



# Identification et Analyse des Mutations Somatiques dans le Cancer à l'aide de Variant Calling

MBISD2

***Réalisé par :***

Khadija Ben Madane

Jamaa Sahraoui

Basma EL Barki

***Encadré par :***

Nouhaila Ennajih

Hasnaa Talimi

**14 janvier 2025**

## Table des matières

<b>1.</b>	<b>Introduction</b>	<b>8</b>
<b>2.</b>	<b>Problématique</b>	<b>8</b>
<b>3.</b>	<b>Spécifications Techniques : Logiciels Utilisés</b>	<b>9</b>
<b>4.</b>	<b>Le flux de travail</b>	<b>11</b>
<b>4.1.</b>	<b>Collecte et Préparation des Données :</b>	<b>12</b>
<b>4.1.1.</b>	<b>Description des données:</b>	<b>12</b>
<b>4.1.2.</b>	<b>Téléchargement des données avec SRA Toolkit :</b>	<b>13</b>
<b>4.2.</b>	<b>Contrôle de qualité des séquences</b>	<b>14</b>
<b>4.2.1.</b>	<b>Contrôle de qualité initial des séquences</b>	<b>14</b>
<b>4.2.2.</b>	<b>Application de Trimomatic</b>	<b>15</b>
<b>4.2.3.</b>	<b>Contrôle de qualité après Trimomatic</b>	<b>15</b>
<b>4.3.</b>	<b>Alignement des lectures sur le génome de référence</b>	<b>17</b>
<b>4.4.</b>	<b>Alignement des séquences</b>	<b>19</b>
<b>4.4.1.</b>	<b>Exécution de BWA MEM :</b>	<b>19</b>
<b>4.4.2.</b>	<b>Structure d'un fichier SAM :</b>	<b>19</b>
<b>4.5.</b>	<b>Conversion, tri et indexation des fichiers BAM</b>	<b>20</b>
<b>4.5.1.</b>	<b>Conversion de SAM vers BAM avec Samtools</b>	<b>20</b>
<b>4.5.2.</b>	<b>Tri et indexation des fichiers BAM (code et taille des fichiers)</b>	<b>20</b>
<b>4.5.3.</b>	<b>Indexation des fichiers BAM</b>	<b>21</b>
<b>4.6.</b>	<b>Indexation du fichier FASTA du génome de référence</b>	<b>21</b>
<b>4.7.</b>	<b>Ajouter ou Remplacer les Groupes et Indexer un fichier BAM</b>	<b>21</b>
<b>4.7.1.</b>	<b>Ajout des Read Groups</b>	<b>21</b>
<b>4.7.2.</b>	<b>Validation des groupes de read ajoutés</b>	<b>22</b>
<b>4.7.3.</b>	<b>Les statistiques de l'alignement</b>	<b>23</b>
<b>4.8.</b>	<b>Marquage des duplicats et indexation du fichier BAM</b>	<b>24</b>
<b>4.8.1.</b>	<b>Marquage des duplicats</b>	<b>24</b>
<b>4.8.2.</b>	<b>Indexation du fichier BAM d'alignement</b>	<b>25</b>
<b>4.9.</b>	<b>Variant Calling (Détection de variants)</b>	<b>25</b>
<b>4.9.1.</b>	<b>Téléchargement et installation de GATK :</b>	<b>25</b>

4.9.2.	Configuration de l'environnement Java:.....	26
4.9.3.	Détection des variants somatiques(variants calling) .....	27
4.10.	Filtrage des variants avec GATK (SelectVariants) .....	28
4.11.	Annotation des SNPs avec SnpEff.....	30
4.11.1.	Téléchargement et extraction de SnpEff .....	31
4.11.2.	Téléchargement du modèle de référence GRCh38.86 .....	31
4.11.3.	Annotation des SNPs avec SnpEff.....	31
5.	Interprétations des résultats.....	32
5.1.	Analyse des Variations Génétiques :.....	32
5.2.	Statistiques d'Alignement et de Cartographie pour les Échantillons de Séquences .....	33
6.	Visualisation et comparaison des resultats .....	34
6.1.	Installation de et Configuration de Pysam .....	35
6.1.1.	Installation de la bibliothèque.....	35
6.1.2.	Objectif de l'utilisation de pysam .....	35
6.2.	Analyse des fichiers VCF .....	36
6.2.1.	Lecture et fusion des fichiers VCF .....	36
6.2.2.	Extraction de la profondeur de lecture (DP) .....	37
6.3.	Filtrage des mutations.....	37
6.3.1.	Filtrage des mutations par profondeur de lecture (DP > 10) .....	37
6.3.2.	Filtrage des mutations ayant passé les critères de qualité (FILTER = PASS) .....	38
6.4.	Exploration des Données.....	38
6.4.1.	Inspection des colonnes et de la colonne INFO .....	38
6.4.2.	Extraction des noms de gènes depuis la colonne INFO .....	39
6.4.3.	Identification des mutations dans les gènes d'intérêt .....	40
6.4.4.	Calcul des fréquences des mutations par gène d'intérêt .....	41
6.4.5.	Analyse des types de mutations .....	41
6.5.	Visualisation des résultats .....	42
6.5.1.	Visualisation du nombre de mutations par gène .....	42
6.5.2.	Visualisation de la distribution de la profondeur de couverture (DP) ...	44

6.5.3.	Sélection des mutations du gène TP53 .....	45
6.5.4.	Visualisation de la distribution des positions des mutations .....	46
6.5.5.	Installation de matplotlib-venn pour visualisation.....	47
6.5.6.	Dessiner un diagramme de Venn pour visualiser les intersections.....	47
6.5.7.	Analyse et Visualisation des SNPs entre Fichiers VCF avec PCA et Heatmap .....	49
6.5.8.	Visualisation des SNPs avec Diagramme de Venn pour 2 ou 3 Fichiers	52
6.5.9.	Visualisation des SNPs avec Diagramme de Venn pour les Fichiers 3, 4 et 5 .....	53
6.5.10.	Analyse des SNPs avec un Diagramme UpSet à partir de Fichiers VCF	54
7.	Conclusion Générale .....	56
8.	Ressources et Références.....	57

## Table de figures

Figure 1 logo du google colab .....	1
Figure 2 logo de python .....	9
Figure 3 logo du NCBI .....	10
Figure 4 logo du BWA .....	10
Figure 5 logo de SAMtools .....	10
Figure 6 logo de Picard .....	10
Figure 7 logo du GATK .....	10
Figure 8 logo du LaTeX .....	11
Figure 9 Workflow bio-informatique de notre projet. ....	12
Figure 10 code d'installation des outils nécessaires. ....	13
Figure 11 Extraction des lectures FASTQ (fastq-dump) .....	14
Figure 12 Exécution du code FastQC pour les fichiers SRR14252108_1.fastq et SRR14252108_2.fastq.....	14
Figure 13 Exécution du code Trimomatic pour le nettoyage des séquences. ....	15
Figure 14 Résultats du contrôle de qualité après le traitement par Trimomatic( per base sequence quality).....	16
Figure 15 Résultats du contrôle de qualité après le traitement par Trimomatic( per sequence quality scores) .....	16
Figure 16 Résultats du contrôle de qualité après le traitement par Trimomatic (per sequence GC content).....	17
Figure 17 Les fichiers de sortie d'indexation de hg38 .....	18
Figure 18 Alignement de séquences avec BWA .....	19
Figure 19 le contenu de fichier SAM .....	20
Figure 20 Conversion du fichier SAM en fichier BAM à l'aide de SAMtools.....	20
Figure 21 Code pour le tri du fichier BAM avec Samtools.....	20
Figure 22 Code pour l'indexation du fichier BAM trié avec Samtools. ....	21
Figure 23 Ce code pour l'extraction du dictionnaire de référence.....	21
Figure 24 l'installation de picard .....	22
Figure 25 Ajout des groupes de lecture .....	22
Figure 26 Vérification de groupe de read de fichier BAM .....	22
Figure 27 les statistiques d'alignement d'un fichier. ....	24
Figure 28 Code pour le marquage des duplicats dans le fichier BAM à l'aide de Samtools. ...	25
Figure 29 Code pour l'indexation du fichier BAM .....	25
Figure 30 l'installation de l'outil GATK.....	26
Figure 31 l'installation du java .....	27
Figure 32 Code pour le variant calling avec GATK Mutect2 .....	28
Figure 33 la commande pour le filtrage des SNPs .....	29
Figure 34 la commande pour le filtrage des INDELS.....	30
Figure 35 Téléchargement et extraction de SnpEff .....	31

Figure 36 Téléchargement du modèle de référence GRCh38.86.....	31
Figure 37 Annotation des SNPs avec SnpEff.....	31
Figure 38 Aperçu de la Fin du Fichier VCF Annotations de Variants .....	32
Figure 39 L'installation de pysam.....	35
Figure 40 Code de lecture et fusion des fichiers VCF .....	36
Figure 41 Affichage des données annotées fusionnées.....	36
Figure 42 Code d'extraction de la profondeur de lecture (DP) .....	37
Figure 43 Affichage des résultats de filtrage des mutations .....	37
Figure 44 Code pour le filtrage des mutations ayant passé les critères de qualité .....	38
Figure 45 Vérification des colonnes disponibles dans le tableau filtered .....	39
Figure 46 Code pour l'extraction des noms de gènes .....	39
Figure 47 Code pour l'identification des mutations dans les gènes d'intérêt.....	40
Figure 48 Affichage des fréquences des mutations par gène d'intérêt.....	41
Figure 49 Code d'affichage de répartition des types de mutation.....	41
Figure 50 L'affichage de la profondeur de couverture moyenne par gène.....	42
Figure 51 Le code de l'affichage de nombre de mutations par gène .....	42
Figure 52 Représentation de nombre de mutations par gène sous forme de graphique en barres.....	43
Figure 53 Code pour la visualisation de la distribution de DP .....	44
Figure 54 La visualisation de la distribution de la profondeur de couverture sous forme d'un histogramme.....	44
Figure 55 Code pour la sélection des mutations associées au gène TP53 .....	45
Figure 56 Code d'affichage de distribution des positions des mutations .....	46
Figure 57 Histogramme de distribution des position des mutation sur le génome .....	46
Figure 58 l'installation de matplotlib-venn .....	43
Figure 59 code de génération du diagramme de venn .....	43
Figure 60 Heatmap : distribution des mutations par type et gène .....	44
Figure 61 Le nombre des SNPs extraits d'apres fichiers les VCF .....	45
Figure 62 Heatmap du distribution des SNPs .....	46
Figure 63 Visualisation de PCA des SNPs entre fichiers .....	47
Figure 64 code de creation de diagramme de Venn pour la comparaison des SNPs extraits	48
Figure 65 Diagramme de Venn des SNPs (fichiers 1,2 et 3) .....	49
Figure 66 Diagramme de Venn des SNPs (fichiers 3,4 et 5) .....	50
Figure 67 UpSet Plot des intersections de SNPs entre six échantillons extraits de fichiers VCF .....	51

## Liste des tableaux

Table 1 Résumé des Données de Séquençage Illumina MiSeq .....	13
Table 2 Comparaison des variation génétique.....	33
Table 3 Tableau des statiqtiques d'alignements .....	34

# 1. Introduction

Le cancer du côlon, également appelé cancer colorectal, est une pathologie majeure dans le domaine de l'oncologie, classée parmi les cancers les plus fréquents à l'échelle mondiale. Il représente la troisième cause de décès par cancer chez les hommes et la deuxième chez les femmes, selon les statistiques de l'Organisation mondiale de la santé (OMS). Ce type de cancer se développe à partir de polypes adénomateux ou de lésions précancéreuses qui subissent des altérations génétiques au fil du temps. Ces modifications génétiques peuvent résulter de mutations somatiques, de l'instabilité microsatellite (MSI) ou d'altérations épigénétiques, conduisant à une prolifération cellulaire incontrôlée. Les mutations somatiques, qui se produisent dans les cellules non germinales, jouent un rôle clé dans l'initiation et la progression du cancer du côlon. Ces mutations touchent souvent des gènes critiques, tels que les gènes suppresseurs de tumeurs (comme APC et TP53) et les oncogènes (comme KRAS). Ces altérations peuvent affecter les mécanismes régulateurs de la division cellulaire, de l'apoptose et de la réparation de l'ADN, contribuant ainsi à la carcinogenèse. Comprendre le paysage des mutations somatiques dans le cancer du côlon est essentiel pour plusieurs raisons :

- **Diagnostic précoce** : Identifier les biomarqueurs génétiques spécifiques peut améliorer le dépistage et la détection précoce.
- **Pronostic** : Les types et la charge mutationnelle peuvent fournir des informations sur l'évolution et la gravité de la maladie.
- **Thérapies ciblées** : De nombreuses mutations somatiques, comme celles affectant KRAS ou BRAF, sont associées à des réponses spécifiques à certaines thérapies, ce qui ouvre la voie à une médecine personnalisée.

Dans ce contexte, l'objectif de ce projet est de mener une analyse approfondie des mutations somatiques présentes dans des échantillons tumoraux de patients atteints de cancer du côlon. En utilisant des outils bio-informatiques avancés et des bases de données génétiques, nous cherchons à explorer la diversité des mutations, leur fréquence et leur impact fonctionnel potentiel. Ce rapport détaille les étapes méthodologiques suivies pour atteindre ces objectifs, en s'appuyant sur des approches robustes et éprouvées en bio-informatique et en génomique.

## 2. Problématique

Le cancer du côlon est un problème de santé publique majeur, souvent diagnostiqué à un stade avancé. Malgré les avancées dans les approches thérapeutiques, les taux de survie restent relativement faibles pour les patients présentant des métastases. Une compréhension approfondie des mutations somatiques récurrentes et de leur impact biologique est essentielle pour améliorer les outils diagnostiques et thérapeutiques. Cependant, l'identification précise de ces mutations dans un large ensemble de données



génomiques reste un défi en raison de la complexité des processus tumoraux et des biais technologiques.

### Questions de Recherche :

1. Quelles sont les mutations somatiques récurrentes dans le cancer du côlon ?
2. Quels gènes liés au cancer, tels que APC, KRAS, ou TP53, sont affectés par ces mutations ?
3. Quels sont les impacts fonctionnels potentiels de ces mutations sur les gènes et les protéines ?

## 3. Spécifications Techniques : Logiciels Utilisés

Dans le cadre de ce projet, plusieurs logiciels et outils ont été utilisés pour assurer la réalisation et l'analyse des données, chacun ayant un rôle spécifique dans la chaîne de traitement. Voici les principaux outils, leur description et leurs illustrations :

### 3.1. Google Colab:



Figure 1 logo du google colab

Une plateforme de notebooks en ligne permettant d'exécuter du code Python dans un environnement cloud. Elle a été utilisée pour le traitement des données de séquençage, le prétraitement, l'alignement, et l'analyse des résultats, grâce à son interface interactive et sa compatibilité avec les bibliothèques scientifiques.

### 3.2. Python:



Figure 2 logo de python

Le langage de programmation principal utilisé pour automatiser les étapes d'analyse des données, incluant le contrôle qualité avec fastQC, le nettoyage des données avec fastp, et la visualisation des résultats. Les bibliothèques comme pandas, matplotlib, et seaborn ont été utilisées pour manipuler les données et générer des graphiques.

### 3.3. NCBI:



Figure 3 logo du NCBI

Une base de données internationale d'accès aux données de séquençage. Elle a servi de source pour le téléchargement des fichiers FASTQ contenant les lectures brutes des échantillons tumoraux. Les outils associés, comme SRA Toolkit, ont facilité l'extraction et la conversion des données.

### 3.4. BWA (Burrows-Wheeler Aligner):



Figure 4 logo du BWA

Un logiciel d'alignement de séquences permettant de mapper les lectures nettoyées sur le génome humain de référence (hg38). Il a joué un rôle central dans l'étape d'alignement.

### 3.5. SAMtools:



Figure 5 logo de SAMtools

Samtools est un ensemble d'outils pour manipuler et analyser les fichiers SAM et BAM de séquençage génétique.

### 3.6. Picard:



Figure 6 logo de Picard

Picard est un ensemble d'outils bio-informatiques pour manipuler et traiter les fichiers de séquençage génétique (BAM, SAM), utilisé pour des tâches comme la déduplication et l'amélioration de la qualité des données.

### 3.7. GATK:



Figure 7 logo du GATK

Le GATK (Genome Analysis Toolkit) est un ensemble d'outils développé par le Broad Institute pour l'analyse des données génomiques issues du séquençage. Il permet principalement de détecter et traiter les variations génétiques (SNPs, indels) et de réaliser des prétraitements comme le recalibrage des données.

### 3.8. LaTeX:



*Figure 8 logo du LaTeX*

Un système de composition de documents utilisé pour la rédaction de ce rapport. Il offre une mise en page professionnelle et des fonctionnalités avancées pour créer des documents structurés et esthétiques.

Chaque outil a contribué à l'accomplissement des différentes étapes méthodologiques du projet, garantissant des résultats fiables et exploitables pour l'analyse des mutations somatiques dans le cancer du côlon.

## 4. Le flux de travail

Le workflow bio-informatique utilisé dans ce projet suit une série d'étapes standard pour l'analyse des données de séquençage issues du cancer colorectal. La Figure 9 illustre les différentes étapes, depuis le contrôle qualité des séquences brutes jusqu'à l'annotation des variants.

