



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

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 (cf. section 1.1) pour déterminer si celles-ci peuvent être utilisées pour rechercher des traces de sélection chez des espèces sauvages apparentées au blé (cf. section 1.2). 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 de plantes apparentées au blé dur 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 le 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 [allogames](#) et, enfin, certaines ont un régime mixte ([autogame](#) et [allogame](#)).

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). A titre indicatif, *Triticum urartu* a un génome [diploïde](#) de 4,8Gpb soit 35,5 fois plus grand qu’*Arabidopsis thaliana* (0,135Gpb). Le génome de *Triticum urartu* correspond à la partie « A » du génome du blé tendre (Froment) qui est [hexaploïde](#) (3 génomes : « A », « B » et « C » cf. [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 que les traces de sélection au travers d’une [phylogénie](#).

Espèce	Mode de reproduction	Nombre d'individus
<i>Aegilops bicornis</i>	Mixte	3
<i>Aegilops caudata</i>	Mixte	3
<i>Aegilops comosa</i>	Mixte	2
<i>Aegilops longissima</i>	Mixte	4
<i>Aegilops mutica</i>	Allogame stricte	7
<i>Aegilops searsii</i>	Mixte	3
<i>Aegilops sharonensis</i>	Mixte	2
<i>Aegilops speltoides</i>	Allogame stricte	10
<i>Aegilops tauschii</i>	Mixte – fortement autogame	21
<i>Aegilops umbellulata</i>	Mixte	3
<i>Aegilops uniaristata</i>	Mixte	3
<i>Eremopyrum bonaepartis</i> (outgroup)	Mixte – fortement autogame	1
<i>Hordeum bulbosum</i> (outgroup)	Allogame	1
<i>Hordeum vulgare</i> (outgroup)	Mixte – fortement autogame	3
<i>Secale strictum</i> (outgroup)	Allogame	3
<i>Secale vavilovii</i> (outgroup)	Allogame	3
<i>Taeniatherum caputmedusae</i> (outgroup)	Mixte – fortement autogame	1
<i>Triticum monococcum</i>	Mixte – fortement autogame	10
<i>Triticum urartu</i>	Mixte – fortement autogame	20

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.

Dès le début de ce stage, nous avons 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 uniformisées à 100 **nucléotides** par **read**. Les fichiers trop gros ont 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](https://ensemblgenomes.org/assembly/Triticum_urartu/IGDB.dna.toplevel.fa.gz)

Le **transcriptome** de référence complet de *Triticum urartu* provient lui aussi de **ensembl biomart**. Voici la requête biomart permettant de récupérer celui-ci :

- Dataset : Triticum urartu genes (Tu2.0)
- Gene type : protein_coding; Ensembl Canonical : Only

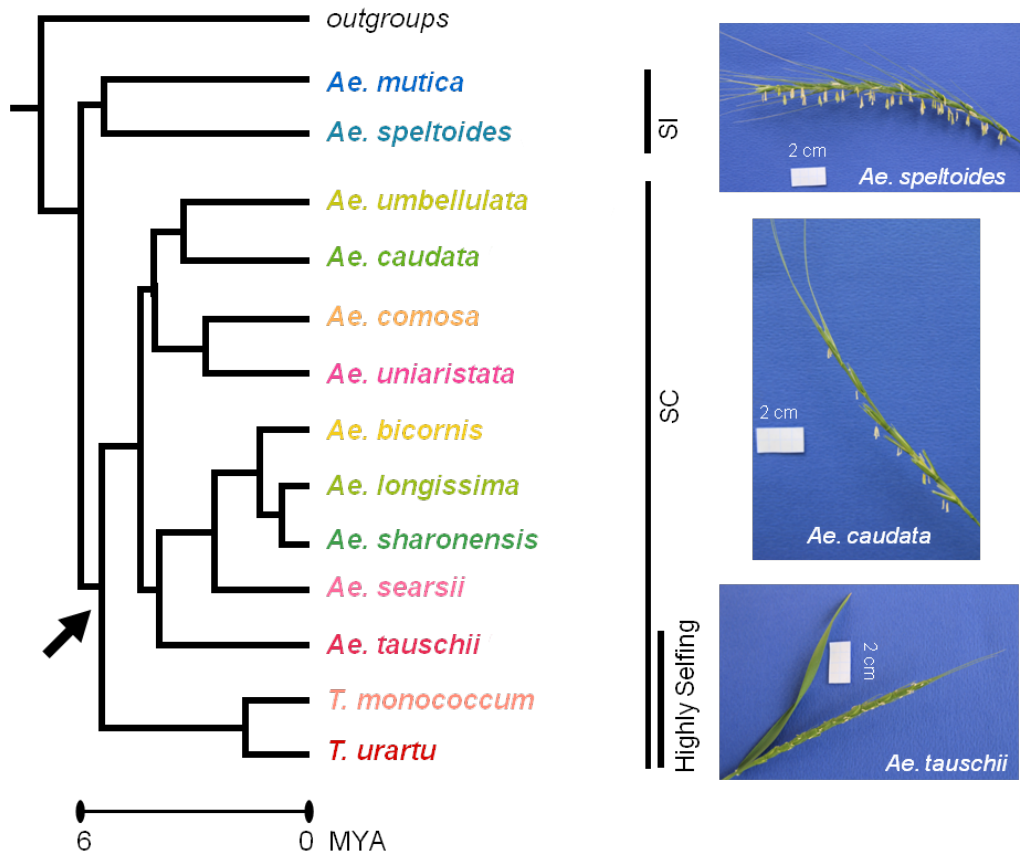


FIGURE 1 – Relation phylogénétique entre les 13 espèces diploïdes du genre *Aegilops* / *Triticum*. Les couleurs représentent un gradient d'auto-fécondation. Les espèces **allogames** (SI) strictes sont bleues, les espèces avec un mode de reproduction mixte (SC) sont en vert / jaune et les espèces **autogames** (Highly Selfing) sont en rouge. Cette figure est issue de [Bur+24] et sa légende a été adaptée et traduite par l'auteur de ce rapport.

— Attributs : Gene stable ID ; Transcript stable ID ; cDNA sequences ; Upstream flank [200] ; Downstream flank [200]

Tableau **SNP** par **contig** 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

Dans les parties codantes du génome on distingue les sites **synonymes** et les sites non **synonymes**. La locution "site **synonyme**" désigne les codons codants pour un même acide aminé (tableau 2). 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**.

Espèce	Individu	Séquence					
espèce_1	individu 1	ATG	AGA	TGC	CGA	CGT	TTT
		M	R	C	R	R	F
espèce_1	individu 2	ATG	AGA	TGC	CGA	CGT	TTT
		M	R	C	R	R	F
espèce_1	individu 3	ATG	AGA	TGC	CGA	CGT	TTT
		M	R	C	R	R	F
espèce_2	individu 1	ATG	CGT	TGC	CCA	TGT	TTT
		M	R	C	R	C	F
espèce_2	individu 2	ATG	CGT	TGC	CCC	TGT	TTT
		M	R	C	R	C	F
espèce_2	individu 3	ATG	CGT	TGC	CCA	TGT	TTT
		M	R	C	R	C	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) contient un exemple de substitution synonyme; le codon n°4 (en orange) contient un exemple de polymorphisme synonyme; le codon n°5 (en rouge) contient un exemple de substitution non synonyme

la **sélection positive** Elle favorise la fixation de mutations dans une population. 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 substitutions synonymes.

