Rapport de stage

Master 2 BI parcours BBS - Bioinformatique et Biologie des Systèmes

---

Analyse et traitement des données de séquençage ARN direct produites par la technologie Nanopore pour l'étude de l'épitranscriptome

---

Emma RODRIGUEZ

Stage co-encadré par :

Christine GASPIN - Nathalie VIALANEIX – Benjamin CHARLIER

2025

Résumé

Plus de 170 modifications chimiques ont été identifiées dans les ARN messagers (ARNm) et non codants (ARNnc), l'une des plus abondantes étant la méthylation en position N6 de l'adénosine (m6A). Ces modifications épitranscriptomiques modulent des propriétés essentielles des ARN (stabilité, structure, localisation, interactions moléculaires) et constituent un niveau supplémentaire de régulation post-transcriptionnelle de l'expression génique. Le séquençage direct de l'ARN par la technologie Oxford Nanopore représente une avancée significative en permettant l'analyse de transcrits natifs en pleine longueur, sans rétrotranscription ni amplification. Cette approche conserve les modifications endogènes, détectables via les altérations du signal électrique générées lors du passage dans le nanopore. Cependant, plusieurs verrous techniques subsistent. La détection conjointe de modifications multiples, leur localisation précise et leur quantification fiable nécessitent encore le développement de méthodes bioinformatiques robustes.

C'est dans ce contexte que s'inscrit mon stage réalisé au sein de l'unité MIAT d'INRAE (équipe SaAB). L'objectif était de développer un pipeline d'analyse pour la détection de modifications d'ARN à partir de données de séquençage ARN direct Nanopore. J'ai initié une chaîne de traitement permettant d'exploiter les variations du signal électrique brut associées aux séquences alignées sur la séquence de référence. Je me suis tout d'abord familiarisé avec la problématique, les formats des données ARN direct et des outils bioinformatiques spécifiques aux données de séquençage ARN direct (long read). J'ai développé un script en Python pour produire des visualisations comparatives entre lectures modifiées et non modifiées et maîtriser les transformations du signal.

Dans le cadre de ce stage, j'ai pu gérer des volumes de données importants (centaines de gigaoctets), mobiliser des outils récents et assurer la reproductibilité de mes analyses (scripts versionnés, environnement Genotoul-Bioinfo). J'ai initié une chaîne de traitement permettant de manipuler et visualiser les caractéristiques du signal brut de séquençage posant les bases pour des développements méthodologiques futurs et des analyses à plus grande échelle.

Abstract

More than 170 chemical modifications have been identified in messenger (mRNA) and non-coding (nnc) RNAs, one of the most abundant being the N6 methylation of adenosine (m6A). These epitranscriptomic modifications modulate essential RNA properties (stability, structure, localization, molecular interactions) and constitute an additional level of post-transcriptional regulation of gene expression. Direct RNA sequencing by the Oxford Nanopore technology represents a significant step forward in allowing full-length native transcripts to be analyzed without retrotranscription or amplification. This approach preserves the endogenous changes, detectable via the alterations of the electrical signal generated during the passage in the nanopore. However, several technical locks remain. The joint detection of multiple changes, their precise location and reliable quantification still require the development of robust bioinformatics methods.

It is in this context that my internship carried out within the MIAT unit of INRAE (SaAB team) fits. The objective was to develop an analysis pipeline for the detection of RNA modifications from direct Nanopore RNA sequencing data. I initiated a processing chain to exploit the variations in the raw electrical signal associated with sequences aligned to the reference sequence. I first became familiar with the problem, the formats of direct RNA data and bioinformatics tools specific to direct RNA sequencing (long read) data. I developed a Python script to produce comparative visualizations between modified and unmodified readings and master the signal transformations.

As part of this internship, I was able to manage large volumes of data (hundreds of gigabytes), mobilize recent tools and ensure the reproducibility of my analyses (versioned scripts, Genotoul-Bioinfo environment). I initiated a processing chain to manipulate and visualize the raw signal characteristics of sequencing laying the foundation for future methodological developments and larger-scale analyses.

Sommaire

I. Introduction

A. Présentation du laboratoire

1. INRAE

INRAE, l'Institut national de recherche pour l'agriculture, l'alimentation et l'environnement, a été créé le 1er janvier 2020. Il résulte de la fusion de deux établissements publics : l'INRA (Institut national de la recherche agronomique) et l'Irstea (Institut national de recherche en sciences et technologies pour l'environnement et l'agriculture).

Cette fusion permet à INRAE de couvrir un spectre de recherche particulièrement large, allant des sciences du vivant à la gestion durable des territoires, en passant par la biodiversité, l'agriculture numérique ou encore la prévention des risques environnementaux. L'institut remplit cinq grandes missions, qui sont : produire des connaissances scientifiques, développer des innovations, apporter un appui aux politiques publiques, participer aux débats scientifiques et former à la recherche.

Aujourd'hui, INRAE s'organise en 14 départements scientifiques répartis sur 18 centres, regroupant 273 unités de recherche, de service ou expérimentales. L'institut se distingue également par sa forte dimension internationale, avec près de six publications sur dix (58 %) co-signées avec des partenaires étrangers.

2. L'unité MIAT

L'unité MIAT (Unité de Mathématiques et Informatique Appliquées de Toulouse, UR875) est une unité de recherche d'INRAE, rattachée au département Mathématiques et Numérique (MATHNUM). Sa mission principale est de développer des approches in *silico* pour répondre aux défis posés par les sciences du vivant, en combinant modélisation, calcul et analyse de données. L'unité travaille sur des problématiques biologiques et agro-environnementales, à différentes échelles : du gène jusqu'à l'écosystème. Elle mobilise des compétences en mathématiques appliquées, en statistique et en informatique pour structurer les données, modéliser les systèmes, simuler leur dynamique et optimiser leur fonctionnement. L'unité s'inscrit à la fois dans une logique de recherche fondamentale et de valorisation appliquée, en lien avec des biologistes, agronomes et écologues.

L'unité MIAT s'articule autour de deux équipes de recherche :

- L'équipe SCIDyn (Simulation, Contrôle et Inférence de Dynamiques Agro-environnementales et Biologiques) travaille sur la modélisation dynamique des agroécosystèmes et systèmes biologiques, avec des outils de simulation, de contrôle et d'inférence pour analyser et prédire leurs évolutions dans un contexte complexe et incertain.

- L'équipe SaAB (Statistique et Algorithmique pour la Biologie) développe des méthodes statistiques et algorithmiques pour l'analyse de données biologiques de grande dimension (transcriptomique, génomique, etc.), avec un accent sur l'intégration de données et la modélisation probabiliste.

L'unité MIAT s'appuie également sur trois plateformes technologiques :

- La plateforme Genotoul-Bioinfo est une plateforme de bio-informatique, membre du GIS GENOTOUL - Génopole Toulouse Midi-Pyrénées et de l'infrastructure nationale de recherche Institut Français de Bio-informatique (IFB) ;

- La plateforme RECORD est une plateforme de modélisation et de simulation des agro-écosystèmes

- La plateforme SIGENAE est une autre plateforme de bio-informatique orientée vers l'analyse des génomes des animaux d'élevage

Mon stage au sein de l'unité MIAT a été encadré par Mme Nathalie VIALANEIX, Mme Christine GASPIN, toutes deux directrices de recherche au sein de l'unité MIAT, et par Benjamin CHARLIER, chargé de recherche. Il a été effectué au sein de l'équipe « Statistique et Algorithmique pour la Biologie » (SaAB) et s'intitule « Analyse et traitement des données de séquençage ARN direct produites par la technologie Nanopore pour l'étude de l'épitranscriptome »

B. Contexte biologique : Etude de l'épitranscriptome

L'acide ribonucléique (ARN) est une molécule issue de la transcription de l'ADN. Il est composé de quatre bases azotées : l'adénine (A), la cytosine (C), la guanine (G) et l'uracile (U), qui remplace la thymine présente dans l'ADN (***figure 1***). L'épitranscriptome correspond aux modifications chimiques qui surviennent sur les ARN après leur transcription, sans modification de leur séquence nucléotidique **(Saletore *et al*, 2012)**. Elle concerne donc des modifications post-transcriptionnelles qui influencent le comportement des ARN (stabilité, localisation dans la cellule, traduction en protéines, dégradation). Il s'agit d'un niveau distinct mais complémentaire, de

régulation de l'expression génique.

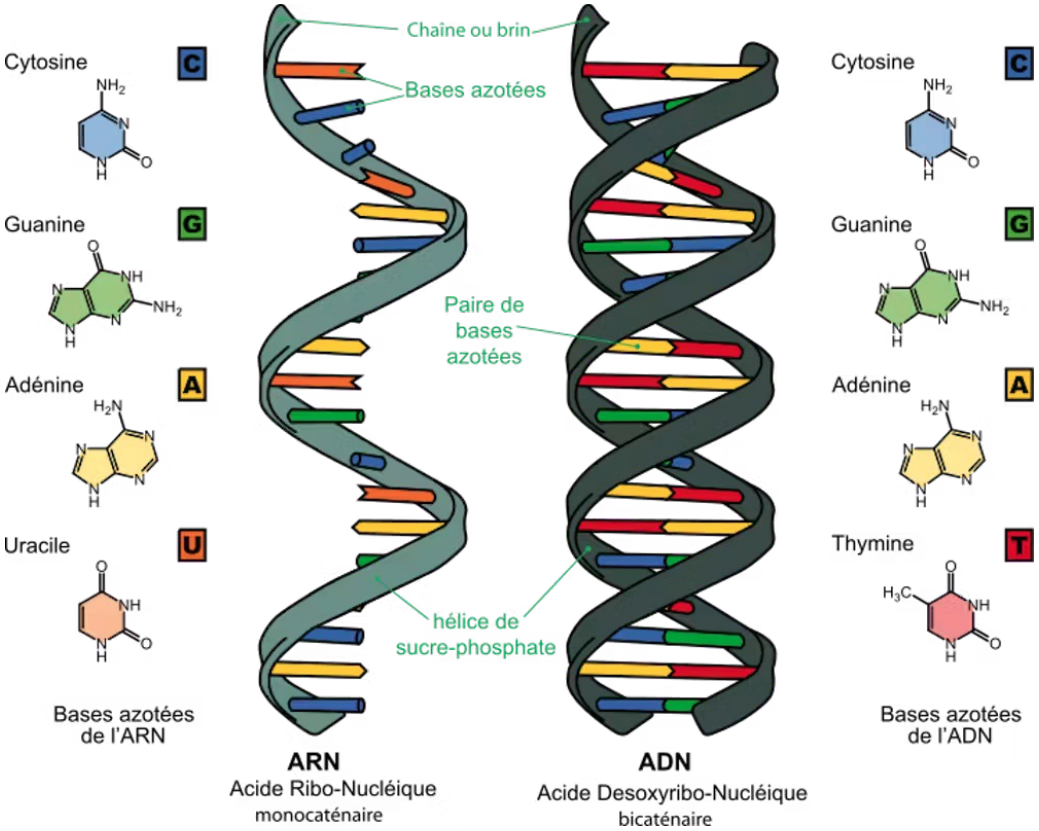


***Figure 1* - <u>Structure comparée de l'ARN et de l'ADN</u>**. L'ARN se distingue de l'ADN par la présence de l'uracile (U) à la place de la thymine (T), ainsi que par un simple brin au lieu d'une double hélice. Le sucre diffère également avec un ribose pour l'ARN contre un désoxyribose pour l'ADN.

La première modification de nucléoside d'ARN a été identifiée dans les années 1950 **(Davis & Allen, 1957)**. À ce jour, plus de 170 modifications distinctes d'ARN sont connues **(Cappannini *et al*, 2024)**, elles se produisent dans les ARN messagers (ARNm), ainsi que dans les ARN non codants tels que les ARN ribosomiques (ARNr), les ARN de transfert (ARNt) ou encore les petits ARN nucléaires (snARN). Leur distribution étendue dans la cellule témoigne de leur importance fonctionnelle. Parmi ces modifications, la N6-méthyladénosine (m6A) est la plus fréquente et la mieux caractérisée ; Elle fera l'objet d'une attention particulière dans la suite de ce travail. D'autres modification sont également répandues comme la 5-méthylcytosine (m5C), la pseudouridine (Ψ), l'inosine (I) et la 1-méthyladénosine (m1A).

Sur le plan biologique, ces modifications influencent des processus comme le développement, la réponse cellulaire au stress, le cycle cellulaire, ainsi que l'apparition de certaines pathologies (notamment le cancer) **(He & He, 2021)**.

C. Identification des modifications d'ARN - Méthodes existantes

Il existe plusieurs méthodes pour détecter les modifications sur l'ARN. Elles n'offrent pas toutes la même précision ni le même niveau d'information Cette section présente les principales méthodes utilisées aujourd'hui, leurs forces et leurs limites.

Approches par recherche de motifs. Cette approche est purement informatique et repose sur l'analyse de séquences génomiques ou transcriptomiques, sans données expérimentales. Elle consiste à rechercher des motifs consensus connus (ex : DRACH pour m6A) **(Hu *et al*, 2022)** associés à certaines modifications afin de prédire des positions candidates dans l'ARN susceptibles d'être modifiées. Cette méthode a notamment contribué à mettre en évidence le grand nombre de 2'_O-méthylations et de pseudoUridylations des ARNr par la recherche de snoARN à boîtes C/D et H/ACA **(Linder *et al*, 2015)**. La méthode est simple, rapide et accessible car elle ne nécessite aucune donnée expérimentale ni traitement biologique. Elle est accessible à l'échelle de la séquence génomique, fonctionne sur n'importe quel jeu de séquences et peut être utilisée en complément de méthodes expérimentales. Tout ceci la rend applicable à grande échelle, permettant l'analyse de transcriptomes entiers ou de bases de données génomiques. Cependant, elle ne permet pas de détecter des modifications non caractérisées : on doit nécessairement connaître le motif associé à la modification pour l'utiliser. De plus, la justesse de l'approche est variable en fonction du motif ; les motifs cibles peuvent être présents sans que la position correspondante sur l'ARN ne soit modifiée. Elle rend impossible l'estimation de la stœchiométrie (proportion de modifications par site) et ne tient pas compte du contexte biologique ou structurel, c'est-à-dire qu'elle ne tient pas compte du fait qu'une même séquence peut être modifiée ou non selon l'environnement cellulaire, le type de tissu, l'état de différenciation ou encore la structure secondaire locale de l'ARN.

Approches par spectrométrie de masse. Ensuite, les premières méthodes de détection directe apparues sont basées sur la spectrométrie de masse et se distinguent par deux approches **(Herbert *et al*, 2024)**. La première approche consiste en un profilage global (*global profiling*), par hydrolyse de l'ARN, qui va séparer et détecter des nucléosides modifiés avec la spectrométrie de masse (LC-MS/MS, UHPLC-MS). La deuxième approche produit une quantification ultrasensible qui utilise des standards isotopiques pour quantifier précisément chaque modification à très faible concentration via la spectrométrie de masse (isotope dilution MS, attomole quantification). Outre la détection directe, ces méthodes basées sur la spectrométrie de masse sont très reproductibles avec une très haute sensibilité (seuils de détection à l'attomole). Elles permettent une quantification

absolue et fiable de chaque modification (grâce aux standards isotopiques) et possèdent un large spectre de détection de modifications pouvant être mesurées simultanément. Leur indépendance vis-à-vis de la séquence leur permet de détecter des modifications sur tous types d'ARN. Malheureusement, le problème majeur est qu'elles ne permettent pas de connaître les positions des modifications dans le transcrit. Ceci notamment dû à la manipulation biologique (hydrolyse) qui engendre une perte d'information structurelle (structure, position, co-modifications).

Approches par séquençage à lectures courtes. L'arrivée du séquençage à haut débit de lecture courte NGS (*next-generation sequencing*) a permis de créer de nouvelles méthodes de détection indirectes des modifications, qui nécessitent un enrichissement ou un traitement chimique/enzymatique avant le séquençage. Ces méthodes peuvent se baser sur l'immunoprécipitation, qui demande l'utilisation d'anticorps spécifiques à une modification pour enrichir les fragments la portant **(Meyer *et al*, 2012)**, suivie d'un séquençage (comme m6A-seq) **(Dominissini *et al*, 2012)**. Certaines méthodes améliorent la résolution en combinant immunoprécipitation et liaisons croisées (*crosslinking*), où des liaisons covalentes ARN-anticorps sont induites par rayonnement ultraviolet. Cela permet de localiser certaines modifications au nucléotide près, comme dans les protocoles miCLIP **(Hussain *et al*, 2013)**. La possibilité de combiner plusieurs approches (immunoprécipitation, traitement chimique **(Schwartz *et al*, 2014)**, enzymes) permet une bonne couverture du transcriptome pour une analyse globale. Ces approches permettent de détecter des modifications *de novo* (Inconnue jusqu'alors). Elles permettent également d'accéder à la stœchiométrie. Cependant, la fragmentation de l'ARN empêche d'observer les relations structurelles (structure secondaire, co-modifications). De plus, des biais expérimentaux sont introduits par les anticorps (spécificité/affinité), les enzymes (activité partielle) ou la PCR (amplification non uniforme). Le fait qu'il existe un protocole spécifique par type de modification engendre une complexité technique importante et si l'on souhaite s'intéresser à plusieurs modifications simultanément, un coût expérimental élevé.