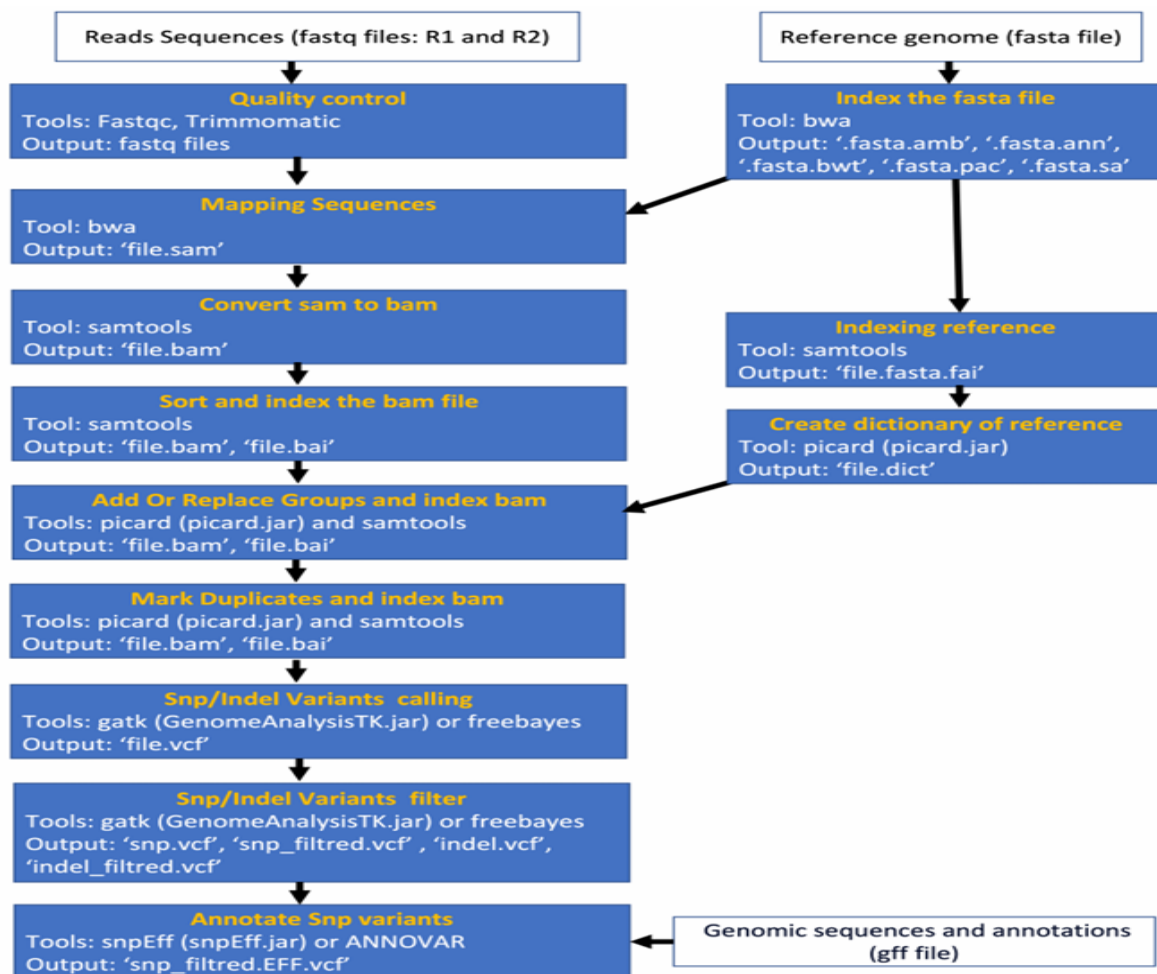


Figure 9 Workflow bio-informatique de notre projet.

## 4.1. Collecte et Préparation des Données :

### 4.1.1. Description des données:

Les données utilisées dans ce projet proviennent de séquençages d'échantillons tumoraux de cancer du côlon accessibles via le NCBI SRA (Sequence Read Archive). Des fichiers au format FASTQ ont été téléchargés pour les échantillons tumoraux. Aucune donnée normale correspondante n'a été incluse dans cette étude.

- **GSM5243673** : Échantillon tumoral de côlon provenant du patient T94.
- **Organisme** : Homo sapiens
- **Instrument** : Illumina MiSeq
- **Stratégie** : Séquençage génomique ciblé

#### Protocole de construction :

- ✚ L'ADN a été extrait des échantillons tumoraux FFPE à l'aide du kit QIAmp DNA FFPE (Qiagen, Allemagne) et quantifié par un test fluorométrique Qubit (Thermo Fisher Scientific, USA).

- Les bibliothèques enrichies ont été préparées à partir de 50 ng d'ADN par échantillon tumoral à l'aide du kit SMARTer ThruPLEX DNA-seq Library Preparation (TaKaRa, Japon) et d'une sonde de capture préconçue IDT xGen Lockdown probes (IDT, USA).
- Les séquençages ont été effectués sur la plateforme MiSeq d'Illumina avec une lecture appariée de 2x130 pb.

### Données expérimentales :

- Nombre de spots : 6 millions
- Nombre de bases: 1,6 milliards
- Taille totale : 663,1 Mo

Ce tableau regroupe les principales informations sur chaque échantillon, telles que l'identifiant, l'organisme, la stratégie de séquençage, les paramètres techniques, et les statistiques de run.

ID de l'Échantillon	GEO Accession	Organisme	Instrument	Stratégie	Source	Sélection	Disposition	Nombre de spots	Nombre de bases	Taille	Date de publication	ID Run
T92	GSM5243672	Homo sapiens	Illumina MiSeq	OTHER	GENOMIC	other	PAIRED	2.7M	708.3M	295.5Mb	03/12/2021	SRR14252111
T86	GSM5243669	Homo sapiens	Illumina MiSeq	OTHER	GENOMIC	other	PAIRED	4.6M	1.2G	517.2Mb	03/12/2021	SRR14252108
T71	GSM5243663	Homo sapiens	Illumina MiSeq	OTHER	GENOMIC	other	PAIRED	4.7M	1.2G	514.8Mb	03/12/2021	SRR14252102
T94	GSM5243673	Homo sapiens	Illumina MiSeq	OTHER	GENOMIC	other	PAIRED	6.0M	1.6G	663.1Mb	03/12/2021	SRR14252112
T09	GSM5243676	Homo sapiens	Illumina MiSeq	OTHER	GENOMIC	other	PAIRED	4.5M	1.2G	508Mb	03/12/2021	SRR14252115
FRH62-T_16S	-	Coral metagenome	Illumina MiSeq	AMPLICON	METAGENOMIC	PCR	PAIRED	192,335	117.5M	66.8Mb	16/04/2021	SRR14252197

Table 1 Résumé des Données de Séquençage Illumina MiSeq

### 4.1.2. Téléchargement des données avec SRA Toolkit :

#### ❖ Mise à jour et installation de SRA Toolkit :

Configuration des paramètres pour permettre le téléchargement depuis NCBI SRA.

```
!apt-get update
!apt-get install -y sra-toolkit bwa samtools fastqc
!pip install matplotlib seaborn pandas
```

Figure 10 code d'installation des outils nécessaires.

#### ❖ Téléchargement des données brutes :

Utilisation de la commande prefetch pour télécharger les fichiers SRA associés à l'identifiant SRR14252115.

### ❖ Extraction des lectures au format FASTQ :

- Commande pour convertir les fichiers SRA en fichiers FASTQ.
- Utilisation de l'option --split-files pour séparer les lectures en deux fichiers distincts (paired-end).

```
!fastq-dump --split-files /content/SRR14252115/SRR14252115.sra  
  
Read 4464831 spots for /content/SRR14252115/SRR14252115.sra  
Written 4464831 spots for /content/SRR14252115/SRR14252115.sra
```

Figure 11 Extraction des lectures FASTQ (fastq-dump)

## 4.2. Contrôle de qualité des séquences

### 4.2.1. Contrôle de qualité initial des séquences

Dans cette première étape, le contrôle de qualité des séquences brutes a été effectué afin d'évaluer la qualité des données avant tout traitement. Le contrôle qualité des séquences brutes est essentiel pour identifier d'éventuels problèmes tels que la présence d'adaptateurs, des bases de mauvaise qualité ou des erreurs de séquençage.

Pour cela, l'outil FastQC a été utilisé pour analyser les fichiers de séquences. FastQC génère un rapport détaillant plusieurs critères de qualité, tels que la distribution des scores de qualité par base, les séquences ayant un faible score de qualité, la présence d'adaptateurs, et bien plus encore. Ce rapport permet de visualiser rapidement la qualité des séquences brutes et d'identifier les problèmes potentiels avant de procéder à des étapes de nettoyage.

```
[ ] !fastqc /content/SRR14252108_1.fastq  
  
Started analysis of SRR14252108_1.fastq  
Approx 5% complete for SRR14252108_1.fastq  
Approx 10% complete for SRR14252108_1.fastq  
Approx 15% complete for SRR14252108_1.fastq  
Approx 20% complete for SRR14252108_1.fastq  
Approx 25% complete for SRR14252108_1.fastq  
Approx 30% complete for SRR14252108_1.fastq  
Approx 35% complete for SRR14252108_1.fastq  
Approx 40% complete for SRR14252108_1.fastq  
Approx 45% complete for SRR14252108_1.fastq  
Approx 50% complete for SRR14252108_1.fastq  
Approx 55% complete for SRR14252108_1.fastq  
Approx 60% complete for SRR14252108_1.fastq  
Approx 65% complete for SRR14252108_1.fastq  
Approx 70% complete for SRR14252108_1.fastq  
Approx 75% complete for SRR14252108_1.fastq  
Approx 80% complete for SRR14252108_1.fastq  
Approx 85% complete for SRR14252108_1.fastq  
Approx 90% complete for SRR14252108_1.fastq  
Approx 95% complete for SRR14252108_1.fastq  
Analysis complete for SRR14252108_1.fastq  
  
[ ] !fastqc /content/SRR14252108_2.fastq  
  
Started analysis of SRR14252108_2.fastq  
Approx 5% complete for SRR14252108_2.fastq  
Approx 10% complete for SRR14252108_2.fastq  
Approx 15% complete for SRR14252108_2.fastq  
Approx 20% complete for SRR14252108_2.fastq  
Approx 25% complete for SRR14252108_2.fastq  
Approx 30% complete for SRR14252108_2.fastq  
Approx 35% complete for SRR14252108_2.fastq  
Approx 40% complete for SRR14252108_2.fastq  
Approx 45% complete for SRR14252108_2.fastq  
Approx 50% complete for SRR14252108_2.fastq  
Approx 55% complete for SRR14252108_2.fastq  
Approx 60% complete for SRR14252108_2.fastq  
Approx 65% complete for SRR14252108_2.fastq  
Approx 70% complete for SRR14252108_2.fastq  
Approx 75% complete for SRR14252108_2.fastq  
Approx 80% complete for SRR14252108_2.fastq  
Approx 85% complete for SRR14252108_2.fastq  
Approx 90% complete for SRR14252108_2.fastq  
Approx 95% complete for SRR14252108_2.fastq  
Analysis complete for SRR14252108_2.fastq
```

Figure 12 Exécution du code FastQC pour les fichiers SRR14252108\_1.fastq et SRR14252108\_2.fastq

### 4.2.2. Application de Trimomatique

Après avoir effectué le contrôle de qualité initial, nous avons appliqué l'outil **Trimomatique** pour nettoyer les séquences. Trimomatique permet de retirer les bases de mauvaise qualité et les adaptateurs présents dans les séquences. Cela permet d'améliorer la précision des analyses ultérieures, telles que l'assemblage ou l'alignement des séquences. Ce nettoyage est particulièrement important pour éviter les biais introduits par des séquences de mauvaise qualité.

```
java -jar Trimomatic-0.39/trimomatic-0.39.jar \
PE -phred33 \
/content/dedup_SRR14252112_1.fastq /content/dedup_SRR14252112_2.fastq \
/content/drive/MyDrive/bioinformatics_project1/output_forward_unpaired-fq.gz \
/content/drive/MyDrive/bioinformatics_project1/output_reverse_unpaired-fq.gz \
/content/drive/MyDrive/bioinformatics_project1/output_forward_unpaired-fq.gz \
/content/drive/MyDrive/bioinformatics_project1/output_reverse_unpaired-fq.gz \
ILLUMINACLIP:Trimomatic-0.39/adapters/TruSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36

TrimomaticPE: Started with arguments:
-phred33 /content/dedup_SRR14252112_1.fastq /content/dedup_SRR14252112_2.fastq /content/drive/MyDrive/bioinformatics_project1/output_forward_unpaired-fq.gz /content/drive/MyDrive/bioinformatics_project1/output_reverse_unpaired-fq.gz /content/drive/MyDrive/bioinformatics_project1/output_forward_unpaired-fq.gz /content/drive/MyDrive/bioinformatics_project1/output_reverse_unpaired-fq.gz
Multiple cores found: Using 2 threads
Using PrefixPair: "TACACTCTTCTCCACAGAGCTCTCCGATCT" and "GTGACTGGAGTTCAGAGCTGTGCTCTCCGATCT"
ILLUMINACLIP: Using 1 prefix pairs, 0 forward/reverse sequences, 0 forward only sequences, 0 reverse only sequences
Input Read Pairs: 1501382 Both Surviving: 1427800 (95.10%) Forward Only Surviving: 60935 (4.06%) Reverse Only Surviving: 12018 (0.80%) Dropped: 629 (0.04%)
TrimomaticPE: Completed successfully
```

Figure 13 Exécution du code Trimomatique pour le nettoyage des séquences.

### 4.2.3. Contrôle de qualité après Trimomatique

Une fois que les séquences ont été traitées par Trimomatique, un nouveau contrôle qualité a été effectué pour évaluer l'impact du nettoyage sur les données. Le processus a été similaire à celui du contrôle initial, en utilisant à nouveau FastQC pour générer des rapports de qualité sur les séquences nettoyées.

Le rapport de qualité généré par FastQC après le nettoyage des séquences par Trimomatique a montré une amélioration notable des scores de qualité, avec des bases de plus haute qualité et l'élimination des adaptateurs.

Voici un exemple du fichier HTML généré par FastQC après le traitement des séquences par Trimomatique. La figure ci-dessous montre les résultats du contrôle de qualité après trimming, indiquant l'amélioration de la qualité des séquences.

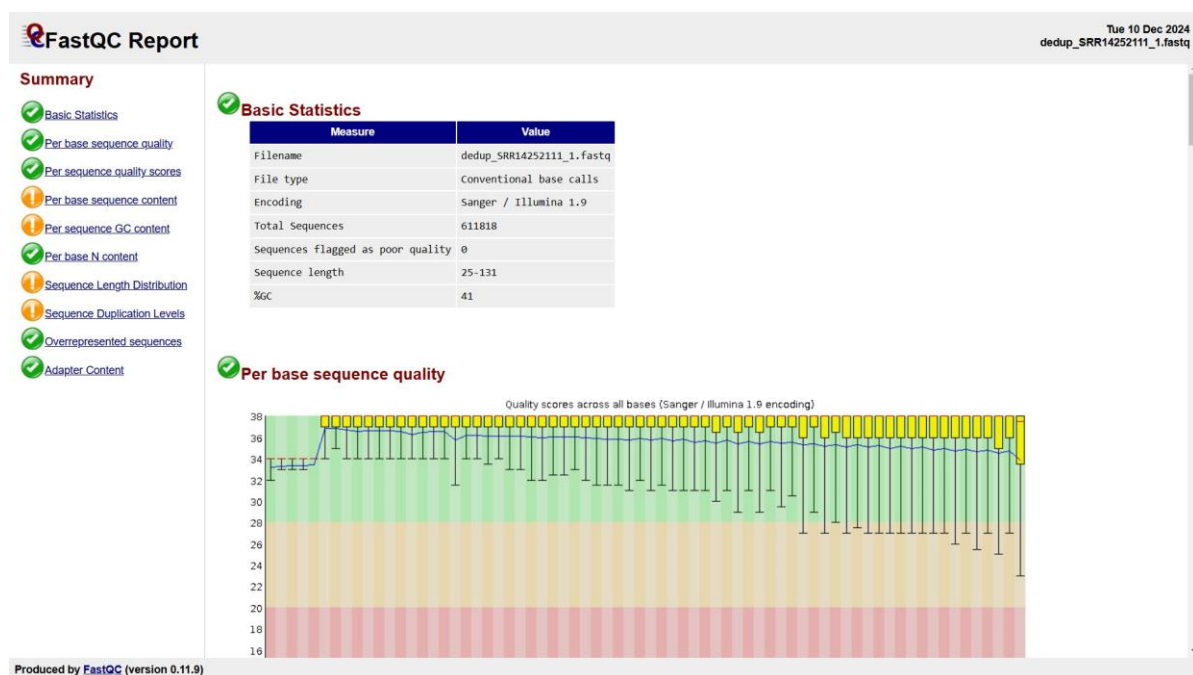


Figure 14 Résultats du contrôle de qualité après le traitement par Trimomatique( per base sequence quality)

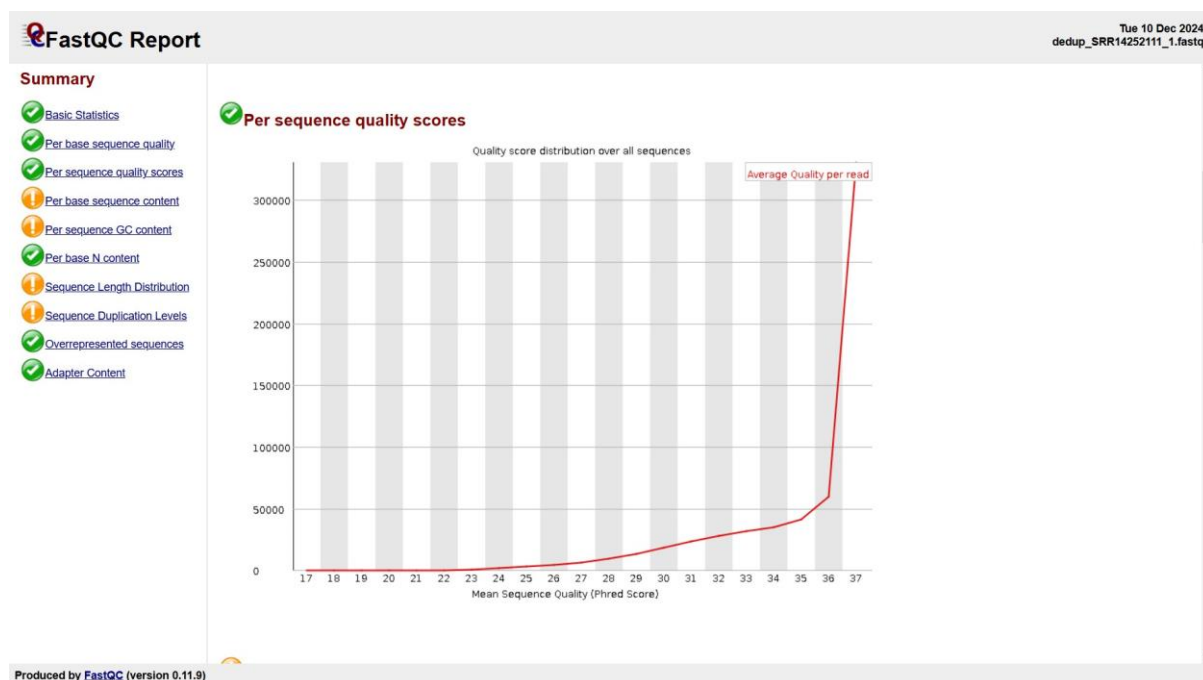


Figure 15 Résultats du contrôle de qualité après le traitement par Trimomatique( per sequence quality scores)