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

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

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 sélection se fait en utilisant les indicateurs suivants :

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 ω 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 substitutions sont probablement éliminées par la sélection naturelle. La significativité du résultats se calcule en utilisant un test du khi deux

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

Dès lors, il est nécessaire de disposer de suffisamment de sites variables pour déterminer si un gène est potentiellement sous sélection via ce type d'analyse. Il est donc requis d'avoir de nombreux gènes avec suffisamment de sites variables. Sans cela, il ne sera pas possible d'estimer correctement ces ratios. L'objectif de ce stage est donc de déterminer si le jeu de données actuel permet cela ou si de nouvelles données doivent être acquises.

Tous les éléments précédemment évoqués 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 : [\[Mar24a\]](#)
- sur le dépôt du projet [\[Mar24b\]](#). 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** ($\text{GNSP} =$). Ex : environ 4000 **contigs** contiennent 1 **SNP** (figure 2).
- le nombre de **contigs** présentant au moins un certain nombre de **SNP** ($\text{GNSP} \geq$). Ex : environ 12 300 ($= 4000 + 3000 + 2000 + 1000 + 800 \dots$) **contigs** contiennent au moins 1 **SNP** (figure 2).

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 (résultats (note sur 20) d'une cohorte d'étudiant, nombre de maladies rare par personne dans la population d'un pays ...).

Ce programme utilise plusieurs bibliothèques externes :

- | | | |
|------------------------------|--------------------------|------------------------|
| — Pytest | — getopt | — os |
| — Matplotlib | — sys | — json |

2.1.1 Données attendues

Les données doivent être contenues dans des fichiers tabulés (exemple : tableau 3). Chaque tableau doit contenir des données liées à 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*.

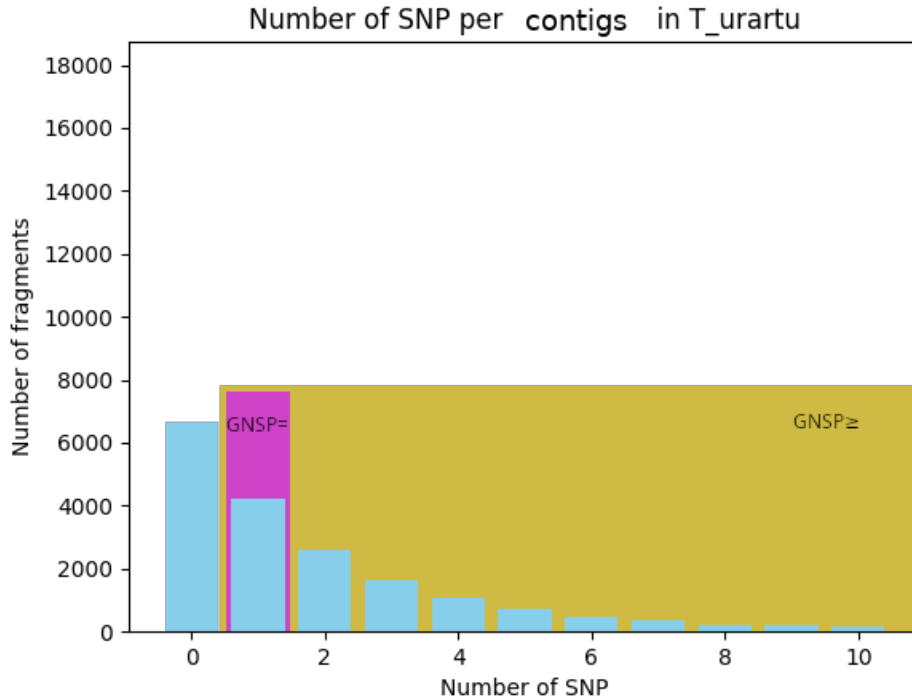


FIGURE 2 – Exemple de sortie de section 2. La zone violette représente la partie prise en compte par $GNSP=$ tandis que la zone jaune représente le $GNSP\geq$. Cette zone s’étend aussi en dehors des limites du graphique. L’abscisse représente le $NbSNP$ et l’ordonnée représente le nombre de *contigs* 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.

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 $GNSP\geq$. 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 $GNSP\geq$ 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 d’heatmaps globales. S’active avec l’option -g

Ces graphiques ont tous, en abscisse ou dans les cellules, le nombre de *SNP* ($NbSNP$).

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 [Mar24b]. La colonne "Contig_name" correspond à la clef primaire, la colonne "SNP" correspond au NbSNP, et la "Autre" correspond à d'autres informations présentes dans le tableau. Ces informations-là ne sont pas utilisées par [Mar24b]. Ce tableau a été généré avec des données aléatoires et correspond aux premières lignes du fichier `tests/data/test1` de la version 1.2.0 de [Mar24b].

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.py::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.py::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 le nombre d'espèces 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 fonctionnement 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 [L^AT_EX](#) 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 : `echo "Clef""NbSnp""Exemple.tsv"| python3 main.py -g` ne fonctionne pas car les arguments doivent être placés avant les options.
 - seuls les entiers supérieurs ou égaux à 0 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 certaines 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 utilisée pour démarrer le programme, mais le dictionnaire d'argument reçu par la fonction qui démarre 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

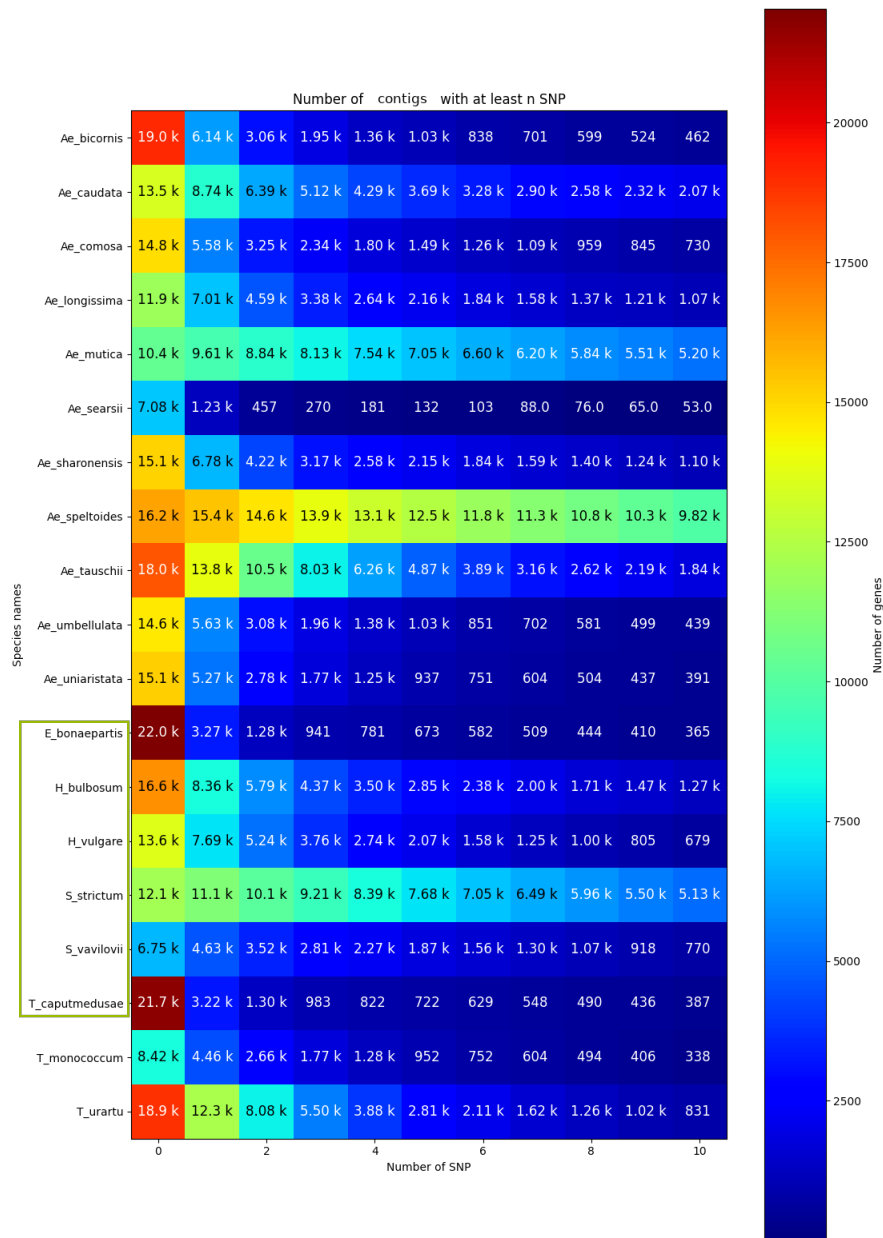
Pour cette analyse, nous avons utilisé les tableaux récapitulant le nombre de *SNP* par *contig* mentionné dans 1.1.2 et le logiciel décrit dans cette partie (section 2). Les résultats décrits dans cette partie correspondent à la figure 3.

