Projet Bioinformatique : samReader

Raphaël Ribes

`https://github.com/RaphaelRibes/samReader`

RaphaelRibes/
samReader

The goal of the project is to extract relevant information from a mapping file (alignment of sequencing reads onto a...

1 Contributor | 0 Issues | 0 Stars | 0 Forks

1 Introduction

Depuis l'émergence des technologies de séquençage de nouvelle génération (NGS), le volume et la complexité des données de séquences ADN ont augmenté de manière exponentielle. Ces technologies, bien que révolutionnaires, génèrent des courts fragments de séquences ADN, appelés reads, qu'il est nécessaire d'aligner sur une séquence de référence pour les analyser efficacement. Afin de standardiser et de faciliter le stockage et la manipulation de ces données d'alignement, le format SAM (Sequence Alignment/Map) a été introduit par Heng Li et Bob Handsaker *et al.* en 2009[1].
Le format SAM offre une solution simple et flexible pour organiser les résultats d'alignement grâce à un fichier tabulé structuré, comprenant une section d'en-tête et une section de données d'alignement. Ce format, ainsi que son grand frère le format BAM, sont devenus indispensable pour des applications variées en biologie plus particulièrement en sciences omique.

Ce Programme vise à lire, analyser et extraire les informations essentielles des fichiers SAM. Le but est de faciliter l'interprétation des alignements, leur profondeur et leur qualité. Cet outil permet une analyse des données issues des technologies de NGS plus simple et compréhensible pour les chercheurs.

2 Présentation du format SAM

Le format SAM comporte une section d'en-tête optionnelle, constituée de lignes commençant par le caractère `@`. Chaque ligne représente un enregistrement spécifique grâce à un code à deux lettres, suivi de champs sous la forme `TAG:VALUE`. Les en-têtes courants incluent `@HD` pour les informations générales du fichier comme la version et l'ordre de tri, `@SQ` pour décrire les séquences de référence (nom, longueur, etc.), `@RG` pour les groupes de lectures (identifiant, échantillon, plateforme), et `@PG` pour tracer les programmes ayant traité les alignements. Enfin, les lignes `@CO` permettent d'ajouter des commentaires libres. Ces en-têtes jouent un rôle essentiel dans l'organisation et l'interprétation des données d'alignement.

Dans le format SAM, chaque ligne d'alignement représente typiquement l'alignement linéaire d'un segment. Chaque ligne est composée de 11 champs ou plus, séparés par des tabulations. Les onze premiers champs sont toujours présents et dans l'ordre suivant ; si l'information représentée par l'un de ces champs est indisponible, une valeur de substitution est utilisée : soit `'0'` soit `'*'`, en fonction du type du champ. Le tableau suivant (tableau 1) donne un aperçu de ces champs obligatoires dans le format SAM :

Col	Champ	Description succincte
1	QNAME	Nom de la requête ou du modèle (chromosome)
2	FLAG	Indicateurs binaires
3	RNAME	Nom de la séquence de référence
4	POS	Position de mappage la plus à gauche (indexée à 1)
5	MAPQ	Qualité du mappage
6	CIGAR	Chaîne CIGAR
7	RNEXT	Nom de la séquence de référence du segment suivant
8	PNEXT	Position du segment suivant
9	TLEN	Longueur observée du modèle
10	SEQ	Séquence du segment
11	QUAL	Qualité des bases

Tableau 1 – Champs obligatoires dans une ligne d'alignement SAM.

— **QNAME :** Nom de la requête ou du modèle. Les segments ayant le même QNAME sont supposés provenir du même modèle. Une valeur `'*'` indique que l'information est indisponible. Dans un fichier SAM, une lecture peut occuper plusieurs lignes d'alignement si son alignement est chimérique ou s'il existe plusieurs mappages.
— **FLAG :** Indicateurs combinés sous forme binaire. Chaque bit a une signification particulière décrit dans le tableau suivant (tableau 2).

Bit	Description
1	Modèle ayant plusieurs segments dans le séquençage
2	Chaque segment correctement aligné selon l'outil d'alignement
4	Segment non aligné
8	Segment suivant dans le modèle non aligné
16	SEQ est inversée complémentaire
32	SEQ du segment suivant est inversée complémentaire
64	Premier segment dans le modèle
128	Dernier segment dans le modèle
256	Alignement secondaire
512	Ne passe pas les filtres (contrôles de qualité)
1024	PCR ou duplicat optique
2048	Alignement supplémentaire

Tableau 2 – Description des valeurs de bits utilisées pour les alignements.

