pipeline a pour but de faciliter les analyses phylogénétiques en les rendant plus « user-friendly ». Il se base sur Conda et SnakeMake pour assurer la portabilité. Celui-ci a été choisi car :

— il est connu l'équipe

— les développeurs se trouvent à proximité

— il avait besoin d'être testé en conditions « réelles »

Nous n'avons utilisé que la partie « READ_MAPPING » du pipeline.

### 3.1.1 Installation

L'installation et la mise en route du pipeline s'est révélée être bien plus compliquée que prévue. Celle-ci a nécessite un fort investissement en temps avant résoudre les différents problèmes rencontrés. (figure 5.6). Les sous-sections suivantes dresseront une liste non exhaustive des problèmes rencontrés et des techniques mises en œuvre pour essayer de les résoudre.

La vaste majorité des problèmes rencontrés sont liés à la création de l'environnement global. En effet, pour fonctionner, GeCKO à besoin de créer un environnement Conda. Ce dernier permet la création des environnements SnakeMake dans lesquels les différentes sous parties du programme vont tourner.

**Slurm n'a pas les droits pour écrire dans le dossier home.** Fort heureusement, ce problème a été anticipé par les développeurs du pipeline. Il suffit de déplacer les dossiers dans lesquels le pipeline souhaite écrire (`~/.conda/` et `~/.cashe/`) dans un dossier où le pipeline a le droit d'écrire puis de créer un lien symbolique dans le home pointant vers ces dossiers.

**Conda n'arrive pas à télécharger certains fichiers (`CondaHTTPError`).** Ce problème est d'autant plus étonnant que l'installation en local du pipeline n'a pas rencontré ce problème et que le lien était parfaitement accessible via un navigateur web. Après un long temps de recherche, il s'est avéré que le problème venait du contrôle ssl. La désactivation de ce dernier a permis de résoudre l'erreur (`conda config --set ssl_verify false`)

**Conflit dans les bibliothèques Conda.** Après plusieurs heures d'installation, Conda s'arrêtait avec un message d'erreur très long, environ $740Ko$, mentionnant un problème de compatibilité entre les versions des différents logiciels et bibliothèques installées par Conda.

**Importation de l'environnement.** Une des stratégies envisagées pour résoudre les problèmes mentionnés précédemment fut l'importation des environnements Conda depuis un autre utilisateur ou depuis mon installation en local. Malheureusement les environnements Conda utilisent les chemins complets pour accéder aux dossiers qui les intéressent. Cela implique que ces chemins contiennent les noms d'utilisateur et autres particularités du système de fichier et sont, de fait, difficilement transférables. J'ai tout de même essayé de modifier les chemins, mais cela n'a pas été concluant.

**Création d'un conteneur Singularity** J'ai envisagé de créer un conteneur Singularity pour pouvoir lancer le pipeline sur le cluster. Cependant, n'ayant pas le « privilege root » sur ma machine, je n'ai pas réussi à créer le conteneur.

**Résolution** Au final, la création des différents environnements a été faite par l'une des développeuse du pipeline. Il s'est avéré qu'il manquait une ligne de configuration dans mon "*.bashrc*" me permettant d'accéder à un grand nombre de logiciels installés sur le cluster. Je ne sais pas si elle a résolu d'autres problèmes.

## 3.2 Prétraitement des données

Les données sur lesquelles j'ai travaillé ont nécessité quelques petits prétraitements avant d'être utilisables par le pipeline.

**Changement du mode de compression** Les "*.fastq*" utilisés pour le mapping sont stockés sur le cluster au format "*.bz2*". Cependant, GeCKO n'accepte pas ce type de compression. J'ai donc fait un petit script (figure 4 parties jaunes) me permettant de décompresser les "*.bz2*" et de les recompresser au format "*.gz*". Comme la première version de ce script a créé des fichiers que GeCKO considérait comme « potentiellement corrompus », j'ai créé une nouvelle version qui vérifie que les fichiers ne le sont pas (figure 4 parties jaunes et bleues). Celle-ci ajoute des contrôles à chaque étape pour s'assurer que tout fonctionne. ([Flo24c] ($gz2\_to\_bz2$))

**Tri des fichiers** Les fichiers "*.fastq*" sur lesquels j'ai travaillé sont tous stockés dans le même dossier, indifféremment de leur espèce. Or, je devais travailler sur les *Triticum urartu* uniquement. Comme j'avais à ma disposition un tableau permettant de faire la correspondance entre l'identifiant du fichier et l'espèce, j'ai créé un petit script. Celui-ci charge simplement les noms de fichiers avant de créer un dossier pour déplacer les fichiers dedans. ([Flo24c] ($Sort\_files\_by\_species$))

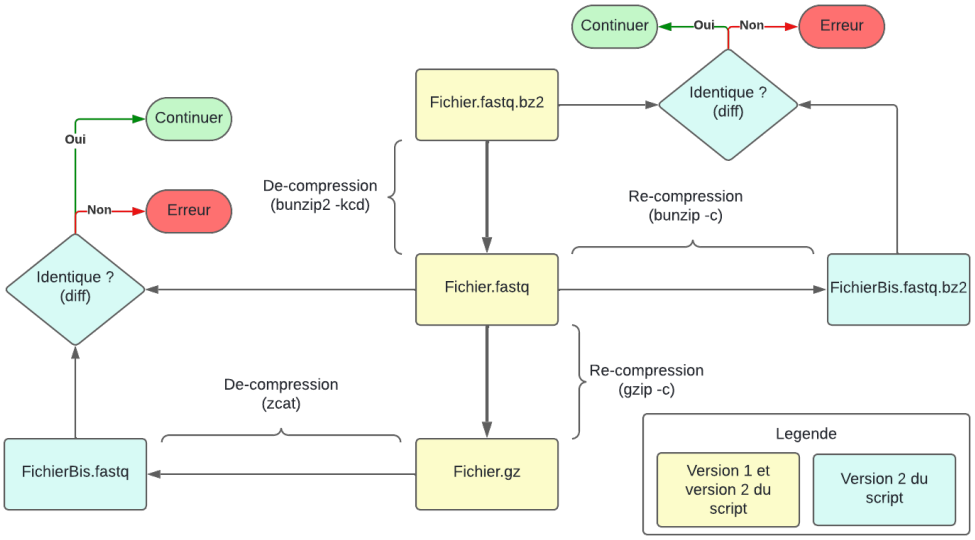



FIGURE 4 – Fonctionnement du programme de changement de compression des "*.fastq*". Les cadres jaunes représentent les étapes communes à la version 1 et à la version 2 tandis que les cadres bleus représentent les étapes spécifiques à la version 2. Le programme est accessible ici : [Flo24c]. Image faite avec Lucidchart.

## 3.3 Déroulement des mapping

Nous aborderons ici les difficultés qui ont eu lieu pendant les mappings ainsi que leurs deroulés.

### 3.3.1 Mapping avec Minimap2 (GeMo)

Le mapper MINIMAP2 a été choisi pour effectuer le mapping des "*.fastq*" sur GeMo. Cependant, ce choix a posé de nombreux problèmes et n'a pas abouti à une solution satisfaisante, malgré un investissement en temps conséquent (figure 5.6). En effet, si nous arrivions à démarrer les mappings, ils se sont sont tous soldés par des erreurs. A titre indicatif voici les types d'erreurs rencontrés sur 17 échantillons :

- 10 `'minimap2: hit.c:210: mm_hit_sort: Assertion ‘has_cigar + no_cigar == 1’failed .'`
- 2 `'minimap2: format.c:380: write_sam_cigar: Assertion ‘clip_len[0] < qlen && clip_len[1] < qlen’failed.'`
- 2 `'Segmentation fault (core dumped)'`
- 3 `'[morecore] insufficient memory'`

Nous avons bien essayé d'augmenter la quantité de mémoire attribuée à Minimap2, mais cela n'a pas permis de résoudre les problèmes (plus de 100*Go* là où 64*Go* était très largement suffisant pour STAR). Nous avons aussi essayé de couper les séquences de chaque chromosome en deux, sans que cela n'aide le mapping.

### 3.3.2 Mapping avec STAR (GeMo)

Comme Minimap2 s'est révélé inexploitable, nous avons choisi d'essayer avec un autre mapper, bien qu'il ne soit pas accessible dans GeCKO. Le mapper choisi a été STAR. Les résultats des mappings ont été obtenus très rapidement et sans grandes difficultés. Cependant, les résultats sont arrivés trop tard (figure 5.6) et n'ont pas put être analysés.