Approches par séquençage direct de l'ARN. En 2014, une nouvelle approche de séquençage direct de l'ADN et de l'ARN a été publiée par Oxford Nanopore **(Deamer *et al*, 2016)**. Cette technologie permet la lecture d'ARN natif (non modifié chimiquement) en lectures longues, sans amplification (PCR) ni traitement enzymatique ou chimique préalable (*a contrario* de l'approche précédente) **(Garalde *et al*, 2018)**. Le brin d'ARN va générer un signal électrique, qui va ensuite être prédit en séquence nucléotidique (étape appelée « appel de base » ou, en anglais, « *base calling* ») à l'aide d'un modèle préalablement entraîné par apprentissage automatique, qu'on utilise pour convertir le signal en séquence **(Van Dijk *et al*, 2018)**. Le séquençage des transcrits entiers

d'ARN (lecture longue) permet de conserver le contexte global (formes alternatives de l'ARN) et local (résolution au nucléotide). Il n'y a pas de bruit associé au biais biologique (a contrario des approches précédentes), car le séquençage se produit sans traitement chimique ni PCR. Cette méthode est aussi avantageuse car elle permet de détecter plusieurs types de modifications connues simultanément, dans la mesure où elles ont été incluses dans l'entraînement du modèle, réduisant théoriquement à la fois la complexité technique comme le coût associé pour l'expérimentation. Cependant, la qualité des résultats dépend fortement du modèle d'apprentissage utilisé pour l'étape d'appel de base et des jeux de données ayant servi à son entraînement. Un signal bruité ou une mauvaise qualité d'entraînement peut entraîner des interprétations erronées des signaux, qui ne correspondent pas nécessairement à des modifications réelles. L'analyse de ce signal brut reste complexe et nécessite un traitement bio-informatique poussé, souvent accompagné de contrôles in vitro ou non modifiés pour valider les prédictions **(Wang *et al*, 2021)**.

D. Traitement et méthodes d'identification avec le séquençage ARN direct nanopore

1. Détection du signal et prédiction de la séquence ARN

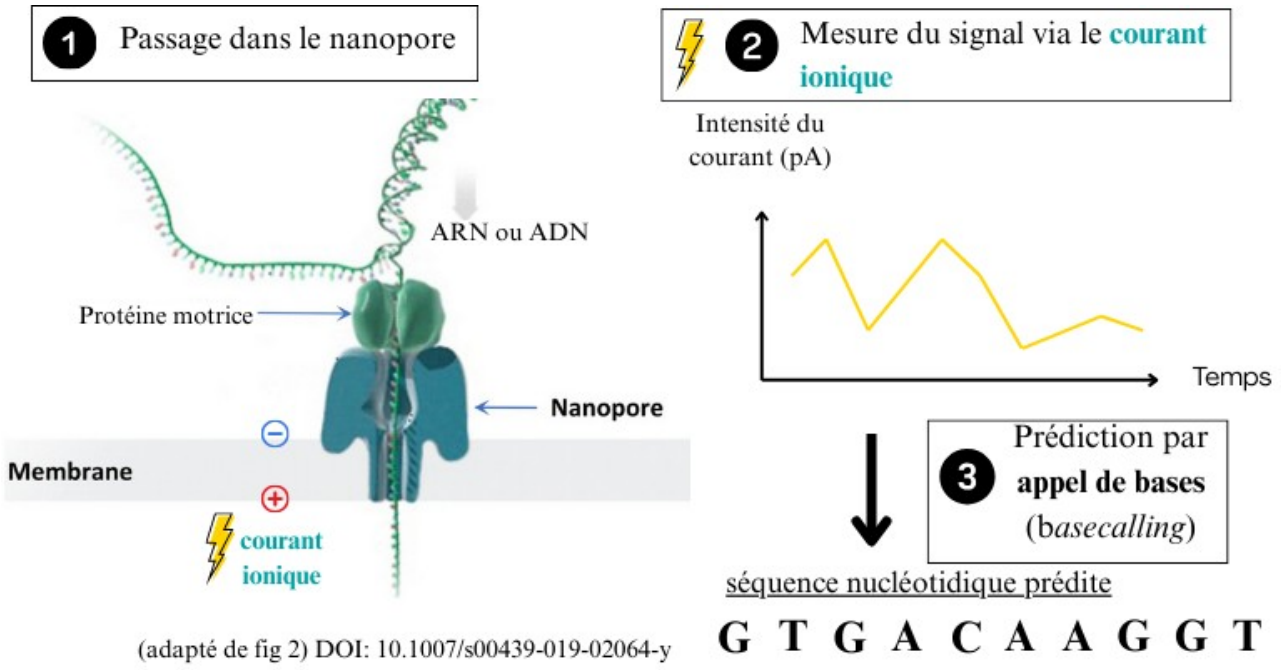


Figure 2* - <u>Principe du séquençage nanopore et de l'appel de bases</u>.** Cette figure illustre le principe du séquençage par nanopore. L'ARN (ou l'ADN) est entraîné à travers un nanopore par une protéine motrice insérée dans une membrane, ce qui perturbe le courant ionique appliqué (1). L'intensité du courant est mesurée en temps réel (2) et varie en fonction des k-mers successifs qui traversent le pore. Le signal électrique ainsi généré est ensuite interprété par un algorithme d'appel de bases (basecalling), qui prédit la séquence nucléotidique correspondante (3). (Adapté de la fig. 2 – **(Midha *et al, 2019))

Lors du séquençage, l'ARN passe à travers un nanopore lui-même inséré dans une membrane électriquement non neutre. Le brin d'ARN va être tracté par une protéine motrice appelée hélicase à travers le pore, ce qui va produire une perturbation du flux ionique. Cela va générer un signal électrique mesuré selon une fenêtre glissante (*k*-mers de 5 bases typiquement), où chaque *k*-mer produit une signature de courant spécifique **(Wang *et al*, 2021)**. Ce signal brut est ensuite interprété en une séquence nucléotidique dans l'étape d' appel de base (***figure 2***). Le signal est analysé sur une fenêtre glissante de 5 nucléotides (k-mers), et chaque motif de courant observé est comparé à des profils appris pour identifier le k-mer le plus probable. Ce n'est donc pas une base unique qui est prédite à chaque instant mais un k-mer, ce qui permet de lisser les erreurs de lecture et de compenser les variations locales de signal. L'appel de base repose sur des modèles préalablement entraînés, principalement à l'aide de réseaux de neurones convolutifs ou récurrents (apprentissage automatique), sur des données d'entraînement connues. Ces modèles sont ensuite utilisés tels quels pour effectuer la prédiction sur les nouveaux signaux.

Les modifications chimiques de l'ARN peuvent perturber le signal brut au moment du passage dans le nanopore et l'appel de base lors de l'interprétation de ce signal. Deux approches principales permettent la détection des modifications : soit basée sur les erreurs d'appel de base, soit basée directement sur les variations du signal brut **(Furlan *et al*, 2021)**.

2. Formats de données générées par le séquençage Nanopore

Format des signaux bruts issus du séquençage

Les signaux bruts générés par les dispositifs de séquençage Oxford Nanopore Technologies (ONT) sont stockés historiquement dans des fichiers au format FAST5, qui reposent sur la structure HDF5 (Hierarchical Data Format version 5). Ces fichiers contiennent des mesures de courant ionique enregistrées au fur et à mesure que des molécules d'ADN ou d'ARN traversent un nanopore (résultant de l'étape 2 de la ***figure 2***). Les données principales incluent l'intensité du courant (en picoampères), la durée de chaque événement (temps de passage dans le nanopore), la fréquence d'échantillonnage (en Hertz) et des métadonnées associées à chaque lecture, telles que les identifiants de lecture et les informations sur l'expérience de séquençage.

Chaque événement de translocation correspond à un segment de la séquence de plusieurs nucleotides passant à travers le nanopore, affectant le courant mesuré. La durée pendant laquelle un k-mer reste dans le nanopore est appelée « temps de séjour », et elle peut varier en fonction de la séquence et de la présence de modifications chimiques telles que la m6A, qui peuvent ralentir ou perturber le passage des molécules. Ces données brutes issus des signaux électriques sont ensuite

prédites en séquences de bases (A, C, G, T) pendant l'appel de bases. Même pour les lectures d'ARN les outils d'appel de base génèrent uniquement des T et non des U, pour représenter les uraciles. C'est probablement le cas pour des raisons de standardisation ou de compatibilité avec les pipelines initialement conçus pour l'ADN.

Format des séquences prédites issues de l'appel de base

Les séquences prédites issues de l'appel de base (résultant de l'étape 3 de la ***figure 2***) sont stockées dans des fichiers au format FASTQ qui est un format texte standard en bioinformatique. Chaque lecture y est décrite sur quatre lignes dont la première commence par le caractère @ et contient un en-tête avec des informations sur la lecture (identifiant unique, run, échantillon, canal, modèle d'appel, etc.). La deuxième ligne contient la séquence nucléotidique prédite (A, C, G, T ou N) et la troisième ligne est un simple + servant de séparateur. La quatrième ligne encode les scores de qualité associés à chaque base, généralement sous forme de caractères ASCII représentant les scores Phred. Ces scores permettent d'estimer la fiabilité de chaque base prédite. Les fichiers FASTQ peuvent être générés par différents outils d'appel de base comme Guppy **(Wick *et al*, 2019)** ou Dorado **(Chen *et al*, 2025)**.

3. Détail des approches pour la détection de modifications

Avant de détailler les approches existantes pour détecter les modifications à partir des données de séquençage direct Nanopore, il est important de souligner une contrainte méthodologique majeure liée à la disponibilité de données de vérité terrain. En effet il n'existe pas dans les échantillons biologiques natifs d'annotations exhaustives et certaines de toutes les positions modifiées **(Furlan *et al*, 2021)**. Si certaines positions sont connues pour être modifiées (grâce à d'autres techniques ou connaissances biologiques), ces informations sont généralement disponibles au niveau du site (position sur le transcrit) alors que les modèles en particulier ceux qui exploitent le signal brut du séquençage, sont souvent entraînés au niveau de la lecture individuelle. Or, même si un site donné est globalement connu pour être modifié cela ne signifie pas que toutes les lectures le couvrant le sont également.

Ces contraintes posent donc problème pour l'apprentissage supervisé classique qui repose sur des données d'entraînement fiables et étiquetées **(Furlan *et al*, 2021)**. En l'absence d'annotations précises au niveau des lectures individuelles les modèles doivent apprendre à partir de données bruitées, incertaines, voire conflictuelles. Cela peut introduire des biais lors de l'entraînement, réduire la performance prédictive et limiter la capacité de généralisation des modèles. De plus la

présence potentielle de co-modifications ou de variations de structure dans les données réelles augmente encore la complexité du signal à interpréter rendant l'apprentissage d'autant plus délicat. En pratique sur données réelles l'information disponible pour l'apprentissage est donc incertaine ou probabiliste, au mieux on dispose d'une stœchiométrie estimée (Proportion de modification par sites), et au pire d'une annotation approximative indiquant simplement qu'un site pourrait être modifié.

Pour contourner cette difficulté, les jeux de données utilisés reposent généralement sur des protocoles expérimentaux de référence comme la transcription in vitro (IVT) d'ARN non modifié, et/ou à l'inverse sur l'incorporation de modifications connues à des positions spécifiques dans des ARN synthétiques. Ces approches permettent de générer des jeux de données contrôlés servant de pseudo-vérité (données artificielles considérées comme vérité de terrain) pour entraîner et valider les modèles. Cependant il est important de souligner que ces données artificielles ne reflètent pas toujours fidèlement la complexité biologique réelle (co-modifications, structures secondaires, bruit technique) et que la généralisation des modèles à des données réelles reste limitée.

a. Méthodes basées sur les erreurs de prédiction et d'alignement

Une approche pour détecter les modifications d'ARN dans les données de séquençage direct Nanopore repose sur l'analyse des erreurs d'appel de base et d'alignement. Les modifications chimiques de l'ARN perturbent le signal électrique généré lors du passage dans le nanopore. En alignant les lectures prédites contre une séquence de référence, on peut observer une accumulation systématique d'erreurs sur certaines positions. Ces erreurs récurrentes à une même position suggèrent la présence possible d'une modification. Un événement isolé ne suffit pas à conclure, mais une fréquence anormalement élevée d'erreurs sur une position donnée peut être exploitée pour identifier des sites candidats. Un échantillon témoin non modifié est généralement utilisé en tant que « vérité » pour distinguer les erreurs liées aux modifications de celles dues aux biais techniques ou de celles correspondant à de vrais événements de mutations (substitutions, insertions ou délétions). **(Furlan *et al*, 2021)**

Pour exploiter ces erreurs, des méthodes d'apprentissage automatique supervisées sont utilisées. Le principe est d'utiliser les erreurs d'appel de base comme signaux d'indication de modifications.

Plusieurs types de données sont extraits des lectures pour servir de variables d'entrée au modèle tels que le taux d'indels (insertions et délétions), le taux de substitutions, les scores de qualité associés aux bases, l'entropie locale mesurant l'incertitude dans les alignements, ainsi que des métriques

issues de l'appel de variants comme les fichiers pileup ou VCF. La fréquence d'erreurs observée pour chaque *k*-mer est également prise en compte, car une modification située sur un nucléotide peut perturber le signal de plusieurs *k*-mers successifs, ce qui se traduit par une accumulation locale d'erreurs dans cette région. Ces données sont annotées à l'aide d'ARN de référence dont l'état de modification est connu, par exemple des transcrits synthétiques obtenus par transcription in vitro. La méthodologie générale repose sur l'alignement des lectures sur une séquence de référence, l'extraction des erreurs de prédiction par position (notamment via un appel de variants avec des outils comme Nanopolish - https://github.com/jts/nanopolish), la construction d'un jeu de données annoté avec des labels binaires (modifié / non modifié), puis l'entraînement d'un modèle supervisé tel qu'un SVM (support vector machine), une forêt aléatoire (random forest) ou un réseau de neurones. (**(Furlan *et al*, 2021)**, **(Pagès-Gallego & de Ridder, 2023)**)

Cette approche est relativement simple à intégrer dans des pipelines standards et fournit une bonne résolution au nucléotide. Toutefois, elle dépend fortement de la qualité de l'appel de base, de l'alignement et de l'appel des positions variantes. Elle reste sensible aux artefacts techniques et aux mutations naturelles, et peut perdre en efficacité si une modification n'induit pas suffisamment d'erreurs visibles.

b. Méthodes basées sur l'intensité et la variation du signal brut

Une autre approche pour détecter les modifications d'ARN dans les données de séquençage direct Nanopore repose sur l'analyse directe du signal électrique brut. Ce signal brut est constitué d'une série temporelle de niveaux de courant mesurés lors du passage de l'ARN à travers le nanopore. Lors du passage dans le nanopore, un k-mer ne reste pas un temps fixe ; la durée de séjour (ou dwell time) varie en fonction de sa séquence et de son environnement. L'appel de base s'effectue au niveau du k-mer en exploitant le signal généré par une fenêtre glissante de k = 5 nucléotides où chaque mesure de courant est associée à un k-mer complet plutôt qu'à une base isolée. Ainsi chaque valeur correspond à l'intensité ionique enregistrée pendant qu'un segment de l'ARN (k-mer de cinq nucléotides) se trouve dans le pore.

Les principales caractéristiques extraites de ce signal sont la moyenne du courant, son écart-type, la durée de séjour (*dwell time*), ainsi que la forme locale du signal (profils de variation). Ces caractéristiques sont sensibles aux modifications chimiques présentes sur l'ARN, qui peuvent altérer le comportement physique du translocat dans le pore. Contrairement à la méthode précédente, il ne s'agit pas ici d'exploiter les erreurs d'appel de base, mais d'observer directement les perturbations du signal physique **(Furlan *et al*, 2021)**.

Le traitement commence par un alignement de la séquence prédite à partir de l'appel de base (*basecalling*). Ensuite, un processus de réalignement du signal brut sur la séquence (*resquiggling*) associe chaque segment du signal à son *k*-mer correspondant. Une fois l'alignement réalisé, le signal observé pour chaque k-mer peut être comparé soit à un signal théorique attendu pour des bases non modifiées (dites canoniques) **(Furlan *et al*, 2021)**, soit à un signal expérimental obtenu à partir d'un échantillon témoin dépourvu de certaines modifications, généré par transcription *in vitro* ou par inactivation d'enzymes de modification **(Furlan *et al*, 2021)**. Dans les deux cas, une différence importante entre les signaux est interprétée comme un indice de modification.

Pour exploiter ces variations du signal, des méthodes d'apprentissage automatique supervisées ou non supervisées peuvent être mobilisées selon les objectifs de l'analyse. Les données d'entrée nécessaires sont extraites après réalignement du signal (*resquiggling*), un processus qui associe à chaque segment du signal brut le *k*-mer correspondant, prédit lors de l'appel de base.

Une fois cette correspondance effectuée, plusieurs types d'informations sont extraits directement du signal brut pour être utilisés comme variables d'entrée dans les modèles de détection. On utilise notamment le niveau moyen du courant électrique, la durée de séjour, l'écart-type ou la distribution locale du signal, ainsi que des descripteurs de forme comme les motifs temporels ou les caractéristiques de la forme du signal. Pour reconstruire la séquence finale, des modèles d'apprentissage (réseau de neurones convolutifs CNN, récurrents RNN, Transformeurs, clustering... (**(Furlan *et al*, 2021)**, **(Pagès-Gallego & de Ridder, 2023)**) utilisent ces signaux successifs pour inférer la suite la plus probable de nucléotides, en tenant compte des recouvrements partiels entre k-mers. Un niveau de courant attendu (*basal level*) est estimé pour chaque k-mer canonique (non modifié) à partir de jeux de données de calibration. La comparaison entre le signal observé et ce signal de référence permet de détecter des écarts caractéristiques d'une éventuelle modification, et ces écarts sont ensuite utilisés comme descripteurs dans les modèles de détection.

Cette méthode repose directement sur les effets physiques que les modifications exercent sur le signal mesuré dans le nanopore **(Furlan *et al*, 2021)**. Elle est potentiellement plus sensible que les approches basées sur les erreurs de prédiction. En contrepartie elle nécessite des jeux de données d'entraînement importants, un prétraitement lourd et reste sensible à la qualité du réalignement du signal sur les séquences prédites.