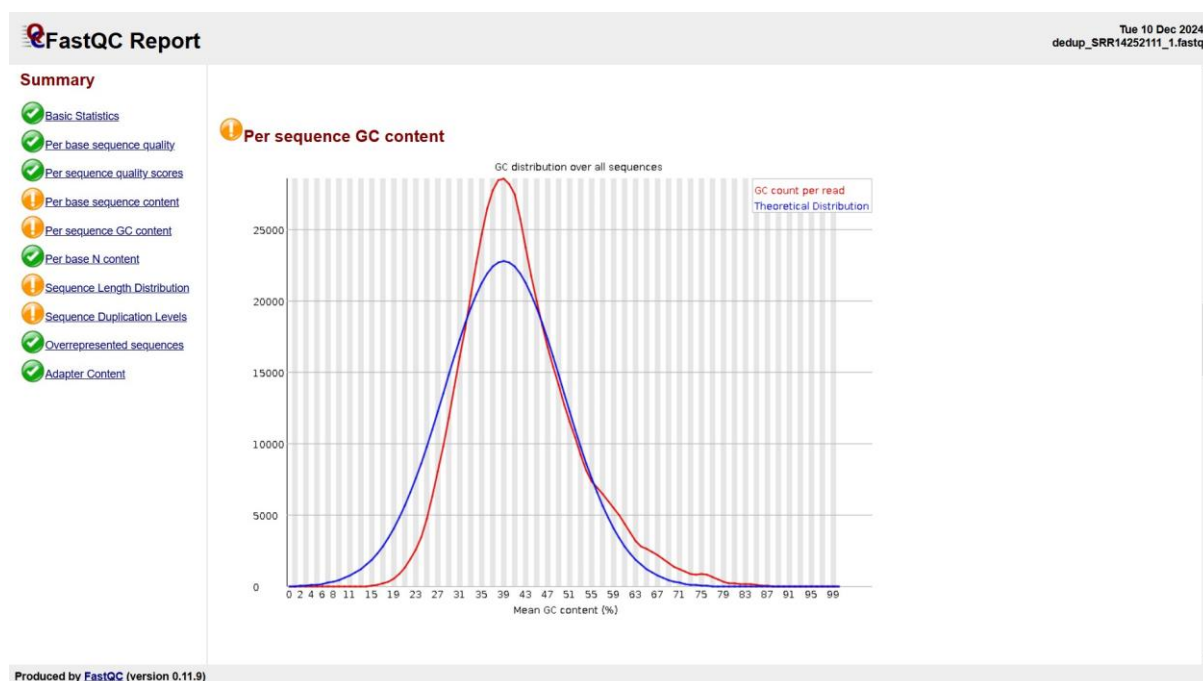


Figure 16 Résultats du contrôle de qualité après le traitement par Trimomatique (per sequence GC content)

### 4.3. Alignement des lectures sur le génome de référence

Une fois les données de séquençage nettoyées, l'étape suivante consiste à aligner les lectures sur le génome humain de référence (hg38). Cette étape est cruciale pour identifier l'emplacement des mutations et pour permettre une analyse des variants somatiques. L'outil utilisé pour cette tâche est BWA (Burrows-Wheeler Aligner), un des logiciels les plus populaires pour l'alignement de séquences courtes en raison de sa rapidité et de sa précision. BWA applique un algorithme basé sur la transformation de Burrows-Wheeler (BWT) pour effectuer l'alignement des lectures. Ce processus se déroule en plusieurs étapes :

#### Construction de l'index du génome de référence :

Avant l'alignement proprement dit, BWA construit un index du génome humain de référence (hg38) pour permettre une recherche rapide des correspondances lors de l'alignement. Cette étape est effectuée une seule fois et génère un fichier indexé pour une utilisation ultérieure. Ce fichier indexé est essentiel pour accélérer le processus d'alignement.

**Résultat :** Fichiers indexés générés dans le même répertoire : hg38.fa.amb, hg38.fa.ann, hg38.fa.bwt, hg38.fa.pac, hg38.fa.sa. Comme il montre la figure17.

	hg38.fa.sa
	hg38.fa.pac
	hg38.fa.fai
	hg38.fa.bwt
	hg38.fa.ann
	hg38.fa.amb

Figure 17 Les fichiers de sortie d'indexation de hg38

#### 1. **hg38.fa.amb**

- Contient une table d'ambiguïté pour les séquences ambiguës dans le génome.
- Utilisé pour gérer les bases ambiguës telles que "N".

#### 2. **hg38.fa.ann**

- Fichier annexe décrivant les annotations associées aux séquences du génome.
- Contient des métadonnées sur les chromosomes ou séquences dans le fichier FASTA.

#### 3. **hg38.fa.bwt**

- Index basé sur la transformation de Burrows-Wheeler (BWT).
- Permet une recherche rapide des séquences lors de l'alignement.

#### 4. **hg38.fa.pac**

- Fichier PAC contenant une représentation compressée des séquences du génome.
- Utilisé conjointement avec BWT pour accélérer les alignements.

#### 5. **hg38.fa.sa**

- Table de suffixes (Suffix Array).
- Permet une recherche efficace des correspondances exactes dans le génome.

## 4.4. Alignement des séquences

### 4.4.1. Exécution de BWA MEM :

BWA-MEM (Burrows-Wheeler Aligner Maximal Exact Matches) est l'un des algorithmes les plus performants pour aligner des lectures de séquençage sur un génome de référence. Il est spécialement conçu pour :

- Les longues lectures (>70 pb), mais fonctionne également bien pour des lectures courtes.
- La gestion des lectures appariées (paired-end) et des lectures non appariées (single-end).
- La détection des alignements secondaires et des insertions/délétions (indels).

La commande illustrée dans la figure 7 a été utilisée pour aligner les lectures nettoyées sur le génome humain de référence hg38.fa. Quatre threads (-t 4) ont été alloués pour accélérer le processus.

```
!bwa mem -t 4 \  
/content/drive/MyDrive/bioinformatics_project/hg38.fa \  
/content/drive/MyDrive/bioinformatics_project/forward_repaired.fq.gz \  
/content/drive/MyDrive/bioinformatics_project/reverse_repaired.fq.gz \  
> /content/drive/MyDrive/bioinformatics_project/aligned.sam
```

Figure 18 Alignement de séquences avec BWA

### 4.4.2. Structure d'un fichier SAM :

Un fichier SAM se compose de deux parties : l'en-tête et les données d'alignement. L'en-tête, marqué par des lignes commençant par @, contient des informations générales sur l'alignement, telles que les séquences de référence (@SQ) et les groupes de lecture (@RG). La section des alignements représente chaque lecture par une ligne tabulée avec des champs clés.

QNAME identifie la lecture, FLAG décrit ses propriétés (orientation, état apparié), et RNAME et POS indiquent la séquence de référence et la position de l'alignement. MAPQ donne le score de qualité, et CIGAR décrit les opérations d'alignement (appariements, insertions, suppressions). Les champs SEQ et QUAL fournissent respectivement la séquence nucléotidique et les scores de qualité Phred. Des champs optionnels, comme le nombre de mismatches (NM) ou le score d'alignement (AS), ajoutent des détails supplémentaires.

Ce format compact et standardisé est essentiel pour analyser et visualiser les alignements, ainsi que pour détecter des variants.

SRR14252097.3	81	chr3	47046512	60	48M	chr14	44847160	0	GGTGCAACTATTGTAGTCACTGCTGCGGCTGGCTGTACCACCACTCCT	<@<<9FCC,
SRR14252097.3	161	chr14	44847160	60	49M	chr3	47046512	0	CGATTAGTCAAATTATGAATCATTTCACATTTAAACACTGTTGTAAA	-8,8B,;CF
SRR14252097.4	99	chr14	16117820	0	131M	=	16117966	241	AGAGTTGGAACCTTGTTTCATTGAGCAGTTTGGCAACAAGTCATTTGTAGAATCTGCAAGGGG	
SRR14252097.4	147	chr14	16117966	0	95M	=	16117820	-241	CTTTGTGATGTTTGCAATTCATCTCACTGAGTTGAACCTTTCTATTCAATTAAGCAGTGTGGAATCA	
SRR14252097.8	97	chr6	152133258	60	80M32S	chr12	130137293	0	TGCTGTTTATTGCTTACCTGCTGACTACTTGACACGCTCTAATAACTTCTCCAGTTCCTTGATAT	
SRR14252097.8	2145	chr12	130137293	60	71H41M	=	130137293	105	TGACCTCCTGGCAGCACAACGACTGAATAACAATTAACA	FC,C,<F0EF808,88F
SRR14252097.8	145	chr12	130137293	60	26S105M	chr6	152133258	0	CTTCTCCAGTTCCTTGATATGACGACTGACCTCCTGGCAGCACAACGACTGAATAACAATTAAC	
SRR14252097.8	2193	chr6	152133303	60	35M96H	=	152133258	-80	CTTCTCCAGTTCCTTGATATGACGACTGACCTCCT	?,,?,?GF8A8<,9EGAF@GF:??
SRR14252097.14	99	chr4	125416459	60	46S85M	=	125416459	116	TGTCGCTTAAGTCTGGAGCAACACAGCTCAATAAATTAGGTGGCATAGATGATCGAGGATCT	
SRR14252097.14	2163	chr1	102992328	60	79H52M	chr4	125416459	0	TCATCTATGCCACCTAATTTATTGAGCTGTGTGCTCCAGACTTAAGCGACA	FCFFFGGG

Figure 19 le contenu de fichier SAM

## 4.5. Conversion, tri et indexation des fichiers BAM

Après avoir effectué l'alignement des séquences contre un génome de référence, les fichiers de sortie sont généralement au format **SAM** (Sequence Alignment/Map). Cependant, pour une manipulation plus efficace et une utilisation optimale dans les étapes suivantes, il est souvent nécessaire de convertir ces fichiers **SAM** en fichiers **BAM** (Binary Alignment/Map), qui sont plus compacts et permettent une gestion plus rapide.

### 4.5.1. Conversion de SAM vers BAM avec Samtools

La première étape consiste à convertir les fichiers **SAM** en **BAM** en utilisant l'outil **Samtools**, qui est couramment utilisé pour manipuler les fichiers de données génomiques. La commande pour cette conversion est la suivante (figure 20):

```
[ ] !samtools view -Sb /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/aligned.sam
> /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/aligned.bam
```

Figure 20 Conversion du fichier SAM en fichier BAM à l'aide de SAMtools.

### 4.5.2. Tri et indexation des fichiers BAM (code et taille des fichiers)

Une fois les fichiers convertis en format **BAM**, la prochaine étape consiste à trier les fichiers. Le tri est nécessaire pour organiser les alignements selon les positions dans le génome de référence, ce qui facilite les analyses ultérieures telles que l'assemblage ou la détection de variants. Le tri peut être effectué en utilisant **Samtools** avec la commande suivante (figure 21):

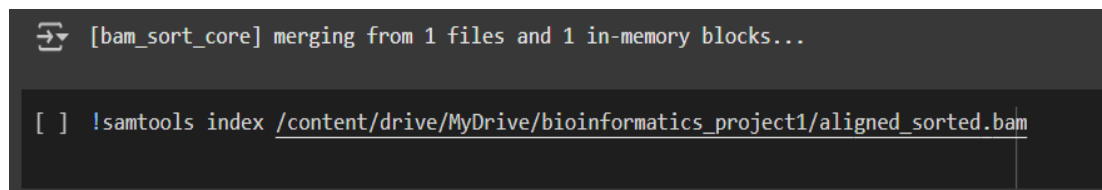
```
[ ] !samtools sort /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/aligned.bam
-o /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/aligned_sorted.bam

[bam_sort_core] merging from 1 files and 1 in-memory blocks...
```

Figure 21 Code pour le tri du fichier BAM avec Samtools.

### 4.5.3. Indexation des fichiers BAM

Une fois le fichier BAM trié, il est important de l'indexer pour permettre un accès rapide aux données lors des analyses. Cela peut être réalisé avec la commande suivante (figure 22) :

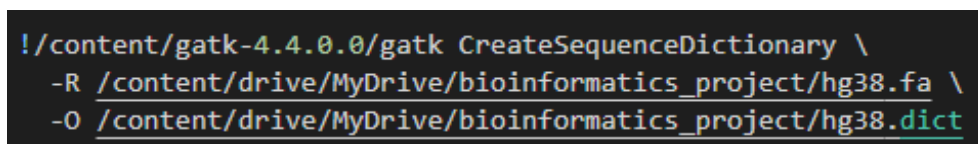


```
[bam_sort_core] merging from 1 files and 1 in-memory blocks...  
[ ] !samtools index /content/drive/MyDrive/bioinformatics_project1/aligned_sorted.bam
```

Figure 22 Code pour l'indexation du fichier BAM trié avec Samtools.

## 4.6. Indexation du fichier FASTA du génome de référence

Le dictionnaire contient des informations importantes sur chaque séquence de référence, telles que les noms des chromosomes et leur longueur. GATK et d'autres outils nécessitent ce dictionnaire pour effectuer des analyses sur les données d'alignement, comme la détection de variants. Voici la commande qui nous fait pour extraire le fichier hg38.dict.



```
!/content/gatk-4.4.0.0/gatk CreateSequenceDictionary \  
-R /content/drive/MyDrive/bioinformatics_project/hg38.fa \  
-O /content/drive/MyDrive/bioinformatics_project/hg38.dict
```

Figure 23 Ce code pour l'extraction du dictionnaire de référence.

Cette étape permet de créer un dictionnaire pour le fichier de séquence de référence (hg38.fa). Ce dictionnaire est essentiel pour que des outils puissent manipuler et analyser efficacement le génome de référence.

Résultat : Cette commande génère le fichier **hg38.dict**, qu'il sera utilisé par GATK et d'autres outils pour effectuer des analyses efficaces du génome de référence.

## 4.7. Ajouter ou Remplacer les Groupes et Indexer un fichier BAM

### 4.7.1. Ajout des Read Groups

Read Groups sont des métadonnées importantes pour certaines étapes d'analyse (par exemple, pour la détection des variants). Picard a été utilisé pour ajouter ces informations :

🚦 Téléchargement de l'outil Picard :

```
!wget https://github.com/broadinstitute/picard/releases/download/2.27.4/picard.jar -P /content/

--2024-12-22 01:28:27-- https://github.com/broadinstitute/picard/releases/download/2.27.4/picard.jar
Resolving github.com (github.com)... 140.82.112.4
Connecting to github.com (github.com)|140.82.112.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/18225913/839cd9dd-e7dc-4
--2024-12-22 01:28:27-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/18225913/
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 18
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 17544754 (17M) [application/octet-stream]
Saving to: '/content/picard.jar'

picard.jar          100%[=====>] 16.73M  --.-KB/s   in 0.1s
2024-12-22 01:28:28 (119 MB/s) - '/content/picard.jar' saved [17544754/17544754]
```

Figure 24 l'installation de picard

🚦 Ajout des groupes de lecture :

```
!java -jar picard.jar AddOrReplaceReadGroups \
I=/content/drive/MyDrive/bioinformatics_project/marked_duplicates1.bam \
O=/content/drive/MyDrive/bioinformatics_project/marked_duplicates_with_RG.bam \
RGID=sample_id \
RGLB=lib1 \
RGPL=illumina \
RGPU=unit1 \
RGSM=tumor_sample_name
```

Figure 25 Ajout des groupes de lecture


### 4.7.2. Validation des groupes de read ajoutés

La commande suivante a permis de vérifier que les groupes de read ont été ajoutés correctement :

```
!samtools view -H /content/drive/MyDrive/bioinformatics_project/marked_duplicates_with_RG.bam | grep '@RG'
```

@RG	ID:sample_id	LB:lib1	PL:illumina	SM:tumor_sample_name	PU:unit1
-----	--------------	---------	-------------	----------------------	----------

Figure 26 Vérification de groupe de read de fichier BAM

 **SAMtools view** : Cette commande affiche un aperçu du fichier BAM en lisant les premiers alignements, ce qui permet de vérifier si les groupes de lecture apparaissent correctement dans les en-têtes.

### 4.7.3. Les statistiques de l'alignement

Pour évaluer la qualité et la couverture des alignements, des statistiques ont été générées à l'aide de SAMtools stats.

Les statistiques générées incluent des métriques telles que :

- Le nombre total de lectures alignées.
- Le pourcentage de lectures correctement appariées.
- Le pourcentage de lectures non alignées.
- Les régions du génome couvertes par les lectures.

Les statistiques essentielles ont été extraites avec la commande suivante :

```
[ ] !cat alignment_stats.txt | grep "^SN" | cut -f 2-
```

```
➡ raw total sequences: 1047740 # excluding supplementary and secondary reads
  filtered sequences: 0
  sequences: 1047740
  is sorted: 1
  1st fragments: 523870
  last fragments: 523870
  reads mapped: 1046790
  reads mapped and paired: 1046110 # paired-end technology bit set + both mates mapped
  reads unmapped: 950
  reads properly paired: 832672 # proper-pair bit set
  reads paired: 1047740 # paired-end technology bit set
  reads duplicated: 0 # PCR or optical duplicate bit set
  reads MQ0: 72931 # mapped and MQ=0
  reads QC failed: 0
  non-primary alignments: 0
  supplementary alignments: 267426
  total length: 130917593 # ignores clipping
  total first fragment length: 65577247 # ignores clipping
  total last fragment length: 65340346 # ignores clipping
  bases mapped: 130847036 # ignores clipping
  bases mapped (cigar): 130696179 # more accurate
  bases trimmed: 0
  bases duplicated: 0
  mismatches: 820239 # from NM fields
  error rate: 6.275922e-03 # mismatches / bases mapped (cigar)
  average length: 124
  average first fragment length: 125
  average last fragment length: 125
  maximum length: 131
  maximum first fragment length: 131
  maximum last fragment length: 131
  average quality: 36.3
  insert size average: 313.3
  insert size standard deviation: 1081.6
  inward oriented pairs: 364856
  outward oriented pairs: 57544
  pairs with other orientation: 3787
  pairs on different chromosomes: 96868
  percentage of properly paired reads (%): 79.5
```

Figure 27 les statistiques d'alignement d'un fichier.

## 4.8. Marquage des duplicats et indexation du fichier BAM

### 4.8.1. Marquage des duplicats

Le marquage des duplicats est une étape essentielle après le tri des fichiers BAM afin d'identifier les lectures redondantes provenant des duplications de fragments d'ADN, principalement introduites lors des cycles de PCR. Ces duplicats, bien que non éliminés du fichier final, sont marqués pour éviter qu'ils ne faussent les analyses en surévaluant la couverture ou en biaisant les résultats des appels de variants. L'outil Samtools a été utilisé pour effectuer cette tâche grâce à la commande illustrée ci-dessous (Figure 23). Cette



commande prend en entrée un fichier BAM trié et produit un fichier où les duplicats sont marqués, facilitant ainsi les étapes d'analyse subséquentes.

```
[ ] # Marquage des doublons
!samtools markdup
/content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/fixed_sorted_aligned11.bam
/content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/marked_duplicates1.bam
```

Figure 28 Code pour le marquage des duplicats dans le fichier BAM à l'aide de Samtools.