### 3.3.3 Mapping avec BWA (TrEx et TrMo)

Les mappings sur les transcriptions de références ont été fais avec BWA *mem*. Il s'agit du mapper utilisé pour réaliser les oldBAM.

**Mapping sur TrEx** Le re-mapping sur TrEx est justifié par la nécessité d'avoir des données comparables au mapping sur TrMo. En effet, nous ne pouvions pas utiliser les oldBAM pour les raisons détaillées ici : 3.4.1.

## 3.4 Analyse des BAMS

Nous détaillerons ici les différentes étapes de l'analyse des "*.bam*". Afin de faciliter la rédaction de ce rapport, des acronymes seront utilisés pour différencier les "*.bam*" en fonction de leur origine :

- oldBAM : fichiers *.bam* générés avec BWA en utilisant transcriptome de référence créé par l'équipe. (*Transcriptome ex-nihilo)* (TrEx) datant d'avant le début du stage (cf. section 1.1.2)
- BamTrMo : fichiers *.bam* générés avec BWA en utilisant transcriptome de référence construit à partir du génome de référence. (*Transcriptome moderne*) (TrMo)
- BamTrEx : fichiers *.bam* générés avec BWA en utilisant TrEx
- BamGeStar : fichiers *.bam* générés avec STAR en utilisant génome de référence. (*Génome Moderne*) (GeMo)

### 3.4.1 Un mot sur les anciens bams (oldBAM)

Comme mentionné dans la section 1.1.2, les oldBAM proviennent d'une étude antérieure. Ces "*.bam*" sont déjà énormément filtrés ce qui implique qu'une partie non négligeable des reads ont été perdus. Cela se voit notamment le tableau 7. Ces "*.bam*" seront tout de même utilisés à titre indicatif dans certaines des analyses.

### 3.4.2 Comparaison des tailles (BamTrEx, BamTrMo)

Une première analyse, très simple a consisté à regarder le tailles global de chaque "*.bam*". Comme les reads font tous la même taille un "*.bam*" plus lourd a tendance à contenir plus de reads.

Dans notre situation, on remarque que les "*.bam*" issus de TrMo sont légèrement plus lourds que ceux issus de TrEx (tableau 4). La différence est généralement comprise entre 1 % et 2 %

| Nom du fichier | TrEx (Mo) | TrMo (Mo) |
|---|---|---|
| Tr207_TAGCTT_L002.bam | 1 866 | 1 903 |
| Tr232_TTAGGC_L003.bam | 2 394 | 2 436 |
| Tr235_TGACCA_L003_00.bam | 1 895 | 1 928 |
| Tr235_TGACCA_L003_01.bam | 624 | 633 |
| Tr245_GATCAG_L003_00.bam | 1 888 | 1 921 |
| Tr245_GATCAG_L003_01.bam | 1 245 | 1 262 |
| Tr246_TAGCTT_L003_00.bam | 1 876 | 1 919 |
| Tr246_TAGCTT_L003_01.bam | 1 605 | 1 635 |
| Tr304_CTTGTA_L004_00.bam | 1 850 | 1 885 |
| Tr304_CTTGTA_L004_01.bam | 1 149 | 1 167 |
| Tr306_ATCACG_L005.bam | 2 152 | 2 189 |
| Tr307_CGATGT_L005.bam | 2 348 | 2 392 |
| Tr309_TTAGGC_L005_00.bam | 1 883 | 1 917 |
| Tr309_TTAGGC_L005_01.bam | 620 | 630 |
| Tr312_TGACCA_L005.bam | 2 011 | 2 049 |
| Tr315_ACAGTG_L005.bam | 1 927 | 1 965 |
| Tr317_GCCAAT_L005.bam | 2 145 | 2 182 |
| Tr318_CAGATC_L005.bam | 1 874 | 1 909 |
| Tr347_CGATGT_L006.bam | 1 843 | 1 879 |
| Tr348_TTAGGC_L006.bam | 2 083 | 2 119 |
| Tr349_TGACCA_L006.bam | 2 220 | 2 259 |

TABLE 4 – Taille des fichiers issus des différents mappings. Chaque taille est comptée en méga octet (Mo) et est arrondie au plus proche. Certains individus sont représentés par deux fichiers du fait des spécificités des "$.fastq$" (Cf. section 1.1.2)

### 3.4.3 Comparaison des FastQcReports (BamTrEx, BamTrMo)

Quand un mapping est lancé avec GeCKO, GeCKO crée de lui même un rapport fastqc(tableau 5, tableau 6). Analysons celui-ci.

**Les taux d'erreurs** sont généralement compris entre 0.5% et 1% pour les deux mappings. Cependant, les taux d'erreurs des BamTrMo sont systématiquement plus élevés que ceux des BamTrEx (0.18 % de plus en moyenne pour chaque).

**Le pourcentage de reads mappés** est compris entre 80 % et 85 % pour TrMo tandis qu'il est compris entre 90 % et 96 % pour TrEx.

**Le pourcentage de brins bien appariés** est compris entre 78 % et 82 % pour TrMo tandis qu'il est compris entre 81 % et 86 % pour TrEx.

Contrairement à nos attententes initiales, il semblerait que TrEx soi plus intéressant pour procéder à nos mappings que TrMo , TrEx maximisant le nombre de reads mappés.

| Nom | Taux d'erreurs | reads mappés (%) | reads appariés (%) |
|---|---|---|---|
| Tr206_GATCAG_L002 | 0.61 | 94.5 | 85.7 |
| Tr207_TAGCTT_L002 | 0.65 | 94.0 | 85.7 |
| Tr232_TTAGGC_L003 | 0.72 | 91.5 | 82.6 |
| Tr306_ATCACG_L005 | 0.61 | 93.9 | 84.6 |
| Tr307_CGATGT_L005 | 0.61 | 93.9 | 85.2 |
| Tr312_TGACCA_L005 | 0.59 | 91.0 | 82.4 |
| Tr315_ACAGTG_L005 | 0.60 | 93.3 | 84.7 |
| Tr317_GCCAAT_L005 | 0.65 | 93.8 | 84.8 |
| Tr318_CAGATC_L005 | 0.65 | 93.3 | 84.4 |
| Tr347_CGATGT_L006 | 0.62 | 92.2 | 83.2 |
| Tr348_TTAGGC_L006 | 0.65 | 90.1 | 81.4 |
| Tr349_TGACCA_L006 | 0.64 | 92.1 | 82.8 |
| Tr235_TGACCA_L003_00 | 0.68 | 91.9 | 82.7 |
| Tr235_TGACCA_L003_01 | 0.93 | 91.9 | 82.7 |
| Tr245_GATCAG_L003_00 | 0.64 | 94.7 | 85.8 |
| Tr245_GATCAG_L003_01 | 0.93 | 94.6 | 85.7 |
| Tr246_TAGCTT_L003_00 | 0.45 | 95.6 | 86.8 |
| Tr246_TAGCTT_L003_01 | 0.74 | 95.5 | 86.7 |
| Tr304_CTTGTA_L004_00 | 0.60 | 93.5 | 85.5 |
| Tr304_CTTGTA_L004_01 | 0.93 | 93.4 | 85.5 |
| Tr309_TTAGGC_L005_00 | 0.63 | 92.9 | 84.2 |
| Tr309_TTAGGC_L005_01 | 0.72 | 92.9 | 84.2 |
| Moyenne | 0.68 | 93.2 | 84.42 |
| Écart type | 0.12 | 1.42 | 1.51 |

TABLE 5 – Statistiques du rapport fastqc des BamTrEx

#### 3.4.4 Nombre de reads mappés par rapport au nombre de contigs (BamTrEx, BamTrMo)

Afin de confirmer ou d'infirmer les bons résultats de TrEx dans la partie précédente (section 3.4.3), nous avons décidé de vérifier que le fort pourcentage de reads mappés n'était pas causé par des reads dont la qualité de la correspondance entre eux et la référence est faible. Pour cela, nous avons divisé le nombre de reads présents dans chaque "*.bam*" par le nombre de reads présents dans le "*.fastq*" correspondant. Pour qu'un read contenu dans un "*.bam*" soi pris en compte, la qualité de la correspondance doit être supérieure ou égale à 30. Cette valeur a été choisies car elle est généralement utilisées comme valeur minimale pour séparer les bonnes correspondances des correspondances insuffisantes.

Dans la tableau 7, on remarque que les BamTrEx sont toujours meilleurs que les BamTrMo. Cela confirme que les résultats (section 3.4.3) ne sont pas causés par des reads ayant une qualité de correspondance faible.