Dans notre cas, pour pouvoir procéder à une analyse de traces de sélection, on souhaiterait avoir 5 *SNP* par *contigs* sur au moins 70% des *contig*. 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.

La figure 3a et la figure 3b permettent aussi de déterminer que :

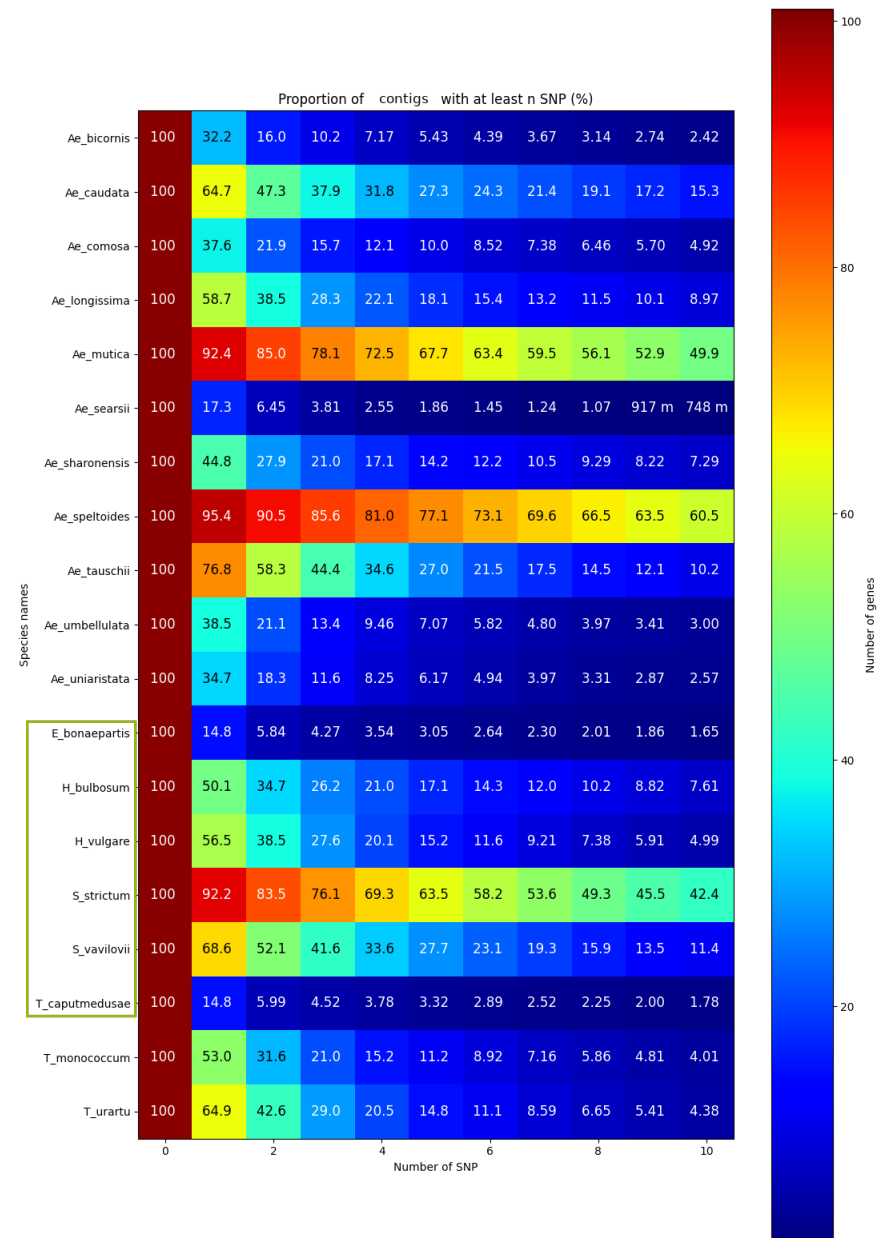
- *Aegilops speltoides* est un bon candidat pour une étude exploratoire, c'est la seule espèce avec 5 *SNP* par *contig* sur au moins 70%.
- les données disponibles sur *Aegilops searsii* sont largement insuffisantes (peu de *reads* et peu de *SNP* par *contig*).

La figure 3 montre clairement que, pour la plupart des espèces, l'on ne dispose pas d'un nombre suffisant de *SNP*. Ces résultats démontrent aussi l'intérêt des figures générées par [Mar24c] pour décider si des analyses de traces de sélection sont envisageables ou non.



(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. Les espèces encadrées en jaune sont les espèces appartenant à l'outgroup (tableau 1). Image modifiée avec Gimp.

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. Les espèces encadrées en jaune sont les espèces appartenant à l'outgroup (tableau 1). Image modifiée avec Gimp.

Commande : `Contig_name BiAllelic_SNP ./TargetedFiles.json -m 11 -kgqcuwt -j Percent --show_values -1 -y --transparent --legends legends.json --start_at_0 --percent`

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 le **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 les résultats soient moins complets que s'il avait été réalisé sur un génome de référence « moderne » (**GeMo**) ou sur **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 nous avons 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 développé dans l'équipe qui m'a accueilli. Il est en libre accès sur **GitHub** [GE224], [Ard+24]. Ce pipeline a pour but de faciliter les analyses de données **NGS** sur de gros génomes en les rendant plus « user-friendly ». Il se base sur **Conda** et **SnakeMake** pour assurer sa portabilité. Celui-ci a été choisi car :

- il est connu de 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 se sont révélées être bien plus compliquées que prévu. Celle-ci a nécessité un fort investissement en temps avant de 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** a 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 dossier "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 absolus 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. Je travaillais en WSL sur une machine Windows.