### 4.8.2. Indexation du fichier BAM d'alignement

Après le marquage des duplicats, l'étape suivante consiste à indexer le fichier BAM. L'indexation est cruciale pour permettre une consultation rapide et efficace des données d'alignement lors des analyses en aval, telles que Variant Calling . Cette étape crée un fichier d'index associé qui permet à des outils comme Samtools d'accéder directement aux alignements d'une position donnée dans le génome, ce qui accélère les processus de recherche et de manipulation des données.

L'indexation du fichier BAM a été effectuée à l'aide de la commande Samtools index (figure 29) :

```
[ ] !samtools index /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/marked_duplicates_with_RG.bam
```

Figure 29 Code pour l'indexation du fichier BAM

- ✚ Cette commande génère un fichier d'index (marked\_duplicates\_with\_RG.bam.bai) qui est utilisé pour des analyses ultérieures, facilitant la gestion des grandes quantités de données génomiques.

## 4.9. Variant Calling (Détection de variants)

Le variant calling est une étape cruciale pour identifier les variations génétiques telles que les SNPs (Single Nucleotide Polymorphisms) et les indels (insertions ou délétions). Après le marquage des duplicats, le fichier BAM corrigé et trié est utilisé comme entrée pour effectuer cette analyse à l'aide de l'outil **GATK** (Genome Analysis Toolkit).

### 4.9.1. Téléchargement et installation de GATK :

GATK (Genome Analysis Toolkit) est un outil clé pour la détection des variants. La version utilisée ici est 4.4.0.0, téléchargée et installée avec les commandes suivantes:

```
[ ] !wget https://github.com/broadinstitute/gatk/releases/download/4.4.0.0/gatk-4.4.0.0.zip

--2024-12-28 14:00:41-- https://github.com/broadinstitute/gatk/releases/download/4.4.0.0/gatk-4.4
Resolving github.com (github.com)... 140.82.114.3
Connecting to github.com (github.com)|140.82.114.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/27452807/0b
--2024-12-28 14:00:41-- https://objects.githubusercontent.com/github-production-release-asset-2e6
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108.133, 185.19
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108.133|:443..
HTTP request sent, awaiting response... 200 OK
Length: 642466069 (613M) [application/octet-stream]
Saving to: 'gatk-4.4.0.0.zip'

gatk-4.4.0.0.zip  100%[=====>] 612.70M  122MB/s  in 4.7s

2024-12-28 14:00:46 (131 MB/s) - 'gatk-4.4.0.0.zip' saved [642466069/642466069]

[ ] !unzip gatk-4.4.0.0.zip
!cd gatk-4.4.0.0

Archive: gatk-4.4.0.0.zip
  creating: gatk-4.4.0.0/
  creating: gatk-4.4.0.0/gatkdoc/
  inflating: gatk-4.4.0.0/gatkdoc/org_broadinstitute_hellbender_tools_walkers_rnaseq_ASEReadCount
  inflating: gatk-4.4.0.0/gatkdoc/picard_analysis_replicates_CollectIndependentReplicateMetrics.h
  inflating: gatk-4.4.0.0/gatkdoc/org_broadinstitute_hellbender_tools_dragstr_CalibrateDragstrMod
  inflating: gatk-4.4.0.0/gatkdoc/org_broadinstitute_hellbender_tools_walkers_annotator_Chromosom
  inflating: gatk-4.4.0.0/gatkdoc/org_broadinstitute_hellbender_engine_filters_LibraryReadFilter.
  inflating: gatk-4.4.0.0/gatkdoc/org_broadinstitute_hellbender_tools_DumpTabixIndex.html
```

Figure 30 l'installation de l'outil GATK

### 4.9.2. Configuration de l'environnement Java:

GATK nécessite Java pour fonctionner. Une version compatible de Java Runtime Environment (JRE) a été installée :

```
[ ] !sudo apt update
!sudo apt install openjdk-17-jre

Hit:1 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 https://cloud.r-project.org/bin/linux/ubuntu jammy-cran40/ InRelease
Hit:4 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:5 https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2204/x86_64 InRelease
Hit:6 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:7 https://r2u.stat.illinois.edu/ubuntu jammy InRelease
Hit:8 https://ppa.launchpadcontent.net/deadsnakes/ppa/ubuntu jammy InRelease
Hit:9 https://ppa.launchpadcontent.net/graphics-drivers/ppa/ubuntu jammy InRelease
Hit:10 https://ppa.launchpadcontent.net/ubuntuGIS/ppa/ubuntu jammy InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
50 packages can be upgraded. Run 'apt list --upgradable' to see them.
W: Skipping acquire of configured file 'main/source/Sources' as repository 'https://r2u.stat.illinois.edu/ubuntu'
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  openjdk-17-jre-headless
Suggested packages:
  libnss-mdns fonts-ipafont-gothic fonts-ipafont-mincho fonts-wqy-microhei
  | fonts-wqy-zenhei fonts-indic
The following NEW packages will be installed:
  openjdk-17-jre openjdk-17-jre-headless
0 upgraded, 2 newly installed, 0 to remove and 50 not upgraded.
Need to get 48.6 MB of archives.
After this operation, 194 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 openjdk-17-jre-headless amd64 17.0.13+1-1
Get:2 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 openjdk-17-jre amd64 17.0.13+1-1
```

Figure 31 l'installation du java

### 4.9.3. Détection des variants somatiques(variants calling)

Après l'indexation du fichier BAM, l'étape suivante consiste à effectuer le **variant calling** pour détecter les variants somatiques, tels que les mutations spécifiques aux échantillons étudiés. Pour cette tâche, nous avons utilisé l'outil **Mutect2** de **GATK (Genome Analysis Toolkit)**, spécifiquement conçu pour le variant calling des mutations somatiques dans les échantillons de cancer.

L'outil **Mutect2** a été utilisé pour identifier les variants somatiques à partir des fichiers BAM d'alignement après le marquage des duplicats et leur indexation. Comme il montre la figure 32.

```
[ ] !/content/gatk-4.4.0.0/gatk Mutect2 \
-R /content/drive/MyDrive/ColabNotebooks/FilesCancer/hg38.fa \
-I /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/marked_duplicates_with_RG.bam \
-tumor-sample tumor_sample_name \
-O /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/only_variants.vcf

14:02:19.082 INFO Mutect2 - HTSJDK Defaults.COMPRESSION_LEVEL : 2
14:02:19.083 INFO Mutect2 - HTSJDK Defaults.USE_ASYNC_IO_READ_FOR_SAMTOOLS : false
14:02:19.083 INFO Mutect2 - HTSJDK Defaults.USE_ASYNC_IO_WRITE_FOR_SAMTOOLS : true
14:02:19.085 INFO Mutect2 - HTSJDK Defaults.USE_ASYNC_IO_WRITE_FOR_TRIBBLE : false
14:02:19.087 INFO Mutect2 - Deflater: IntelDeflater
14:02:19.087 INFO Mutect2 - Inflater: IntelInflater
14:02:19.088 INFO Mutect2 - GCS max retries/reopens: 20
14:02:19.088 INFO Mutect2 - Requester pays: disabled
14:02:19.089 INFO Mutect2 - Initializing engine
14:02:20.598 INFO Mutect2 - Done initializing engine
14:02:20.698 INFO NativeLibraryLoader - Loading libgkl_utils.so from jar:file:/content/gatk-4.4.0.0/
14:02:20.723 INFO NativeLibraryLoader - Loading libgkl_pairhmm_omp.so from jar:file:/content/gatk-
14:02:20.749 INFO IntelPairHmm - Flush-to-zero (FTZ) is enabled when running PairHMM
14:02:20.753 INFO IntelPairHmm - Available threads: 2
14:02:20.754 INFO IntelPairHmm - Requested threads: 4
14:02:20.755 WARN IntelPairHmm - Using 2 available threads, but 4 were requested
14:02:20.756 INFO PairHMM - Using the OpenMP multi-threaded AVX-accelerated native PairHMM impleme
14:02:20.873 INFO ProgressMeter - Starting traversal
14:02:20.875 INFO ProgressMeter - Current Locus Elapsed Minutes Regions Processed Re
14:02:30.915 INFO ProgressMeter - chr1:1961905 0.2 6750
14:02:40.946 INFO ProgressMeter - chr1:5046863 0.3 17470
14:02:50.982 INFO ProgressMeter - chr1:7831966 0.5 27300
14:03:00.983 INFO ProgressMeter - chr1:11012182 0.7 38500
14:03:10.991 INFO ProgressMeter - chr1:15101127 0.8 52770
```

Figure 32 Code pour le variant calling avec GATK Mutect2

- ✚ -R spécifie le fichier de référence utilisé pour l'alignement des séquences.
- ✚ -I désigne le fichier BAM d'entrée, qui contient les alignements après le marquage des duplicates et l'indexation.
- ✚ -O indique le fichier de sortie au format VCF, qui contient les variants somatiques détectés.

**Mutect2** génère un fichier VCF qui comprend des informations sur les variants somatiques, notamment les positions génomiques des mutations, ainsi que des scores de qualité pour évaluer la fiabilité des appels de variants.

Cette étape est essentielle pour identifier les mutations spécifiques à un échantillon, notamment dans les études sur le cancer ou d'autres maladies liées à des variations somatiques.

## 4.10. Filtrage des variants avec GATK (SelectVariants)

Après avoir effectué le **variant calling** avec **Mutect2**, nous avons procédé à un filtrage des variants en séparant les SNPs (Single Nucleotide Polymorphisms) et les INDELs (Insertions/Deletions) dans deux fichiers distincts. Cette étape permet de se concentrer sur des types de variants spécifiques, en fonction des besoins de l'analyse.

Nous avons utilisé l'outil **SelectVariants** de **GATK** pour filtrer et extraire les SNPs et les INDELs à partir du fichier VCF généré précédemment. Les commandes de la figure 33 et 34 ont été utilisées pour effectuer ce filtrage .

```
[ ] !/content/gatk-4.4.0.0/gatk SelectVariants \
-R /content/drive/MyDrive/ColabNotebooks/FilesCancer/hg38.fa \
-V /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252111/only_variants.vcf \
--select-type-to-include SNP \
-O /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252111/snps.vcf

Using GATK jar /content/gatk-4.4.0.0/gatk-package-4.4.0.0-local.jar
Running:
  java -Dsamjdk.use_async_io_read_samtools=false -Dsamjdk.use_async_io_write_samtools=true
12:04:14.875 INFO NativeLibraryLoader - Loading libgkl_compression.so from jar:file:/content
12:04:14.932 INFO SelectVariants - -----
12:04:14.939 INFO SelectVariants - The Genome Analysis Toolkit (GATK) v4.4.0.0
12:04:14.939 INFO SelectVariants - For support and documentation go to https://software.broad
12:04:14.939 INFO SelectVariants - Executing as root@b0185f9a31ff on Linux v6.1.85+ amd64
12:04:14.940 INFO SelectVariants - Java runtime: OpenJDK 64-Bit Server VM v17.0.13+11-Ubuntu
12:04:14.940 INFO SelectVariants - Start Date/Time: December 28, 2024 at 12:04:14 PM UTC
12:04:14.940 INFO SelectVariants - -----
12:04:14.940 INFO SelectVariants - HTSJDK Version: 3.0.5
12:04:14.942 INFO SelectVariants - Picard Version: 3.0.0
12:04:14.942 INFO SelectVariants - Built for Spark Version: 3.3.1
12:04:14.943 INFO SelectVariants - HTSJDK Defaults.COMPRESSION_LEVEL : 2
12:04:14.943 INFO SelectVariants - HTSJDK Defaults.USE_ASYNC_IO_READ_FOR_SAMTOOLS : false
12:04:14.943 INFO SelectVariants - HTSJDK Defaults.USE_ASYNC_IO_WRITE_FOR_SAMTOOLS : true
12:04:14.944 INFO SelectVariants - HTSJDK Defaults.USE_ASYNC_IO_WRITE_FOR_TRIBBLE : false
12:04:14.944 INFO SelectVariants - Deflater: IntelDeflater
12:04:14.944 INFO SelectVariants - Inflater: IntelInflater
12:04:14.945 INFO SelectVariants - GCS max retries/reopens: 20
12:04:14.945 INFO SelectVariants - Requester pays: disabled
12:04:14.946 INFO SelectVariants - Initializing engine
12:04:15.271 INFO FeatureManager - Using codec VCFCodec to read file file:///content/drive/M
12:04:15.446 INFO SelectVariants - Done initializing engine
```

Figure 33 la commande pour le filtrage des SNPs



```
[ ] !/content/gatk-4.4.0.0/gatk SelectVariants \
-R /content/drive/MyDrive/ColabNotebooks/FilesCancer/hg38.fa \
-V /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/only_variants.vcf \
--select-type-to-include INDEL \
-O /content/drive/MyDrive/ColabNotebooks/FilesCancer/SRR14252108/indels.vcf

Using GATK jar /content/gatk-4.4.0.0/gatk-package-4.4.0.0-local.jar
Running:
  java -Dsamjdk.use_async_io_read_samtools=false -Dsamjdk.use_async_io_write_samtools=
15:22:19.464 INFO NativeLibraryLoader - Loading libgkl_compression.so from jar:file:/c
15:22:19.535 INFO SelectVariants - -----
15:22:19.540 INFO SelectVariants - The Genome Analysis Toolkit (GATK) v4.4.0.0
15:22:19.540 INFO SelectVariants - For support and documentation go to https://software.broadinstitute.org/gatk/
15:22:19.541 INFO SelectVariants - Executing as root@040b704df52e on Linux v6.1.85+ am
15:22:19.541 INFO SelectVariants - Java runtime: OpenJDK 64-Bit Server VM v17.0.13+11-l
15:22:19.541 INFO SelectVariants - Start Date/Time: December 28, 2024 at 3:22:19 PM UT
15:22:19.541 INFO SelectVariants - -----
15:22:19.541 INFO SelectVariants - -----
15:22:19.543 INFO SelectVariants - HTSJDK Version: 3.0.5
15:22:19.543 INFO SelectVariants - Picard Version: 3.0.0
15:22:19.543 INFO SelectVariants - Built for Spark Version: 3.3.1
15:22:19.544 INFO SelectVariants - HTSJDK Defaults.COMPRESSION_LEVEL : 2
15:22:19.544 INFO SelectVariants - HTSJDK Defaults.USE_ASYNC_IO_READ_FOR_SAMTOOLS : fa
15:22:19.544 INFO SelectVariants - HTSJDK Defaults.USE_ASYNC_IO_WRITE_FOR_SAMTOOLS : t
15:22:19.545 INFO SelectVariants - HTSJDK Defaults.USE_ASYNC_IO_WRITE_FOR_TRIBBLE : fa
15:22:19.545 INFO SelectVariants - Deflater: IntelDeflater
15:22:19.545 INFO SelectVariants - Inflater: IntelInflater
15:22:19.545 INFO SelectVariants - GCS max retries/reopens: 20
15:22:19.546 INFO SelectVariants - Requester pays: disabled
15:22:19.546 INFO SelectVariants - Initializing engine
15:22:19.913 INFO FeatureManager - Using codec VCFCodec to read file file:///content/d
15:22:20.091 INFO SelectVariants - Done initializing engine
```

Figure 34 la commande pour le filtrage des INDELS

- ✚ -R : Spécifie le fichier de référence, qui est nécessaire pour interpréter les variants.
- ✚ -V : Le fichier VCF d'entrée contenant tous les variants appelés.
- ✚ --select-type-to-include SNP : Sélectionne uniquement les SNPs dans le fichier de sortie.
- ✚ --select-type-to-include INDEL : Sélectionne uniquement les INDELS dans le fichier de sortie.
- ✚ -O : Définit le fichier de sortie filtré, contenant les variants filtrés par type (SNP ou INDEL).

Ces étapes permettent d'extraire les différents types de variants (SNPs et INDELS) pour une analyse plus ciblée dans le contexte de notre étude.

## 4.11. Annotation des SNPs avec SnpEff

Après avoir extrait les SNPs dans un fichier distinct, nous avons procédé à l'annotation des variants pour obtenir des informations supplémentaires sur leurs impacts biologiques, tels que l'effet sur les gènes ou les protéines. Cette étape permet de mieux comprendre les variants et leur rôle potentiel dans les maladies ou les traits phénotypiques à l'aide de l'outil SnpEff.