#### 3.4.5 Nombre de reads mappés par contigs (BamTrEx, BamTrMo)

Afin de savoir si les bons résultats de TrEx (section 3.4.3) ne sont pas causés par des contigs sur-représentés, c'est à dire des contigs sur lesquels une proportion trop grande de reads sont mis en correspondance, il a été décidé de faire une analyse supplémentaire.

| Nom | Taux d'erreurs | reads mappés (%) | reads appariés (%) |
|---|---|---|---|
| Tr206_GATCAG_L002 | 0.78 | 84.7 | 80.2 |
| Tr207_TAGCTT_L002 | 0.82 | 84.4 | 79.8 |
| Tr232_TTAGGC_L003 | 0.89 | 82.1 | 77.8 |
| Tr306_ATCACG_L005 | 0.75 | 84.3 | 79.3 |
| Tr307_CGATGT_L005 | 0.79 | 84.3 | 79.4 |
| Tr312_TGACCA_L005 | 0.77 | 81.3 | 76.3 |
| Tr315_ACAGTG_L005 | 0.79 | 83.4 | 78.4 |
| Tr317_GCCAAT_L005 | 0.82 | 84.1 | 79.1 |
| Tr318_CAGATC_L005 | 0.82 | 84.0 | 79.2 |
| Tr347_CGATGT_L006 | 0.79 | 82.9 | 77.7 |
| Tr348_TTAGGC_L006 | 0.80 | 82.0 | 77.0 |
| Tr349_TGACCA_L006 | 0.79 | 83.2 | 78.4 |
| Tr235_TGACCA_L003_00 | 0.87 | 82.5 | 78.1 |
| Tr235_TGACCA_L003_01 | 1.11 | 82.3 | 78.0 |
| Tr245_GATCAG_L003_00 | 0.82 | 85.4 | 81.4 |
| Tr245_GATCAG_L003_01 | 1.10 | 85.2 | 81.2 |
| Tr246_TAGCTT_L003_00 | 0.77 | 84.9 | 80.8 |
| Tr246_TAGCTT_L003_01 | 1.05 | 84.7 | 80.5 |
| Tr304_CTTGTA_L004_00 | 0.79 | 83.7 | 79.5 |
| Tr304_CTTGTA_L004_01 | 1.11 | 83.6 | 79.4 |
| Tr309_TTAGGC_L005_00 | 0.81 | 83.4 | 78.4 |
| Tr309_TTAGGC_L005_01 | 0.90 | 83.4 | 78.4 |
| Moyenne | 0.86 | 83.63 | 79.01 |
| Écart type | 0.12 | 1.11 | 1.32 |

TABLE 6 – Statistiques du rapport fastqc des BamTrMo

Celle-ci se base sur le comptage du nombre de reads mappés sur chaque contigs ([Flo24c] (*Read_per_contig*) ). Pour chaque référence, le script utilisé parcourt les contigs contenus dans la référence. Pour chaque contig, les "*.bam*" correspondant à la référence sont ouverts et le nombre d'occurrences de ce contig est compté (`samtools view -c $BAM_FILE $CONTIG_NAME`). Le résultat de ce comptage est inscrit dans un "*.tsv*". Les valeurs du "*.tsv*" sont reformatées ([Flo24c] `Refomat_bam-contig_read`). Cela permet de séparer chaque "*.bam*" dans des fichiers différents. Au passage, le nombre de reads est divisé par 10. Cela permet de visualiser les résultats avec le logiciel présenté dans la section 2 (figure 5.7, figure 5.8).

Une autre étape de pré-traitement a permis de générer la figure 5 ([Flo24c] `Boxplot`).

La figure 5.7 nous montre que le nombre de reads par contigs est plus élevé dans les BamTrEx que dans les BamTrMo, et ce, malgré le fait que TrMo possèdent un nombre de contigs largement plus élevé que TrEx. Dans le même temps, la figure 5 nous montre clairement que, en moyenne, les contig dans BamTrEx possèdent largement plus de reads que BamTrMo (741,35 contre seulement 620,02). L'écart type entre les deux conditions est tout a fait similaire ($2495,10 \approx 2481.73$).

| Individu | BamTrEx | BamTrMo | oldBAM |
|---|---|---|---|
| Tr206 | 92.98 | 77.07 | 66.72 |
| Tr207 | 89.27 | 73.36 | 65.43 |
| Tr232 | 90.38 | 75.27 | 64.90 |
| Tr235 | 92.27 | 76.03 | 67.14 |
| Tr245 | 89.65 | 74.00 | 62.33 |
| Tr246 | 94.03 | 76.57 | 68.89 |
| Tr304 | 91.79 | 75.45 | 68.27 |
| Tr306 | 92.76 | 76.42 | 70.38 |
| Tr307 | 91.76 | 75.83 | 66.11 |
| Tr309 | 92.43 | 76.33 | 70.32 |
| Tr312 | 90.19 | 74.55 | 64.53 |
| Tr315 | 92.37 | 76.44 | 66.43 |
| Tr317 | 92.22 | 76.07 | 66.50 |
| Tr318 | 90.34 | 74.64 | 67.13 |
| Tr347 | 91.75 | 75.22 | 67.53 |
| Tr348 | 91.30 | 75.39 | 66.67 |
| Tr349 | 88.44 | 74.26 | 64.00 |

TABLE 7 – Tableau indiquant le pourcentage de reads se trouvant dans le(s) "*.fastq*" correspondant à un individu se trouvant aussi dans les "*.bam*" de cet individu avec une qualité de correspondance supérieure ou égale à 30. Le nombre de reads a été extrait des "*.fastq*" en utilisant glsSeqkit (`seqkit stats`). Le nombre de reads a été extrait des "*.bam*" avec Samtools (`samtools stats -c -F 4 -q 30`).

### 3.5 Conclusion

Au vu des analyses réalisées précédemment, il est claire que le mapping des "*.fastq*" sur TrEx est meilleur que le mapping des "*.fastq*" sur TrMo :

— le nombre de reads par contigs est plus élevé

— le nombre de contigs ayant recu des reads est plus élevé

— la qualité du mapping est plus élevée

Ces résultats vont à l'encontre de notre intuition initiale mais restent incomplets puisque que les BamGeStar n'ont pas été analysés (section 3.3.2).

## 4 Pipeline pour la recherche de traces de sélections

La création de cette pipeline a débuté pendant le mapping mentionné dans la section 3.3.1 (cf. figure 5.6). Trop peu de temps a été consacré à cette pipeline pour qu'elle soit dans un état utilisable. Cependant, le terrain a bien été débroussaillé. Cette section a pour but de présenter les concepts théoriques et les avancements de la pipeline.

### 4.1 Objectif et contexte

L'objectif de cette pipeline est de détecter automatiquement des traces de sélection sur des gènes. Pour cela la pipeline doit pouvoir accepter deux types de sources :

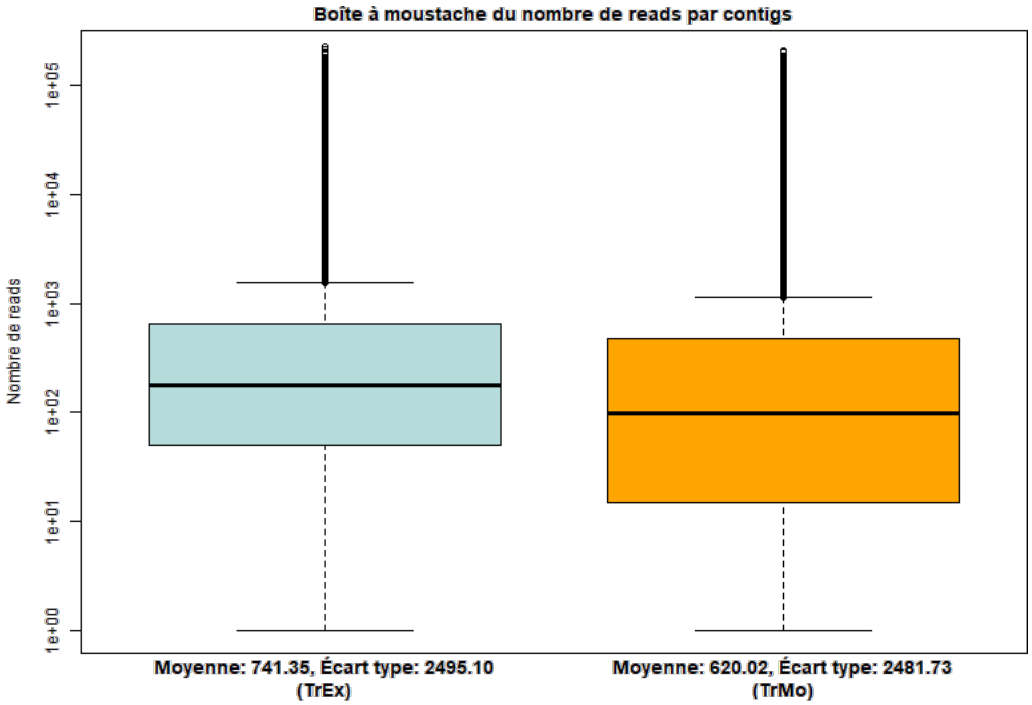



FIGURE 5 – Boite à moustache du nombre de reads par contigs. Les données relatives à BamTrEx sont a gauche tandis que les données relatives a BamTrMo sont à droite. Seuls les contigs ayant au moins 1 reads ont été pris en compte. Figure générée sur R puis modifiée avec Gimp pour fusionner les deux boites à moustaches sur le même graphique.