Résolution La création des différents environnements a été faite par Johanna GIRONOLLE, l'une des développeuses 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. Il s'agit de 44 fichiers *.fastq* contenant, en moyenne, 23 917 794 reads.

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. ([Mar24d] \$gz2_to_bz2\$)

Tri des fichiers Les fichiers *.fastq* sur lesquels j'ai travaillé sont tous stockés dans le même dossier, indépendamment 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. ([Mar24d] Sort_files_by_species)

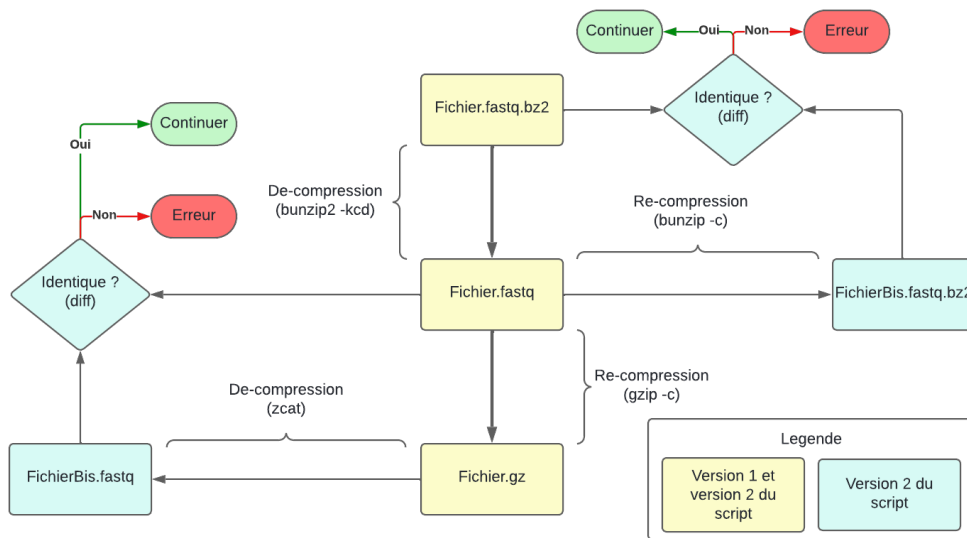


FIGURE 4 – Fonctionnement du programme de changement de compression des ".fastq". Les cadres jaunes représentent les étapes communes à la version 1 et à la version 2 tandis que les cadres bleus représentent les étapes spécifiques à la version 2. Le programme est accessible ici : [Mar24d]. Image faite avec Lucidchart.

3.3 Déroulement des mappings

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

3.3.1 Mapping avec Minimap2 (GeMo)

Le mappeur 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 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 100Go là où 64Go 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 `mappeur`, bien qu'il ne soit pas accessible dans `GeCKO`. Le `mappeur` choisi a été `STAR`. Ce choix fut motivé par les conseils d'Anthony BOUREUX, mon tuteur pédagogique. 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 pu être analysés.

A la suite de ce stage, il est prévu d'ajouter `STAR` dans `GeCKO`.

3.3.3 Mapping avec BWA (TrEx et TrMo)

Les `mappings` sur les `transcriptomes` de références ont été faits avec `BWA mem`. Il s'agit du `mappeur` 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 `TrEx` datant d'avant le début du stage (cf. section 1.1.2)

`BamTrMo` fichiers `.bam` générés avec `BWA` en utilisant `TrMo`

`BamTrEx` fichiers `.bam` générés avec `BWA` en utilisant `TrEx`

`BamGeStar` fichiers `.bam` générés avec `STAR` en utilisant `GeMo`

Avec notre hypothèse de départ (section 3) nous attendons à ce que le nombre de `reads` mappés suivent la distribution suivante : $BamGeStar \geq oldBAM > BamTrMo \gg oldBAM$.

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 la taille globale 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 %. Ce premier résultat nous permet de supposer que les deux `mapping` sont

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)

équivalents ou très légèrement en faveur de **TrEx**. Cela est conforme à nos attendus de départ.

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 reads bien appariés est compris entre 78 % et 82 % pour **TrMo** tandis qu’il est compris entre 81 % et 86 % pour **TrEx**.

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

Nom	Taux d'erreurs	reads mappés (%)	reads bien 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"` soit pris en compte, la qualité de la correspondance doit être supérieure ou égale à 30. Cette valeur a été choisie car elle est généralement utilisée 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` surreprésentés, 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 bien 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` ([Mar24d] (*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 fichier tabulé ("*.tsv*"). Les valeurs du "*.tsv*" sont reformatées ([Mar24d] `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 prétraitement a permis de générer la figure 5 ([Mar24d] `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 à 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 clair 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 reçu 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).

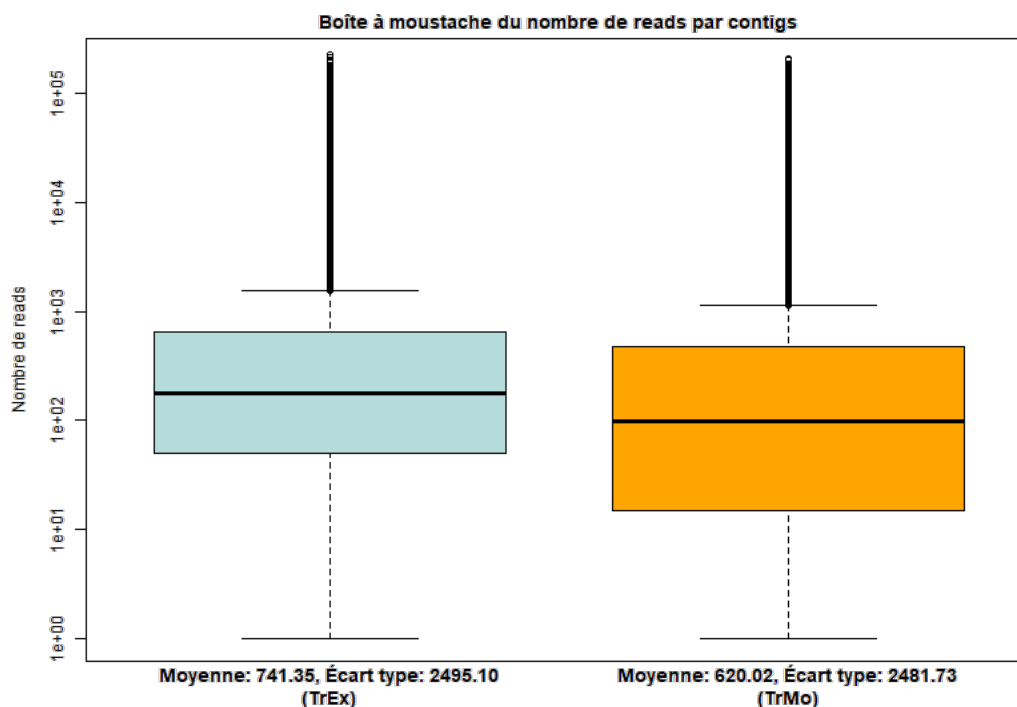


FIGURE 5 – Boîte à moustache du nombre de **reads** par **contigs**. Les données relatives à **BamTrEx** sont à gauche tandis que les données relatives à **BamTrMo** sont à droite. Seuls les **contigs** ayant au moins 1 **reads** ont été pris en compte. Figure générée sur **R** puis modifiée avec **Gimp**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

4 Pipeline pour la recherche de traces de sélection

La création de ce 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 ce pipeline pour qu’il soit dans un état utilisable. Cependant, le terrain a bien été débroussaillé. Cette section a pour but de présenter les concepts théoriques et l’état d’avancement du pipeline.