E. Objectifs du stage

Pendant mon stage je devais initier le développement d'un pipeline complet d'analyse de données

afin de disposer d'un outil permettant de tester de nouvelles stratégies d'apprentissage automatique visant à améliorer la précision de prédiction. J'avais pour objectif de maîtriser l'ensemble des étapes de transformation du signal pour pouvoir extraire les données pertinentes associées aux modifications à partir des signaux bruts pré-traités et des métriques d'alignement, afin de les structurer de manière exploitable pour l'apprentissage d'un modèle. Mon objectif était aussi d'analyser les choix méthodologiques faits par les outils existants en particulier sur la manière dont le signal brut est prétraité, aligné, segmenté en k-mers et comparé à des signaux de référence. Afin de mieux comprendre les hypothèses intégrées dans chaque étape du traitement (comme l'utilisation d'un signal moyen attendu pour chaque k-mer non modifié) et de voir comment elles influencent l'entraînement et la performance des modèles.

J'ai d'abord évalué les erreurs produites lors de l'appel de base. Ensuite, je me suis concentrée sur les variations du signal brut généré par le passage de l'ARN dans le nanopore. J'ai aussi porté attention à la qualité du traitement en amont pour garantir la cohérence des résultats.

II. Matériel et Méthodes

A. Données utilisées

Les données que j'ai utilisées au cours de ce stage sont issues de l'article « m6ATM: a deep learning framework for demystifying m6A epitranscriptome via Nanopore long read RNA-seq data » **(Yu *et al*, 2024)**. Celles-ci étaient disponibles sur GEO et sur la base de données SRA. L'article d'origine visait à apprendre un modèle capable de détecter les sites de modification m6A à l'échelle du transcriptome en utilisant le séquençage direct ARN d'Oxford Nanopore Technologies. Les échantillons analysés sont des constructions synthétiques obtenues par transcription *in vitro* (IVT), conçues pour contenir des pourcentages contrôlés de modification m6A. Pour créer ces IVT, les ARN ont été extraits à partir de constructions synthétiques polyadénylées puis capés en 5' pour imiter des ARN messagers naturels. Les échantillons modifiés ont également été ligaturés à des codes-barres uniques pour permettre le séquençage multiplexé. La qualité des ARN a été contrôlée à l'aide des instruments NanoDrop ND1000 et Qubit Fluorometer. Pour chaque condition, 500 ng d'ARN poly(A)+ ont été séquencés sur un MinION (ONT) à l'aide du kit RNA-SQK002 et de flowcells R9.4.1 (protocole ONT).

J'ai donc travaillé sur deux échantillons, d'abord un témoin non modifié (IVT_unmod, identifiant GEO : GSM8228252 et SRA : SRR28789938), contenant uniquement des adénosines non

modifiées, puis un échantillon modifié où chaque lecture contient en moyenne 50 % de bases modifiées (m6A sur les A) aléatoirement réparties (IVT_m6A_50, identifiant GEO : GSM8228254 et SRA : SRR28789936). La référence utilisée est un fichier FASTA compressé (GSE265754_IVTR.fa.gz) contenant 16 séquences d'ARN synthétiques, d'une longueur moyenne d'environ 2 200 nucléotides. Les signaux bruts issus du séquençage sont disponibles au format FAST5 et les séquences prédites issues de l'appel de base au format FASTQ sur la base de données SRA, avec respectivement 367 312 et 159 867 lectures pour l'échantillon contrôle et le modifié à 50 %.

B. Méthode de travail et gestion de projet

Pour le suivi de mon travail je participais chaque semaine à une réunion avec mes encadrant(e)s pour présenter l'avancement de mon travail, discuter des questions rencontrées et fixer les objectifs pour la suite. Pour permettre à mes encadrant(e)s d'accéder facilement à mon travail, j'ai mis en place un projet GitLab dès le début du stage (sur la forgeMIA, l'instance GitLab interne d'INRAE). J'y ai organisé mes scripts en fonction des jeux de données utilisés, chaque ensemble ayant son propre répertoire. Dans chacun, j'ai ajouté un fichier README.md pour documenter mes analyses, expliquer les étapes suivies et commenter les résultats obtenus. J'ai adopté une démarche itérative où j'explorais les données et testais différentes approches, puis revenais sur certaines étapes en fonction des résultats. Pour tout ce qui concernait la gestion de la bibliographie, j'utilisais à la fois Nextcloud (le cloud interne d'INRAE) pour le partage de fichiers et Zotero (https://www.zotero.org) pour centraliser les références bibliographiques. J'ai utilisé les outils spécifiques d'INRAE pour respecter la politique interne de l'organisme et travailler dans des environnements sécurisés. Pour exécuter les scripts, j'utilisais le cluster de la plateforme Genotoul-Bioinfo. J'ai demandé un compte personnel au début du stage pour organiser mes données et soumettre mes tâches soit directement sur un nœud (en mode interactif) soit à l'aide de Slurm.

Pour organiser mon travail sur le cluster Genotoul-Bioinfo j'ai monté le répertoire */work/user/erodriguez* sur ma machine locale via sshfs. Cela m'a permis de faciliter le transfert de fichiers entre mon poste et l'espace de travail distant. J'ai ensuite structuré mon projet en créant une arborescence standardisée comprenant les répertoires suivants : data pour les données brutes, analysis pour les fichiers produits, reference pour les fichiers de référence, et script pour les scripts exécutés. J'ai défini une variable d'environnement ROOT pour pointer vers le chemin principal */home/erodriguez/work/m6ATM* afin de rendre mes scripts reproductibles.

Mon stage s'est déroulé sur une durée de six mois, selon le calendrier présenté dans la ***figure 3***, qui retrace l'organisation temporelle des différentes étapes du travail réalisé.

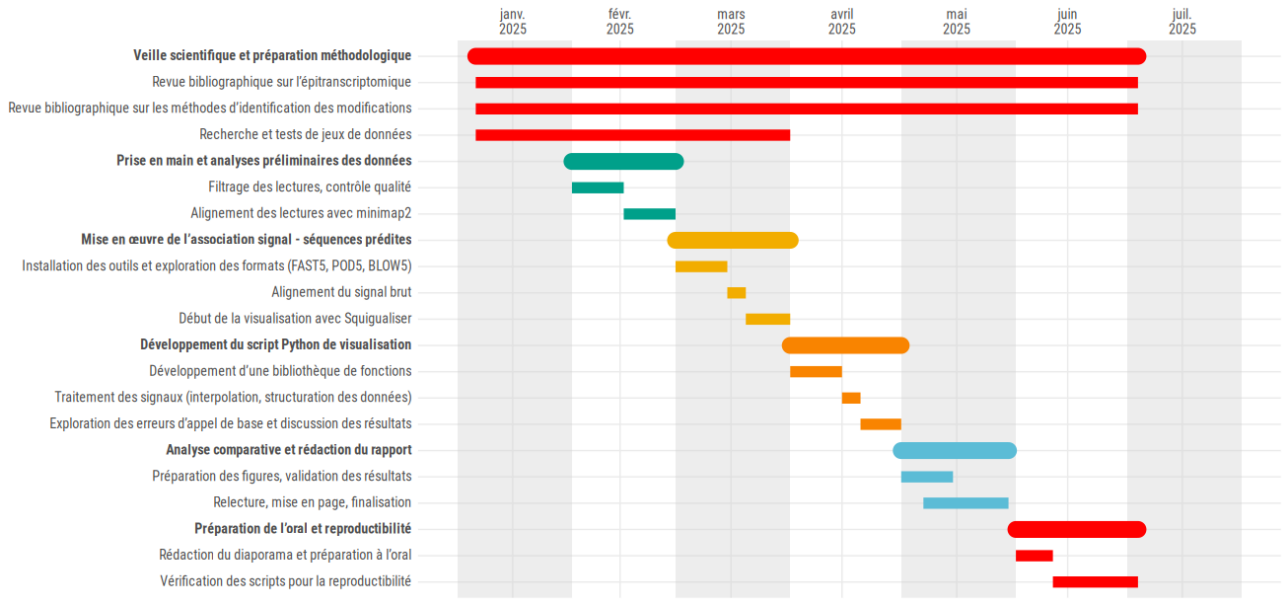


Figure 3 **- <u>Calendrier du programme de travail durant le stage (janvier - juillet 2025)</u>.** Ce diagramme de Gantt présente la planification des différentes étapes du stage depuis la veille scientifique et la prise en main des données jusqu'à l'analyse des résultats, la rédaction du rapport et la préparation de la soutenance.

C. Liste, description et version des outils

J'ai utilisé un ensemble d'outils pour traiter mes données dont certains étaient disponibles directement via les modules du cluster Genotoul-Bioinfo. J'ai chargé SRA-Toolkit (v3.0.2, https://github.com/ncbi/sra-tools) pour le téléchargement des fichiers depuis la base SRA. J'ai utilisé samtools (v1.20) pour manipuler les fichiers d'alignement SAM et BAM. J'ai filtré les lectures en fonction de leur qualité avec SeqKit (v2.9.0, https://github.com/shenwei356/seqkit). J'ai aussi utilisé NCBI EDirect **(Scalfani, 2021)** (v20.5.20231007, https://github.com/Klortho/edirect) pour récupérer les métadonnées associées aux jeux de données.

Les autres outils ont été installés manuellement dans mon environnement utilisateur (via pip ou directement en clonant les répertoires GitHub officiels). J'ai compilé bioawk (version 20110810, https://github.com/lh3/bioawk) pour extraire des informations à partir des fichiers FASTA. J'ai utilisé minimap2 **(Li, 2018)** (v2.28-r1209, https://github.com/lh3/minimap2) pour aligner les lectures RNA sur la séquence de référence. J'ai également utilisé IGV **(Thorvaldsdottir *et al*, 2013)** (Integrative Genomics Viewer, version 2.16.0, https://github.com/igvteam/igv) pour visualiser les alignements générés par minimap2 et identifier des positions variantes entre les échantillons. Pour calculer des statistiques sur les données de séquençage Oxford Nanopore ainsi que sur les alignements j'ai utilisé l'outil NanoStat **(De Coster & Rademakers, 2023)** (version

1.6.0, https://github.com/wdecoster/nanostat) et NanoPlot du même outil (v1.44.1, https://github.com/wdecoster/NanoPlot) pour visualiser les statistiques.

Pour manipuler les fichiers contenant les signaux bruts, j'ai utilisé ont_fast5_api (release_4.1.3, https://github.com/nanoporetech/ont_fast5_api). J'ai converti les fichiers FAST5 au format POD5 avec pod5 (v0.3.23, https://github.com/nanoporetech/pod5-file-format). Ensuite j'ai converti les fichiers POD5 en BLOW5 avec blue-crab (v0.5.1, https://github.com/Psy-Fer/blue-crab), et j'ai fusionné ces BLOW5 avec slow5tools **(Samarakoon *et al*, 2023)** (v1.3.0, https://github.com/hasindu2008/slow5tools). J'ai utilisé f5c **(Gamaarachchi *et al*, 2020)** (v1.5, https://github.com/hasindu2008/f5c) pour aligner les signaux bruts sur les lectures alignées. Enfin, j'ai utilisé squigualiser **(Samarakoon *et al*, 2024)** (v0.6.3, https://github.com/hiruna72/squigualiser) pour visualiser et comparer les signaux associés aux différentes conditions expérimentales. Pour l'analyse Python j'ai utilisé Python 3.12.3 avec les bibliothèques pandas (2.2.3), IPython (8.31.0), scipy (1.15.1), matplotlib (3.9.1) et numpy (2.2.2).

D. Organisation et étapes de traitement

La ***figure 4*** ci-dessous présente les différentes étapes que j'ai suivies durant mon stage pour traiter et analyser les données de séquençage direct de l'ARN. La partie en haut à gauche (« Détection du signal et prédiction de la séquence ARN ») correspond à la phase de compréhension du sujet, des formats et des données. Elle regroupe les éléments de bibliographie que j'ai consultés pour me familiariser avec le fonctionnement de la technologie Nanopore, l'étape d'appel de base pour la prédiction des sequences (voir ***figure 2)*** . Cette étape m'a aussi permis d'identifier le jeu de données public fiable basé sur des transcrits synthétiques IVT **(Yu *et al*, 2024)**. Cela m'a demandé un certain temps, car plusieurs jeux de données publiques étaient obsolètes ou partiellement exploitables à cause de changements de formats ou de compatibilités (plusieurs mises en œuvre infructueuses, parfois avancées).

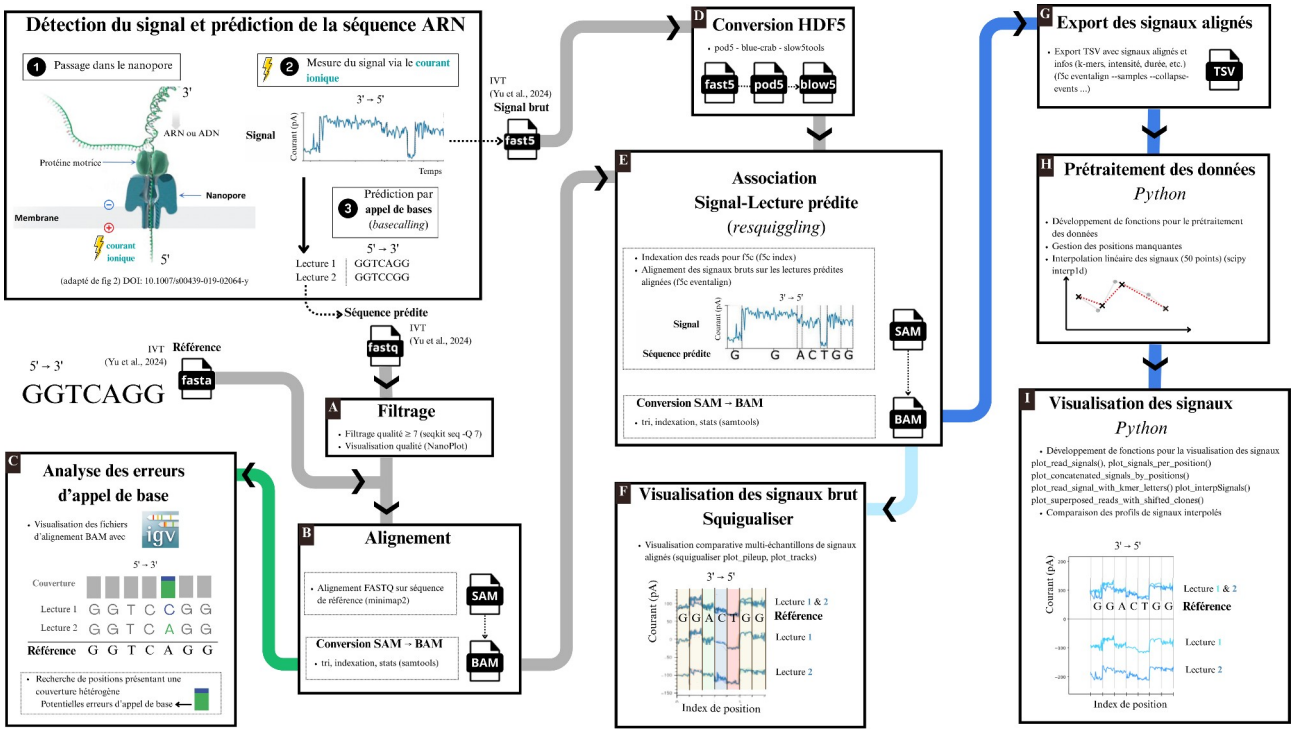


Figure 4* - <u>Vision globale des étapes de traitement des données Nanopore réalisées durant le stage</u>.** Cette figure résume les différentes étapes que j'ai suivies durant mon stage pour traiter et analyser les données de séquençage direct de l'ARN. La partie en haut à gauche (« Détection du signal et prédiction de la séquence ARN ») correspond à la phase de compréhension du sujet explicité dans la ***figure 2. Les blocs A à I décrivent les étapes de traitement réalisées de manière chronologique. J'ai commencé par filtrer les lectures de mauvaise qualité (A), puis les ai alignées à la séquence de référence (B). J'ai ensuite utilisé IGV pour repérer des erreurs d'appel de base spécifiques à l'échantillon modifié (C). Après cela, j'ai converti les signaux bruts du format FAST5 vers BLOW5 (D), afin de pouvoir les exploiter avec des outils récents. J'ai ensuite aligné ces signaux sur les séquences appelées (E), pour relier chaque segment de signal à un k-mer. J'ai visualisé une première fois les signaux avec l'outil Squigualiser (F), puis exporté les signaux alignés au format TSV (G). À partir de là, j'ai développé mes propres scripts Python pour prétraiter les signaux (H) et créer des visualisations comparatives entre échantillons (I). Les couleurs indiquent les types de visualisation : en vert les analyses basées sur les erreurs d'appel de base, en bleu les analyses sur les signaux. Le bleu clair correspond à un outil existant (Squigualiser), et le bleu foncé à mes développements personnels en Python.

Les blocs A à I décrivent ensuite les étapes de traitement réalisées, de manière chronologique. J'ai commencé par filtrer les lectures de mauvaise qualité (A), puis les ai alignées à la séquence de référence (B). J'ai ensuite utilisé IGV pour repérer des erreurs d'appel de base spécifiques à l'échantillon modifié (C). Après cela, j'ai converti les signaux bruts du format FAST5 vers BLOW5 (D), afin de pouvoir les exploiter avec des outils récents. J'ai ensuite aligné ces signaux sur les séquences appelées (E), pour relier chaque segment de signal à un k-mer. J'ai visualisé une première fois les signaux avec l'outil Squigualiser (F), puis exporté les signaux alignés au format TSV (G). À partir de là, j'ai développé mes propres scripts Python pour prétraiter les signaux (H) et créer des visualisations comparatives entre échantillons (I). Les couleurs indiquent les types de visualisation : en vert les analyses basées sur les erreurs d'appel de base, en bleu les analyses sur les signaux. Le bleu clair correspond à un outil existant (Squigualiser), et le bleu foncé à mes développements personnels en Python. Toutes ces étapes seront explicitées précisément dans la suite de la section matériel et méthodes.