**alignements de gènes orthologues** Ces alignements doivent être réalisés au niveau des codons et non des nucléotides, la sélection se faisant principalement au niveau de la protéine. Lorsque ce type de fichier est fournis, une analyse utilisant CODEML doit être réalisée.

**fichier de statistiques** Ces fichiers tabulés doivent contenir des informations concernant le $P_n$, le $P_s$, le $D_n$, et le $D_s$. Dans ce type de situation un test de McDonald–Kreitman est réalisé.

Dans le cadre de ce stage, le but est d'identifier des traces de sélection sur les gènes orthologues identifiés chez les différents individus mentionnés dans la figure 1.

## 4.2 Fonctionnement théorique

Le Fonctionnement théorique du pipeline est le suivant :

— Analyse avec CODEML / EggLib.

1. Chargement du fichier d'alignement (`egglib.io.from_fasta`)
2. Chargement de l'arbre ayant permis la création de l'alignement (`egglib.Tree`)
3. Déclenchement de CODEML sur les modèles $M1a$, $M2a$, $M7$ et $M8$ (`egglib.wrappers.codeml`).
4. Test du khi carré sur les résultats : $M1a$ contre $M2a$ et $M7$ contre $M8$.
5. Si un des tests est significatif, le résultat doit être écrit dans un fichier de sortie.

— Analyses avec test de McDonald–Kreitman.

1. Chargement du fichier contenant les statistiques
2. Pour chaque fichier : test de Fisher pour identifier si $\frac{D_n}{D_s}$ est significativement supérieur à $\frac{P_n}{P_s}$
3. Si le test est significatif, écriture du résultat dans un fichier de sortie.

Dans l'idéal pour chaque fichier traité et pour chaque modèle, le pipeline doit utiliseé du multiprocessing, du thtreading ou Slurm.

## 4.3 Les outils

Présentons succinctement les logiciels utilisés.

### 4.3.1 CODEML

CODEML est un outil disponible dans PAML (Phylogenetic Analysis by Maximum Likelihood), un ensemble d'outil permettant de faire des analyses relatives aux phylogénies. CODEML permet d'analyser les pressions sélectives agissant sur des séquences codantes. Dans notre cas, la "sélection positive" est définie par la présence de codons où $\omega > 1$ [ÁKY23]. Pour cela, plusieurs modèles sont disponibles. Dans le cadre de ce stage nous ne nous intéresserons qu'aux modèles "sites" :

— $M1a$ Modèle "quasiment neutre". [ÁKY23]

— $M2a$ Modèle "sélection positive". [ÁKY23]

— $M7$ suit une distribution Beta, ce modèle ne permet pas d'attribuer une sélection positive à un site. [Yan+00]

— $M8$ modèle similaire à $M7$, ce modèle ajoute un paramètre permettant d'attribuer une sélection positive à un site. La comparaison des résultats de ce modèle avec $M7$ permet de détecter les sites subissant une sélection positive [Yan+00]

### 4.3.2 EggLib

EggLib est un package Python proposant des objets permettant la manipulation d'objets biologiques (alignements, arbres phylogénétiques, séquences ...). En plus de ces objets, EggLib propose des wrappers pour différents logiciels. Nous utiliserons principalement EggLib pour son wrapper CODEML. Il permet, en théorie, d'exécuter CODEML, de lire automatiquement le fichier de sortie et de charger les informations de celui-ci dans un dictionnaire Python. Cela facilite grandement l'exploitation des résultats de CODEML.

Afin de me familiariser avec CODEML, j'ai suivi un "Beginner's Guide" ([ÁKY23]). Celui-explique les différents types de tests réalisables avec cet outil et propose des données permettant de reproduire les exemples utilisés dans le guide. Celles-ci se trouvent sur un dépôt GitHub[pam].

La reproduction des résultats s'est faite sans accrocs. Seule la partie $02\_branch\_models$ a été utilisée car elle contient les tests que nous souhaitons effectuer.

## 4.4 Familiarisation avec EggLib

Pour me familiarisé avec EggLib, et plus particulièrement avec le wrapper de CODEML, j'ai tenté de reproduire les manipulations faites avec CODEML (section 4.3.2) dans un script Python.

### 4.4.1 Problèmes avec EggLib

Lors de ma reproduction des résultats, j'ai découvert deux problème majeurs.

**Gestion des séquences** Lorsque la première séquence de l'alignement contient des délétions ("− − −"), la fonction `egglib.wrappers._codeml._helper_rst` n'arrive pas à extraire les données. Celui-ci considére la délétion comme des caractères invalides.

**Arguments du wrapper** L'argument "codon_freq" été limité à une valeur de 3 tandis que CODEML peut accepter plus de valeur. A titre indicatif, [ÁKY23] utilise une valeur de 7.

Ces deux problèmes ont été rapportés au développeur et ont été résolus avec le passage à la version 3.3.3.

**REGEX incomplète** La REGEX utilisée pour exploiter le fichier de sortie de CODEML ne fonctionne pas correctement :

1. la présence de délétions ("− − −") dans l'alignement casse la REGEX (le symbole "-" n'est pas reconnu par celle-ci)
2. la "log-likelihood" qui est requise par certains tests n'est pas récupérée par le wrapper
3. la valeurs de $P(w) > 1$ associé aux sites potentiellement sélectionnés ne sont pas récupérés.

J'ai tenté de corriger la *REGEX* (cf. [Flo24c] *Regex*) mais je ne suis pas arrivé au bout de la correction du point n°3.

**[ Je souhaite envoyer les modifications que j'ai apporté à la REGEX à Stéphane De Mita, pensez-vous que cela soit utile / nécessaire ? ]**

## 4.5 Conclusion

Si ce stage n'a pas permis de construire cette pipeline, il aura permis de mettre en évidence des problèmes liée à EggLib. L'identification de ces problèmes et la construction théorique de la pipeline devrait permettre dans l'avenir de reprendre la construction de la pipeline.

# 5 Conclusion générale

Le jeu de données étudié risque fortement de ne pas convenir pour réaliser des recherches des traces de sélection chez des espèces apparentées au blé. En effet, comme vu dans la section 2.2, le nombre de SNP par contigs est trop faible. De plus, les re-mappings (section 3.5) montrent que la référence (TrEx) utilisée pour identifier les SNP est la meilleure.

On notera tout de même qu'il existe deux points à poursuivre pour avoir une idée plus certaine des résultats :

— les "*.bam*" issus des mappings sur GeMo n'ont pas été analysé (section 3.3.2). Il est tout a fait possible que ce mappings soit meilleur et plus complet que les précédents.

— les tableaux présentant le nombre de SNP par contigs (cf. (section 1.1.2)) n'ont été fait qu'a partir des oldBAM. Si les tableaux on été générés après le filtrage drastique qu'ils ont subi, le nombre de SNPpeu avoir été grandement affecté. Refaire ces tableaux avec les nouveaux fichiers (BamGeStar, BamTrEx, BamTrMo) pourrait permettre de grandement augmenter le nombre de SNPtrouvé.