Par exemple `163` convertit en binaire donne `000010100011` ce qui signifie que le segment est aligné, est le dernier segment dans le modèle et que le prochain segment est aligné et inversé complémenté.
— **RNAME :** Nom de la séquence de référence de l'alignement. Si des lignes d'en-tête @SQ sont présentes, RNAME (si différent de `'*'`) doit être défini dans l'un des tags `SN` des lignes `@SQ`.
— **POS :** Position de mappage la plus à gauche, indexée à 1, de la première opération CIGAR qui "consomme" une base de la séquence de référence.
— **MAPQ :** Qualité du mappage, calculée comme $-10 \log_{10} Pr\{\text{la position de mappage est erronée}\}$, arrondie à l'entier le plus proche. Une valeur de 255 indique que la qualité n'est pas disponible.
— **CIGAR :** Chaîne CIGAR qui décrit l'alignement entre la séquence de la requête et la séquence de référence. Chaque opération (Op) est représentée par un chiffre suivi d'une lettre (tableau 3).

Op	Description	Consomme la requête	Consomme la référence
M	Correspondance/Discordance d'alignement	oui	oui
I	Insertion dans la référence	oui	non
D	Suppression de la référence	non	oui
N	Région ignorée dans la référence	non	oui
S	Recouvrement souple (séquences coupées présentes dans SEQ)	oui	non
H	Recouvrement dur (séquences coupées absentes dans SEQ)	non	non
P	Remplissage (suppression silencieuse de la référence remplie)	non	non
=	Correspondance de séquence	oui	oui
X	Discordance de séquence	oui	oui

Tableau 3 – Description des opérations d'alignement

"Consomme la requête" et "Consomme la référence" indiquent si l'opération CIGAR provoque l'avancement de l'alignement respectivement le long de la séquence de requête et de la séquence de référence (figure 1).

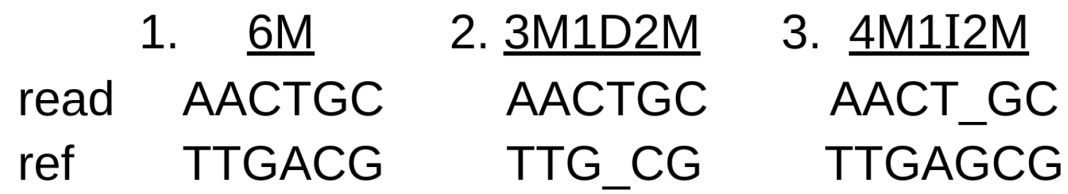

FIGURE 1 – Exemple d'alignement avec la chaîne CIGAR complètement alignée (1), partiellement alignée avec une délétion (2) et partiellement alignée avec une insertion (3).

— **RNEXT et PNEXT :** Indiquent respectivement le nom et la position de la séquence de référence du segment suivant dans le modèle.
— **TLEN :** Longueur observée du modèle. Positive pour le segment le plus à gauche, négative pour le segment le plus à droite. Il est défini à 0 pour un modèle à un seul segment ou lorsque l'information est indisponible.
— **SEQ :** Séquence du segment. Peut être `'*'` si la séquence n'est pas stockée.
— **QUAL :** Qualité des bases, encodée en ASCII.

3 Exemple d'application

L'utilisation de technologies de séquençage de NGS a considérablement révolutionné la recherche biologique, notamment en génétique, écologie et santé. Les séquences d'ADN issues de ces technologies permettent de mieux comprendre la diversité génétique au sein des populations, ainsi que l'identification des variations qui peuvent influencer des traits biologiques spécifiques. Par exemple, dans un contexte médical, la détection de polymorphismes (variations génétiques) pourrait aider à identifier des prédispositions à des maladies génétiques, ou à comprendre les mécanismes de résistance aux traitements[2].

En utilisant les technologies de NGS, on peut également mieux comprendre les processus d'évolution et de spéciation, en identifiant les différences génétiques qui caractérisent les différentes populations ou espèces. Cela permet de poser des hypothèses sur les mécanismes biologiques à l'origine de la variation génétique et de mieux appréhender les dynamiques évolutives[3].

Dans ce cadre, l'analyse des séquences d'ADN, et en particulier la détection des polymorphismes, est devenue un outil central pour étudier des questions biologiques fondamentales, telles que l'adaptation à l'environnement, les interactions entre les espèces, et les bases génétiques des maladies.

4 Description du programme (figure 2)

Le programme commence en premier avec le script `samReader.sh`, qui avant de lancer le programme, réalise plusieurs étapes préliminaires :

- La lecture et validation des options et arguments en ligne de commande.
- La vérification de la sélection d'une version de SAM.
- Si l'option –trusted est activée, le fichier est directement traité sans validation supplémentaire.
- La vérification du fichier d'entrée (existence, non-vide, conformité au format SAM).

Ensuite, le script `main.py` est exécuté. Le programme va charger les modules correspondant à la version de SAM sélectionnée. Il va ensuite créer un répertoire `temp` dans lequel il va stocker les fichiers temporaires nécessaires à la génération des rapports. Enfin, `samReader` va créer un répertoire avec le nom du fichier de sorti si précisé, ou en utilisant le nom du fichier d'entrée.