Une difficulté importante du stage était que personne dans l'équipe ne connaissait en détail la technologie Nanopore, ni les formats spécifiques de fichiers utilisés (FAST5, POD5, BLOW5). J'ai donc dû identifier les formats (FAST5, POD5, BLOW5), comprendre les conversions nécessaires, tester de nombreux outils et résoudre de nombreux problèmes liés aux versions ou à l'incompatibilité des formats (L'ensemble du traitement a demandé un gros travail d'apprentissage). La documentation des outils étant souvent fragmentaire ou obsolète, j'ai dû croiser plusieurs sources, tester de nombreuses configurations, et adapter les paramètres en fonction des erreurs rencontrées. Cette phase a représenté un investissement conséquent en temps et en efforts, et a nécessité une forte capacité d'autonomie et de résolution de problèmes techniques.

E. Prétraitement des données

J'ai commencé par télécharger les données depuis le projet SRA PRJNA1104150 (SRR28789938 pour le contrôle et SRR28789936 pour l'échantillon m6A 50 % modifié). J'ai d'abord récupéré manuellement les fichiers de séquençage au format FAST5 contenant le signal brut depuis les liens publics SRA sous forme d'archives compressées, que j'ai ensuite extraites dans des dossiers organisés par condition expérimentale. J'ai téléchargé les fichiers de séquences prédites suite a l'appel de base à l'aide de `prefetch`, puis j'ai extrait les lectures au format FASTQ avec `fastq-dump`. Les identifiants de lectures issues du téléchargement SRA `prefetch` étaient préfixés par l'identifiant SRA ce qui ne correspondait pas aux identifiants des fichiers FAST5. Pour corriger ça, j'ai utilisé awk pour reformater les entêtes FASTQ afin de conserver uniquement l'identifiant UUID (Universal Unique IDentifier).

Ensuite j'ai filtré les lectures (***figure 4,*** bloc A) en conservant uniquement celles ayant un score de qualité supérieur ou égal à 7, en utilisant `seqkit seq -Q 7`. Ce score `-Q` correspond à un score Phred (échelle logarithmique utilisée pour estimer la probabilité d'erreur d'un appel de base, calculé avec $Q = -10 \times \log10(p)$ où p est la probabilité que la base soit incorrecte). Ce seuil, établi à 7, correspond à celui utilisé dans l'article de référence **(Yu *et al*, 2024)** pour garantir une bonne qualité des lectures. Pour évaluer l'impact du filtrage, j'ai utilisé NanoPlot (un outil spécialisé dans les données nanopore) pour visualiser les distributions de qualité et de longueur des lectures avant et après le filtrage.

F. Alignement des lectures FASTQ

Une fois les lectures de bonne qualité récupérées, je les ai alignées sur les séquences de référence pour obtenir leur localisation, condition nécessaire pour aligner ensuite les signaux bruts (***figure 4,***

bloc B). J'ai utilisé minimap2 pour aligner les lectures filtrées sur le fichier de référence GSE265754_IVTR.fa . Pour minimap2 j'ai utilisé les mêmes paramètres que ceux décrits dans l'article de référence **(Yu *et al*, 2024)** `-ax splice -k14 -uf --secondary=no` . Le paramètre `-a` force la sortie au format SAM et `-x splice` active les réglages adaptés aux lectures longues avec épissage (adaptés au séquençage direct d'ARN). Le paramètre `-k14` réduit la taille des k-mers utilisés pour l'indexation pour améliorer la sensibilité et `-uf` indique que les lectures ne sont pas orientées. Enfin `--secondary=no` désactive les alignements secondaires pour ne conserver que les alignements principaux.

J'ai ensuite converti les fichiers SAM en BAM (version binaire compressée plus rapide à traiter que le format SAM) avec samtools view `-hbS` (où `-h` conserve les en-têtes, `-b` spécifie la sortie au format BAM et `-S` indique une entrée au format SAM), puis trié les alignements avec samtools sort et généré les index avec samtools index (***figure 4,*** bloc B) . Pour vérifier la qualité des alignements, j'ai généré des statistiques globales avec samtools flagstat (taux de lectures alignées, lectures secondaires, etc.). Après cela, j'ai utilisé IGV (Integrative Genomics Viewer) pour visualiser les alignements générés par minimap2 et identifier des positions variantes entre les échantillons. (***figure 4,*** bloc C)

G. Conversion des signaux FAST5 en format BLOW5

À l'origine les signaux bruts du séquençage Nanopore étaient stockés dans des fichiers au format FAST5 qui est basé sur HDF5 et structuré de manière hiérarchique (type dictionnaire). Ces fichiers peuvent contenir à la fois le signal brut issu du nanopore et le résultat de l'appel de bases (*basecalling*), ce qui en fait un format complet mais également très lourd. Leur structure nécessite des ressources importantes pour la lecture et rend difficile une analyse rapide et parallèle sur de gros volumes de données.

Pour pallier ces limitations des formats plus récents et plus légers ont été développés dont le format POD5 (nouveau standard de Nanopore, fondé sur Apache Arrow) et BLOW5 (issu de la recherche académique, format binaire optimisé pour la vitesse et la compression). Contrairement au format FAST5, les formats POD5 et BLOW5 sont spécialisés dans la représentation du signal brut uniquement. Le format BLOW5 en particulier est optimisé pour les performances d'entrée/sortie (I/O). Il permet une lecture séquentielle plus rapide, réduit considérablement l'encombrement mémoire grâce à une meilleure compression et est compatible avec une lecture parallèle multi-thread, ce qui accélère fortement le traitement de grands volumes de données.

Certains outils récents que j'utilise ensuite comme f5c (version optimisé de Nanopolish développé par Oxford Nanopore Technologies) ou squigualiser ne prennent plus en charge les fichiers FAST5 et nécessitent l'utilisation de fichiers BLOW5 pour fonctionner.

En théorie il est possible de convertir directement les fichiers FAST5 en fichiers BLOW5 à l'aide de slow5tools. Mais je n'ai jamais pu utiliser cet outil en pratique : il a systématiquement échoué sur mes données sans que je parvienne à résoudre ces erreurs de conversion directe FAST5 vers BLOW5. Ce problème s'explique en partie par le fait que le format FAST5 a évolué au fil des années : selon la date de production, un fichier FAST5 (unique ou multiple respectivement) ne suit pas forcément la même structure même s'il porte le même nom. Les anciens fichiers peuvent contenir des schémas HDF5 obsolètes ou non pris en charge et les outils de mise à jour de ces fichiers ne sont plus maintenus. J'ai donc dû contourner cette limitation en passant par une conversion intermédiaire en POD5.

J'ai d'abord converti les fichiers multi-FAST5 (contenant plusieurs lectures par fichier) en fichiers POD5 en utilisant la commande `pod5 convert fast5`. Il est important de noter que cette conversion n'est possible que si les fichiers FAST5 sont bien en format multi-FAST5. De plus, j'ai utilisé le paramètre `--one-to-one` pour obtenir un fichier POD5 par lecture car la conversion suivante vers BLOW5 ne fonctionne qu'à partir de fichiers POD5 unitaires. Enfin j'ai converti ces fichiers POD5 en fichiers BLOW5 avec la commande `blue-crab p2s`, et j'ai fusionné tous les fichiers BLOW5 générés en un seul fichier BLOW5 par échantillon avec `slow5tools merge`**.** Ce traitement a été réalisé séparément pour mes deux échantillons . (***figure 4,*** bloc D)

H. Alignement des signaux avec f5c eventalign

Une fois les lectures alignées et les signaux bruts disponibles au format BLOW5 j'ai utilisé l'outil f5c pour aligner les signaux bruts sur les séquences alignés précédemment (***figure 4,*** bloc E). Cette étape permet d'associer chaque segment du signal à un *k*-mer de la séquence de référence, pour détecter des différences de signal liées à la présence ou non de modifications m6A.

J'ai d'abord indexé les fichiers FASTQ avec `f5c index` en spécifiant le fichier BLOW5 correspondant avec `--slow5`. Ensuite j'ai utilisé `f5c eventalign` avec les paramètres `-a -b -r -g --slow5` pour produire un fichier SAM contenant l'alignement du signal brut. Le paramètre `-a` active l'annotation des bases alignées dans la sortie, `-b` indique le fichier BAM des lectures alignées, `-r` spécifie le fichier FASTQ des lectures, `-g` fournit la séquence de référence et `--slow5` permet de charger les signaux bruts à partir du fichier BLOW5. Pour finir, j'ai converti ces fichiers en BAM

triés et indexés avec samtools.

I. Visualisation des signaux avec Squigualiser

Pour comparer les signaux bruts entre l'échantillon modifié et le contrôle j'ai utilisé squigualiser, afin de détecter visuellement d'éventuels décalages ou différences d'intensité de signal caractéristiques d'une modification m6A (***figure 4,*** bloc F).

Squigualiser est un outil open-source pour visualiser les signaux bruts produits par le séquençage nanopore. Il aligne les signaux soit à la séquence lue (*signal-to-read*), soit à une séquence de référence (*signal-to-reference*) avec une résolution à l'échelle du nucléotide. Il fonctionne avec des données ADN ou ARN issues des nanopore et est compatible avec plusieurs logiciels d'alignement de signal comme f5c, Nanopolish. Il produit une interface interactive au format HTML dans laquelle on peut naviguer sur une région, afficher ou masquer des lectures et personnaliser l'affichage. Les lectures sont empilées sous forme d'empilement visuel (*pileup*) et plusieurs échantillons peuvent être affichés en parallèle (*parallel tracks*) pour comparer des conditions. Ça permet de repérer visuellement des décalages ou des variations du signal liées à des modifications chimiques comme m6A. Squigualiser intègre plusieurs fonctions avancées dans une seule interface comme la visualisation multi-lectures, l'affichage des insertions et suppressions, l'ajout de signaux simulés, etc. Contrairement à d'autres outils limités ou statiques il est conçu spécifiquement pour l'exploration des signaux bruts. Il utilise Python 3.8 et s'installe avec pip, bioconda ou peut être utilisé directement via des exécutables pré-compilés (*prebuilt binaries*). Par défaut cet outil représente simultanément toutes les lectures présentes dans les fichiers BLOW5 et BAM, ce qui rend les visualisations illisibles car trop denses, j'ai donc dû filtrer mes fichiers avant de lancer l'analyse.

Pour chaque échantillon, j'ai extrait un fichier BLOW5 et un fichier BAM ne contenant que les lectures d'intérêt. Celles-ci étaient situées dans une région où des écarts de position (nucléotides différents de la référence) étaient visibles dans IGV notamment IVT_seq_2_2201:1221-1260. J'ai choisi cette région comme exemple et sélectionné deux lectures par échantillon qui présentaient une variation dans l'échantillon modifié (positions 1236, 1241 et 1245) par rapport à la référence et au témoin. J'ai isolé les signaux bruts correspondants avec `slow5tools get` à partir des fichiers BLOW5 séparément pour chaque condition. J'ai ensuite extrait les alignements associés avec `samtools view -N`, en fournissant un fichier texte listant les identifiants des lectures à conserver (`-N` permet de ne garder que les alignements correspondant à ces identifiants). J'ai indexé les fichiers BAM générés

puis j'ai vérifié que seuls les reads attendus étaient bien présents avec `samtools view | cut -f1 | sort | uniq` (`cut -f1 | uniq` extrait les identifiants de lecture uniques à partir de la première colonne).

Ensuite, j'ai créé un fichier d'entrée listant deux commandes `squigualiser plot_pileup` (une pour chaque échantillon) en spécifiant la région IVT_seq_2_2201:1221-1260, la séquence de référence, les fichiers BLOW5 extraits, les fichiers BAM associés ainsi qu'un nom de tag pour distinguer les courbes. J'ai ensuite utilisé `squigualiser plot_tracks` avec l'option `--shared_x` (qui aligne les graphiques sur un axe des abscisses commun pour la comparaison directe des signaux) et ce fichier de commandes pour générer une visualisation comparative des signaux modifiés et non modifiés.

J. Export des signaux alignés au format TSV

J'avais pour objectif de maîtriser l'ensemble des étapes de transformation du signal et de comparaison entre conditions à la fois pour reproduire en Python la visualisation proposée par Squigualiser. Ainsi que pour pouvoir extraire les données pertinentes associées aux modifications et aux témoins, à partir des signaux bruts pré-traités et des métriques d'alignement, afin de les structurer de manière exploitable pour l'apprentissage d'un modèle. (***figure 4,*** bloc G)

Pour ça, j'ai dû exporter les alignements de `f5c eventalign` au format TSV pour chacun des échantillons. J'ai utilisé les paramètres `--samples` pour inclure les signaux bruts associés à chaque événement dans le fichier de sortie, `--print-read-names` pour afficher les identifiants complets des lectures au lieu de simples index numériques, `--signal-index` pour récupérer les indices de début et de fin des signaux associés à chaque événement, et `--collapse-events` pour fusionner les événements redondants. Ce paramètre `--collapse-events` est nécessaire car il permet de fusionner pour une même lecture les doublons d'événements alignés sur une même position de référence et un même *k*-mer. Lors de l'alignement du signal brut `f5c eventalign` (utilisé précédemment) segmente le signal continu en plusieurs événements, chacun étant censé correspondre à un *k*-mer donné. Mais, en pratique, cette segmentation n'est pas toujours parfaitement alignée, ou bien, pour une même lecture, plusieurs événements peuvent être associés à une même position de référence avec des indices de début et de fin légèrement différents, en raison du bruit du signal ou d'imprécisions dans la segmentation. Ces redondances entraînent la présence de plusieurs lignes pour une même position et une même lecture dans le fichier TSV. Pour chaque groupe d'événements fusionnés `--collapse-events` prend l'intervalle de signal le plus large, calcule la moyenne de l'intensité, son écart-type ainsi que la durée de l'événement. Le fichier TSV résultant contient pour chaque ligne le *contig* (nom de la séquence de référence), la position du k-mer, *reference kmer* (*k*-mer attendu dans la

séquence de référence), le *read name* (identifiant de la lecture), *strand* (t pour template, c pour complement). Ensuite *model kmer* correspond au k-mer sur lequel l'événement observé a été aligné dans le modèle (pore-model) fourni par ONT. Ce modèle associe à chaque k-mer un niveau de courant attendu représenté par une moyenne ($model_{mean}$) et un écart-type ($model_{stdv}$). Ces valeurs ne sont pas brutes mais mises à l'échelle à l'aide des paramètres de calibration propres à chaque lecture (scaling.scale, scaling.shift, scaling.var) :

$$model_{mean} = scaling.scale \times level_{mean} + scaling.shift$$

$$model_{stdv} = level_{stdv} \times scaling.var$$

où $level_{mean}$ et $level_{stdv}$ sont les valeurs de courant théoriques associées au k-mer dans le pore-model brut, permettant d'adapter dynamiquement le modèle aux conditions de chaque séquençage. Le champ $event\ level_{mean}$ représente la moyenne du courant mesuré pendant l'événement détecté (c'est-à-dire une portion du signal associée à un k-mer donné) et $event_{stdv}$ son écart-type, avec la durée de l'événement donnée par $event_{length}$ exprimée en secondes. Ces valeurs sont calculées directement à partir des échantillons bruts de signal.

La valeur de *standardized level* (niveau standardisé) exprime la différence entre le signal observé et le signal attendu sous une forme normalisée tenant compte de la variabilité estimée du modèle. Cette valeur permet de comparer les événements de manière standardisée indépendamment des variations entre lectures, et est définie telle que :

$$standardized\ level = \frac{event\ level_{mean} - model_{mean}}{\sqrt{(scaling.var)} \times model_{stdv}}$$

Enfin start_idx et end_idx ajoutés par l'option `--signal-index` donnent respectivement les indices de début et de fin du signal associé à l'événement dans la série de courant brut (index 0-based, format fermé/ouvert). Le champ samples activé par `--samples` contient la séquence complète des mesures de courant (en pico-Ampere) correspondant à ce segment sous forme d'un liste de valeurs séparées par des virgules.

Pour chaque échantillon le fichier TSV généré avec f5c eventalign occupait plusieurs dizaines de gigaoctets. J'ai donc filtré ces fichiers pour ne conserver que les lectures alignées sur une région d'intérêt « IVT_seq_2_2201 » que j'avais choisie et ciblée dans IGV. J'ai utilisé awk pour extraire uniquement les lignes dont la première colonne correspondait à cette région, et j'ai réinséré la ligne d'en-tête d'origine avec `head -n 1`. Une fois formatés, j'ai utilisé ces fichiers TSV dans un notebook

jupyter que j'ai créé pour visualiser les distributions de signal de la région d'intérêt.

K. Développement d'un script python pour la visualisation des signaux

Pour permettre la visualisation comparative du signal brut entre l'échantillon témoin et l'échantillon modifié (50 % m6A) dans la région IVT_seq_2_2201:1221-1260 j'ai d'abord préparé les fichiers TSV générés par f5c eventalign. Je les ai traités avec Python à l'aide des bibliothèques pandas, numpy, scipy et matplotlib.