## 5.1 Apprentissage personnel

Enfin ce stage à été l'occasion pour moi de développer de nouvelles compétences et d'apprendre à me servir de nouveaux logiciels. La liste ci-dessous fourni une liste non exhaustive des outils que j'ai manipulés et des compétences que j'ai acquises ou commencé à acquérir.

| *Logiciel* | Example |
|---|---|
| Python | section 2 |
| Doxygen | section 2.1.5 |
| LaTeX | ce rapport, cf [Flo24c] |
| CODEML | section 4.3.1 |
| EggLib | section 4.3.2 |
| Slurm | section 3 |
| GeCKO | section 3 |
| Bash | [Flo24c] |
| R | figure 5 |
| Conda | - |
| GitHub | section 2 |
| *Bonne pratique* | Example |
| Documentation des outils | section 2.1.5 |
| Création de tests unitaires | section 2.1.5 |
| Gestion de la mémoire | Slurm, section 2 |
| Usage de Git | [Flo24c], [Flo24d] |
| Assurer une traçabilité | Usage de Git, ce rapport |
| Tenue d'un journal de bord | - |
| Assurer la reproductibilité des résultats | Légende des figures |
| Parallélisation des tâches quand possible | Usage de Slurm, section 4.2 |
| Organisation personnelle | du travail et de l'espace de travail |
| *Apprentissage* | Example |
| Usage d'un cluster | section 3 |
| Usage d'environnements WSL | - |
| Création d'environnements Conda | - |
| Connaissances sur l'usage des mappers | section 3 |
| Connaissances sur la recherche de traces de sélection | - |
| Connaissances sur les "*.bam*" | - |

TABLE 8 – Récapitulatif non exhaustif, des logiciels utilisés, bonnes pratiques acquises et autres apprentissages

# Annexes

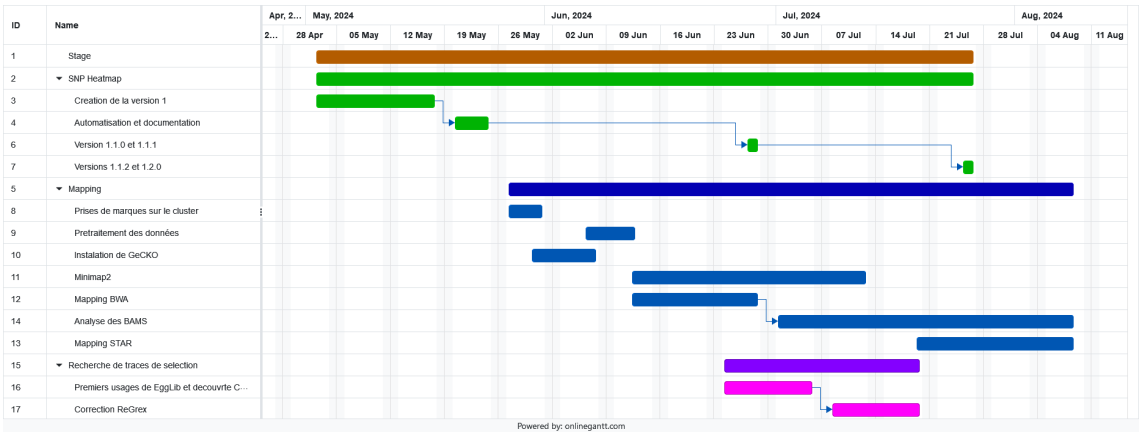
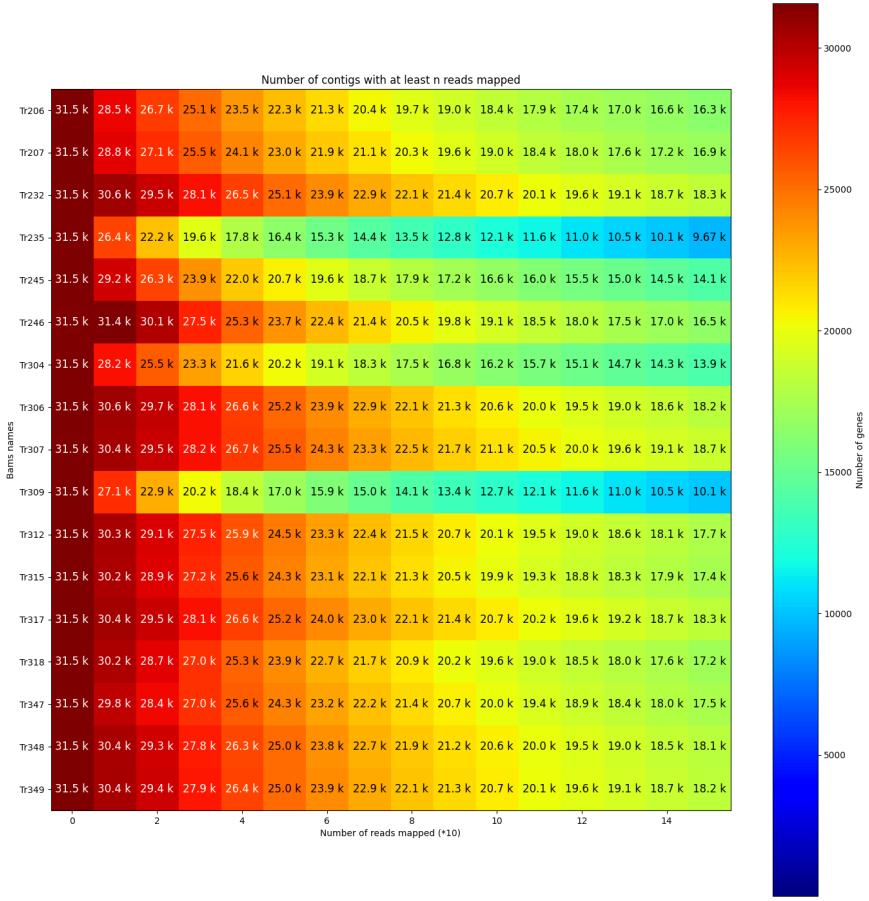
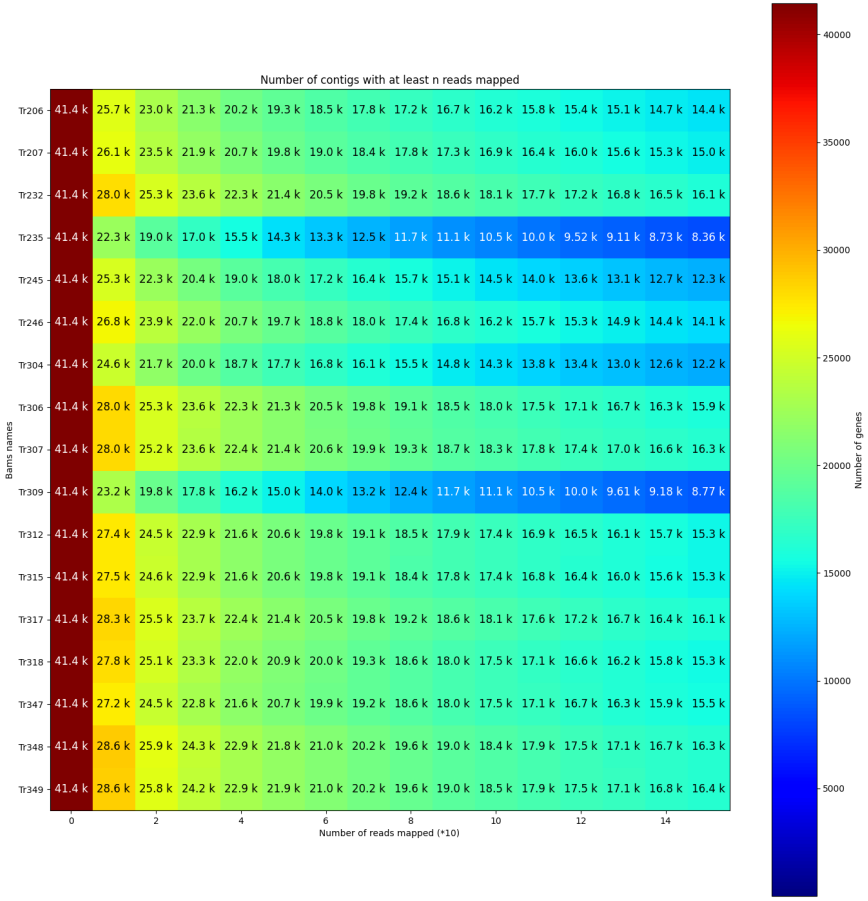



FIGURE 5.6 – Diagramme de gantt illustrant l'usage du temps pendant le stage. Les jours fériés et jours ne sont pas affichés et les dates de début et de fin peuvent être légèrement décalée par rapport à la réalité. Figure générée avec OnlineGant.