### 4.11.1. Téléchargement et extraction de SnpEff

Cette figure (figure 35) montre les commandes utilisées pour télécharger et extraire le package **SnpEff** nécessaire à l'annotation des variants.

```
[ ] # Téléchargement et extraction de SnpEff
!wget http://sourceforge.net/projects/snpeff/files/snpeff_latest_core.zip -O snpEff.zip
!unzip snpEff.zip -d /content/drive/MyDrive/bioinformatics_project1/

# Passage dans le répertoire de SnpEff
%cd /content/drive/MyDrive/bioinformatics_project1/snpEff/

inflating: /content/drive/MyDrive/bioinformatics_project1/snpEff/scripts/countColumns.py
inflating: /content/drive/MyDrive/bioinformatics_project1/snpEff/scripts/vcfAnnFirst.py
inflating: /content/drive/MyDrive/bioinformatics_project1/snpEff/scripts/join.pl
inflating: /content/drive/MyDrive/bioinformatics_project1/snpEff/scripts/txt2vcf.py
inflating: /content/drive/MyDrive/bioinformatics_project1/snpEff/scripts/fastqSplit.pl
inflating: /content/drive/MyDrive/bioinformatics_project1/snpEff/scripts/cgShore.sh
inflating: /content/drive/MyDrive/bioinformatics_project1/snpEff/scripts/annotate_demo_GATK.sh
inflating: /content/drive/MyDrive/bioinformatics_project1/snpEff/scripts/vcfInfoOnePerLine.pl
```

Figure 35 Téléchargement et extraction de SnpEff

### 4.11.2. Téléchargement du modèle de référence GRCh38.86

Cette figure (figure 36) présente la commande utilisée pour télécharger le modèle de référence **GRCh38.86**, qui est essentiel pour l'annotation des variants à l'aide de **SnpEff**.

```
[ ] !java -jar /content/drive/MyDrive/bioinformatics_project1/snpEff/snpEff.jar download GRCh38.86
```

Figure 36 Téléchargement du modèle de référence GRCh38.86

### 4.11.3. Annotation des SNPs avec SnpEff

Cette figure montre la commande utilisée pour annoter les SNPs présents dans le fichier **snps.vcf**, en utilisant le modèle **GRCh38.86** pour produire le fichier annoté **snps\_annotated.vcf**.

```
!java -jar /content/drive/MyDrive/bioinformatics_project1/snpEff/snpEff.jar GRCh38.86 /content/drive/MyDrive/bioinformatics_project1/snps.vcf > /content/drive/MyDrive/bioinformatics_project1/snps_annotated.vcf
```

Figure 37 Annotation des SNPs avec SnpEff

Le fichier **snps\_annotated.vcf** (figure 38) contient des variantes génétiques annotées sous forme de SNPs (Single Nucleotide Polymorphisms) détectées sur le chromosome Y. Chaque ligne représente une variante avec des informations telles que la position génomique, l'allèle de référence et alternatif, des annotations fonctionnelles (gènes affectés, effets prédits), ainsi que des statistiques comme la profondeur de lecture (DP), la fréquence allélique (AF),

et des données de génotype. Les annotations sont basées sur des bases de données génomiques pour indiquer les impacts biologiques potentiels.

```
tail -n 20 /content/drive/MyDrive/bioinformatics_project1/snps_annotated.vcf
```

chrY	56847120	.	A	G	.	.	AS_SB_TABLE=0,0,0,0;DP=2;ECNT=2;MBQ=0,28;MFRL=0,143;MQ=60,60;MPOS=10;POPAF=7.30;TLOD=4.62;ANN=6 intergenic_region MODIFIER PARP4P1-CTBP2P1
chrY	56847928	.	T	C	.	.	AS_SB_TABLE=0,0,0,0;DP=1;ECNT=2;MBQ=0,38;MFRL=0,375;MQ=60,55;MPOS=7;POPAF=7.30;TLOD=4.28;ANN=C intergenic_region MODIFIER PARP4P1-CTBP2P1
chrY	56847949	.	G	A	.	.	AS_SB_TABLE=0,0,0,0;DP=1;ECNT=2;MBQ=0,38;MFRL=0,375;MQ=60,55;MPOS=28;POPAF=7.30;TLOD=4.28;ANN=A intergenic_region MODIFIER PARP4P1-CTBP2P1
chrY	56848245	.	T	A	.	.	AS_SB_TABLE=0,0,0,0;DP=1;ECNT=1;MBQ=0,38;MFRL=0,375;MQ=60,55;MPOS=50;POPAF=7.30;TLOD=3.98;ANN=A intergenic_region MODIFIER PARP4P1-CTBP2P1
chrY	56850800	.	G	T	.	.	AS_SB_TABLE=0,0,0,0;DP=2;ECNT=6;MBQ=0,28;MFRL=0,158;MQ=60,47;MPOS=50;POPAF=7.30;TLOD=8.70;ANN=T upstream_gene_variant MODIFIER CTBP2P1 EN
chrY	56850821	.	G	A	.	.	AS_SB_TABLE=0,0,0,0;DP=2;ECNT=6;MBQ=0,28;MFRL=0,158;MQ=60,47;MPOS=52;POPAF=7.30;TLOD=8.70;ANN=A upstream_gene_variant MODIFIER CTBP2P1 EN
chrY	56850839	.	G	A	.	.	AS_SB_TABLE=0,0,0,0;DP=2;ECNT=6;MBQ=0,28;MFRL=0,158;MQ=60,47;MPOS=42;POPAF=7.30;TLOD=8.70;ANN=A upstream_gene_variant MODIFIER CTBP2P1 EN
chrY	56850859	.	G	A	.	.	AS_SB_TABLE=0,0,0,0;DP=2;ECNT=6;MBQ=0,28;MFRL=0,158;MQ=60,47;MPOS=22;POPAF=7.30;TLOD=8.70;ANN=A upstream_gene_variant MODIFIER CTBP2P1 EN
chrY	56850875	.	C	A	.	.	AS_SB_TABLE=0,0,0,0;DP=1;ECNT=6;MBQ=0,38;MFRL=0,158;MQ=60,54;MPOS=19;POPAF=7.30;TLOD=8.70;ANN=A upstream_gene_variant MODIFIER CTBP2P1 EN
chrY	56861065	.	C	A	.	.	AS_SB_TABLE=0,0,2,1;DP=4;ECNT=1;MBQ=0,28;MFRL=0,98;MQ=60,25;MPOS=8;POPAF=7.30;TLOD=7.57;ANN=A intergenic_region MODIFIER CTBP2P1-CHR_END
chrY	56864388	.	A	G	.	.	AS_SB_TABLE=0,0,0,0;DP=2;ECNT=2;MBQ=0,28;MFRL=0,147;MQ=60,47;MPOS=50;POPAF=7.30;TLOD=4.62;ANN=6 intergenic_region MODIFIER CTBP2P1-CHR_END
chrY	56866207	.	T	C	.	.	AS_SB_TABLE=0,0,0,0;DP=1;ECNT=1;MBQ=0,38;MFRL=0,200;MQ=60,60;MPOS=33;POPAF=7.30;TLOD=3.98;ANN=C intergenic_region MODIFIER CTBP2P1-CHR_END
chrY	56873763	.	G	T	.	.	AS_SB_TABLE=0,0,0,0;DP=1;ECNT=6;MBQ=0,38;MFRL=0,219;MQ=60,60;MPOS=15;POPAF=7.30;TLOD=3.98;ANN=T intergenic_region MODIFIER CTBP2P1-CHR_END
chrY	56873831	.	G	C	.	.	AS_SB_TABLE=0,0,0,0;DP=2;ECNT=6;MBQ=38,38;MFRL=0,219,0;MQ=60,60;MPOS=59;POPAF=7.30;TLOD=3.50;ANN=C intergenic_region MODIFIER CTBP2P1-CHR_END
chrY	56873882	.	T	A	.	.	AS_SB_TABLE=0,0,0,0;DP=2;ECNT=6;MBQ=38,38;MFRL=0,219,0;MQ=60,40;MPOS=46;POPAF=7.30;TLOD=3.50;ANN=A intergenic_region MODIFIER CTBP2P1-CHR_END
chrY	56873899	.	C	G	.	.	AS_SB_TABLE=0,0,0,0;DP=2;ECNT=6;MBQ=38,34;MFRL=0,219,0;MQ=60,60;MPOS=3;POPAF=7.30;TLOD=3.10;ANN=6 intergenic_region MODIFIER CTBP2P1-CHR_END
chrY	56873923	.	G	A	.	.	AS_SB_TABLE=0,0,0,0;DP=1;ECNT=6;MBQ=0,31;MFRL=0,219;MQ=60,40;MPOS=43;POPAF=7.30;TLOD=3.28;ANN=A intergenic_region MODIFIER CTBP2P1-CHR_END
chrY	56873952	.	C	T	.	.	AS_SB_TABLE=0,0,0,0;DP=1;ECNT=6;MBQ=0,38;MFRL=0,219;MQ=60,40;MPOS=14;POPAF=7.30;TLOD=3.98;ANN=T intergenic_region MODIFIER CTBP2P1-CHR_END
chrY	56878338	.	T	A	.	.	AS_SB_TABLE=0,0,0,0;DP=2;ECNT=1;MBQ=0,38;MFRL=0,120;MQ=60,33;MPOS=19;POPAF=7.30;TLOD=6.88;ANN=A intergenic_region MODIFIER CTBP2P1-CHR_END
chrY_K1270740v1_random	10781	.	C	T	.	.	AS_SB_TABLE=0,0,0,0;DP=1;ECNT=2;MBQ=0,37;MFRL=0,154;MQ=60,49;MPOS=7;POPAF=7.30;TLOD=3.25;ANN=T MODIFIER ERROR_CHROM

Figure 38 Aperçu de la Fin du Fichier VCF Annotations de Variants

## 5. Interprétations des résultats

### 5.1. Analyse des Variations Génétiques :

Ce tableau présente la distribution des variations génétiques sous forme de SNPs (Single Nucleotide Polymorphisms) et d'indels (insertions et délétions) dans sept séquences différentes. Le nombre de SNPs varie considérablement, avec la séquence 6 présentant la plus grande quantité (267,910 SNPs), suivie de la séquence 5 (178,604 SNPs), tandis que la séquence 7 en contient un nombre beaucoup plus faible (1,413 SNPs). Concernant les indels, la séquence 6 a également le plus grand nombre (74,196 indels), tandis que la séquence 7 affiche le plus faible (877 indels). Le ratio SNPs/indels, qui reflète la dominance des SNPs par rapport aux indels, varie entre les séquences. Les séquences 1, 2, 4, 5 et 6 montrent un ratio relativement similaire, autour de 3, indiquant une prévalence des SNPs. La séquence 7, cependant, affiche un ratio beaucoup plus faible (1.61), ce qui suggère que les indels y sont relativement plus fréquents par rapport aux SNPs. Cette variation pourrait être le résultat de différences biologiques entre les séquences ou d'effets expérimentaux. Ces informations sont cruciales pour mieux comprendre les mécanismes sous-jacents de la variation génétique dans ces séquences spécifiques.



Séquence ▼	Nombre de SNPs ▼	Nombre d'indels ▼	Ratio SNPs/indels ▼
Séquence 1	187,698	47,106	3.98
Séquence 2	110,396	33,469	3.3
Séquence 3	5,617	1,607	3.49
Séquence 4	86,305	19,650	4.39
Séquence 5	178,604	45,822	3.9
Séquence 6	267,910	74,196	3.61
Séquence 7	1,413	877	1.61

Table 2 Comparaison des variation génétique

## 5.2. Statistiques d'Alignement et de Cartographie pour les Échantillons de Séquences

Ce tableau présente des statistiques d'alignement pour huit échantillons. La plupart des échantillons (1 à 6) ont des taux de mappage supérieurs à 99%, indiquant une bonne qualité d'alignement, tandis que les échantillons 7 et 8 montrent des taux de mappage beaucoup plus faibles (0,21% et 86,3%). Les alignements primaires sont nombreux pour les échantillons 1 à 6, avec des duplications observées. Le taux de lectures correctement appariées varie entre 72,64% et 80,09%, sauf pour les échantillons 7 et 8, où il est très faible. Les singletons et les mates mappées sur des chromosomes différents sont également notés. Ces données reflètent la qualité générale des alignements pour chaque échantillon.

Les statistiques des séquences 7 et 8 montrent des anomalies significatives par rapport aux autres échantillons. Le taux de mappage est extrêmement faible pour la séquence 7 (0,21%) et modérément faible pour la séquence 8 (86,3%), ce qui indique une mauvaise qualité d'alignement ou un contenu non pertinent dans ces échantillons. De plus, les "properly paired reads" sont presque inexistants pour la séquence 7 (0,10%), et bien que la séquence 8 ait un taux relativement meilleur (85,31%), elle reste en dehors des standards observés pour les autres échantillons. Ces résultats suggèrent que les séquences 7 et 8 ne sont pas fiables pour une analyse ultérieure et doivent être éliminées afin de garantir la robustesse et la qualité des conclusions.

Statistiques ▼	Échantillon 1 ▼	Échantillon 2 ▼	Échantillon 3 ▼	Échantillon 4 ▼	Échantillon 5 ▼	Échantillon 6 ▼
Total (QC-passed + QC-failed)	1,252,060	2,652,259	1,315,166	2,839,959	2,684,843	2,766,215
Primary alignments	893.11	2,071,616	1,047,740	2,126,358	2,040,770	2,036,324
Secondary alignments	0	0	0	0	0	0
Supplementary alignments	358.95	580.643	267.426	713.601	644.073	729.891
Duplicates	183.236	328.383	222.009	254.946	400.462	296.638
Primary duplicates	183.236	328.383	222.009	254.946	400.462	296.638
Mapped reads	1,251,277	2,650,749	1,314,216	2,836,976	2,682,748	2,764,822
Mapped (%)	99.94%	99.94%	99.93%	99.89%	99.92%	99.95%
Paired in sequencing	893.11	2,071,616	1,047,740	2,126,358	2,040,770	2,036,324
Read1	446.555	1,035,808	523.87	1,063,179	1,020,385	1,018,162
Read2	446.555	1,035,808	523.87	1,063,179	1,020,385	1,018,162
Properly paired	715.292	1,620,312	832.672	1,544,618	1,537,074	1,541,868
Properly paired (%)	80.09%	78.21%	79.47%	72.64%	75.32%	75.72%
Singletons	665	1.344	680	2.357	1.747	1.311
Singletons (%)	0.07%	0.06%	0.06%	0.11%	0.09%	0.06%
Mate mapped to a different chr	159.098	401.342	193.736	524.096	459.118	446.27
Mate mapped to a different chr (mapQ≥5)	132.223	328.517	168.706	445.107	405.337	369.835

Table 3 Tableau des statistiques d'alignements

## 6. Visualisation et comparaison des résultats

Dans cette section, nous utilisons la bibliothèque pysam pour analyser et comparer les résultats des alignements des séquences de nos échantillons. La bibliothèque pysam est un outil puissant pour manipuler les fichiers BAM/SAM/CRAM, souvent utilisés pour stocker les données d'alignement dans le cadre d'analyses bioinformatiques.

## 6.1. Installation de et Configuration de Pysam

### 6.1.1. Installation de la bibliothèque

```
!pip install pysam

Collecting pysam
  Downloading pysam-0.22.1-cp310-cp310-manylinux_2_28_x86_64.whl.metadata (1.5 kB)
  Downloading pysam-0.22.1-cp310-cp310-manylinux_2_28_x86_64.whl (22.0 MB)
    2K |#####| 22.0/22.0 MB 69.9 MB/s eta 0:00:00
?25hInstalling collected packages: pysam
Successfully installed pysam-0.22.1
```

Figure 39 l'installation de pysam

Pour commencer, pysam a été installé dans l'environnement Python à l'aide de la commande suivante :

La sortie de cette commande montre que pysam, version 0.22.1, a été téléchargé et installé avec succès :

- Le fichier correspondant à la bibliothèque a été récupéré (22,0 Mo).
- Les dépendances nécessaires ont été installées automatiquement.

### 6.1.2. Objectif de l'utilisation de pysam

L'objectif principal est d'exploiter pysam pour :

1. Extraire des statistiques sur les alignements, par exemple :
  - ✚ Le nombre total de lectures alignées,
  - ✚ Le nombre de duplications,
  - ✚ Le nombre de lectures correctement appariées, etc.
2. Visualiser les données sous forme de tableaux comparatifs pour identifier les différences entre les échantillons.
3. Interpréter les résultats en fonction des métriques calculées (taux d'alignement, taux de duplication, etc.).

Ces résultats permettront d'évaluer la qualité des alignements et de détecter d'éventuelles anomalies ou variations entre les différents échantillons.

## 6.2. Analyse des fichiers VCF

### 6.2.1. Lecture et fusion des fichiers VCF

Cette section lit plusieurs fichiers VCF et en extrait les informations principales, telles que le chromosome, la position, les allèles, la qualité, etc., à l'aide de la bibliothèque pysam. Les données de tous les fichiers sont ensuite fusionnées dans un seul tableau pour faciliter leur analyse.

```
import pysam
import pandas as pd

# Liste des chemins de vos fichiers VCF
file_paths = [
    '/content/drive/MyDrive/comparaison_vcf/com_ano_snp/Copie de snps_annotated.vcf',
    '/content/drive/MyDrive/comparaison_vcf/com_ano_snp/Copie de snps_filtered.eff.vcf',
    '/content/drive/MyDrive/comparaison_vcf/com_ano_snp/snps_annotatedjamaa (1).vcf',
    '/content/drive/MyDrive/comparaison_vcf/com_ano_snp/snps_annotatedjamaa.vcf',
    '/content/drive/MyDrive/comparaison_vcf/com_ano_snp/snps_annotatedkhadija(1).vcf',
    '/content/drive/MyDrive/comparaison_vcf/com_ano_snp/snps_annotatedkhadija(2).vcf',
    '/content/drive/MyDrive/comparaison_vcf/Copie de snps_annotated.vcf'
]

# Fonction pour extraire les données d'un fichier VCF
def parse_vcf(file_path):
    data = []
    with pysam.VariantFile(file_path) as vcf:
        for record in vcf.fetch():
            data.append({
                'CHROM': record.chrom,
                'POS': record.pos,
                'ID': record.id,
                'REF': record.ref,
                'ALT': ','.join(str(alt) for alt in record.alts),
                'QUAL': record.qual,
                'FILTER': ','.join(record.filter.keys()),
                'INFO': record.info
            })
    return pd.DataFrame(data)

# Lire et fusionner tous les fichiers VCF
data_frames = [parse_vcf(file) for file in file_paths]
mutations = pd.concat(data_frames, ignore_index=True)

# Afficher un aperçu des données fusionnées
print("Données annotées fusionnées :")
print(mutations.head())
```

Figure 40 Code de lecture et fusion des fichiers VCF

```
Données annotées fusionnées :
  CHROM  POS  ID REF ALT  QUAL  FILTER  \
0  chr1  75026  None  G  A  None  LowQual
1  chr1  437636  None  C  A  None  LowQual
2  chr1  634112  None  T  C  None    PASS
3  chr1  645257  None  T  C  None  LowQual
4  chr1  645261  None  T  C  None  LowQual

                                INFO
0  [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...
1  [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...
2  [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...
3  [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...
4  [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...
```

Figure 41 Affichage des données annotées fusionnées

### 6.2.2. Extraction de la profondeur de lecture (DP)

Cette étape extrait la métrique DP (Depth of Coverage) depuis la colonne INFO des fichiers VCF. DP indique le nombre de lectures soutenant une variation génétique, ce qui permet d'évaluer sa fiabilité. La valeur extraite est ajoutée comme une nouvelle colonne au tableau.

```
import pysam

# Fonction pour extraire DP à partir de VariantRecordInfo
def extract_dp_from_variant_info(info):
    try:
        # Vérifiez si l'objet est une instance de pysam.libcbcf.VariantRecordInfo
        if isinstance(info, pysam.libcbcf.VariantRecordInfo):
            # Extraire le DP de l'objet VariantRecordInfo
            return info.get('DP', None) # Retourne None si DP n'est pas présent
    except Exception as e:
        print(f"Erreur lors de l'extraction de DP: {e}")
    return None

# Appliquer la fonction d'extraction sur la colonne INFO
mutations['DP'] = mutations['INFO'].apply(extract_dp_from_variant_info)

# Afficher un aperçu des résultats
print(mutations[['CHROM', 'POS', 'DP']].head(10))
```