1. Préparation et pré-traitement des jeux de données

Cette sous-partie correspond au bloc H de la ***figure 4***

Chargement des jeux de données

J'ai d'abord défini une fonction **load_data()** pour charger les fichiers TSV et corriger l'indexation de la colonne position (les positions étant initialisées à 0 dans la sortie de f5c, j'ai appliqué un décalage de +1 pour correspondre aux coordonnées séléctionnées dans IGV). Ensuite, j'ai créé la fonction **filter_reads()** qui a permis de sélectionner uniquement les quatre lectures d'intérêt identifiées visuellement via IGV (deux dans l'échantillon modifié et deux dans le témoin). Ma fonction **filter_by_position()** a ensuite extrait les lignes correspondant aux positions 1221 à 1260 puis j'ai concaténé les deux jeux de données résultants (modifié et non modifié) en un seul fichier, trié selon la position.

Interpolation linéaire

Afin d'uniformiser la structure des signaux bruts pour la visualisation, j'ai défini une fonction **interpolation_lineaire()** qui convertit chaque signal initialement représenté par une liste de valeurs inégales, en un vecteur de 50 points via une interpolation linéaire. Le choix de 50 points s'appuie sur la distribution observée des longueurs des signaux bruts alignés aux k-mers dans mon fichier TSV où en moyenne les signaux font 42 points et en tout 75 % des événements contiennent au maximum 51 valeurs. Fixer la taille à 50 permet donc de couvrir la majorité des cas sans extrapoler fortement les signaux courts, ni perdre trop d'information dans les signaux longs. En effet cette interpolation est nécessaire car la longueur des signaux peut varier d'un événement à l'autre (à cause de la vitesse de passage dans le Nanopore qui peut varier ou à cause de décalages lors de la segmentation), rendant impossible leur comparaison directe. Un passage lent produit un signal plus

long (plus de 50 points mesurés) et un passage rapide un signal plus court. Dans le cas d'un signal court (moins de 50 points), la méthode que j'ai implémentée insère des valeurs intermédiaires entre les quelques points disponibles par interpolation linéaire des points observés, ce qui ajoute des points d'observations factices intermédiaires. Dans le cas d'un signal long, l'approche réduit le signal à 50 points en sous-échantillonnant les points d'observation, ce qui a pour effet de lisser les variations fines. La fonction applique la méthode *interp1d* de la bibliotheque *scipy* pour lisser et normaliser chaque signal extrait de la colonne « samples ».

Uniformisation du jeu de données

Après un pré-traitement, j'ai vérifié que toutes les positions entre 1221 et 1260 contenaient bien les quatre lectures attendues et constaté que certaines positions intermédiaires (notamment autour des nucléotides modifiés 1236, 1241 et 1245) étaient incomplètes. Pour garantir l'uniformité du tableau de données, j'ai explicitement identifié les positions manquantes et généré les lignes correspondantes pour chaque lecture absente. Les valeurs manquantes ont été initialisées avec des vecteurs de NaN pour les signaux (samples et interpolated) et par la chaîne « ----- » pour les k-mers (reference_kmer, model_kmer) pour éviter les erreurs d'affichage lors de la phase de visualisation.

Ce traitement assure que toutes les positions comprises entre 1221 et 1260 sont représentées pour chaque lecture, y compris celles sans couverture, pour aligner correctement les signaux. Pour finir, j'ai sauvegardé le jeu de données final au format TSV. Celui-ci servira de point de départ pour la visualisation des signaux présentée dans la section suivante.

2. Représentation graphique des signaux

À partir du fichier TSV final contenant les signaux interpolés j'ai développé plusieurs fonctions de visualisation en Python pour comparer visuellement les signaux bruts des lectures modifiées et non modifiées, en m'inspirant des représentations proposées par squigualiser (***figure 4,*** bloc I). Mon objectif était d'identifier visuellement des différences systématiques entre les deux échantillons soit à une position donnée (c'est-à-dire sur une portion de signal associée à un k-mer), soit sur une portion plus étendue du transcrit (sur une fenêtre glissante couvrant plusieurs k-mers). Pour cela les lectures de l'échantillon témoin (non modifié) sont affichées en bleu et celles de l'échantillon modifié en rouge, avec des nuances différentes attribuées à chaque lectures individuelles. Dans certaines fonctions j'ai également appliqué un décalage vertical entre les courbes de signal afin d'éviter les chevauchements tout en conservant leur alignement par position de référence, ce qui permet de les visualiser clairement les unes en dessous des autres.

J’ai d’abord écrit la fonction **plot_read_signals()** pour afficher, pour un read donné, à chaque position, la courbe du signal brut issu de la colonne samples ainsi que sa version interpolée à 50 points. Cette fonction m’a permis de vérifier visuellement la cohérence du processus d’interpolation et de confirmer que les signaux originaux étaient correctement transformés sans perte d’information. Pour voir position par position si les lectures issues de l’échantillon modifié présentent un profil de signal systématiquement différent de celles du témoin j’ai défini la fonction **plot_signals_per_position()**. Elle trace, pour chaque position de la région étudiée, les signaux interpolés de tous les reads superposés sur un même graphe.

Ensuite pour évaluer l’évolution du signal sur une portion plus étendue du transcrit et comparer les lectures dans leur continuité, j’ai créé la fonction **plot_concatenated_signals_by_positions()**. Cette fonction récupère pour une lecture donnée les signaux interpolés correspondant à une liste de positions successives (triées par ordre croissant ou décroissant), et les affiche les uns à la suite des autres sur un même graphique. Cela permet de reconstituer le profil de signal d’une lecture sur une région définie en plaçant bout à bout les segments correspondant à chaque position, selon l’ordre dans lequel ils apparaissent sur le transcrit. J’ai aussi créé la fonction **plot_read_signal_with_kmer_letters()** qui affiche le signal interpolé d’une lecture et y ajoute, sous forme d’annotations, la première lettre du reference_kmer associé à chaque position. J’ai choisi d’afficher uniquement cette première lettre car elle correspond à la base présente dans la séquence de référence à la position considérée. Cela permet de visualiser directement sur le graphique du signal la séquence de référence associée à chaque segment de signal, sans surcharger l’affichage avec les k-mers complets. Cette logique est en option dans le reste de mes fonctions.

Ma fonction **plot_interpSignals()** permet d’afficher sur un même graphique les signaux interpolés de plusieurs lectures sélectionnées. Les signaux sont superposés sans décalage vertical ce qui permet de visualiser dans un premier temps l’évolution de leur profil de signal, les uns par rapport aux autres sur toute la longueur des lectures. L’axe des abscisses correspond aux positions de référence et l’axe des ordonnées à l’intensité du signal (en picoampères, pA). Ma dernière fonction **plot_superposed_reads_with_shifted_clones()** permet de tracer chaque lecture une première fois à son niveau d’intensité réel, et une seconde fois en la décalant verticalement vers le bas. Ce décalage vertical artificiel permet de mieux distinguer les courbes lorsque plusieurs lectures se superposent dans la même zone de signal. Le signal est reconstruit en rassemblant pour chaque lecture les segments de signal interpolé extraits du fichier TSV généré par f5c eventalign, en suivant l’ordre des positions (croissant ou décroissant). Ces segments sont alignés bout à bout pour former une seule courbe continue représentant l’évolution du signal sur toute la longueur des lectures de la

même manière que la fonction précédente. Le décalage vertical artificiel est calculé dynamiquement en commençant à -50 pA en dessous du maximum de la première lecture, puis un espacement fixe est appliqué entre chaque courbe décalée pour éviter tout chevauchement. L'axe des abscisses est ajusté pour afficher les positions génomiques associées extraites des données originales.

III. Résultats

A. Prétraitement des données : Filtrage des lectures selon leur qualité

Sur la ***tableau 1*** on voit que, après filtrage à un seuil de qualité Phred >= 7, 87,0 % des lectures ont été conservées pour l'échantillon contrôle (319 544 sur 367 312) et 77,1 % pour l'échantillon modifié (123 269 sur 159 867). La longueur moyenne des lectures augmente dans les deux cas passant de 1540,2 à 1629,0 pb (+ 5,8 %) pour le contrôle, et de 1206,1 à 1299,6 pb (+ 7,8 %) pour l'échantillon modifié. Cette augmentation est attendue car les lectures de faible qualité sont souvent plus courtes et leur élimination permet donc de conserver des lectures plus longues et plus fiables pour l'analyse. La longueur médiane progresse également avec une hausse de + 9,0 % pour le contrôle (de 1553 à 1693 pb) et de + 11,9 % pour le modifié (de 1079 à 1207 pb). Le N50 (longueur à partir de laquelle 50 % du nombre total de bases est atteint : une valeur élevée de ce critère traduit une meilleure proportion de longues lectures) suit la même tendance, augmentant de 2195 à 2207 pb pour le contrôle et de 1825 à 1903 pb pour le modifié. Pour la qualité, le score Phred moyen passe de 8,8 à 9,6 (+ 9,1 %) pour le contrôle et de 7,9 à 8,7 (+ 10,1 %) pour le modifié. Les scores de qualité médiane progressent plus modérément avec + 3,1 % pour le contrôle (de 9,6 à 9,9) et + 4,8 % pour le modifié (de 8,3 à 8,7). On voit que le nombre total de bases diminue légèrement mais reste néanmoins majoritairement conservées, passant de 565 747 911 à 520 536 987 bases pour le contrôle (soit 92,0 %), et de 192 820 431 à 160 194 697 bases pour le modifié (soit 83,1 %).

La qualité légèrement inférieure observée pour l'échantillon modifié à 50 % peut s'expliquer par la présence de modifications m6A qui vont perturber le signal électrique mesuré lors du séquençage Nanopore, ce qui peut générer davantage d'erreurs lors de l'appel de base et donc abaisser les scores de qualité. Il est possible que ces modifications influencent la stabilité ou la structure locale des ARN, ce qui peut affecter leur comportement dans le nanopore. Une part de cette différence peut aussi relever de la variabilité technique naturelle entre les échantillons au niveau de la transcription in vitro ou de la préparation des bibliothèques.

Échantillon	Nombre de lectures	Longueur moyenne (pb)	Longueur médiane (pb)	N50	Qualité moyenne	Qualité médiane	Nombre total de bases
Contrôle non filtré	367312	1540.2	1553	2195	8.8	9.6	565 747 911
Contrôle filtré	319544 (87.0 %)	1629 (+5.8 %)	1693 (+9.0 %)	2207	9.6 (+9.1 %)	9.9 (+3.1 %)	520 536 987 (92.0 %)
Modifié non filtré	159867	1206.1	1079	1825	7.9	8.3	192 820 431
Modifié filtré	123269 (77.1 %)	1299.6 (+7.8 %)	1207 (+11.9 %)	1903	8.7 (+10.1 %)	8.7 (+4.8 %)	160 194 697 (83.1 %)

Tableau 1 **- Impact du filtrage qualité (seuil Phred >= 7) sur les lectures séquencées.** Ce tableau présente les caractéristiques des lectures avant et après filtrage pour les deux échantillons (contrôle et modifié contenant 50 % de modifications m6A). Les longueurs moyenne et médiane (exprimées en paires de bases) indiquent la taille des lectures. Le N50 correspond à la longueur à partir de laquelle 50 % du nombre total de bases est atteint (une valeur élevée traduit une meilleure proportion de longues lectures). Les qualités moyenne et médiane sont exprimées en scores Phred, qui estiment la fiabilité des bases (plus le score est élevé, moins l'erreur est probable). Le filtrage entraîne une amélioration globale d'environ 10 % des scores de qualité dans les deux conditions, avec un taux de rétention plus élevé pour l'échantillon contrôle (87 %) par rapport à l'échantillon modifié (~ 77 %).

Ensuite on voit sur la ***figure 5*** qui montre la distribution des lectures (qualité moyenne en fonction de la longueur des lectures) pour les deux échantillons (contrôle et modifié à 50 % m6A) avant et après filtrage, que les données non filtrées présentent une forte hétérogénéité. Avant filtrage, les deux échantillons montrent une grande densité de lectures courtes (inférieures à 500 pb) et de faible qualité (Phred < 7), traduisant un ensemble de séquences bruitées ou dégradées.

Après le filtrage les lectures restantes se concentrent dans une zone plus restreinte et homogène entre 800 et 1800 pb en longueur avec une qualité moyenne comprise majoritairement entre 8 et 13. On voit visuellement que le filtrage permet d'éliminer les lectures de très faible qualité ou de très courte longueur. Le nuage de points devient plus dense et resserré, et les distributions marginales (les histogrammes) montrent une amélioration globale dans les deux dimensions.

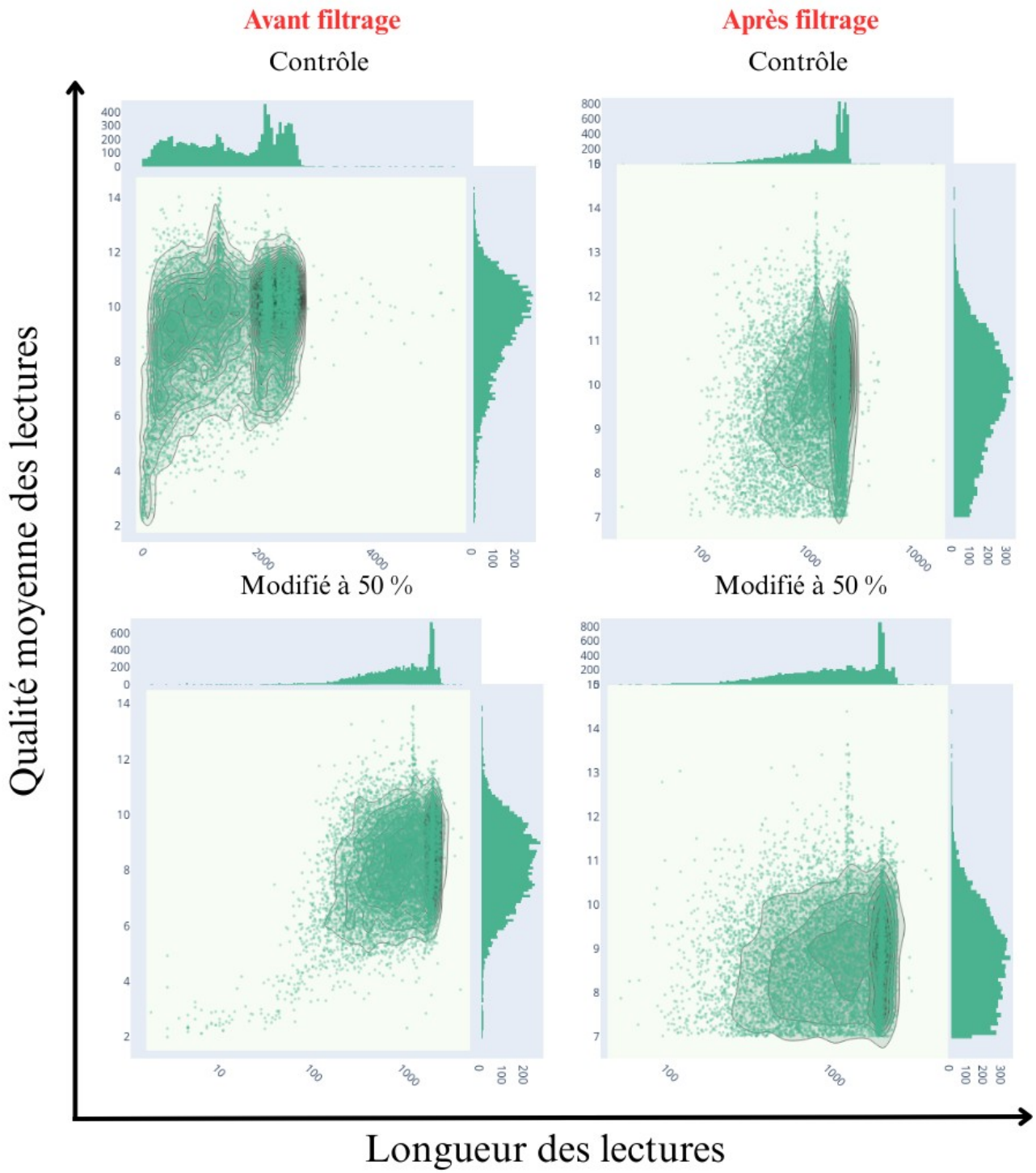


***Figure 5* - Graphiques comparatifs de la qualité moyenne des lectures en fonction de leur longueur avant et après filtrage (seuil Phred >= 7) réalisé avec NanoPlot.** Chaque graphique montre la distribution des lectures pour les deux échantillons (contrôle et modifié à 50 % m6A), avant et après filtrage. L'axe des ordonnées représente la qualité moyenne des lectures (score Phred) et l'axe des abscisses leur longueur (en paires de bases). Les nuages de points et les histogrammes marginaux permettent d'évaluer la densité et la répartition des lectures selon ces deux dimensions. Avant filtrage, une proportion importante de lectures courtes et de faible qualité est visible, notamment dans l'échantillon modifié. Après filtrage, les lectures se concentrent dans une zone plus homogène, plus longue et de meilleure qualité. Ceci confirme que le filtrage a permis de supprimer les lectures dégradées tout en conservant l'essentiel des données informatives.

B. Alignement des lectures sur la référence

1. Alignement avec minimap2

Les résultats de l'alignement des lectures avec minimap2, résumés dans le ***tableau 2***, ont donné un taux élevé de lectures alignées dans les deux échantillons. Pour l'échantillon contrôle, 301 925 lectures sur 319 544 ont été correctement alignées (94,5 %), contre 115 702 sur 123 269 pour l'échantillon modifié (93,9 %). Le nombre de lectures non alignées reste faible avec 5,5 % et 6,1 % respectivement. Ces bons résultats sont attendus dans un contexte de transcription in vitro en particulier pour l'échantillon non modifié, puisque les transcrits correspondent directement à la séquence de référence sans variations ni événements biologiques non canoniques susceptibles de compliquer l'alignement.