4.1 Objectif et contexte

L’objectif de ce pipeline est de détecter automatiquement des traces de sélection sur des gènes. Pour cela le pipeline doit pouvoir accepter deux types de sources :

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 (hors **out-group**).

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`). Les modèles sont expliqués dans la section 4.3.1
4. Test du khi deux 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 utiliser du multiprocessing, du multi-threading 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'outils 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 bêta, 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 classes 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-ci 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 accroc. Seule la partie 02_branch_models a été utilisée, celle-ci contenant l'ensemble des tests que nous souhaitons effectuer.

4.4 Familiarisation avec EggLib

Pour me familiariser 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 problèmes 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 considère la délétion comme des caractères invalides.

Arguments du wrapper L'argument "codon_freq" était limité à une valeur de 3 tandis que CODEML peut accepter des valeurs plus hautes. À titre indicatif, [ÁKY23] utilise une valeur de 7.

Ces deux problèmes ont été rapportés à Stéphane DE MITA (développeur de EggLib) et ont été résolus avec le passage à la version 3.3.3.

REGEX incomplète Une REGEX est utilisée pour extraire efficacement les données clefs du fichier de sortie de CODEML. Malheureusement, elle 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. les valeurs de $P(w) > 1$ associées aux sites potentiellement sélectionnés n'est pas récupérées.

J'ai tenté de corriger la REGEX (cf. [Mar24d] *Regex*) mais je ne suis pas arrivé au bout de la correction du point n°3. Mon ébauche de correction a été portée à l'attention Stéphane DE MITA.

4.5 Conclusion

Si ce stage n'a pas permis de construire ce pipeline, il aura permis de mettre en évidence des problèmes liés à [EggLib](#). L'identification de ces problèmes et la construction théorique du pipeline devrait permettre à l'avenir de reprendre la construction du pipeline.

5 Conclusion générale

Le jeu de données étudié risque fortement de ne pas convenir pour réaliser des recherches de traces de sélection chez des espèces apparentées au blé. En effet, comme vu dans la section 2.2, le nombre de **SNP** par **contig** 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 confirmer ces résultats :

- les ".bam" issus des **mappings** sur **GeMo** n'ont pas été analysés (section 3.3.2). Il est tout à fait possible que ce **mappings** soit meilleur et plus complet que les précédents.
- les tableaux présentant le nombre de **SNP** par **contig** (cf. (section 1.1.2)) n'ont été faits qu'à partir des **oldBAM**. Si les tableaux ont été générés après le filtrage drastique qu'ils ont subi (section 3.4.1), le nombre de **SNP** peu avoir été grandement affecté. Refaire ces tableaux avec les nouveaux fichiers (**BamGeStar**, **BamTrEx**, **BamTrMo**) pourrait permettre de grandement augmenter le nombre de **SNP** trouvés.

5.1 Apprentissage personnel

Enfin ce stage a été l'occasion pour moi de développer de nouvelles compétences et d'apprendre à me servir de nouveaux logiciels. Le tableau 8, en annexe, fournit une liste non exhaustive des outils que j'ai manipulés et des compétences que j'ai acquises ou commencé à acquérir.

Annexes

<i>Logiciel</i>	Exemple
Python	section 2
Doxygen	section 2.1.5
L ^A T _E X	ce rapport, cf. [Mar24d]
CODEML	section 4.3.1
EggLib	section 4.3.2
Slurm	section 3
GeCKO	section 3
Bash	[Mar24d]
R	figure 5
Conda	-
GitHub	section 2
<i>Bonne pratique</i>	Exemple
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	[Mar24d], [Mar24c]
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
<i>Apprentissage</i>	Exemple
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