Pour chaque chromosome présent dans le fichier SAM, le programme va effectuer les étapes suivantes :

- Convertit la liste des reads en un tableau NumPy pour effectuer des opérations numériques de manière efficace.
- Calcule la longueur du chromosome en fonction de la position maximale et de la longueur de la chaîne CIGAR.
- Initialise deux tableaux NumPy, mapq et depth, remplis de zéros pour stocker respectivement les informations sur la qualité de mappage et la profondeur.
- Lance l'étape d'analyse préliminaire, qui va générer les statistiques générales sur les reads en créant des fichiers fasta comprenant les reads alignés, partiellement alignés et non alignés. Cette étape compte également la quantité de paires de reads alignées, partiellement alignées et non alignées.
- Lance l'analyse des mutations, qui va remplir les tableaux mapq et depth avec les informations sur la qualité de mappage et la profondeur ainsi que générer des statistiques globales sur les mutations présentes dans les reads.
- Créer les graphiques de profondeur et de qualité de mappage.
- Génère un rapport PDF contenant les informations sur les reads, les mutations, la qualité de mappage et la profondeur par chromosome.

Une fois que le programme à itérer sur tous les chromosomes, il va générer un rapport global contenant les informations sur les reads, les mutations ainsi qu'un graphique representation le ratio de reads mappés sur ceux non mappés et partiellement mappés.

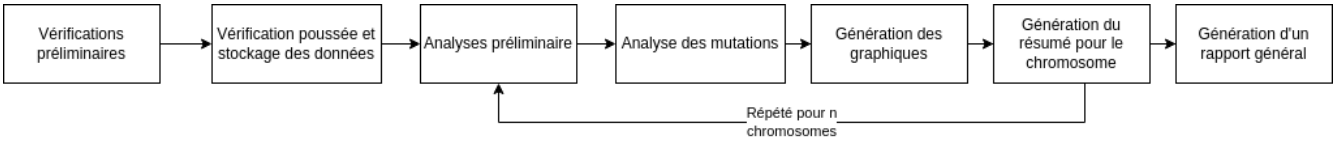

FIGURE 2 – Pipeline de traitement des données.

5 Discussion

Le projet `samReader` présente des points forts significatifs, notamment la capacité à générer des rapports détaillés et personnalisés. Ces rapports, produits sous forme de fichiers PDF, synthétisent les analyses de manière claire et exhaustive. L'analyse approfondie des mutations fournit des statistiques générales sur les anomalies présentes dans les reads. De plus, la segmentation des données par chromosomes améliore la lisibilité en regroupant les reads alignés, partiellement alignés et non alignés dans des répertoires distincts. L'évaluation de la profondeur de couverture et la représentation de l'évolution de la qualité d'alignement sont aussi des fonctionnalités précieuses pour les chercheurs. La détection des erreurs dans le fichier SAM est un atout majeur, permettant de garantir la fiabilité des données analysées.

Cependant, plusieurs limitations freinent l'efficacité du programme. La lenteur de son exécution constitue un obstacle majeur, en grande partie en raison de la génération des rapports et des graphiques, particulièrement via LaTeX. La représentation base par base de la qualité et de la profondeur de mappage est également problématique, limitant la précision des analyses. Par ailleurs, le traitement des données de qualité de mappage n'est pas optimal : lorsque deux reads se superposent, l'information précédente est écrasée, ce qui peut entraîner une perte de données essentielles.

Pour améliorer ces aspects, des pistes d'optimisation sont envisageables. La vitesse d'exécution pourrait être augmentée par une relecture plus efficace des reads et une implémentation dynamique des graphiques. Une meilleure gestion des données MAPQ est également recommandée, en stockant séparément les valeurs pour chaque base et en calculant ultérieurement une moyenne ou une médiane. L'implémentation des différentes versions de SAM reste encore perfectible. Bien que les idées soient présentes, leur réalisation nécessite des ajustements et des améliorations pour atteindre une architecture satisfaisante. Ces ajustements permettraient d'accroître la précision et la rapidité du programme tout en conservant sa flexibilité.

Références

[1] Heng Li et al. "The Sequence Alignment/Map format and SAMtools". In : *Bioinformatics* 25.16 (15 août 2009), p. 2078-2079. ISSN : 1367-4811, 1367-4803. DOI : 10.1093/bioinformatics/btp352. URL : https://academic.oup.com/bioinformatics/article/25/16/2078/204688 (visité le 16/12/2024).

[2] Elaine R. Mardis. "Next-Generation DNA Sequencing Methods". In : *Annual Review of Genomics and Human Genetics* 9.1 (1er sept. 2008), p. 387-402. ISSN : 1527-8204, 1545-293X. DOI : 10.1146/annurev.genom.9.081307.164359. URL : https://www.annualreviews.org/doi/10.1146/annurev.genom.9.081307.164359 (visité le 14/12/2024).

[3] Jay Shendure et Hanlee Ji. "Next-generation DNA sequencing". In : *Nature Biotechnology* 26.10 (oct. 2008), p. 1135-1145. ISSN : 1087-0156, 1546-1696. DOI : 10.1038/nbt1486. URL : https://www.nature.com/articles/nbt1486 (visité le 16/12/2024).