Les lectures alignées dans l'échantillon contrôle sont en moyenne plus longues (~1662 pb) et plus homogènes (longueur médiane : 1820 pb) que celles de l'échantillon modifié (1344 pb en moyenne et 1248 pb en médiane). Cette différence était déjà présente avant l'alignement ce qui indique qu'elle est liée aux lectures elles-mêmes, et non au processus d'alignement. On peut toutefois noter que l'alignement a eu tendance à accentuer légèrement l'écart de longueur médiane entre les deux échantillons, qui passe de 486 pb avant alignement (***tableau 1*** : 1693pb contre 1207pb) à 572 pb après alignement (***tableau 2*** : 1820pb contre 1248pb). En revanche, l'écart de qualité médiane se réduit légèrement après alignement entre les échantillons passant de 1,2 (***tableau 1*** : 9,9 contre 8,7) à 1,0 (***tableau 2*** : 9,8 contre 8,8), ce qui peut traduire une sélection plus marquée des lectures de meilleure qualité dans l'échantillon modifié. Le N50 suit la même tendance avec 2219 pb pour le contrôle contre 1946 pb pour le modifié. Au niveau de la qualité les scores Phred sont également plus élevés dans l'échantillon contrôle (qualité moyenne à 9,0 et médiane à 9,8) que dans l'échantillon modifié (respectivement 8,2 et 8,8). Les taux d'identité mesurent la proportion de bases identiques entre les lectures prédites alignées et la séquence de référence. Ils sont plus élevés dans l'échantillon contrôle,avec 87,1 % en moyenne et 88,0 % en médiane, contre 82,8 % et 83,4 % pour l'échantillon modifié. Cela indique que les séquences prédites dans l'échantillon modifié présentent davantage de divergences vis-à-vis de la référence, ce qui peut s'expliquer par une baisse de fidélité de l'appel de base induite par la présence de modifications chimiques comme la m6A. Cette observation est cohérente avec l'idée que ces modifications peuvent perturber localement le signal brut. On peut relever le fait que la qualité générale des deux échantillons baisse après alignement (-6 % de qualité moyenne après alignement par rapport aux données filtrées). Cela peut s'expliquer par le fait que avec le paramétrage spécifique aux données long-read, minimap2 privilégie les lectures plus longues même un peu bruitées, au détriment des lectures courtes qui ne

couvrent pas suffisamment la référence même si elle sont de meilleure qualité a priori.

Paramètre	Contrôle	Modifié
Nombre total de lectures	319544	123269
Non alignées (unmapped)	17619 (5,5 %)	7567 (6,1%)
Alignées (mapped)	301925 (94,5 %)	115702 (93,9 %)
Longueur moyenne (pb)	1662,1	1344,0
Longueur médiane (pb)	1820,0	1248,0
N50	2219,0	1946,0
Qualité moyenne	9,0	8,2
Qualité médiane	9,8	8,8
Identité moyenne (%)	87,1	82,8
Identité médiane (%)	88,0	83,4
Nombre total de bases	501 835 467	155 504 671

***Tableau 2* – Statistiques d'alignement des lectures filtrées sur la référence avec minimap2.** Cette table présente les résultats de l'alignement des lectures de chaque échantillon (contrôle et modifié à 50 % m6A) sur la séquence de référence. Le nombre total de lectures correspond aux séquences en entrée. Les lectures alignées (mapped) et non alignées (unmapped) sont indiquées avec leurs proportions respectives. Les longueurs moyenne et médiane (en paires de bases) décrivent la distribution des tailles des lectures alignées. Le N50 correspond à la longueur à partir de laquelle 50 % du nombre total de bases est atteint (une valeur élevée traduit une meilleure proportion de longues lectures). Les scores de qualité correspondent aux valeurs Phred moyennes et médianes des lectures alignées. L'identité (en %) mesure la similarité des lectures avec la séquence de référence (valeurs moyenne et médiane). La ligne Nombre total de bases donne le nombre total de bases alignées pour chaque échantillon. L'alignement est globalement très satisfaisant avec plus de 93 % des lectures alignées dans les deux conditions. Ce bon taux est attendu dans le cas de l'ARN IVT car les transcrits sont produits à partir d'une séquence de référence connue, sans variabilité naturelle. Ils sont synthétisés de manière contrôlée sans modifications imprévues ni hétérogénéité, ce qui ne reflète pas la complexité des conditions biologiques réelles.

2. IGV (Integrative Genomics Viewer) : Visualisation des alignements générés par minimap2 et détection de divergences locales entre échantillons

Dans la ***figure 6,*** les alignements des deux échantillons sont comparés dans IGV (Integrative Genomics Viewer) en prenant pour exemple la sous séquence de référence IVT_seq_2_2201 entre les positions 1221 et 1260. J'ai choisi cette région car elle montre des différences localisées dans les bases prédites, unique à l'échantillon modifié à 50 %. Les fichiers BAM des deux conditions obtenus après alignement avec minimap2, ainsi que la séquence de référence (GSE265754_IVTR.fa) ont été chargés dans IGV. Ce dernier affiche la couverture des lectures à chaque position sous forme de barres empilées. Une barre entièrement grise signifie que toutes les bases prédites dans les lectures correspondent à la base de la référence. En revanche, des barres colorées (rouge, vert, bleu) indiquent qu'à cette position plusieurs types de bases sont prédites parmi les lectures, ce qui reflète une couverture hétérogène en nucléotides. Dans le contexte d'ARN synthétiques (IVT) il est très peu probable que ces divergences correspondent à de véritables mutations, les transcrits étant produits à partir d'une séquence de référence fixe. Elles sont donc très

probablement dues à des perturbations du signal brut causées par la présence de m6A dans l'échantillon modifié.

Dans mon exemple, on observe dans l'échantillon modifié que certaines positions présentent des barres partiellement colorées (1236, 1241 et 1245), révélant une répartition différente des nucléotides dans les lectures à ces positions (où la séquence de référence contient respectivement les nucléotides A, A et C). Dans des séquences naturelles, ce type de phénomène peut correspondre à des mutations biologiques (insertion, délétion, substitution, duplication). Toutefois, nous sommes ici en présence de séquences contrôlées donc ces variations peuvent s'expliquer par des erreurs d'alignement ainsi que des erreurs techniques lors de l'appel de base (qui pourrait expliquer la position 1245 qui ne pourrait pas correspondre a une modification m6A comme la référence est un C), ou encore par une variation du signal engendrée par une modification chimique comme la m6A qui aurait perturbé les prédictions de l'appel de base. Le témoin ne présente aucune variation à ces positions (en concordance avec la séquence de référence), ce qui suggère une perturbation du signal dans l'échantillon modifié compatible avec l'effet local d'une potentielle modification sur l'ARN.

Cependant il faut remarquer le fait que la position 1245 correspond à une cytosine (C) dans la séquence de référence, ce qui exclut une modification de type m6A qui ne peut concerner que les adénosines. La variation observée dans l'échantillon modifié à cette position pourrait alors s'expliquer autrement. Une première hypothèse est une erreur d'appel de base induite de manière indirecte par une modification chimique située à proximité (une m6A dans un k-mer chevauchant cette position). Comme l'appel de base repose sur des k-mers (généralement des groupes de 5) une modification située en amont ou en aval peut perturber le signal global du k-mer, et fausser la prédiction même sur une base non modifiée. Une autre possibilité est une erreur de segmentation du signal ou un alignement approximatif lié à une baisse locale de qualité, qui pourrait introduire une base incorrecte à cette position dans certaines lectures. Le fait que cette variation soit absente du témoin renforce l'idée qu'elle est liée au contexte perturbé par la modification, même si elle ne touche pas directement une adénosine.

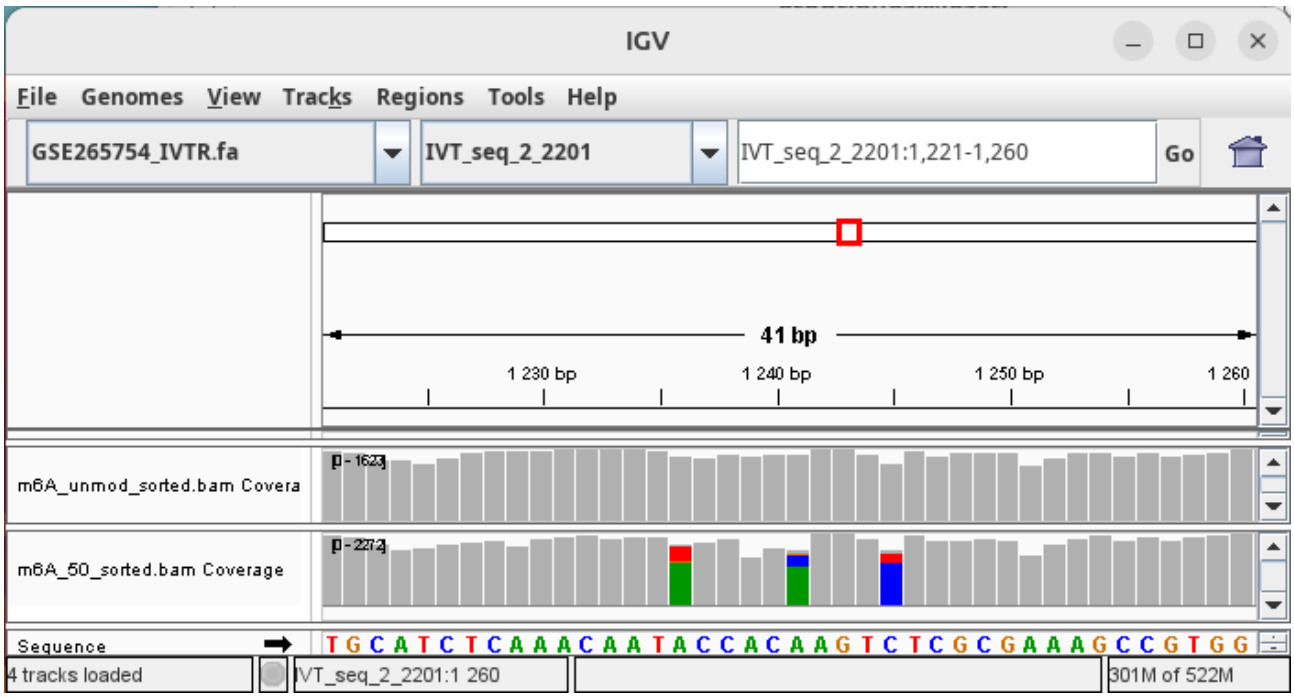


***Figure 6* – Visualisation avec IGV (Integrative Genomics Viewer) des alignements minimap2 sur la séquence IVT_seq_2_2201 : mise en évidence de variations spécifiques à l'échantillon modifié**. Cette figure montre la visualisation dans IGV des alignements générés avec minimap2 entre les positions 1221 et 1260 de la séquence IVT_seq_2_2201. J'ai choisi cette région comme exemple car elle illustre la présence de positions présentant des différences de base uniquement dans l'échantillon modifié, Les fichiers d'alignement (au format BAM) des deux échantillons (témoin non modifié m6A_unmod et modifié à 50 % de m6A m6A_50) sont chargés et comparés à la séquence de référence (GSE265754_IVTR.fa). L'affichage représente la couverture des lectures à chaque position où une barre entièrement grise indique qu'un seul type de nucléotide est observé dans toutes les lectures à cette position et qu'il est identique au nucléotide de la référence. Les barres colorées (rouge, vert et bleu) indique une position variante où plusieurs nucléotides différents sont retrouvés dans les lectures à cette position.

C. Association des signaux bruts aux lectures prédites alignées et comparaison des effets de la modification m6A sur la qualité des lectures associées

J'ai associé les signaux bruts (format BLOW5) aux lectures alignées à l'aide de f5c eventalign, pour les deux échantillons. Les résultats du ***tableau 3*** montrent que l'échantillon modifié présente une association signal–lecture légèrement moins efficace, ainsi qu'une qualité globale de lecture et d'alignement légèrement inférieure à celle du témoin.

Sur les 301 925 lectures alignées dans l'échantillon contrôle, 294 160 signaux ont été correctement associés donc 80,1 % des signaux bruts disponibles. Dans l'échantillon modifié, 110 870 signaux ont pu être associés à une lecture alignée ce qui représente 69,3 % des 160 135 signaux bruts. Il y a une bonne compatibilité entre les données de signal brut et les lectures alignées car le taux de lectures alignées conservées avec signal reste élevé dans les deux cas avec 97,4 % pour le contrôle et 95,8 % pour le modifié. Cette perte apparente de certains signaux, bien qu'ayant permis de prédire une séquence ne soient finalement associés à aucune lecture alignée, peut sembler

surprenante. Cette différence peut s'expliquer par le fait qu'une partie des lectures a déjà été éliminée lors du filtrage qualité (environ 20 % ± 5 % selon le ***tableau 1***), puis une autre partie n'a pas été alignée (environ 5,75 % ± 0,25 % selon le ***tableau 2***). À cela s'ajoutent d'autres facteurs comme de potentielles incohérences d'identifiants entre les fichiers de signal brut et les lectures alignées, des échecs dans le réalignement du signal (resquiggling) en particulier au moment de la segmentation du signal, ou encore par des lectures partiellement alignées. Des erreurs ponctuelles dans les fichiers d'indexation ou de métadonnées peuvent également empêcher certaines correspondances, même lorsque les signaux et lectures sont valides individuellement.

La comparaison des N50 entre les ***tableau 2*** et ***3*** permet d'évaluer l'effet de l'association signal-lecture sur la représentativité des longues lectures. Dans l'échantillon contrôle le N50 reste stable (***tableau 2*** : 2219 pb après alignement, ***tableau 3*** : 2221 pb après association), ce qui indique que les longues lectures sont bien conservées à cette étape. Dans l'échantillon modifié le N50 augmente légèrement (***tableau 2*** : 1946 pb, tableau 3 : 1966 pb), ce qui suggère que l'association avec le signal brut n'a pas dégradé la structure des lectures et pourrait même avoir favorisé une légère surreprésentation des lectures plus longues. Cette légère augmentation du N50 après l'association signal-lecture pourrait s'expliquer par la manière dont les lectures sont conservées au moment du réalignement du signal brut (resquiggling). Les lectures plus longues tendent à produire un signal plus stable et plus informatif, ce qui facilite leur traitement lors du réalignement (resquiggling). À l'inverse, les lectures courtes ou partielles sont plus sensibles aux effets de bord, aux erreurs de segmentation ou à des anomalies dans le signal, ce qui peut entraîner leur exclusion ou leur échec d'association. Par conséquent l'étape d'association peut introduire un biais en faveur des lectures plus longues, légèrement enrichies dans les données finales avec signal associé, ce qui se traduit par une augmentation du N50 dans l'echantillon modifié.

On remarque toutefois que l'augmentation du N50 ne s'accompagne pas nécessairement d'une amélioration de la qualité. En effet après l'association signal-lecture (resquiggling) même si le N50 augmente dans l'échantillon modifié dans le ***tableau 3***, la qualité médiane des lectures ne s'améliore pas et reste identique a celle des résultats du ***tableau 2*** (alignement minimap2). Cela reste cohérent car une lecture longue peut être correctement alignée et associée au signal, mais avoir un score Phred faible si l'appel de base est rendu incertain par la présence de modifications comme la m6A dans l'échantillon modifié.

Paramètre	Contrôle	Modifié à 50 %
Nombre de lectures alignées	301925	115702
Nombre de signaux bruts individuels	367 312	160 135
Signaux bruts associés à une lecture alignée	294 160 (80,1 %)	110 870 (69,3 %)
% de lectures alignées conservées avec signal	97,4 %	95,8 %
Longueur moyenne (pb)	1688,6	1376,7
Longueur médiane (pb)	1871,0	1296,0
N50	2221,0	1966,0
Qualité moyenne	9,1	8,2
Qualité médiane	9,9	8,8
Identité moyenne (%)	87,1	82,9
Identité médiane (%)	88,0	83,5
Nombre total de bases	496 725 826	152 630 126

***Tableau 3* – Résultat de l'association des signaux bruts aux lectures alignées avec f5c eventalign.** Ce tableau présente les statistiques issues de l'alignement des signaux bruts (au format BLOW5) sur les lectures alignées à l'aide de l'outil f5c eventalign, pour les deux échantillons (contrôle et modifié à 50 % m6A). Le nombre de lectures alignées correspond aux lectures préalablement alignées avec minimap2. Le nombre de signaux bruts individuels représente les lectures ayant un signal associé dans les fichiers BLOW5. La ligne « Signaux bruts associés à une lecture alignée » indique combien de ces signaux ont pu être associés avec succès à une lecture alignée. Le pourcentage correspondant indique la proportion de signaux correctement alignés. Le taux de conservation des lectures alignées avec signal indique la fraction des lectures pour lesquelles un signal a été retrouvé et utilisé parmi les lectures alignées initiales. Les longueurs moyenne et médiane (exprimées en paires de bases) décrivent la distribution des tailles des lectures associées à un signal. Le N50 correspond à la longueur à partir de laquelle 50 % du nombre total de bases est atteint (une valeur élevée traduit une meilleure proportion de longues lectures). Les valeurs de qualité moyenne et médiane (scores Phred) estiment la fiabilité de l'appel de base dans les lectures concernées. Les taux d'identité moyen et médian mesurent le pourcentage de correspondance entre les lectures alignées et la séquence de référence et reflètent la fidélité de l'alignement dans les régions associées aux signaux. L'association entre signal brut et lecture alignée est plus efficace dans l'échantillon contrôle (80,1 %) que dans l'échantillon modifié (69,3 %), mais le taux de conservation des lectures alignées reste élevé dans les deux cas (97,4 % et 95,8 % respectivement).