	CHROM	POS	DP
0	chr1	75026	1
1	chr1	437636	1
2	chr1	634112	12
3	chr1	645257	1
4	chr1	645261	1
5	chr1	818596	1
6	chr1	818602	1
7	chr1	818802	1
8	chr1	818812	1
9	chr1	821925	1

Figure 42 Code d'extraction de la profondeur de lecture (DP)

## 6.3. Filtrage des mutations

### 6.3.1. Filtrage des mutations par profondeur de lecture (DP > 10)

Un filtre est appliqué pour conserver uniquement les mutations ayant une profondeur de lecture (DP) supérieure à 10. Cela élimine les variations soutenues par un faible nombre de lectures, souvent peu fiables.

```
# Appliquer un filtre pour ne garder que les mutations avec DP > 10
mutations_filtrées = mutations[mutations['DP'] > 10]

# Afficher un aperçu des résultats filtrés
print(mutations_filtrées[['CHROM', 'POS', 'DP']].head())
```

	CHROM	POS	DP
2	chr1	634112	12
1618	chr1	20868717	13
2132	chr1	26729701	67
2134	chr1	26732760	60
2137	chr1	26760756	12

Figure 43 Affichage des résultats de filtrage des mutations

### 6.3.2. Filtrage des mutations ayant passé les critères de qualité (FILTER = PASS)

Cette section isole les mutations marquées comme PASS dans la colonne FILTER, indiquant qu'elles répondent aux critères de qualité définis dans les fichiers VCF. Les mutations filtrées sont sauvegardées dans un fichier CSV pour une utilisation ultérieure.

```
# Filtrer uniquement les mutations avec FILTER = PASS
filtered_mutations = mutations[mutations['FILTER'] == 'PASS']

print(f"Nombre de mutations après filtrage sur FILTER : {filtered_mutations.shape[0]}")
print(filtered_mutations.head())
output_path = '/content/drive/MyDrive/comparaison_vcf/com_ano_snp/mutations_filtrees.csv'
filtered_mutations.to_csv(output_path, index=False)
print(f"Mutations filtrées enregistrées dans : {output_path}")
```

```
Nombre de mutations après filtrage sur FILTER : 6629
  CHROM    POS  ID REF ALT  QUAL FILTER  \
2   chr1  634112  None  T  C   None   PASS
1618 chr1  28868717  None  T  A   None   PASS
2132 chr1  26729701  None  C  T   None   PASS
2134 chr1  26732760  None  C  T   None   PASS
2137 chr1  26760756  None  T  C   None   PASS
```

```
                INFO  DP
2   [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...  12
1618 [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...  13
2132 [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...  67
2134 [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...  60
2137 [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...  12
Mutations filtrées enregistrées dans : /content/drive/MyDrive/comparaison_vcf/com_ano_snp/mutations_filtrees.csv
```

Figure 44 Code pour le filtrage des mutations ayant passé les critères de qualité

Les étapes précédentes permettent d'obtenir un ensemble de mutations de haute qualité, issues de plusieurs fichiers VCF, avec des filtres basés sur la couverture (DP > 10) et la qualité (FILTER = PASS). Ces données sont désormais prêtes pour des analyses comparatives ou des recherches approfondies.

## 6.4. Exploration des Données

### 6.4.1. Inspection des colonnes et de la colonne INFO

Cette étape vérifie les colonnes disponibles dans le tableau filtered mutations et explore la colonne INFO. Cette dernière contient des métadonnées détaillées sur chaque mutation, qui seront exploitées pour extraire des informations supplémentaires.

```
print(filtered_mutations.columns)

Index(['CHROM', 'POS', 'ID', 'REF', 'ALT', 'QUAL', 'FILTER', 'INFO', 'DP'], dtype='object')

# Afficher les premières valeurs de la colonne 'INFO'
print(filtered_mutations['INFO'].head())

2      [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...
1618   [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...
2132   [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...
2134   [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...
2137   [AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...
Name: INFO, dtype: object
```

Figure 45 Vérification des colonnes disponibles dans le tableau filtre

### 6.4.2. Extraction des noms de gènes depuis la colonne INFO

Cette étape extrait les noms des gènes associés aux mutations en analysant la colonne INFO, qui contient des annotations détaillées. Une nouvelle colonne GENE est ajoutée au tableau filtered mutations, permettant de lier chaque variation génétique à son gène correspondant. Cela facilite l'identification des mutations dans des gènes spécifiques.

```
import pysam

# Fonction pour extraire le gène à partir de la clé 'ANN' de VariantRecordInfo
def extract_gene_from_info(info_object):
    if isinstance(info_object, pysam.libcbcf.VariantRecordInfo):
        for key, value in info_object.items():
            if key == 'ANN': # Si la clé est 'ANN', extraire l'information sur le gène
                annotation = value[0] # Prendre la première annotation
                parts = annotation.split('|') # Séparer par '|'
                if len(parts) > 3: # Vérifier si la structure est correcte
                    return parts[3] # Le gène est généralement à l'index 3
        return None # Retourner None si aucun gène n'est trouvé

# Appliquer cette fonction à la colonne 'INFO' pour créer la colonne 'GENE'
filtered_mutations.loc[:, 'GENE'] = filtered_mutations['INFO'].apply(lambda x: extract_gene_from_info(x))

# Afficher les résultats
print(filtered_mutations[['CHROM', 'POS', 'GENE', 'REF', 'ALT']].head())

   CHROM  POS  GENE REF ALT
2    chr1  634112  MIR6723  T  C
1618 chr1  20868717  EIF463  T  A
2132 chr1  26729701  ARIID1A  C  T
2134 chr1  26732760  ARIID1A  C  T
2137 chr1  26760756  ARIID1A  T  C

<ipython-input-11-4fc72b531cc0>:15: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
filtered_mutations.loc[:, 'GENE'] = filtered_mutations['INFO'].apply(lambda x: extract_gene_from_info(x))
```

Figure 46 Code pour l'extraction des noms de gènes

### 6.4.3. Identification des mutations dans les gènes d'intérêt

Cette étape cible les mutations affectant des gènes d'intérêt (comme TP53, KRAS, BRCA1, et BRCA2), connus pour leur implication dans des maladies graves, notamment le cancer. Les mutations identifiées sont sauvegardées dans un fichier CSV pour des analyses plus approfondies.

```
# Liste des gènes d'intérêt
genes_of_interest = ['TP53', 'KRAS', 'BRCA1', 'BRCA2']

# Filtrer les mutations pour les gènes d'intérêt
mutations_significatives = filtered_mutations[filtered_mutations['GENE'].isin(genes_of_interest)]

# Afficher un aperçu des mutations significatives
print(mutations_significatives[['CHROM', 'POS', 'GENE', 'REF', 'ALT']].head())
output_path = '/content/drive/MyDrive/comparaison_vcf/com_ano_snp/mutations_significatives.csv'
filtered_mutations.to_csv(output_path, index=False)
```

	CHROM	POS	GENE	REF	ALT
21164	chr12	25209414	KRAS	T	C
21165	chr12	25209531	KRAS	G	A
21166	chr12	25209618	KRAS	A	C
21167	chr12	25209843	KRAS	A	G
21172	chr12	25225537	KRAS	A	T

Figure 47 Code pour l'identification des mutations dans les gènes d'intérêt



#### 6.4.4. Calcul des fréquences des mutations par gène d'intérêt

Cette étape calcule la fréquence des mutations pour chaque gène d'intérêt identifié dans l'étape précédente. Cela permet de quantifier combien de mutations affectent chaque gène prioritaire, fournissant ainsi une mesure comparative de leur implication potentielle dans les phénomènes étudiés.

```
gene_counts = mutations_significatives['GENE'].value_counts()
print("Fréquence des mutations par gène d'intérêt :")
print(gene_counts)
```

```
Fréquence des mutations par gène d'intérêt :
GENE
BRCA2    108
TP53      32
KRAS      28
Name: count, dtype: int64
```

Figure 48 Affichage des fréquences des mutations par gène d'intérêt

#### 6.4.5. Analyse des types de mutations

Cette étape vise à classer les mutations dans deux catégories : Substitution et Insertion/Deletion (indel). La classification est effectuée en fonction de la longueur des séquences de nucléotides de référence (REF) et alternative (ALT). Si les longueurs sont égales, la mutation est une Substitution, sinon elle est une Insertion/Deletion. Ensuite, le code calcule la répartition des mutations en comptabilisant le nombre d'occurrences de chaque type.

```
mutations_significatives['MUTATION_TYPE'] = mutations_significatives.apply(lambda row: 'Substitution' if len(row['REF']) == len(row['ALT']) else 'Insertion/Deletion', axis=1)

mutation_type_counts = mutations_significatives['MUTATION_TYPE'].value_counts()
print("Répartition des types de mutations :")
print(mutation_type_counts)
```

```
Répartition des types de mutations :
MUTATION_TYPE
Substitution    167
Insertion/Deletion    1
Name: count, dtype: int64
```

<ipython-input-14-54e33879651e>:1: SettingWithCopyWarning:  
A value is trying to be set on a copy of a slice from a DataFrame.  
Try using .loc[row\_indexer,col\_indexer] = value instead

See the caveats in the documentation: [https://pandas.pydata.org/pandas-docs/stable/user\\_guide/indexing.html#returning-a-view-versus-a-copy](https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)

```
mutations_significatives['MUTATION_TYPE'] = mutations_significatives.apply(lambda row: 'Substitution' if len(row['REF']) == len(row['ALT']) else 'Insertion/Deletion', axis=1)
```

Figure 49 Code d'affichage de répartition des types de mutation

Cette section calcule la profondeur de couverture moyenne (DP) pour chaque gène dans le DataFrame mutations significatives. Le calcul est réalisé en regroupant les données par gène à l'aide de la fonction groupby de pandas, puis en calculant la moyenne de la colonne DP pour chaque groupe. Cette information permet d'évaluer la qualité de la couverture des gènes étudiés.

```
dp_mean = mutations_significatives.groupby('GENE')['DP'].mean()
print("Profondeur de couverture moyenne par gène :")
print(dp_mean)
```

```
Profondeur de couverture moyenne par gène :
GENE
BRCA2    55.555556
KRAS     68.357143
TP53     57.750000
Name: DP, dtype: float64
```

Figure 50 L'affichage de la profondeur de couverture moyenne par gène

## 6.5. Visualisation des résultats

### 6.5.1. Visualisation du nombre de mutations par gène

Ce code crée un graphique en barres pour visualiser le nombre de mutations par gène dans le DataFrame mutations significatives. Le graphique est généré à l'aide de la bibliothèque seaborn et matplotlib, où chaque barre représente un gène et sa hauteur correspond au nombre de mutations observées pour ce gène. L'axe des x affiche les noms des gènes, et un titre est ajouté pour mieux comprendre ce qui est représenté. Les étiquettes de l'axe x sont également pivotées à 45 degrés pour faciliter leur lecture.

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
sns.countplot(data=mutations_significatives, x='GENE', order=gene_counts.index)
plt.title('Nombre de mutations par gène')
plt.xticks(rotation=45)
plt.show()
```

Figure 51 Le code de l'affichage de nombre de mutations par gène

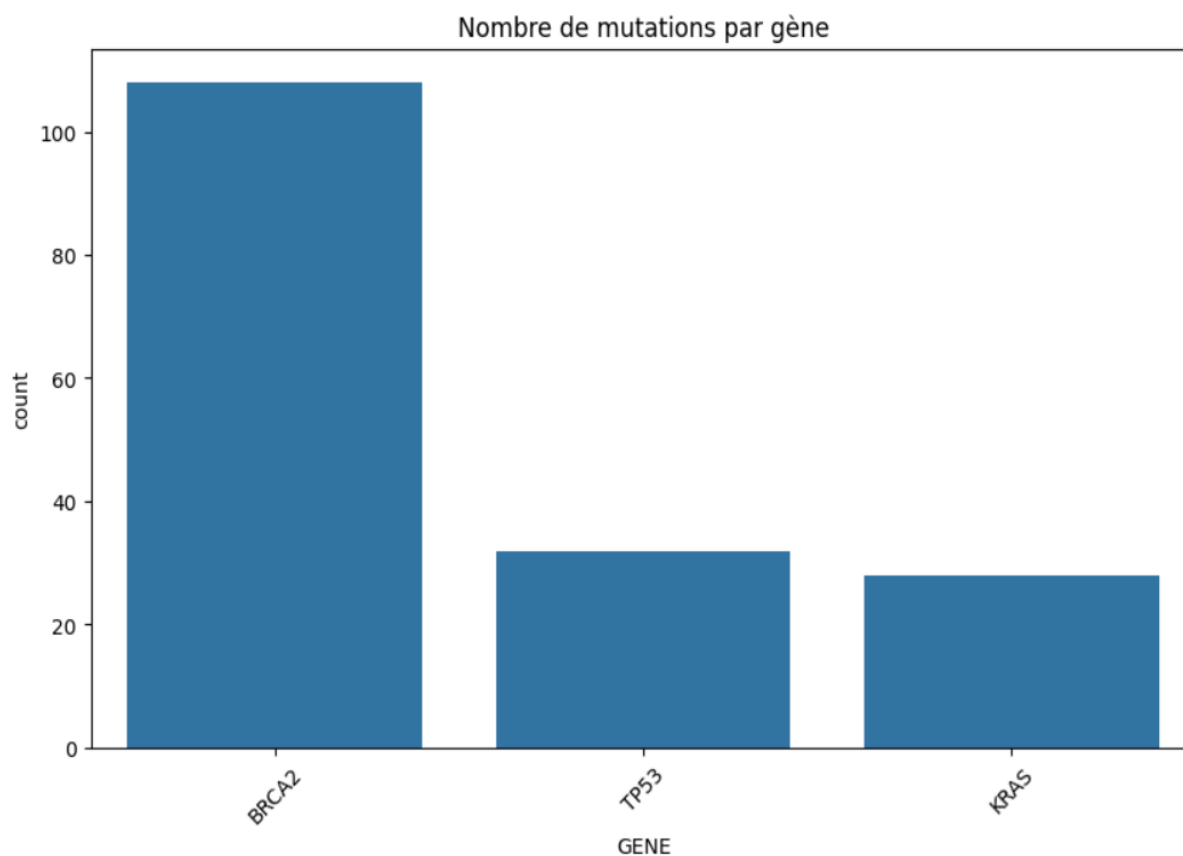


Figure 52 Représentation de nombre de mutations par gène sous forme de graphique en barres

### Interprétation :

Ce diagramme en barres montre la répartition du nombre de mutations significatives détectées pour trois gènes spécifiques : **BRCA2**, **TP53**, et **KRAS**. Le gène **BRCA2** présente un nombre nettement plus élevé de mutations significatives par rapport aux deux autres gènes, suivi par **TP53**, puis **KRAS** avec le nombre le plus bas. Cette distribution pourrait refléter une prévalence accrue de mutations dans le gène **BRCA2** dans les échantillons étudiés, ce qui est cohérent avec son rôle connu dans la susceptibilité au cancer, en particulier dans le cancer du sein et de l'ovaire. Le gène **TP53**, souvent décrit comme le "gardien du génome", montre également une proportion importante de mutations, ce qui correspond à son implication fréquente dans divers cancers. Enfin, **KRAS**, bien qu'ayant un nombre relativement plus faible de mutations, reste un acteur clé dans la signalisation oncogénique, particulièrement dans les cancers du pancréas, du poumon et colorectal.

### 6.5.2. Visualisation de la distribution de la profondeur de couverture (DP)

Ce code génère un histogramme pour visualiser la distribution de la profondeur de couverture (DP) dans les mutations. Le graphique affiche la fréquence des différentes valeurs de DP, avec une courbe de densité (KDE) superposée pour mieux représenter la distribution des données. Le titre, ainsi que les labels des axes, permettent de clarifier les informations montrées dans le graphique, qui aide à analyser la répartition de la profondeur de couverture dans les données de mutations.

```
plt.figure(figsize=(10, 6))
sns.histplot(mutations_significatives['DP'], bins=30, kde=True)
plt.title('Distribution de la profondeur de couverture (DP)')
plt.xlabel('Profondeur de couverture (DP)')
plt.ylabel('Nombre de mutations')
plt.show()
```

Figure 53 Code pour la visualisation de la distribution de DP

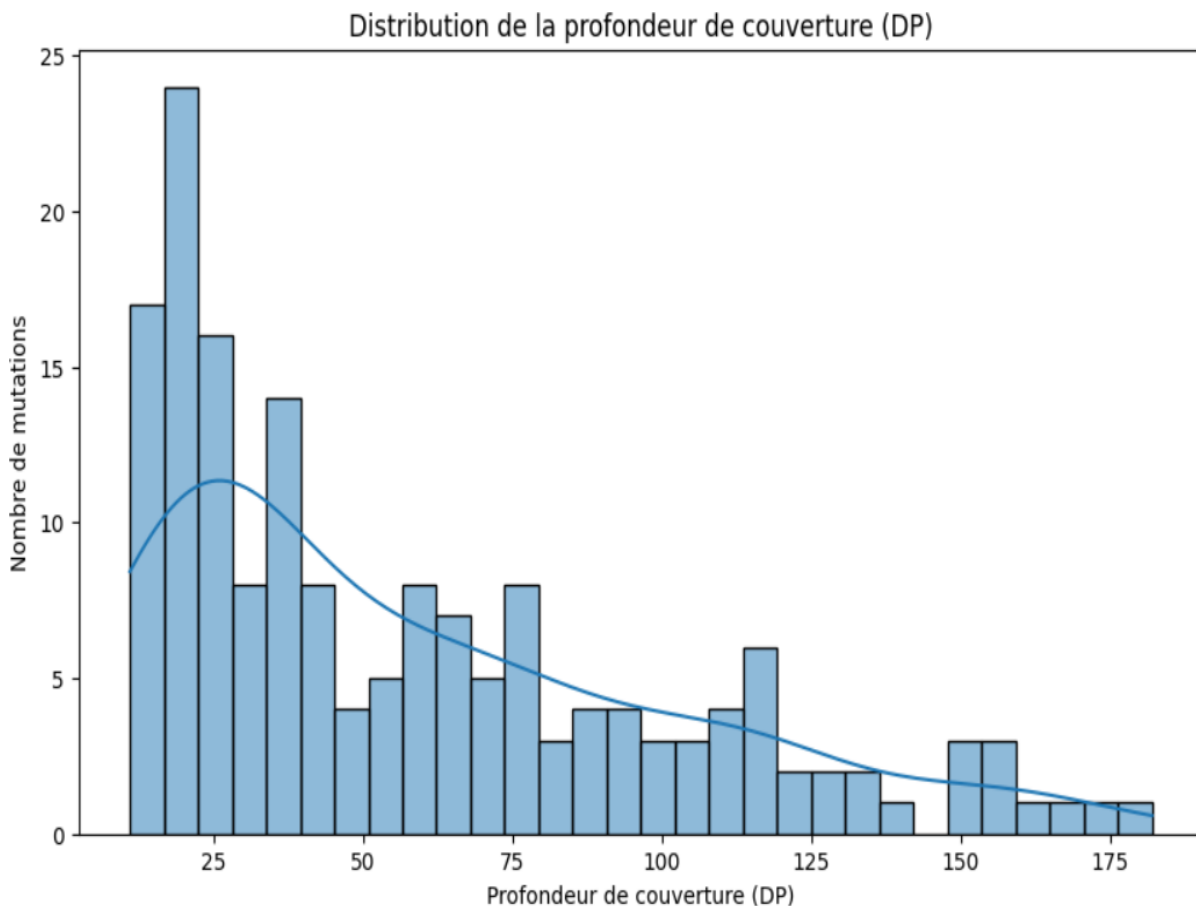


Figure 54 La visualisation de la distribution de la profondeur de couverture sous forme d'un histogramme

## Interpretation :

Ce graphique histogramme montre la distribution de la profondeur de couverture 'DP' pour les mutations détectées. La majorité des mutations ont une profondeur de couverture située autour de 25, avec une décroissance progressive pour des profondeurs plus élevées. La superposition de la courbe KDE permet de visualiser cette tendance avec davantage de clarté. Une profondeur de couverture plus élevée indique une plus grande confiance dans les mutations détectées, bien que cela puisse également dépendre des spécifications techniques du séquençage. La forte concentration de valeurs autour de 25 pourrait être liée aux paramètres de l'analyse bio-informatique ou à la technologie utilisée pour générer les données.

### 6.5.3. Sélection des mutations du gène TP53

Dans ce code, les mutations associées au gène TP53 sont extraites du DataFrame mutations significatives en appliquant un filtre sur la colonne GENE. Cela permet de se concentrer uniquement sur les mutations affectant ce gène spécifique, essentiel dans de nombreuses études sur le cancer.