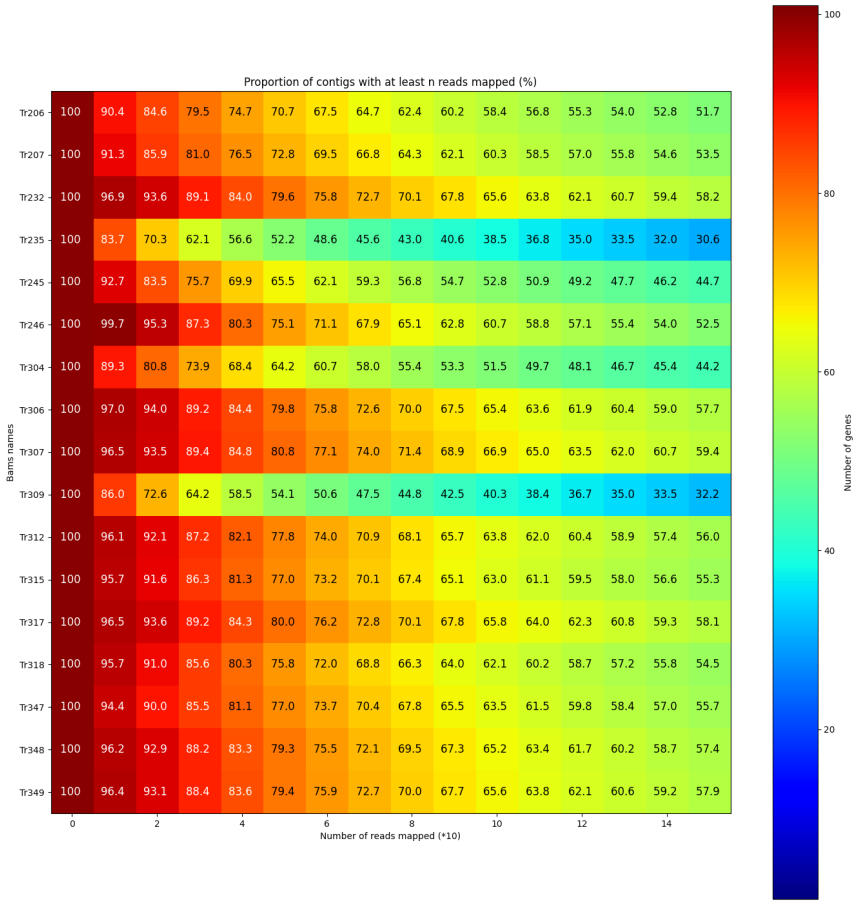



(a) Résultat de l'analyse des BamTrEx (3.4.5).
Commande : `Contig Number ./TargetedFiles.json -m 16 -kgqcuwt -j Classic --show_values -1 -y --transparent --legends legends.json --start_at_0`

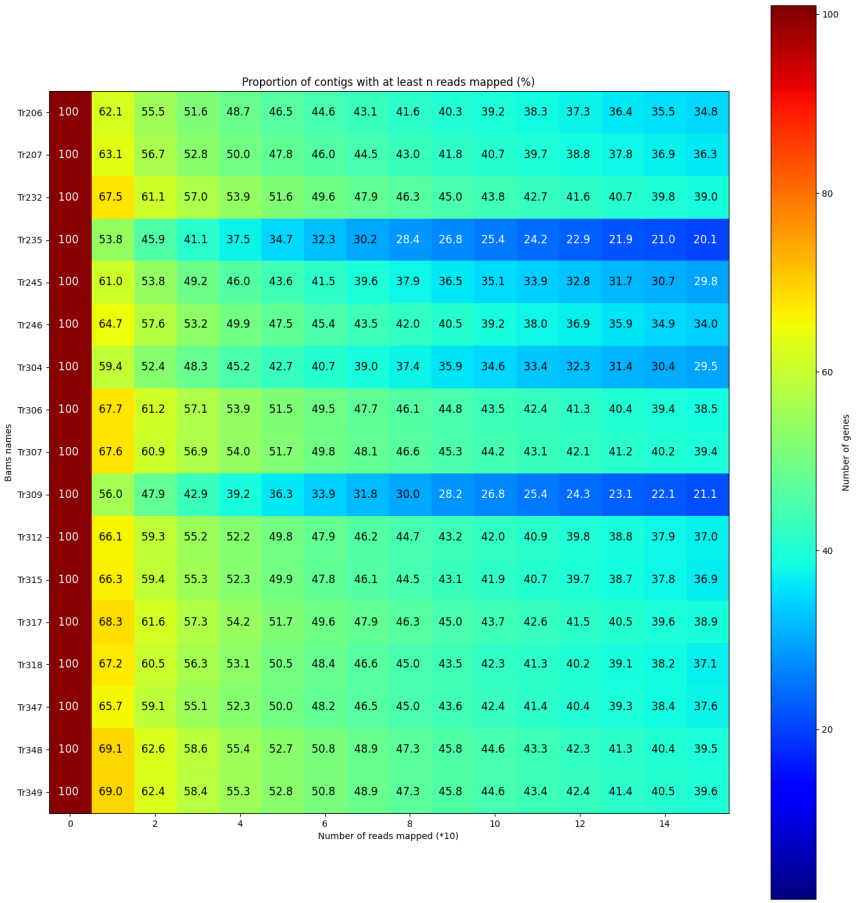



(b) Résultat de l'analyse des BamTrMo (3.4.5).
Commande : `Contig Number ./TargetedFiles.json -m 16 -kgqcuwt -j Classic --show_values -1 -y --transparent --legends legends.json --start_at_0`

FIGURE 5.7 – Résultat de l'analyse des "*.bam*" effectuée dans la 3.4.5. Présente le nombre de contigs avec au moins $n*10$ reads dans BamTrEx et BamTrMo. L'ordonnée indique le nom de l'individu et l'abscisse le nombre de reads considéré ($n$). Un incrémentassions de 1 dans l'axe de l'abscisse équivaut à une augmentation de 10 reads. L'échelle de couleur indique le nombre de contigs contenant ce nombre de reads.
Fait avec le logiciel mentionné dans la 2. La version utilisée est la 1.2.0.




(a) Résultat de l'analyse des BamTrEx (3.4.5).
Commande : `Contig Number ./TargetedFiles.json -m 16 -kgqcuwt -j Classic --show_values -1 -y --transparent --legends legends.json --start_at_0 --percent`




(b) Résultat de l'analyse des BamTrMo (3.4.5).
Commande : `Contig Number ./TargetedFiles.json -m 16 -kgqcuwt -j Classic --show_values -1 -y --transparent --legends legends.json --start_at_0 --percent`

FIGURE 5.8 – Résultat de l'analyse des "*.bam*" effectuée dans la 3.4.5. Présente le pourcentage de contigs avec au moins $n * 10$ reads dans BamTrEx et BamTrMo. L'ordonnée indique le nom de l'individu et l'abscisse le nombre de reads considéré ($n$). Un incrémentassions de 1 dans l'axe de l'abscisse équivaut à une augmentation de 10 reads. L'échelle de couleur indique le pourcentage de contigs contenant ce nombre de reads.
Fait avec le logiciel mentionné dans la 2. La version utilisée est la 1.2.0.

## Acronymes

$D_n$ nombre de substitutions non synonymes entre deux populations. C'est le nombre de sites différents entraînant un changement dans la séquence d'acides aminés de la protéine. 5, 20

$D_s$ nombre de substitutions synonymes entre deux populations. C'est le nombre de sites différents n'entraînant pas de changements dans la séquence d'acides aminés de la protéine. 5, 20

$P_n$ nombre de sites polymorphes non synonymes au sein d'une population. C'est le nombre de mutations entraînant un changement dans la séquence d'acides aminés de la protéine. 4, 5, 20

$P_s$ nombre de sites polymorphes synonymes au sein d'une population. C'est le nombre de mutations n'entraînant aucun changement dans la séquence d'acides aminés de la protéine. 5, 20

$\frac{D_n}{D_s}$ aussi appelé $\omega$ ou $\frac{K_a}{K_s}$, ce ratio représente la balance entre les substitutions synonymes ($D_s$) et non synonymes ($D_n$) entre deux populations. Si $\frac{D_n}{D_s} > 1$ la substitution est probablement conservée par la sélection naturelle. A l'inverse, si $\frac{D_n}{D_s} < 1$ les substituions sont probablement éliminées par la sélection naturelle. La significativité du résultats se calcule en utilisant un test du khi carré. 5, 21

$\frac{P_n}{P_s}$ représente la balance entre les mutations synonymes ($P_s$) et non synonymes ($P_n$) dans une unique population. 5, 21