D. Visualisation et analyse des signaux

Pour les figures de résultats Python qui suivent, j'ai utilisé le jeu de données au format TSV obtenu à partir des alignements des signaux bruts avec l'outil `f5c eventalign` (voir partie *Matériel et Méthodes : Visualisation des signaux que j'ai développée avec Python*).

1. Interpolation linéaire des signaux bruts sur Python

La ***figure 7*** illustre l'effet de l'interpolation linéaire sur le signal brut, en prenant pour exemple la position génomique 1221. Elle montre l'intensité du signal (en pA) en fonction de l'index des points du segment de signal brut associés à cette position, avant et après interpolation, dans une lecture témoin et une lecture modifiée. Le choix d'interpoler chaque signal à 50 points (voir Matériel et

Méthodes) avait pour objectif de couvrir la majorité des cas sans extrapoler fortement les signaux courts, ni perdre trop d'information dans les signaux longs. Cet exemple semble confirmer la pertinence de ce choix.

On observe dans la ***figure 7*** deux cas où le nombre de points s'écarte fortement de la médiane du jeu de données (42 points) : un signal court de 11 points pour la lecture témoin, et un signal long de 86 points pour la lecture modifiée. Cette figure permet de visualiser la différence de longueur des signaux bruts, et montre la nécessité de réaliser une interpolation linéaire pour pouvoir les comparer sur une échelle commune. Il serait en effet impossible de comparer directement ces signaux sans interpolation puisque à cette même position, le signal extrait de la lecture de l'échantillon modifié est environ huit fois plus long que celui du contrôle, alors que ces signaux représentent exactement la même portion de k-mer.

On pourrait supposer que cette différence de longueur est liée au ralentissement de la vitesse de translocation dans l'échantillon modifié, induit par la présence de modifications chimiques comme la m6A, qui sont connues pour ralentir le passage des lectures dans le nanopore. Cependant, à la position 1221 la couverture est uniforme et la base prédite est identique à la référence dans les deux échantillons (voir ***figure 6***), ce qui ne suggère pas de modification à cet endroit. La différence de longueur ne peut donc probablement pas s'expliquer par un effet d'une m6A à cette position ni dans son voisinage immédiat, car les positions autour de 1221 (visibles ou non sur la ***figure 7***) ne présentent pas d'hétérogénéité de couverture sur IGV, ce qui suggère l'absence de modification susceptible d'influencer le signal dans cette région.

Cela suggère que des écarts de longueur aussi importants que dans cet exemple pourraient être liés soit à des différences réelles de vitesse de translocation dans les régions voisines (non influencé par une modification), soit à des erreurs de segmentation du signal brut lors du réalignement (*resquiggling*).

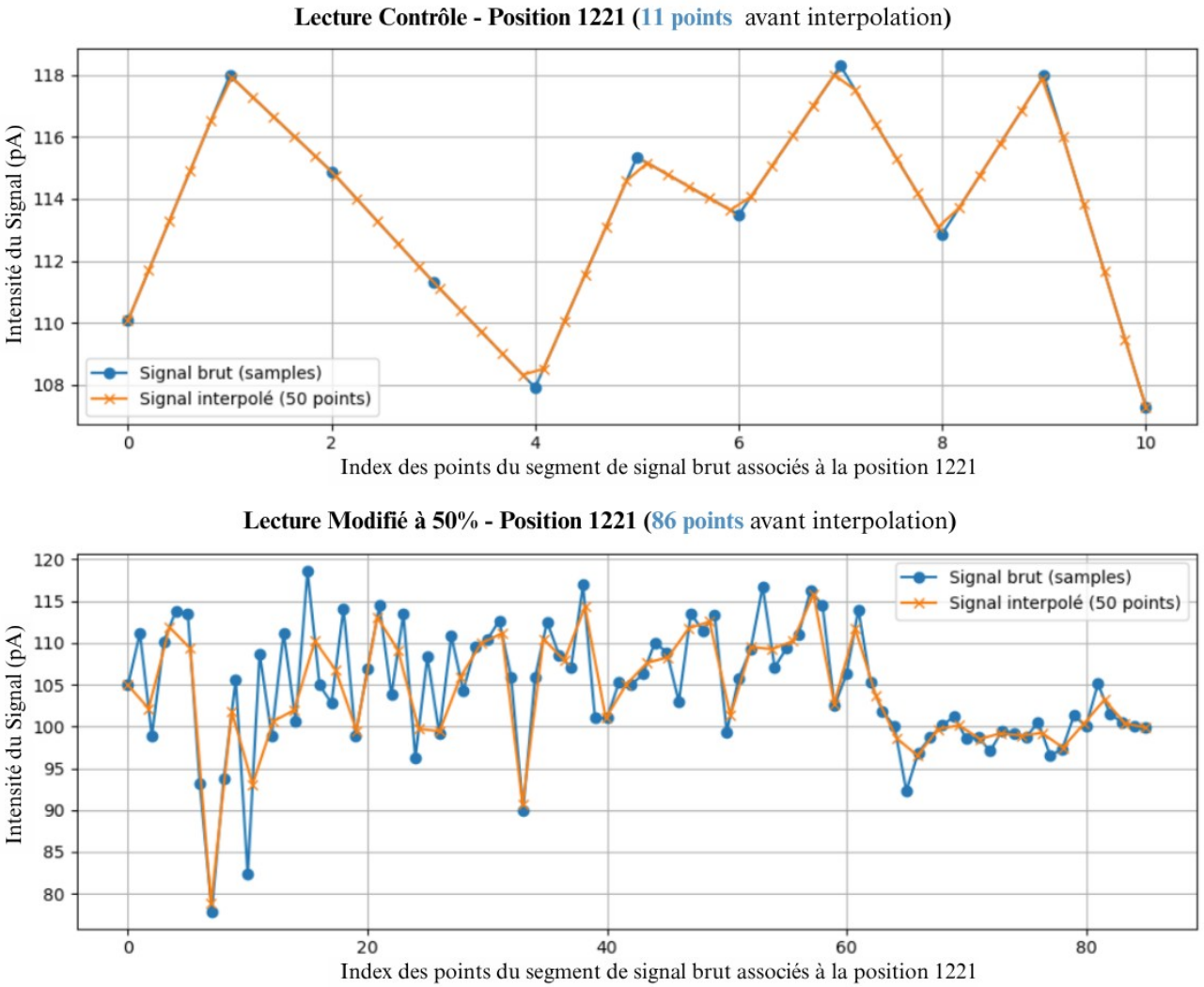


***Figure 7* - Visualisation graphique de l'effet de l'interpolation linéaire sur les signaux bruts pour homogénéiser leurs longueurs entre lectures sur une position commune.** Cette figure montre le signal brut mesuré à la position 1221 pour deux lectures dont une lecture témoin (haut) avec 11 points et une lecture modifiée à 50 % (bas) avec 86 points. L'axe des abscisses correspond à l'index des points mesurés dans le segment de signal brut associé à cette position, et l'axe des ordonnées à l'intensité du signal (en pA). Les signaux bruts associés à une même position n'ont pas la même longueur selon les lectures et cette variation empêche toute comparaison directe entre lectures. L'interpolation linéaire convertit chaque segment en un vecteur de 50 points. Dans la lecture témoin (signal court), l'interpolation ajoute des points intermédiaires. Dans la lecture modifiée (signal long) elle lisse et réduit le nombre de points. Cette normalisation permet ensuite d'aligner les signaux entre lectures et conditions.

2. Visualisation du signal brut interpolé sur une fenêtre glissante centrée sur un site candidat m6A

La ***figure 8*** est une comparaison directe des signaux interpolés entre lectures contrôle et modifiées à 50 % m6A sur une fenêtre de cinq positions décroissantes de 1236 à 1232. Cette région inclut la position 1236 identifiée précédemment comme potentiellement modifiée (voir ***figure 6***), où les lectures modifiées présentent un nucléotide C au lieu du A attendu dans la référence. J'ai assemblé les segments de signal interpolé à 50 points pour chacune des cinq positions, en suivant une fenêtre glissante triée par ordre de position génomique décroissante (Les signaux sont mesurés dans le sens

3’ → 5’ correspondant à la direction de translocation de l’ARN dans le pore). Cela donne une courbe continue de 250 points correspondant à cinq 5-mers. Une des lectures modifiées (ID : 4c51a7a2-81cf-4ae2-9b08-a2a2f854de10) n’a pas de signal associé aux positions 1233 et 1232, ce qui provoque une interruption nette dans sa courbe. Ce manque de données à ces positions peut être dû à une absence de couverture, un problème durant la segmentation du signal ou encore à une difficulté d’alignement du signal.

Le *k*-mer de référence assigné par f5c eventalign par segment de signal à chaque position est indiqué en haut et une astérisque jaune marque les occurrences du nucléotide potentiellement modifié (ici un Å à la position 1236). Représenter le signal sur une fenêtre glissante centrée autour de ce Å permet d’évaluer si la modification chimique supposée (ici m6A) a un effet local ponctuel ou si elle perturbe également le signal mesuré dans les k-mers voisins, où ce même Å apparaît dans un contexte différent. Cela permet de détecter une éventuelle propagation de l’effet de la modification sur plusieurs positions consécutives, ce qui renforcerait l’hypothèse d’un impact réel sur le signal brut.

Pour conclure sur les résultats de la ***figure 8***, malgré la mise en évidence d’une position potentiellement modifiée (1236) et l’intégration d’une fenêtre glissante couvrant plusieurs *k*-mers successifs, la visualisation des signaux interpolés ne révèle pas de perturbation franche, systématique ou homogène dans les courbes des lectures modifiées par rapport aux contrôles. Les profils de signal présentent une variabilité importante entre lectures même au sein d’un même groupe, ce qui rend difficile la mise en évidence d’un effet net attribuable à la modification.

Néanmoins on peut noter visuellement que dans les k-mers 3 à 5, les courbes des lectures modifiées semblent globalement décalées par rapport à celles des témoins. Cette différence reste à confirmer de manière quantitative (par un test statistique ou une autre méthode d’analyse), mais elle pourrait indiquer un effet local de la modification chimique sur la forme du signal dans cette région.

Il convient toutefois de rappeler que dans l’échantillon modifié à 50 %, les adénosines sont modifiées de manière aléatoire. On ne peut donc pas garantir que les lectures utilisées dans cette figure sont effectivement modifiées à cet endroit précis. Cependant au vu des résultats précédent de l’alignement des séquences prédites visibles dans la ***figure 6***, qui montre une hétérogénéité de l’appel de base à la position 1236 uniquement dans l’échantillon modifié, on suppose a priori que les lectures présentant cette hétérogénéité (et que j’ai sélectionnées spécifiquement dans les ***figure 7*** , ***8*** et ***9***) portent bien une modification, ce qui justifie l’analyse visuelle centrée sur cette position.

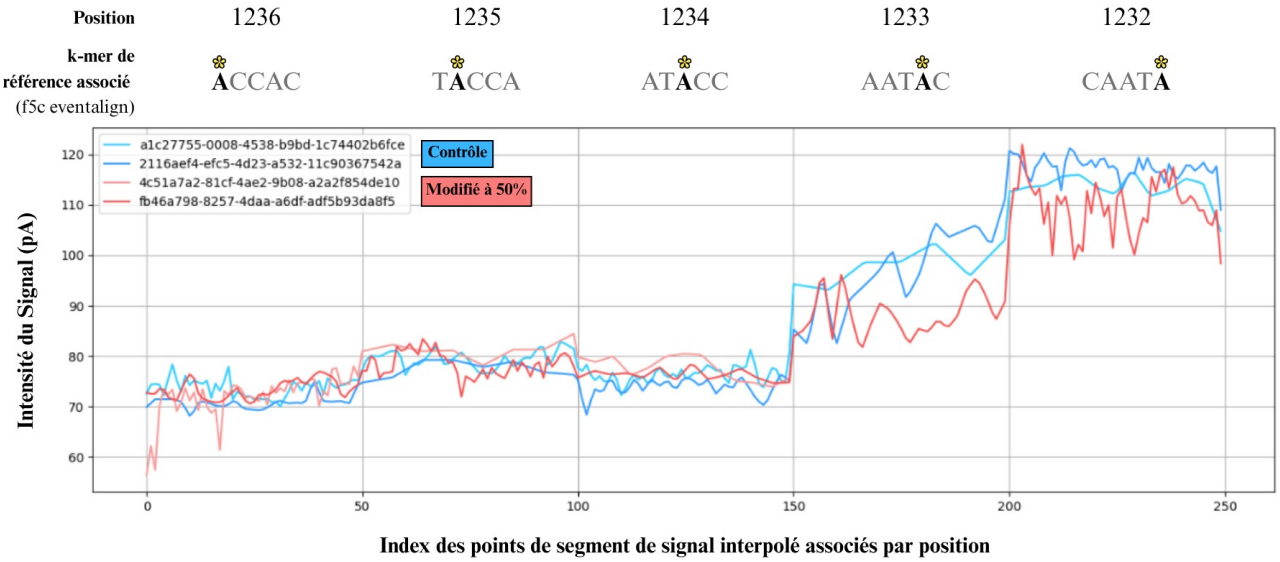


***Figure 8* – Comparaison des signaux interpolés (3' → 5') sur une fenêtre de positions décroissantes autour d'un site potentiellement modifié.** Cette figure montre l'évolution du signal interpolé (en pA) pour deux lectures contrôle (en bleu) et deux lectures modifiées à 50 % m6A (en rouge) sur une fenêtre de 5 positions génomiques comprises entre 1236 et 1232. Sur la visualisation IGV de la Figure 2, ces lectures (en rouge) de l'échantillon modifié présentaient des nucléotides différents (C) de la référence (A) à la position 1236 . Les signaux sont affichés en concaténant les segments interpolés à 50 points par position (250 points par lecture au total). Les positions sont représentées dans l'ordre décroissant (de 1236 à 1232) conformément à la direction 3' → 5' du signal dans les données Nanopore. Cette inversion reflète la manière dont les événements sont alignés et prédits par f5c eventalign. Les k-mers de référence assignés à chaque segment de position sont indiqués en haut de la figure, avec une astérisque jaune signalant la présence du nucléotide potentiellement modifié (ici un A à la position 1236). Ce même nucléotide est également présent dans les k-mers suivants ce qui permet d'observer son impact au niveau de la variation du signal sur plusieurs positions successives. Un espace vide apparaît sur la courbe quand aucun signal n'est associé à une position, comme pour la lecture modifiée en rouge clair (ID : 4c51a7a2-81cf-4ae2-9b08-a2a2f854de10) qui est absente aux positions 1233 et 1232.

3. Développement d'un outil de visualisation du signal en Python et comparaison avec la visualisation de l'outil Squigualiser

La ***figure 9*** compare la visualisation du signal brut obtenue avec Squigualiser (A) et celle générée avec mon script Python (B) pour deux lectures contrôle et deux lectures modifiées à 50 % m6A dans la région IVT_seq_2_2201:1221–1260. Pour faciliter la comparaison j'ai annoté les lectures de 1 à 4 et surligné les positions 1236, 1241 et 1245 identifiées comme présentant une variation de couverture dans IGV (voir ***figure 6***).

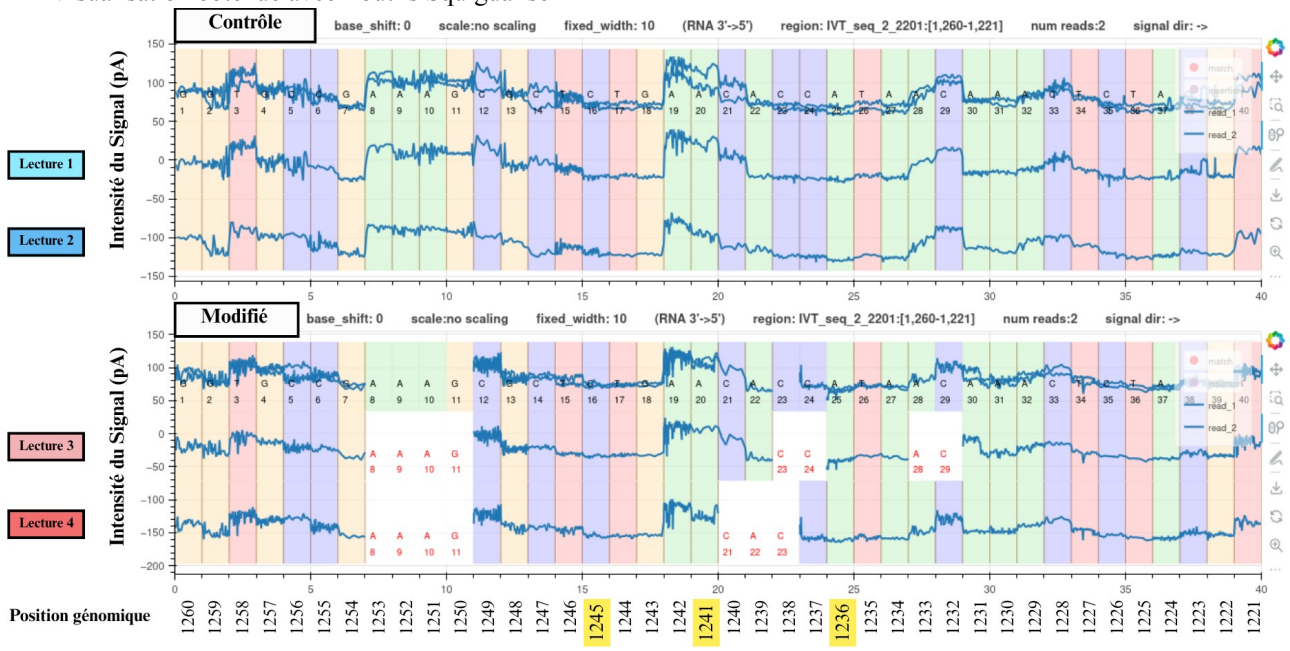
Dans la ***figure 9-***A, on peut voir que Squigualiser affiche les signaux alignés à la séquence de référence à l'échelle du nucléotide. Les lectures contrôle (lectures 1 et 2) présentent un signal continu sur l'ensemble de la région mais les lectures modifiées (lectures 3 et 4) présentent plusieurs absences de signal. Ces absences sont représentées par des rectangles blancs annotés avec les bases correspondantes de la séquence de référence (en rouge), indiquant qu'aucun signal n'a été associé à ces positions dans la lecture alignée. Une absence commune pour les lectures 3 et 4 est observée entre les positions 1253 et 1250, la lecture 3 présente deux autres zones non couvertes entre 1238 et

1237, et entre 1233 et 1232 et la lecture 4 présente une absence de signal entre 1240 et 1238 (à peu près dans la même zone que dans la lecture 3).