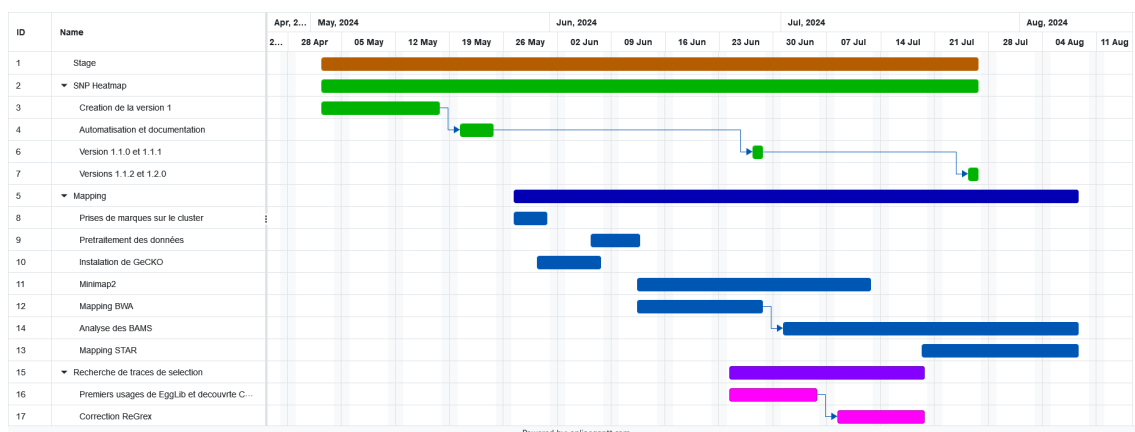
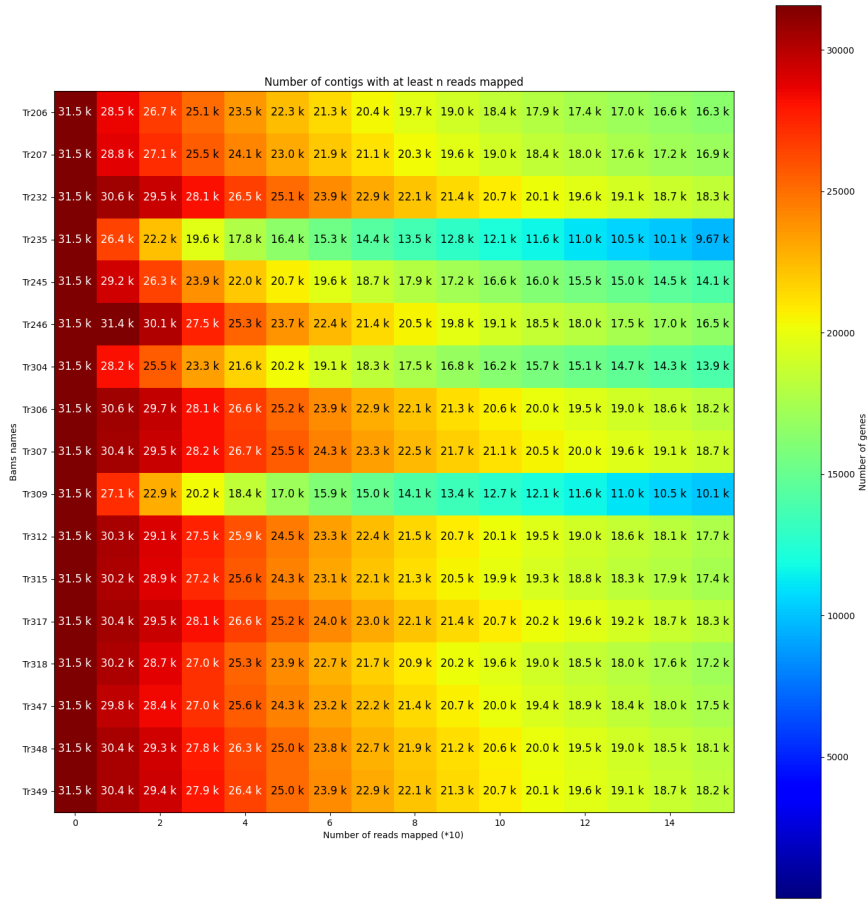
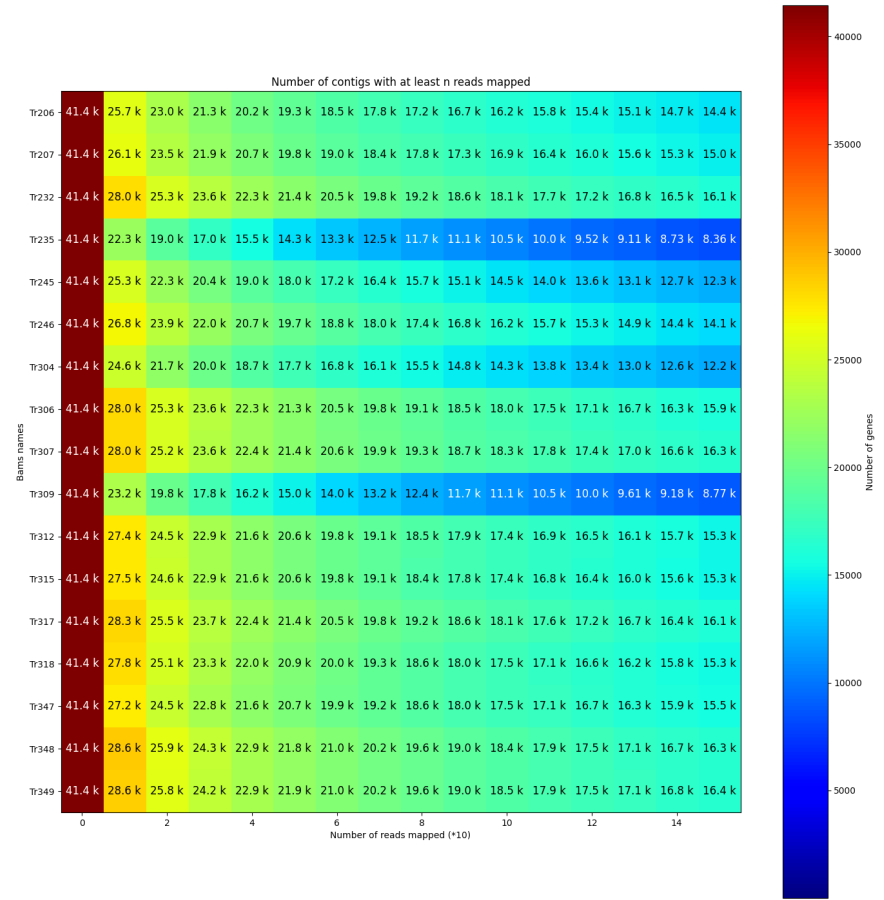


FIGURE 5.6 – Diagramme de gantt illustrant l’usage du temps pendant le stage. Les jours fériés ne sont pas affichés et les dates de début et de fin peuvent être légèrement décalées par rapport à la réalité. Figure générée avec [OnlineGantt](#).



(a) Résultat de l'analyse des BamTrEx (3.4.5).

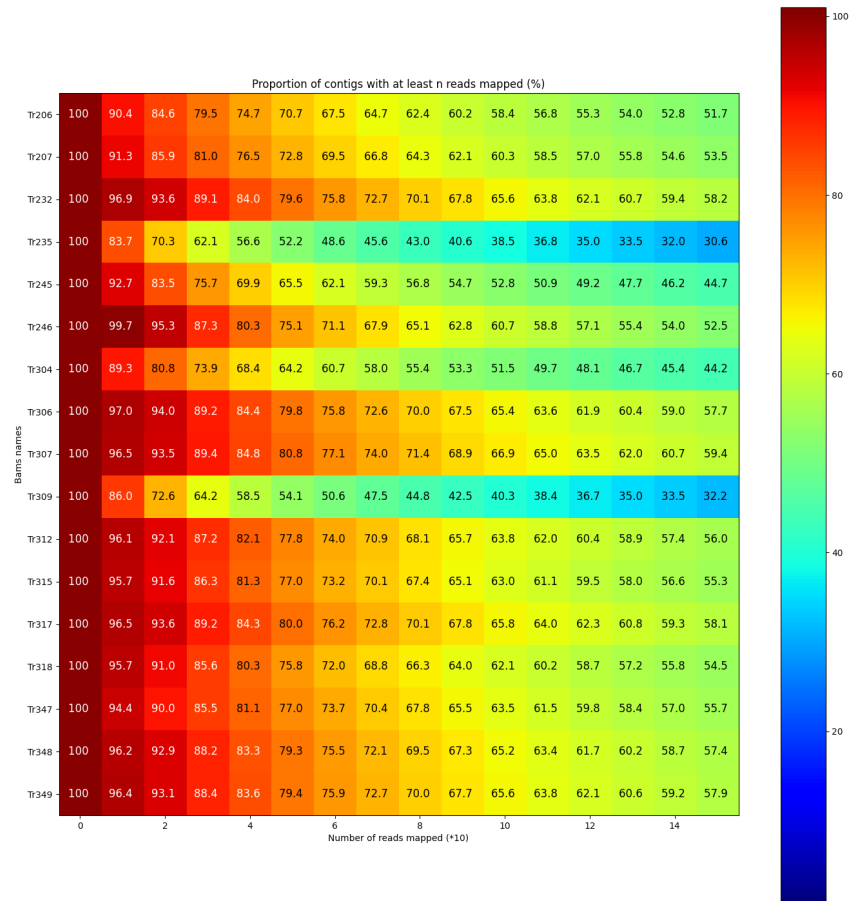
Commande : `Contig Number ./TargetedFiles.json -m 16 -kgqcuwt -j Classic --show_values -1 -y --transparent --legends legends.json --start_at_0`



(b) Résultat de l'analyse des BamTrMo (3.4.5).