$\omega$ autre écriture de $\frac{D_n}{D_s}$. 5, 21

**BamGeStar** fichiers *.bam* générés avec STAR en utilisant GeMo. 15, 19, 23

**BamTrEx** fichiers *.bam* générés avec BWA en utilisant TrEx. 15–20, 23, 26, 27

**BamTrMo** fichiers *.bam* générés avec BWA en utilisant TrMo. 15–20, 23, 26, 27

**GeMo** génome de référence. (*Génome Moderne*). 2, 12, 14, 15, 23

**GNSP=** Nombre de gènes présentant au moins un certain nombre de SNP. 5–7

**GNSP≥** nombre de gènes présentant un certain nombre de SNP. 6–8

**NbSNP** Nombre de SNP. 6–8, 10–12

**oldBAM** fichiers *.bam* générés avec BWA en utilisant TrEx datant d'avant le début du stage (cf. section 1.1.2). 2, 15, 19, 23

**REGEX** suite de caractère spécifique décrivant une liste de caractères possible. Exemple `^re.*` correspond à tous les mots (ou toutes les lignes dépendant du contexte) qui commencent par "re". 22

**SNP** polymorphisme Nucléotidique Simple. 3, 5–7, 9–12, 23

**TrEx** transcriptome de référence créé par l'équipe. (*Transcriptome ex-nihilo)*. 2, 12, 15–19, 23

**TrMo** transcriptome de référence construit à partir du génome de référence. (*Transcriptome moderne*). 12, 15, 16, 18, 19

## Glossaire

*.bashrc* fichier de configuration utilisé par les environnements Bash type. 13

*.pdf* type de fichier standardisé correspondant à la norme ISO 32000 type. 8

*.png* format d'image numérique. type. 7

*.svg* format d'image numérique utilisant des vecteurs pour représenter une image. Ce type de fichiers assure que l'image pourra être étendue sans perte de qualité type. 7

*.tsv* type de fichier texte représentant un tableau. Les données y sont séparées par des tabulations. type. 7, 18

*README* fichier texte utilisé pour décrire une application aux utilisateurs. type. 9

*pyproject.toml* fichier utilisé dans les projets Python pour préciser les prérequis et les caractéristiques d'une application. type. 9

**LaTeX** système de développement de documents techniques et scientifiques [Lat] type. 8, 24

**Bash** langage de programmation présent par défaut sur les sytémes d'exploitation linux. [**bash**] type. 24

**brins bien appariés** deux brins d'ADN (ou d'ARN) sont bien appariés lorsque les nucléotides qui les composent sont correctement appariés selon la complémentarité des bases. type. 16

**BWA** mappeur efficace pour aligner de l'ARN sur des références de petite taille [Li13] type. 15

**cluster** ensemble d'ordinateur permettant de réaliser des calculs gourmands en ressource. type. 13, 24

**CODEML** programme contenu dans le package PAML (cf. section 4.3.1, [pam]) type. 20–22, 24

**commit** opération qui enregistre un ensemble de modifications apportées au code dans l'historique du projet. type. 8, 9

**Conda** gestionnaire de paquets et d'environnements virtuels isolés. [Ana] type. 9, 12, 13, 24

**diploïde** "se dit d'une cellule qui possède un jeu double de chromosomes semblables" [Rob] type. 1

**dNdSpNpS** outil permettant d'extraire les $D_n$, $D_s$, $P_n$, $P_s$ de données liées au génome. [dNd] type. 3

**Doxygen** Outil de documentation automatique [Dox] type. 8, 24

**EggLib** bibliothèque Python proposant des wrappers pour divers outils de bioinformatique et des objets permettant de manipuler des séquences biologiques (cf. section 4.3.2) [Sio+22] type. 20–22, 24

**GeCKO** pipeline facilitant les analyses phylogénétiques (cf. section 3.1) ([Ard+24]) type. 12, 13, 15, 16, 24

**getopt** bibliothèque Python permettant de lire les arguments donnés à un script au moment du lancement [Get] type. 6, 9

**Gimp** Outil de modification d'image [Gim] type. 6, 20

**Git** système de contrôle de version distribué qui permet de suivre les modifications d'un code source. [Git] type. 24

**GitHub** plate-forme en ligne permettant de centraliser des dépôts Git. [Git24] type. 5, 8, 12, 21, 24

**hexaploïde** se dit d'une cellule ou d'un organisme qui possède six jeux complets de chromosomes (6n), soit six copies de chaque chromosome. type. 1

**hétérogame** mode de "reproduction sexuée par deux gamètes de morphologie différente (par ex. ovule et spermatozoïde)" [Rob] type. 1–3

**inflorescence** "mode de groupement des fleurs d'une plante, ou groupe de fleurs" [Dic] type. 1

**json** bibliothèque Python permettant d'exploiter des *.json* [**json**] type. 6

**Lucidchart** éditeur de diagrammes en ligne.[Luc] type. 14

**MatPlotLib** bibliothèque Python permettant de générer des graphiques 2D [Hun07] type. 6, 7

**Minimap2** mappeur efficace pour aligner de l'ARN sur de l'ADN [Min] type. 14, 15

**OnlineGant** logiciel en ligne permettant de faire des diagrammes de Gantt [Gan] type. 25

**os** bibliothèque Python permettant de communiquer avec le système d'exploitation [OS] type. 6

**PAML** logiciel utilisé pour l'analyse phylogénétique (cf. section 4.3.1, [Yan20]) (*Phylogenetic Analysis by Maximum Likelihood*) type. 21

**pipe** ou "tube" est un "mécanisme qui permet de chaîner des processus de sorte que la sortie d'un processus (stdout) alimente directement l'entrée (stdin) du suivant." [Dic] type. 9

**polymorphisme** présence de plusieurs allèles (version d'un gène) au sein d'une population type. 4

**Pytest** bibliothèque Python permettant de tester les résultat des fonctions d'un projet [PyT] type. 6, 8

**Python** langage de programmation grand publique [Pyt24] type. 21, 22, 24

**R** langage de programmation utilisé pour les analyses statistiques [R] type. 20, 24

**Samtools** outil permettant l'analyse et la manipulation de fichiers *.bam* type. 19

**Singularity** sytéme donnant la possibilité de créer des "conteneurs" permettant à une application de tourner dans un environnement virtuelle et indépendant. Ce système permet d'assurer une forme de reproductibilité et de probabilité type. 13

**Slurm** logiciel permettant de répartir l'exécution de programmes sur les différents noeuds d'un cluster [YJG03] type. 12, 21, 24

**SnakeMake** système de gestion de workflow permettant d'assurer une forme de reproductibilité et de portabilité [Möl+21] type. 12

**STAR** mappeur efficace pour aligner de l'ARN sur de l'ADN [Dob+13] type. 14, 15

**sys** bibliothèque Python permettant d'accéder aux variables de l'interpréteur Python [Sys] type. 6

**test de Fisher** test statistique permettant de déterminer si deux variables d'un tableau de contingence sont indépendantes type. 21

**test de McDonald–Kreitman** test utilisé pour distinguer les effets de la sélection naturelle des variations génétiques neutres. type. 20, 21

**test du khi carré** test statistique permettant de déterminer si les différences observées entre les fréquences attendues et les fréquences observées sont significatives type. 5, 20

**wrapper** code informatique permettant d'encapsuler un logiciel tiers dans un autre logiciel. type. 21, 22

**WSL** sous systéme Linux pour Windows. [22] type. 24

## Références


[Bur+24] Concetta Burgarella et al. « Mating systems and recombination landscape strongly shape genetic diversity and selection in wheat relatives ». In : *Evolution Letters* (août 2024), qrae039. ISSN : 2056-3744. DOI : 10.1093/evlett/qrae039. URL : https://doi.org/10.1093/evlett/qrae039 (visité le 17/08/2024).

[Glé+19] Sylvain Glémin et al. « Pervasive hybridizations in the history of wheat relatives ». In : *Science Advances* 5.5 (mai 2019). Publisher : American Association for the Advancement of Science, eaav9188. DOI : 10.1126/sciadv.aav9188. URL : https://www.science.org/doi/10.1126/sciadv.aav9188 (visité le 02/08/2024).

[Wik24] Wikipedia. *Blé tendre*. fr. Page Version ID : 217016404. Juill. 2024. URL : https://fr.wikipedia.org/w/index.php?title=Bl%C3%A9_tendre&oldid=217016404#G%C3%A9nome (visité le 03/08/2024).

[Sar+17] Gautier Sarah et al. « A large set of 26 new reference transcriptomes dedicated to comparative population genomics in crops and wild relatives ». en. In : *Molecular Ecology Resources* 17.3 (mai 2017), p. 565-580. ISSN : 1755-098X, 1755-0998. DOI : 10.1111/1755-0998.12587. URL : https://onlinelibrary.wiley.com/doi/10.1111/1755-0998.12587 (visité le 02/08/2024).