```
tp53_mutations = mutations_significatives[mutations_significatives['GENE'] == 'TP53']
print(tp53_mutations.head())
```

	CHROM	POS	ID	REF	ALT	QUAL	FILTER	\
35804	chr17	7673776	None	G	A	None	PASS	
35805	chr17	7674326	None	C	G	None	PASS	
35808	chr17	7674797	None	T	C	None	PASS	
35809	chr17	7674978	None	G	A	None	PASS	
35810	chr17	7675327	None	C	T	None	PASS	

	INFO	DP	GENE	\
35804	[AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...	76	TP53	
35805	[AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...	15	TP53	
35808	[AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...	17	TP53	
35809	[AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...	59	TP53	
35810	[AS_SB_TABLE, DP, ECNT, MBQ, MFRL, MMQ, MPOS, ...	88	TP53	

	MUTATION_TYPE
35804	Substitution
35805	Substitution
35808	Substitution
35809	Substitution
35810	Substitution

Figure 55 Code pour la sélection des mutations associées au gène TP53

#### 6.5.4. Visualisation de la distribution des positions des mutations

Un histogramme est généré pour visualiser la distribution des positions des mutations sur le génome. L'axe des x représente les positions sur le chromosome, tandis que l'axe des y affiche le nombre de mutations dans chaque position. Une courbe de densité (KDE) est superposée pour mieux observer la concentration des mutations sur le génome.

```
plt.figure(figsize=(10, 6))
sns.histplot(mutations_significatives['POS'], bins=50, kde=True)
plt.title('Distribution des positions des mutations sur le génome')
plt.xlabel('Position sur le chromosome')
plt.ylabel('Nombre de mutations')
plt.show()
```

Figure 56 Code d'affichage de distribution des positions des mutations

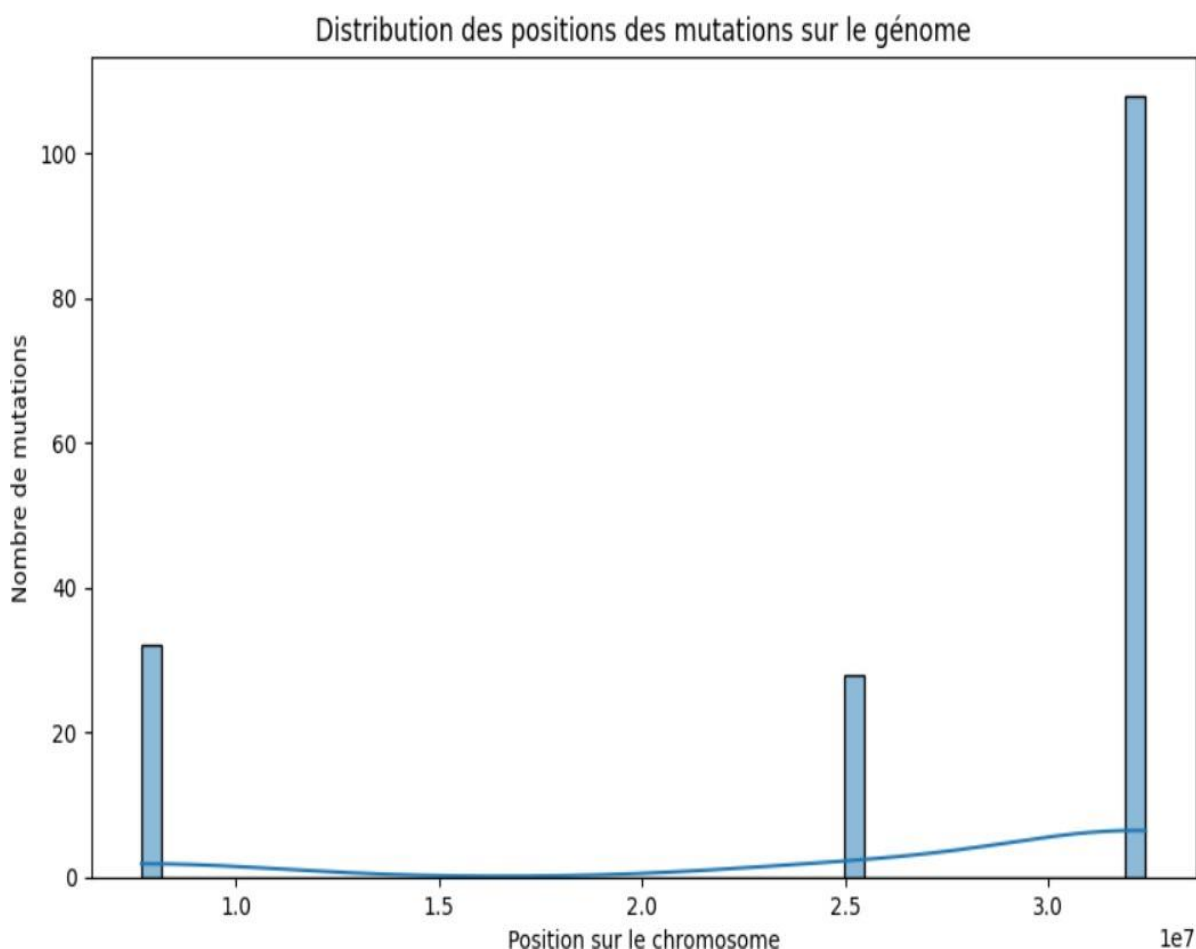


Figure 57 Histogramme de distribution des position des mutation sur le génome

### 6.5.5. Installation de matplotlib-venn pour visualisation

Cette ligne installe la bibliothèque matplotlib-venn, qui permet de créer des diagrammes de Venn pour visualiser les relations entre les ensembles de données.

```
!pip install matplotlib-venn

Requirement already satisfied: matplotlib-venn in /usr/local/lib/python3.10/dist-packages (1.1.1)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from matplotlib-venn) (3.10.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from matplotlib-venn) (1.26.4)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from matplotlib-venn) (1.13.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->matplotlib-venn) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->matplotlib-venn) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->matplotlib-venn) (4.55.3)
Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->matplotlib-venn) (1.4.8)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->matplotlib-venn) (24.2)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.10/dist-packages (from matplotlib->matplotlib-venn) (11.1.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->matplotlib-venn) (3.2.1)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib->matplotlib-venn) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib->matplotlib-venn) (1.17.0)
```

Figure 58 l'installation de matplotlib-venn

### 6.5.6. Dessiner un diagramme de Venn pour visualiser les intersections

Ce code génère un diagramme de Venn pour comparer les ensembles de mutations dans mutations\_significatives et all\_genes\_data. La fonction venn2 prend comme entrée la taille de chaque ensemble et de leur intersection. Elle produit un visuel facile à interpréter des relations entre les ensembles.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Créer un tableau croisé de la distribution des mutations (types de mutations x gènes)
mutation_type_gene_freq = mutations_significatives.groupby(['GENE_SYMBOL', 'MUTATION_TYPE']).size().unstack(fill_value=0)

# Afficher une Heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(mutation_type_gene_freq, annot=True, cmap="YlGnBu", cbar=True)
plt.title("Heatmap: Distribution des mutations par type et gène")
plt.xlabel("Type de mutation")
plt.ylabel("Gène")
plt.show()
```

Figure 59 code de génération du diagramme de venn

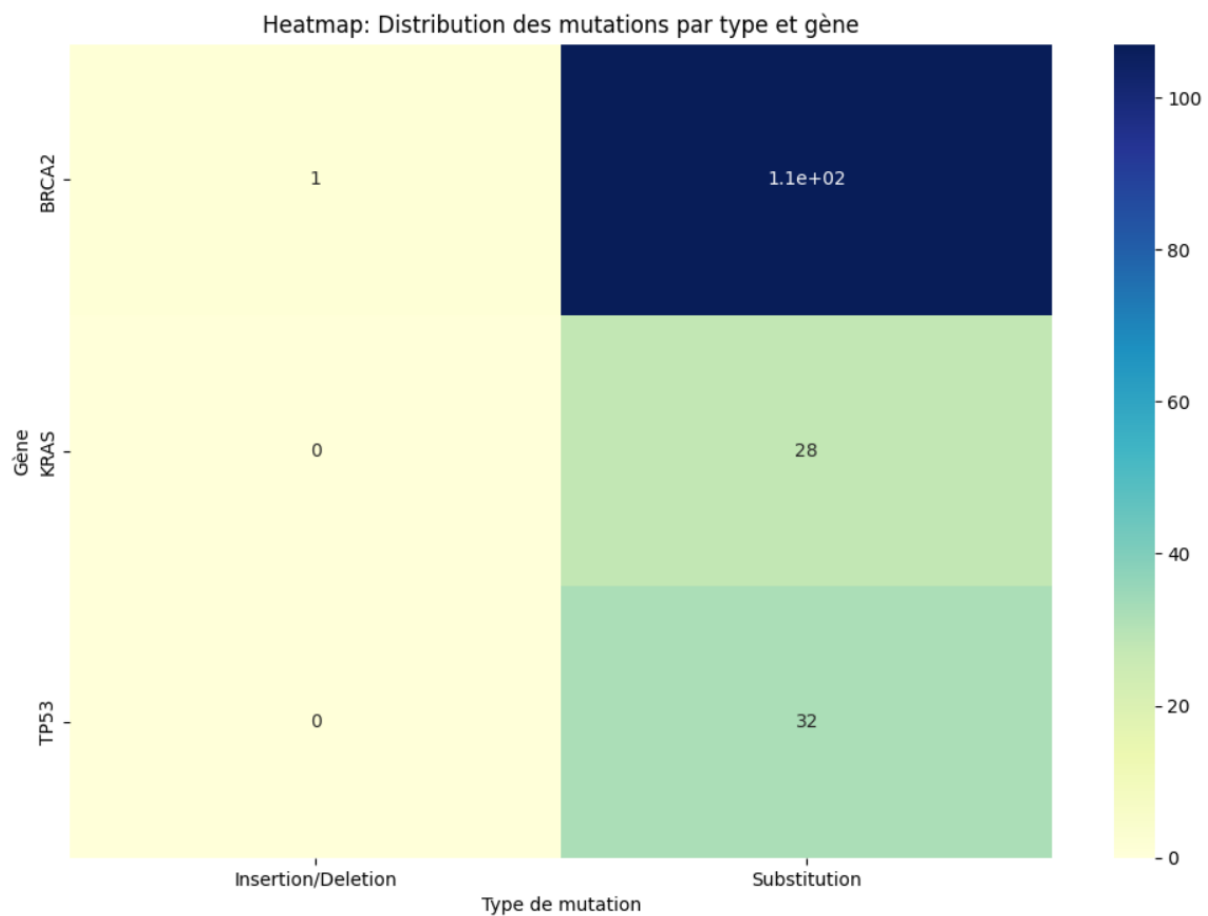


Figure 60 Heatmap : distribution des mutations par type et gène

### Interpretation :

La heatmap présente la distribution des mutations pour trois gènes clés 'BRCA2, KRAS, et TP53' en fonction de deux types de mutations : les substitutions et les insertions/délétions. Les résultats montrent une prédominance claire des substitutions comme principal type de mutation pour ces gènes. Le gène BRCA2 se distingue par un nombre élevé de mutations par substitution 'environ 110', tandis que KRAS et TP53 présentent respectivement 28 et 32 substitutions. Les insertions/délétions sont très rares, n'apparaissant qu'une seule fois pour le gène BRCA2 et étant totalement absentes pour KRAS et TP53. Ces observations suggèrent que les mécanismes sous-jacents aux mutations varient selon les gènes, BRCA2 étant particulièrement vulnérable aux substitutions, probablement en raison de son rôle dans la réparation de l'ADN. Cette visualisation met en évidence la spécificité des mutations selon les gènes, soulignant l'importance des substitutions dans les processus pathologiques, en particulier dans le contexte des maladies comme le cancer.



### 6.5.7. Analyse et Visualisation des SNPs entre Fichiers VCF avec PCA et Heatmap

Ce code traite des fichiers VCF (Variant Call Format) contenant des données sur des mutations génétiques, telles que des SNPs (Single Nucleotide Polymorphisms). Il commence par lister les chemins des fichiers VCF à analyser. Ensuite, il définit une fonction pour extraire les SNPs à partir de ces fichiers en parcourant chaque ligne et en extrayant les informations chromosomiques et de position. Après avoir extrait les SNPs, le code les compare entre les différents fichiers, en identifiant les SNPs communs et uniques. Une DataFrame est ensuite construite pour visualiser la présence des SNPs dans chaque fichier, et une heatmap est générée pour afficher cette comparaison de manière visuelle. Enfin, le code effectue une analyse en composantes principales (PCA) pour réduire la dimensionnalité des données et obtenir une représentation graphique des relations entre les fichiers, en projetant les résultats dans un espace de deux dimensions.

```
File 1: 5617 SNPs extraits  
File 2: 86305 SNPs extraits  
File 3: 187698 SNPs extraits  
File 4: 110396 SNPs extraits  
File 5: 178604 SNPs extraits  
File 6: 267910 SNPs extraits  
File 7: 1413 SNPs extraits  
Total SNPs (tous fichiers confondus) : 826981
```

Figure 61 Le nombre des SNPs extraits d'après fichiers les VCF

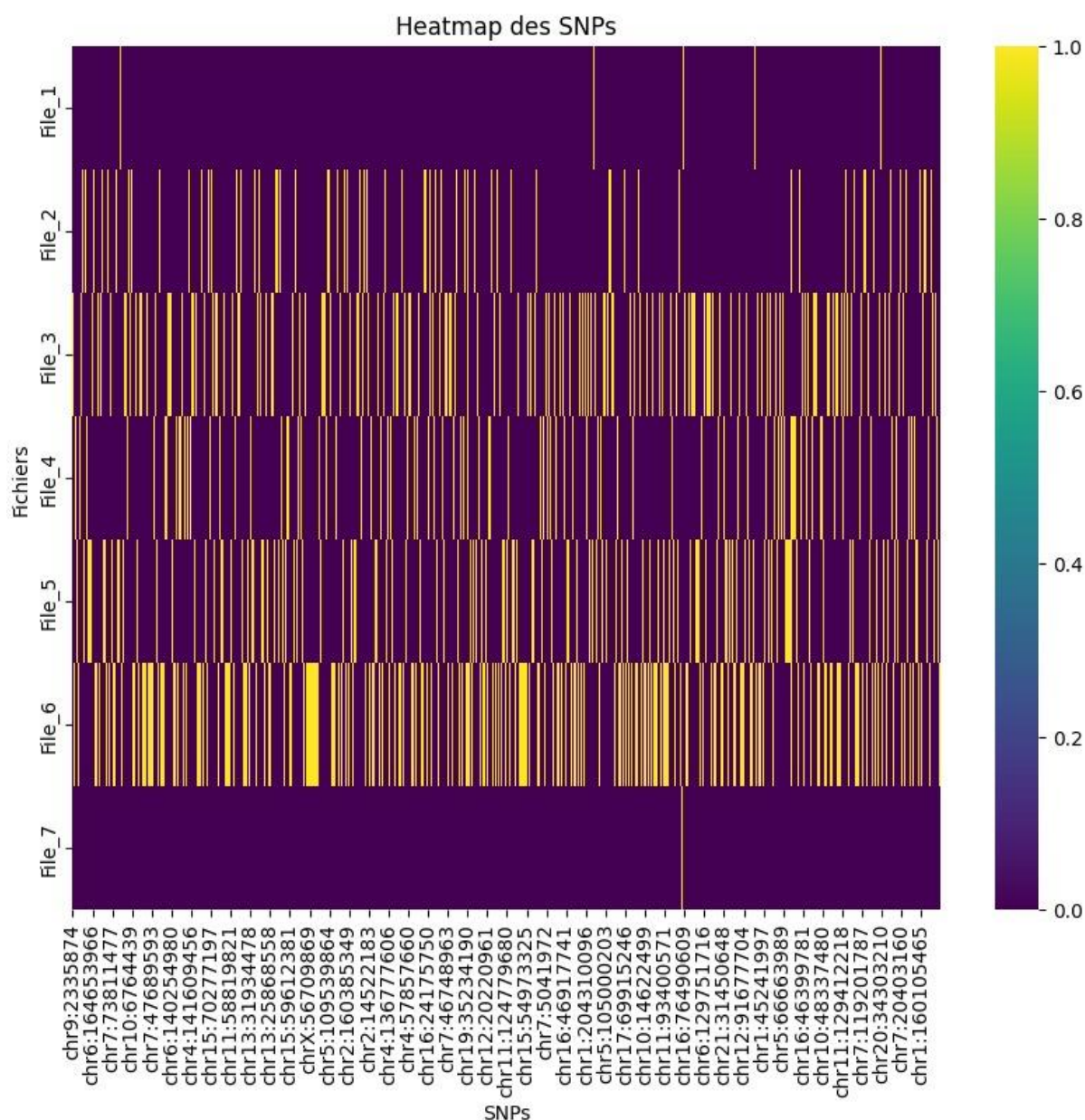


Figure 62 Heatmap du distribution des SNPs



### Interprétation :

La heatmap illustre la distribution des SNPs 'Single Nucleotide Polymorphisms' parmi plusieurs fichiers 'File1 à File7'. Chaque colonne représente un SNP identifié par son chromosome et sa position, tandis que chaque ligne correspond à un fichier spécifique. Les couleurs varient du violet 'absence de SNP, valeur proche de 0' au jaune 'présence de SNP, valeur proche de 1', permettant de visualiser la présence ou l'absence de chaque SNP dans les fichiers. L'analyse révèle une variabilité significative des SNPs entre les fichiers : certains SNPs sont partagés par plusieurs fichiers, tandis que d'autres sont spécifiques à un ou quelques fichiers. Notamment, le fichier File7 se distingue par une absence quasi totale de

SNPs, ce qui pourrait indiquer une faible couverture génomique ou des caractéristiques particulières de cet échantillon. De plus, certains SNPs apparaissent fréquemment dans plusieurs fichiers, suggérant des variations génétiques communes ou conservées. Cette visualisation offre un aperçu clair des similarités et différences entre les échantillons, et peut servir de base pour des études génétiques approfondies sur les variations communes et uniques.