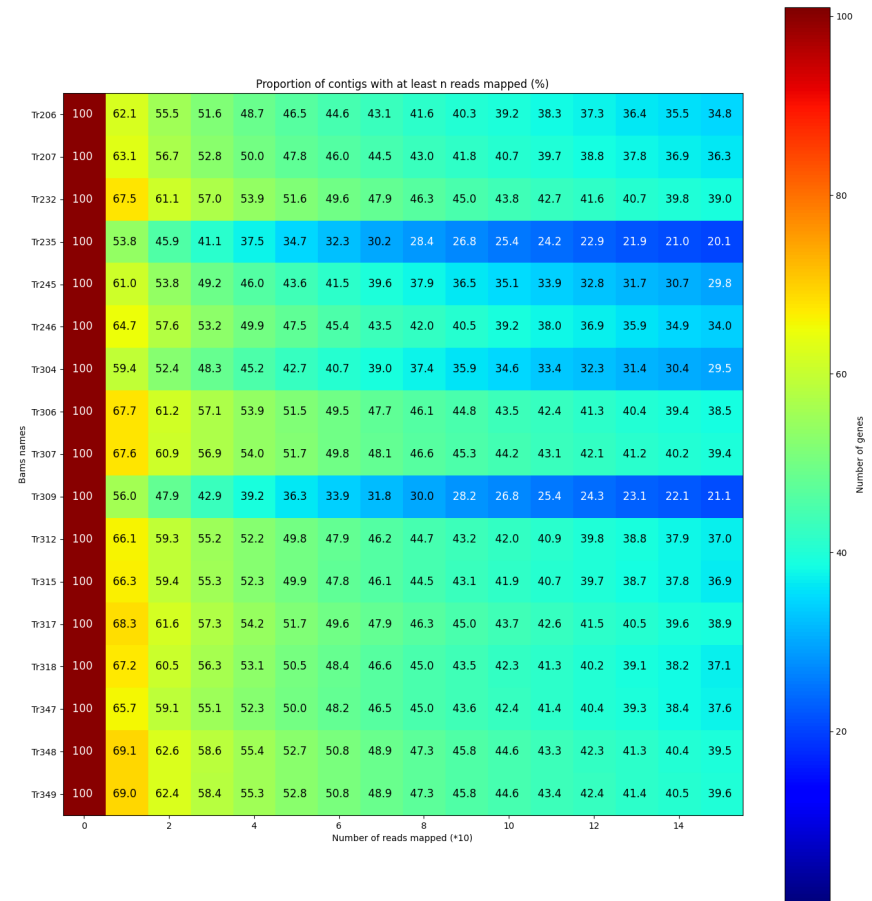
Commande : `Contig Number ./TargetedFiles.json -m 16 -kgqcuwt -j Classic --show_values -1 -y --transparent --legends legends.json --start_at_0`

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). Une incrémentation 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). Une incrémentation 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. 4, 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. 4, 5, 20

$\frac{D_n}{D_s}$ aussi appelé ω 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 substitutions sont probablement éliminées par la sélection naturelle. La significativité du résultat se calcule en utilisant un **test du khi deux**. 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

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

BamGeStar fichiers **.bam** générés avec STAR en utilisant GeMo. 15, 19, 24

BamTrEx fichiers **.bam** générés avec BWA en utilisant TrEx. 15–20, 24, 27, 28

BamTrMo fichiers **.bam** générés avec BWA en utilisant TrMo. 15–20, 24, 27, 28

GeMo génome de référence. (*Génome Moderne*). 2, 12, 14, 15, 24

GNSP= Nombre de gènes présentant au moins un certain nombre de SNP. 6, 7

GNSP \geq nombre de gènes présentant un certain nombre de SNP. 6–8

NbSNP Nombre de SNP. 6–8, 10–12

NGS Designe les technologies de **Séquençage** Nouvelle Generation. (*Next-Generation Sequencing*). 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, 24

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, 6, 7, 9–12, 24

TrEx **transcriptome** de référence créé par l'équipe. (*Transcriptome ex-nihilo*). 2, 12, 15–19, 24

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

Glossaire

- .bam* fichier de cartographie d'alignement binaire. Ce type de fichier est généralement créé à la suite d'un [mapping](#) ou de la compression d'un fichier *".sam"* (fichier de cartographie d'alignement de séquences). [2](#), [6](#), [15–19](#), [24](#), [25](#), [27](#), [28](#)
- .bashrc* fichier de configuration utilisé par les environnements [Bash](#). [13](#)
- .bz2* fichier compressé au format *bz2*. [13](#)
- .fastq* type de fichier texte permettant de stocker des séquences nucléotidiques (relatif aux [nucléotides](#)) et leur qualité. Contient généralement des [reads](#) issues d'un [séquençage](#). [2](#), [13](#), [14](#), [16](#), [17](#), [19](#)
- .gz* fichier compressé au format *gz*. [13](#)
- .json* fichier texte structuré permettant de stocker des informations. Ce type de fichier est souvent utilisé pour les fichiers de configuration. [6](#)
- .pdf* type de fichier standardisé correspondant à la norme ISO 32000. [9](#)
- .png* format d'image numérique supportant les fonds transparents. [8](#)
- .svg* format d'image numérique utilisant des vecteurs pour représenter une image. Ce type de fichier assure que l'image pourra être étendue sans perte de qualité . [8](#)
- .tsv* type de fichier texte représentant un tableau. Les données y sont séparées par des tabulations ("t"). [8](#), [18](#)
- README* fichier texte utilisé pour décrire une application aux utilisateurs. [10](#)
- pyproject.toml* fichier utilisé dans les projets [Python](#) pour préciser les prérequis et les caractéristiques d'une application. [10](#)
- ~~TeX~~ système permettant composition de documents de haute qualité utilisé pour la production de textes scientifiques et techniques [[Lat](#)]. Par de nombreux aspect, ~~TeX~~ est assimilable à un langage de programmation. [9](#), [25](#)
- allogame** "caractérisé par l'allogamie, par la fécondation du pistil d'une fleur par le pollen d'une autre fleur." [[Rob](#)]. [1–3](#)
- assemblage** utilisation des [reads](#) pour reconstruire la séquences originale dont ils sont issus. Ce processus donne naissance à des [contigs](#) et peut requérir une référence. On parle d'assemblage *ex-nihilo* si aucune référence n'est utilisée. [2](#)
- autogame** (ou autogamie) "mode de reproduction par union de gamètes provenant du même individu, observé surtout dans le règne végétal (algues, champignons...), plus rare dans le règne animal" [[Rob](#)]. [1–3](#)
- Bash** langage de programmation présent par défaut sur les systèmes d'exploitation Linux. [[bash](#)]. [25](#)
- bien appariés** ce dit de deux brins d'ADN, d'ARN, de [reads](#), ou de [contigs](#) lorsque les [nucléotides](#) qui les composent forment la séquence complémentaire d'une référence. [17](#), [18](#)
- BWA** [mappeur](#) efficace pour aligner de l'ARN sur des références de petite taille [[Li13](#)]. [15](#)
- cluster** ensemble d'ordinateur permettant de réaliser des calculs gourmands en ressource. [13](#), [25](#)
- CODEML** programme contenu dans le package [PAML](#) (cf. section [4.3.1](#), [[pam](#)]). [20–22](#), [25](#)

codon "triplet de [nucléotide](#) [...] qui détermine la synthèse cellulaire des acides aminés" [[Rob](#)]. 4, 20, 21

commit opération qui enregistre un ensemble de modifications apportées au code dans l'historique du projet. 9, 10