[Flo24a] Marchal Florent. *F-Marchal/M1BioinfoInternship2024-INRAE_AGAP_GE2POP*. original-date : 2024-07-26T07 :35 :22Z. Juill. 2024. URL : https://github.com/F-Marchal/M1BioinfoInternship2024-INRAE_AGAP_GE2POP (visité le 05/08/2024).

[Flo24b] Marchal Florent. *F-Marchal/SnpHeatMap*. original-date : 2024-05-04T14 :18 :46Z. Juill. 2024. URL : https://github.com/F-Marchal/SnpHeatMap (visité le 05/08/2024).

[GE224] GE2POP. *GE2POP/GeCKO*. original-date : 2022-01-31T14 :38 :02Z. Mai 2024. URL : https://github.com/GE2POP/GeCKO (visité le 02/08/2024).

[Ard+24] Morgane Ardisson et al. « GeCKO : user-friendly workflows for genotyping complex genomes using target enrichment capture. A use case on the large tetraploid durum wheat genome. » In : (mars 2024). DOI : 10.21203/rs.3.rs-4123643/v1.

[Flo24c] Marchal Florent. *F-Marchal/M1BioinfoInternship2024-INRAE_AGAP_GE2POP*. original-date : 2024-07-26T07 :35 :22Z. Juill. 2024. URL : https://github.com/F-Marchal/M1BioinfoInternship2024-INRAE_AGAP_GE2POP (visité le 02/08/2024).

[ÁKY23] Sandra Álvarez-Carretero, Paschalia Kapli et Ziheng Yang. « Beginner's Guide on the Use of PAML to Detect Positive Selection ». In : *Molecular Biology and Evolution* 40.4 (avr. 2023), msad041. ISSN : 1537-1719. DOI : 10.1093/molbev/msad041. URL : https://doi.org/10.1093/molbev/msad041 (visité le 02/08/2024).

[Yan+00] Ziheng Yang et al. « Codon-Substitution Models for Heterogeneous Selection Pressure at Amino Acid Sites ». In : *Genetics* 155.1 (mai 2000), p. 431-449. ISSN : 1943-2631. DOI : 10.1093/genetics/155.1.431. URL : https://doi.org/10.1093/genetics/155.1.431 (visité le 15/08/2024).

[pam] PAML-TUTORIAL. *paml-tutorial/positive-selection at main · abacus-gene/paml-tutorial*. en. URL : https://github.com/abacus-gene/paml-tutorial/tree/main/positive-selection (visité le 15/08/2024).

[Flo24d] Marchal FLORENT. *F-Marchal/SnpHeatMap*. original-date : 2024-05-04T14 :18 :46Z. Juill. 2024. URL : https://github.com/F-Marchal/SnpHeatMap (visité le 02/08/2024).

[Lat] LATEX. *LaTeX - A document preparation system*. URL : https://www.latex-project.org/ (visité le 19/08/2024).

[Li13] Heng LI. *Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM*. arXiv :1303.3997 [q-bio]. Mai 2013. DOI : 10.48550/arXiv.1303.3997. URL : http://arxiv.org/abs/1303.3997 (visité le 19/08/2024).

[Ana] ANACONDA. *About Anaconda*. en-US. URL : https://www.anaconda.com/about-us (visité le 19/08/2024).

[Rob] Le ROBERT. *Dictionnaire français en ligne Le Robert*. fr-FR. URL : https://dictionnaire.lerobert.com/ (visité le 03/08/2024).

[dNd] DNDSPINPIS. *PopPhyl*. URL : https://kimura.univ-montp2.fr/PopPhyl/index.php?section=tools (visité le 19/08/2024).

[Dox] DOXYGEN. *Doxygen homepage*. URL : https://doxygen.nl/ (visité le 19/08/2024).

[Sio+22] Mathieu SIOL et al. « EggLib 3 : a Python package for population genetics and genomics ». In : *Molecular Ecology Resources* 22 (juin 2022). DOI : 10.1111/1755-0998.13672.

[Get] GETOPT. *getopt — C-style parser for command line options*. en. URL : https://docs.python.org/3/library/getopt.html (visité le 19/08/2024).

[Gim] GIMP. *GIMP*. en. URL : https://www.gimp.org/ (visité le 19/08/2024).

[Git] GIT. *Git*. URL : https://git-scm.com/ (visité le 19/08/2024).

[Git24] GITHUB. *GitHub : Let's build from here*. en. 2024. URL : https://github.com/ (visité le 19/08/2024).

[Dic] Le DICTIONNAIRE. *LE DICTIONNAIRE - Dictionnaire français en ligne gratuit*. fr. URL : https://www.le-dictionnaire.com (visité le 03/08/2024).

[Luc] LUCIDCHART. *Visualisez et mettez en œuvre vos projets | Lucidchart*. URL : https://www.lucidchart.com/pages/fr/landing?km_CPC_CampaignId=369475290&km_CPC_AdGroupID=1240249291937118&km_CPC_Keyword=lucidchart&km_CPC_MatchType=e&km_CPC_ExtensionID=%7Bextensionid%7D&km_CPC_Network=s&km_CPC_AdPosition=&km_CPC_Creative=&km_CPC_TargetID=kwd-77515723735585:loc-66&km_CPC_Country=125950&km_CPC_Device=c&km_CPC_placement=&km_CPC_target=&mkt_query=Lucidchart&msclkid=6af96a7994ab1f22ed690cbb635ad09e (visité le 19/08/2024).

[Hun07] John D. HUNTER. « Matplotlib : A 2D Graphics Environment ». In : *Computing in Science & Engineering* 9.3 (mai 2007). Conference Name : Computing in Science & Engineering, p. 90-95. ISSN : 1558-366X. DOI : 10.1109/MCSE.2007.55. URL : https://ieeexplore.ieee.org/document/4160265 (visité le 19/08/2024).

[Min] MINIMAP2. *Minimap2 : pairwise alignment for nucleotide sequences | Bioinformatics | Oxford Academic*. URL : https://academic.oup.com/bioinformatics/article/34/18/3094/4994778?login=false (visité le 19/08/2024).

[Gan] GANTT. *Free Online Gantt Chart Software*. URL : https://www.onlinegantt.com (visité le 19/08/2024).

[OS] OS. *os — Miscellaneous operating system interfaces*. en. URL : https://docs.python.org/3/library/os.html (visité le 19/08/2024).

[Yan20] Ziheng YANG. *PAML MANUAL - User Guide - PAML : Phylogenetic Analysis by Maximum Likelihood*. Fév. 2020. URL : http://abacus.gene.ucl.ac.uk/software/pamlDOC.pdf (visité le 02/08/2024).

[PyT] PYTEST. *pytest documentation*. URL : https://docs.pytest.org/en/stable/ (visité le 19/08/2024).

[Pyt24] PYTHON. *Welcome to Python.org*. en. Août 2024. URL : https://www.python.org/ (visité le 19/08/2024).

[R] R. *R : The R Project for Statistical Computing*. URL : https://www.r-project.org/ (visité le 19/08/2024).

[YJG03] Andy B. YOO, Morris A. JETTE et Mark GRONDONA. « SLURM : Simple Linux Utility for Resource Management ». en. In : *Job Scheduling Strategies for Parallel Processing*. Sous la dir. de Dror FEITELSON, Larry RUDOLPH et Uwe SCHWIEGELSHOHN. Berlin, Heidelberg : Springer, 2003, p. 44-60. ISBN : 978-3-540-39727-4. DOI : 10.1007/10968987_3.

[Möl+21] Felix MÖLDER et al. *Sustainable data analysis with Snakemake*. en. Jan. 2021. DOI : 10.12688/f1000research.29032.1. URL : https://f1000research.com/articles/10-33 (visité le 19/08/2024).

[Dob+13] Alexander DOBIN et al. « STAR : ultrafast universal RNA-seq aligner ». In : *Bioinformatics* 29.1 (jan. 2013), p. 15-21. ISSN : 1367-4803. DOI : 10.1093/bioinformatics/bts635. URL : https://doi.org/10.1093/bioinformatics/bts635 (visité le 19/08/2024).

[Sys] SYS. *sys — System-specific parameters and functions*. en. URL : https://docs.python.org/3/library/sys.html (visité le 19/08/2024).

[22] *Windows Subsystem for Linux Documentation*. en-us. Juin 2022. URL : https://learn.microsoft.com/en-us/windows/wsl/ (visité le 19/08/2024).