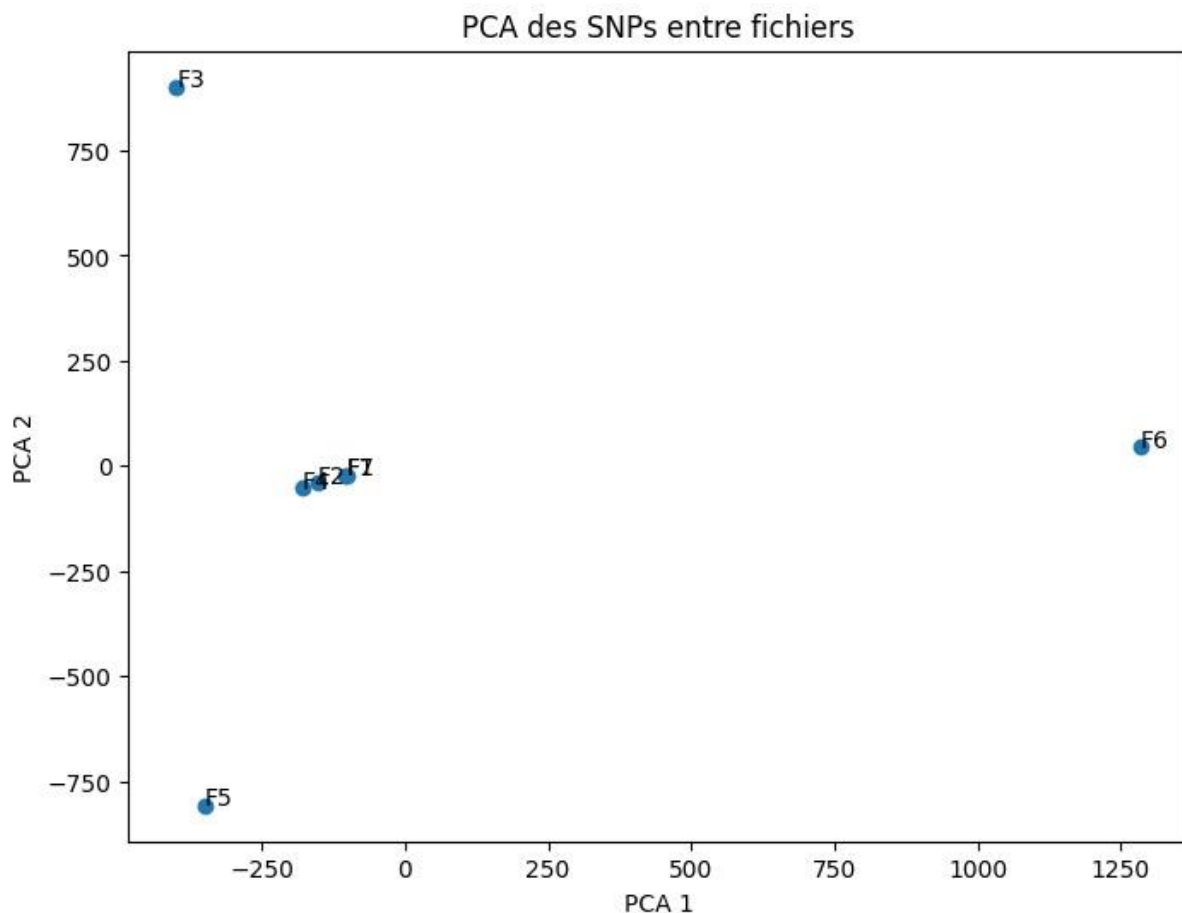


Figure 63 Visualisation de PCA des SNPs entre fichiers

#### **Interprétation :**

L'analyse réalisée porte sur les SNPs extraits de plusieurs fichiers VCF. Après avoir identifié les SNPs communs et uniques entre les différents fichiers, une Analyse en Composantes Principales a été effectuée pour visualiser les relations entre ces fichiers. Les résultats du PCA, projetés dans un espace bidimensionnel, révèlent des regroupements et des distinctions importantes. Les fichiers F1, F2 et F7 forment un cluster proche, indiquant une forte similarité dans leurs SNPs, probablement due à une origine commune ou des

caractéristiques génétiques similaires. À l'inverse, les fichiers F3, F5 et F6 se distinguent nettement des autres, avec des positions éloignées dans le graphique, suggérant des variations génétiques uniques ou une diversité marquée dans leurs SNPs respectifs. Cette analyse met en évidence la diversité génétique entre les fichiers étudiés et permet d'identifier des groupes d'échantillons partageant des similitudes génétiques.

### 6.5.8. Visualisation des SNPs avec Diagramme de Venn pour 2 ou 3 Fichiers

Ce code crée un diagramme de Venn pour comparer les SNPs extraits de 2 ou 3 fichiers VCF. La fonction `plot_venn` prend en entrée une liste d'ensembles de SNPs et les étiquettes correspondantes aux fichiers. Selon le nombre de fichiers (2 ou 3), elle génère un diagramme de Venn adapté. Le titre du graphique est "Diagramme de Venn des SNPs", et les intersections des SNPs entre les fichiers sont visualisées, ce qui permet d'examiner les similitudes et les différences dans les données.

```
from matplotlib_venn import venn2, venn3
from itertools import combinations

# Fonction pour tracer un diagramme de Venn
def plot_venn(snps_sets, labels):
    num_files = len(snps_sets)

    if num_files == 2:
        # Diagramme de Venn pour 2 fichiers
        venn = venn2([snps_sets[0], snps_sets[1]], set_labels=labels)
    elif num_files == 3:
        # Diagramme de Venn pour 3 fichiers
        venn = venn3([snps_sets[0], snps_sets[1], snps_sets[2]], set_labels=labels)
    else:
        print("Diagramme de Venn supporté seulement pour 2 ou 3 ensembles.")
        return

    plt.title("Diagramme de Venn des SNPs")
    plt.show()

# Exemple pour 3 fichiers (vous pouvez changer selon vos besoins)
plot_venn(snps_sets[:3], labels=["Fichier 1", "Fichier 2", "Fichier 3"])
```

Figure 64 code de creation de diagramme de Venn pour la comparaison des SNPs extraits

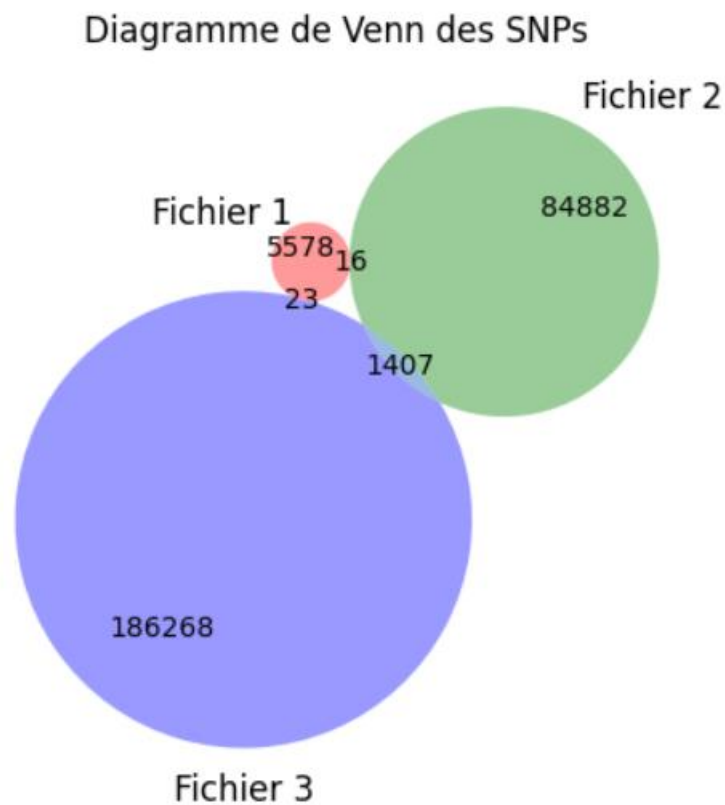


Figure 65 Diagramme de Venn des SNPs (fichiers 1,2 et 3)

### 6.5.9. Visualisation des SNPs avec Diagramme de Venn pour les Fichiers 3, 4 et 5

Ce code génère un diagramme de Venn pour comparer les SNPs extraits des fichiers 3, 4 et 5. La fonction `plot_venn` prend en entrée une liste contenant les ensembles de SNPs de ces trois fichiers et les étiquettes correspondantes. Elle crée un diagramme de Venn pour afficher les intersections entre ces trois ensembles de données. Le titre du graphique est "Diagramme de Venn des SNPs pour les fichiers 3, 4 et 5", et il permet de visualiser les SNPs partagés ou uniques entre ces fichiers.

### Diagramme de Venn des SNPs pour les fichiers 3, 4 et 5

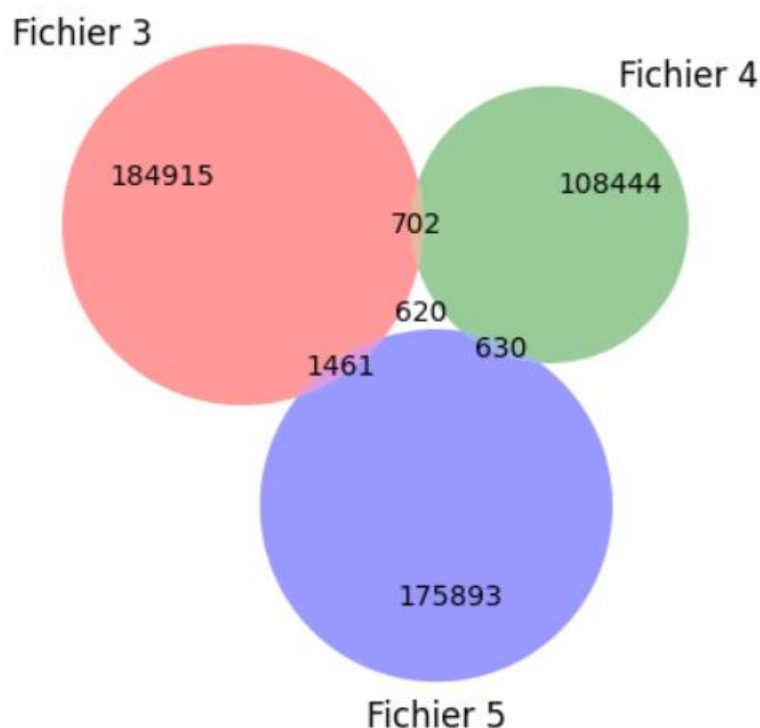


Figure 66 Diagramme de Venn des SNPs (fichiers 3,4 et 5)

#### 6.5.10. Analyse des SNPs avec un Diagramme UpSet à partir de Fichiers VCF

Ce code effectue une analyse des SNPs extraits de plusieurs fichiers VCF en utilisant la bibliothèque cyvcf2. La fonction `extract_snps_from_vcf` extrait les SNPs de chaque fichier VCF en les identifiant par leur chromosome et leur position. Ensuite, la fonction `prepare_upset_data` génère les données nécessaires pour un diagramme UpSet, qui visualise les intersections entre les SNPs présents dans les différents échantillons. Le graphique UpSet ainsi généré montre la fréquence des SNPs partagés entre les échantillons, avec des annotations sur les barres indiquant le nombre d'occurrences pour chaque combinaison de SNPs. Le titre du graphique est "UpSet Plot of SNPs from VCF Files", fournissant une représentation claire des SNPs communs et uniques dans les fichiers VCF.

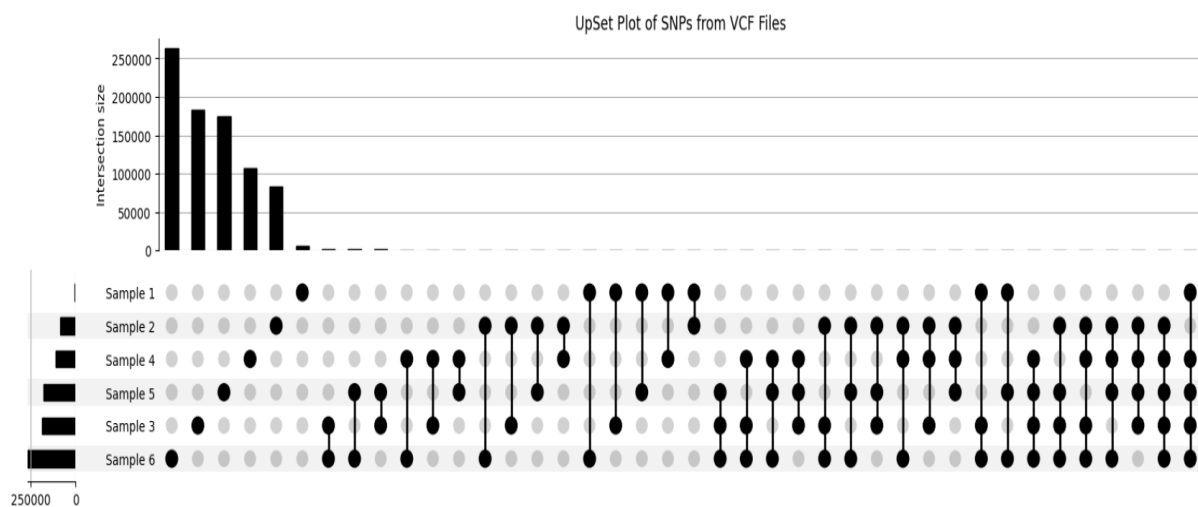


Figure 67 UpSet Plot des intersections de SNPs entre six échantillons extraits de fichiers VCF

### **Interprétation :**

L'UpSet Plot présenté illustre les intersections de SNPs entre six échantillons extraits de fichiers VCF. Les barres verticales, situées en haut du graphique, indiquent la taille des intersections, c'est-à-dire le nombre de SNPs partagés entre différents groupes d'échantillons. Par exemple, la plus grande intersection '263,416 SNPs' correspond aux SNPs spécifiques à un seul échantillon, Sample 1. Les intersections suivantes montrent des SNPs partagés entre deux ou plusieurs échantillons, avec des tailles décroissantes.

Le graphique inférieur, constitué de points connectés, montre visuellement les échantillons impliqués dans chaque intersection. Les SNPs de la deuxième barre verticale '174,250 SNPs' sont partagés uniquement entre Sample 1 et Sample 2. Les SNPs communs à tous les échantillons 'intersection impliquant les six points connectés' sont très rares, indiquant une diversité génétique significative entre les échantillons.

Enfin, les barres horizontales à gauche du graphique indiquent la taille totale des SNPs pour chaque échantillon. Sample 1 présente le plus grand nombre de SNPs individuels, tandis que Sample 6 en contient le moins. Cette visualisation met en évidence des groupes d'échantillons partageant des SNPs spécifiques, tout en soulignant les distinctions génétiques entre les échantillons, rendant cette analyse utile pour explorer la diversité génétique et les relations entre ces derniers.

## 7. Conclusion Générale

En conclusion, ce projet d'Identification et Analyse des Mutations Somatiques dans le Cancer Colorectal met en évidence le rôle crucial des approches bio-informatiques dans l'étude des variations génétiques. En combinant des techniques avancées, telles que l'analyse des SNPs et indels, la cartographie génomique, les heatmaps, l'Analyse en Composantes Principales (PCA) et les UpSet plots, nous avons pu explorer en profondeur la diversité génétique et identifier des mutations significatives dans des gènes clés (BRCA2, TP53, KRAS). Ces outils ont permis d'extraire des données précises et fiables, malgré des défis liés à la qualité variable des échantillons, et d'identifier des biomarqueurs potentiels qui pourraient transformer les stratégies de diagnostic et de traitement du cancer colorectal.

L'utilisation de la bio-informatique a non seulement optimisé l'analyse massive des données génomiques, mais elle a également contribué à réduire le temps et les ressources nécessaires pour atteindre des résultats exploitables. Ce projet souligne ainsi l'importance de la bio-informatique dans l'évolution de la recherche médicale, en permettant une meilleure compréhension des mécanismes biologiques complexes et en favorisant le développement de thérapies ciblées et personnalisées. En intégrant des techniques bio-informatiques de pointe dans la recherche médicale, nous contribuons directement à l'amélioration des soins aux patients et au progrès global de la médecine de précision.



## 8. Ressources et Références

### Bases de Données

- ✚ NCBI SRA (Sequence Read Archive): <https://www.ncbi.nlm.nih.gov/sra>
- ✚ ENA (European Nucleotide Archive): <https://www.ebi.ac.uk/ena>

### Outils Bioinformatiques

- ✚ FastQC: <https://www.bioinformatics.babraham.ac.uk/projects/fastqc/>
- ✚ Trimmomatic: <http://www.usadellab.org/cms/?page=trimmomatic>
- ✚ BWA (Burrows-Wheeler Aligner): <http://bio-bwa.sourceforge.net/>
- ✚ Samtools: <http://www.htslib.org/>
- ✚ GATK (Genome Analysis Toolkit): <https://gatk.broadinstitute.org/>
- ✚ Picard: <https://broadinstitute.github.io/picard/>