Conda gestionnaire de paquets et d'environnements virtuels isolés. [[Ana](#)]. 10, 12, 13, 25

contig séquence de [nucléotides](#) obtenue à partir de l'[assemblage](#) de [reads](#). Dans le cadre de ce rapport, comme nous travaillons sur des données [transcriptomiques](#), un [contig](#) correspond à un cadre ouvert de lecture. L'on assimilera donc "contigs" et "gènes". 3, 6–12, 17–20, 24, 27, 28

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

dNdSpNpS outil permettant de calculer les D_n , D_s , P_n , P_s de chaque [contig](#) [[dNd](#)]. 3

Doxygen Outil de documentation automatique [[Dox](#)]. 9, 25

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](#)]. 21–23, 25

fastqc outil d'analyse des fichiers [.bam](#), [.sam](#) et [.fastq](#). [[Fas](#)]. 16–18

GeCKO pipeline facilitant les analyses phylogénétiques (cf. section 3.1) ([[Ard+24](#)]). 12, 13, 15, 16, 25

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

Gimp Outil de modification d'image [[Gim](#)]. 7, 11, 20

Git système de contrôle de version distribué permettant de suivre les modifications d'un code source. [[Git](#)]. 25

GitHub plate-forme en ligne permettant de centraliser des dépôts [Git](#). [[Git24](#)]. 6, 9, 12, 22, 25

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

inflorescence "mode de groupement des fleurs d'une plante, ou groupe de fleurs" [[Dic](#)]. 1

json bibliothèque [Python](#) permettant d'exploiter des [.json](#) [[jso](#)]. 6

Lucidchart éditeur de diagrammes en ligne [[Luc](#)]. 14

mappeur logiciel permettant de réaliser des [mappings](#). 14, 15

mapping alignement de [reads](#) sur une séquence de référence. Cela permet d'identifier la position du [read](#) dans un génome ainsi que les différences qu'il présente par rapport à l'original (ex: [SNP](#)). 2, 12–17, 19, 20, 24

Matplotlib bibliothèque [Python](#) permettant de générer des graphiques 2D [[Hun07](#)]. 6, 8

Minimap2 [mappeur](#) efficace pour aligner de l'ARN sur de l'ADN [[Min](#)]. 14, 15

nucléotide "constituant élémentaire des acides nucléiques (ADN et ARN), formé par un nucléoside associé à un phosphate". [Rob] (modifiée). 2, 4, 20

OnlineGant logiciel en ligne permettant de faire des diagrammes de Gantt [Gan]. 26

orthologue désigne des séquences héritées d'un même ancêtre. 6, 20

os bibliothèque [Python](#) permettant de communiquer avec le système d'exploitation [OS]. 6

outgroup groupe d'espèces lié, mais extérieur aux espèces dont on souhaite construire la phylogénie. 2, 11, 20

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

phylogénie "étude de l'évolution des êtres vivants afin de déterminer leurs liens de parenté" [Dic]. 1, 2, 6, 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]. 9

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

Pytest bibliothèque [Python](#) permettant de tester les résultats des fonctions d'un projet [PyT]. 6, 9

Python langage de programmation grand public, [Py24]. 21, 22, 25

R langage de programmation utilisé pour les analyses statistiques [R]. 20, 25

read séquence de [nucléotide](#) obtenue directement à partir du [séquençage](#) de l'ADN ou de l'ARN. 2, 10, 13, 15–20, 27, 28

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

Singularity système 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 portabilité. 13

Slurm logiciel permettant de répartir l'exécution de programmes sur les différents nœuds d'un [cluster](#) [YJG03]. 12, 21, 25

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

STAR [mappeur](#) efficace pour aligner de l'ARN sur de l'ADN [Dob+13]. 15

substitution désigne, dans le contexte de ce stage, un changement entre les séquences [nucléotide](#) de deux espèces (section 1.2). 4, 5

synonyme dans le contexte de ce rapport, désigne les sites ou les mutations qui produisent le même acide aminé. En effet, deux [codons](#) différents peuvent produire le même acide aminé. A l'inverse, des sites "non synonymes" désignent les sites ou les mutations qui ne produisent pas le même acide aminé. 3–5

sys bibliothèque [Python](#) permettant d'accéder aux variables de l'interpréteur [Python](#) [Sys]. 6

séquençage détermination de l'ordre des [nucléotide](#) composant une molécule d'ADN ou d'ARN. 1, 2

test de Fisher test statistique permettant de déterminer si deux variables d'un tableau de contingence sont indépendantes. [21](#)

test de McDonald-Kreitman test utilisé pour distinguer les effets de la sélection naturelle des variations génétiques neutres. [20](#), [21](#)

test du khi deux test statistique permettant de déterminer si les différences observées entre les fréquences attendues et les fréquences observées sont significatives. [5](#), [21](#)

transcriptome "ensemble des molécules d'ARN résultant de l'expression d'une partie du génome d'un tissu cellulaire ou d'un type de cellule" [[Dic](#)]. [1](#), [2](#), [12](#), [15](#)

transcriptomique qui est relatif au [transcriptome](#). [1](#)

wrapper code informatique permettant d'encapsuler un logiciel tiers dans un autre logiciel. [22](#)

WSL sous système Linux pour Windows. [[22](#)]. [13](#), [25](#)

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](https://doi.org/10.1093/evlett/qrae039). URL : <https://doi.org/10.1093/evlett/qrae039> (visité le 17/08/2024).
- [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](https://doi.org/10.1111/1755-0998.12587). URL : <https://onlinelibrary.wiley.com/doi/10.1111/1755-0998.12587> (visité le 02/08/2024).
- [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](https://doi.org/10.1126/sciadv.aav9188). URL : <https://www.science.org/doi/10.1126/sciadv.aav9188> (visité le 02/08/2024).
- [Mar24a] Florent MARCHAL. *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).
- [Mar24b] Florent MARCHAL. *F-Marchal/SnpHeatMap*. original-date : 2024-05-04T14:18:46Z. Juill. 2024. URL : <https://github.com/F-Marchal/SnpHeatMap> (visité le 05/08/2024).
- [Mar24c] Florent MARCHAL. *F-Marchal/SnpHeatMap*. original-date : 2024-05-04T14:18:46Z. Juill. 2024. URL : <https://github.com/F-Marchal/SnpHeatMap> (visité le 02/08/2024).
- [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 ARDISON 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](https://doi.org/10.21203/rs.3.rs-4123643/v1).
- [Mar24d] Florent MARCHAL. *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](https://doi.org/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](https://doi.org/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).
- [Lat] LATEX. *LaTeX - A document preparation system*. URL : <https://www.latex-project.org/> (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).
- [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](https://doi.org/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).
- [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](https://doi.org/10.1111/1755-0998.13672).
- [Fas] FASTQC. *Babraham Bioinformatics - FastQC A Quality Control tool for High Throughput Sequence Data*. URL : <https://www.bioinformatics.babraham.ac.uk/projects/fastqc/> (visité le 19/08/2024).
- [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).
- [jso] JSON. *json — JSON encoder and decoder*. en. URL : <https://docs.python.org/3/library/json.html> (visité le 20/08/2024).
- [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](https://doi.org/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](https://doi.org/10.1007/10968987_3).
- [Möl+21] Felix MÖLDER et al. *Sustainable data analysis with Snakemake*. en. Jan. 2021. DOI : [10.12688/f1000research.29032.1](https://doi.org/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](https://doi.org/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).