Dans la ***figure 9-***B cette visualisation est reproduite par la fonction développée en Python, avec les signaux interpolés superposés pour l'ensemble des lectures que j'ai tracé une première fois à son niveau réel, puis une seconde fois en dessous avec un décalage vertical (voir Matériel et Méthodes - Représentation graphique des signaux). Pour cela, j'ai calculé le décalage artificiel dynamiquement à partir de -50 pA sous le maximum de la première lecture, avec un espacement fixe entre les courbes. Les zones sans signal sont visibles sous forme d'interruptions nettes dans les courbes, aux mêmes positions que celles observées dans la visualisation Squigualiser.

La comparaison entre les deux visualisations montre que les principales caractéristiques du signal affiché par Squigualiser sont correctement reproduites. Cela valide encore une fois le choix que j'ai fait d'utiliser une interpolation à 50 points pour comparer les signaux, ainsi que l'ensemble de mon développement concernant le prétraitement, l'affichage et la comparaison visuelle des signaux. En particulier les éléments liés à la distinction entre lectures, à la représentation des absences de signal et à la lisibilité des profils y compris en cas de superposition, semblent bien restitués.

A - Visualisation obtenue avec l'outils Squigualiser



B - Visualisation obtenue avec mon script Python - plot_superposed_reads_with_shifted_clones()

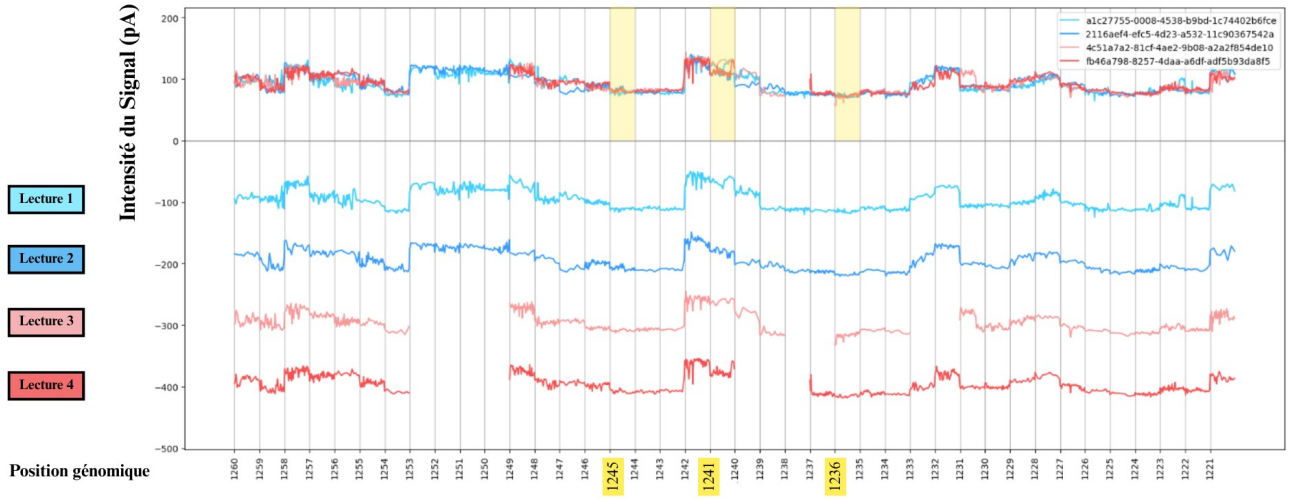


***Figure 9* - Comparaison de deux approches de visualisation du signal brut entre conditions expérimentales.** Cette figure compare deux approches de visualisation des signaux bruts issus du séquençage direct d'ARN par nanopore pour deux lectures contrôle et deux lectures modifiées à 50 % m6A, dans la région IVT_seq_2_2201:1221–1260. L'enchaînement des lectures est identique entre les deux visualisations : les deux premières correspondent à l'échantillon contrôle et les deux suivantes à l'échantillon modifié, dans le même ordre d'affichage que dans Squigualiser. J'ai annoté les lectures directement dans la figure (étiquettes bleues pour les contrôles avec les lectures 1 et 2, rouges pour les modifiées avec les lectures 3 et 4), afin d'en faciliter le repérage. Les positions génomiques d'intérêt (1236, 1241 et 1245), identifiées comme présentant une variation de couverture dans IGV, sont surlignées en jaune dans « Position génomique » (ainsi que sur le graphique pour la sous figure B).

(A) - La sous-figure A correspond à la visualisation obtenue avec l'outil interactif Squigualiser (doi:10.1093/bioinformatics/btae501). Le signal est aligné à la séquence de référence (signal-to-reference), affiché à résolution nucléotidique dans une vue pileup (empilement de lectures). Les lectures modifiées et non modifiées sont affichées sur deux pistes distinctes (parallel tracks) pour faciliter la comparaison. Le premier nucléotide des k-mers de référence associés à chaque position est indiqué en haut de chaque colonne colorée. L'absence de signal est modélisé par des rectangles blancs avec les bases manquantes dans les lectures annotées en rouge (correspondants aux bases de la reference).

(B) - La sous-figure B montre le signal brut interpolé obtenu à partir d'un fichier TSV généré par f5c eventalign pour les lectures sélectionnées. Le signal est affiché en utilisant la fonction Python *plot_superposed_reads_with_shifted_clones()* (voir Matériel et Méthodes) que j'ai développée pour superposer les lectures tout en appliquant un décalage vertical permettant de mieux distinguer les courbes. Chaque lecture est une première fois à son niveau d'intensité réel en superposition avec les autres lectures et une seconde fois en dessous en décalé afin de visualiser et comparer plus facilement les profils des différentes lectures. Le décalage vertical est calculé dynamiquement en commençant à -50 pA en dessous du maximum de la première lecture, puis un espacement fixe est appliqué entre chaque courbe décalée pour éviter tout chevauchement.

IV. Discussion

Le filtrage qualité (***tableau 1,figure 5***) a permis d'éliminer efficacement les lectures courtes et peu informatives, majoritairement présentes dans l'échantillon modifié, tout en préservant l'essentiel des données utiles. Ces lectures de faible qualité sont susceptibles de perturber l'analyse en raison d'un signal de séquençage plus bruité. Après filtrage entre les deux échantillons les lectures restantes sont plus homogènes en longueur et en qualité.(***tableau 2***).

La longueur médiane des lectures est systématiquement plus faible dans l'échantillon modifié et cet écart s'accentue légèrement après alignement, tandis que la différence de qualité médiane se réduit entre les deux échantillons. Par contre le fait que la qualité générale baisse pour les deux échantillons après alignement par rapport a celle des données juste filtrée montre un effet du paramétrage spécifique pour les lectures longues que j'ai utilisé dans minimap2, qui semble privilégier les lectures plus longues même un peu bruitées, au détriment des lectures courtes qui ne couvrent pas suffisamment la référence (***tableau 2***). En parallèle l'étape d'association entre les signaux et les séquences peut introduire un biais en faveur des lectures plus longues, contribuant à l'augmentation du N50 observée dans l'échantillon modifié (***tableau 3***). Toutefois, cette augmentation de longueur ne s'accompagne pas d'une amélioration de la qualité des lectures, la médiane des scores Phred restant stable entre l'alignement et le resquiggling (***tableau 2*** et ***3***). Cela suggère qu'une lecture longue peut être bien alignée et associée à un signal, tout en présentant une faible qualité si l'appel de base est incertain en raison de la présence de modifications comme la m6A, connues pour perturber le signal électrique du séquençage Nanopore. Une part de cette variabilité peut également provenir des conditions expérimentales liées à la transcription in vitro. De plus le fait que l'étape d'association signal-lecture prédite puisse introduire un biais en faveur des lectures plus longues dans l'échantillon modifié (***tableau 3***), possiblement en raison de la difficulté à associer un signal altéré par les modifications, pourrait masquer des lectures pertinentes. Ce phénomène est à considérer lors de l'apprentissage d'un modèle visant à identifier les modifications, car il pourrait influencer la représentativité des données d'entraînement.

L'hétérogénéité observée dans la composition des bases prédites par rapport à la référence dans l'échantillon modifié dans mon analyse IGV (***figure 6***) peut s'expliquer par l'influence directe d'une modification chimique si celle-ci est portée par la position concernée. Cependant, lorsque la base de référence ne peut pas être modifiée (comme à la position 1245 qui est une cytosine qui ne peut pas porter de modification de type m6A) cette variation peut résulter de l'effet indirect d'une modification voisine. En effet, une m6A présente dans un k-mer chevauchant pourrait perturber le

signal au voisinage et fausser l'appel de base sur une position non modifiée. Alternativement, une erreur technique telle qu'une mauvaise segmentation du signal ou un alignement approximatif dans une zone de moindre qualité, pourrait également expliquer cette hétérogénéité apparente. Il convient toutefois de souligner que ce scénario s'inscrit dans un contexte expérimental particulier basé sur une transcription in vitro, qui ne reflète pas directement la complexité ni la distribution naturelle des modifications dans les ARN biologiques.

La différence importante de points captés entre les deux échantillons illustré dans la ***figure 7*** à la position 1221 alors qu'il n'y a ni variation de base prédite ni signe de modification, peut s'expliquer par différentes hypothèses. Il peut s'agir d'une vraie différence de vitesse de translocation entre les lectures ce qui peut arriver même sans modification. Mais il peut aussi s'agir d'une erreur de segmentation du signal lors du resquiggling. Dans un échantillon modifié où le signal est plus bruité, ce genre d'erreur est plus fréquent. Ces deux hypothèses restent possibles et montrent qu'une différence de longueur n'indique pas forcément une modification. On peut également noter que la transformation appliquée dans la ***figure 7***, visant à normaliser la longueur des signaux pour les rendre comparables, est certes indispensable, mais elle peut aussi masquer certaines micro-variations et ainsi réduire la sensibilité de l'analyse.

La visualisation des signaux interpolés sur une fenêtre centrée autour de la position 1236 (***figure 8***) révèle une forte variabilité entre lectures, y compris au sein d'un même groupe. Aucune signature stable ou reproductible du signal associé à la modification n'émerge visuellement. Les courbes restent hétérogènes et aucune tendance cohérente ne se dégage, ce qui rend toute interprétation robuste difficile à partir de ces profils seuls. De plus, même si j'ai spécifiquement sélectionné des lectures potentiellement modifiées aux trois positions et sur lesquelles on observe des erreurs d'appel de base dans la condition modifiée sur IGV, l'analyse effectuée n'est basée que sur deux lectures par condition (à titre illustratif), alors que l'ensemble des données comprend plusieurs milliers de lectures (et plusieurs séquences de référence, pas uniquement la séquence 2 utilisée dans mon exemple). Dans ce contexte, toute tentative de comparaison visuelle exhaustive et précise des variations fines de signal est irréaliste. Bien que certaines fluctuations locales observées puissent évoquer une différence liée à la modification, l'ensemble des données représentées ici ne permet pas de conclure de manière fiable à une perturbation systématique ou à une propagation claire de l'effet du nucléotide modifié sur les *k*-mers voisins. Une piste d'amélioration serait de générer des courbes moyennes ou médianes avec des intervalles de confiance dans les deux conditions, à partir d'un plus grand nombre de lectures pour mieux explorer d'éventuelles différences de signal.

La comparaison avec Squigualiser (***figure 9***) confirme que l' outil de visualisation développé en Python reproduit les principales caractéristiques attendues en alignant les signaux à la référence et en modélisant correctement les absences de signal. Les comparaisons visuelles seules ne permettent pas de conclure à une différence significative entre les signaux des échantillons. Pour objectiver ces différences, il me semble nécessaire d'extraire des métriques numériques et de les analyser par des tests statistiques. Cela permettrait non seulement d'objectiver les différences, mais aussi de rendre les données utilisables pour l'apprentissage d'un modèle visant à identifier la modification, qui était l'objectif initial. À noter que Squigualiser se limite à la visualisation et ne permet pas directement d'extraire les données prétraitées pour ce type d'analyse.

La prochaine étape pour la suite serait d'effectuer un appel de variants (variant calling) à partir des fichiers BAM générés par f5c eventalign pour récupérer les métriques par position comme la fréquence des bases par position ou le score de qualité de l'appel (**(Furlan *et al*, 2021)**, **(Pagès-Gallego & de Ridder, 2023)**). À chaque site ces métriques d'alignement auraient pu être croisées avec des métriques dérivées du signal brut comme la moyenne, l'écart-type ou la durée de l'événement segmenté par f5c. L'analyse conjointe de ces indicateurs avec des tests statistiques entre conditions pourrait permettre de détecter des écarts systématiques au niveau du signal ou de la séquence et d'identifier des positions potentiellement modifiées de manière plus précise. L'idée serait, à partir de ces métriques d'appel de variant associées aux valeurs de signal brut par position, de séparer les positions en deux groupes annotés comme modifiées ou non. À partir de ce jeu étiqueté, un modèle d'apprentissage automatique supervisé pourrait être entraîné en utilisant les métriques extraites pour chaque position (fréquence des bases, scores de qualité, moyenne du signal, écart-type, durée des événements) comme variables d'entrée pour prédire la probabilité qu'une position donnée porte une modification.

Un des facteurs limitants important durant mon stage a été la taille très volumineuse des données issues du séquençage Nanopore. À titre d'exemple, les fichiers TSV générés à partir des signaux associés aux séquences alignées avec f5c eventalign dépassaient chacun la centaine de gigaoctets. Cette contrainte technique m'a conduite à me concentrer sur la seule séquence 2 pour mes analyses exploratoires. De manière générale les données Nanopore, notamment les fichiers de signal brut (fast5/pod5/blow5), sont particulièrement lourdes à manipuler (jusqu'à plusieurs centaines de gigaoctets par échantillon). Un autre facteur limitant est que la technologie étant relativement récente et qu'elle évolue donc très rapidement. De ce fait, certains outils deviennent rapidement obsolètes ou ne sont plus maintenus, ce qui complexifie leur utilisation ou limite leur compatibilité avec les dernières versions des formats de données ou des logiciels d'analyse.

Conclusion personnelle

Ce stage a été une expérience particulièrement enrichissante, tant sur le plan technique que scientifique. Il m'a permis de développer des compétences en gestion de données massives et en traitement de signaux complexes. Travailler sur une technologie de pointe comme le séquençage direct Nanopore m'a confrontée à des défis concrets liés à l'évolution rapide de cette méthode, tout en stimulant ma curiosité scientifique et ma capacité à résoudre des problèmes. Au-delà des aspects techniques, ce projet m'a permis de mieux appréhender les enjeux de l'épitranscriptomique et la richesse des questions de recherche encore ouvertes dans ce domaine. J'ai particulièrement apprécié l'autonomie qui m'a été accordée ainsi que la possibilité de m'impliquer dans une thématique à la fois actuelle et innovante.

Cette expérience a renforcé mon intérêt pour ce champ de recherche et m'a clairement donné envie de poursuivre dans cette voie. J'aimerais approfondir ce projet dans le cadre d'une thèse, afin de contribuer au développement de nouvelles approches pour la détection des modifications de l'ARN.

Références

Cappannini A, Ray A, Purta E, Mukherjee S, Boccaletto P, Moafinejad SN, Lechner A, Barchet C, Klaholz BP, Stefaniak F, *et al* (2024) MODOMICS: a database of RNA modifications and related information. 2023 update. *Nucleic Acids Res* 52: D239–D244

Chen H-X, Liu Z-D, Bai X, Wu B, Song R, Yao H-C, Chen Y, Chi W, Hua Q, Cheng L, *et al* (2025) Accurate cross-species 5mC detection for Oxford Nanopore sequencing in plants with DeepPlant. *Nat Commun* 16: 3227

Davis FF & Allen FW (1957) RIBONUCLEIC ACIDS FROM YEAST WHICH CONTAIN A FIFTH NUCLEOTIDE. *J Biol Chem* 227: 907–915

De Coster W & Rademakers R (2023) NanoPack2: population-scale evaluation of long-read sequencing data. *Bioinformatics* 39: btad311

Deamer D, Akeson M & Branton D (2016) Three decades of nanopore sequencing. *Nat Biotechnol* 34: 518–524

Dominissini D, Moshitch-Moshkovitz S, Schwartz S, Salmon-Divon M, Ungar L, Osenberg S, Cesarkas K, Jacob-Hirsch J, Amariglio N, Kupiec M, *et al* (2012) Topology of the human and mouse m6A RNA methylomes revealed by m6A-seq. *Nature* 485: 201–206

Furlan M, Delgado-Tejedor A, Mulroney L, Pelizzola M, Novoa EM & Leonardi T (2021) Computational methods for RNA modification detection from nanopore direct RNA sequencing data. *RNA Biol* 18: 31–40

Gamaarachchi H, Lam CW, Jayatilaka G, Samarakoon H, Simpson JT, Smith MA & Parameswaran S (2020) GPU accelerated adaptive banded event alignment for rapid comparative nanopore signal analysis. *BMC Bioinformatics* 21: 343

Garalde DR, Snell EA, Jachimowicz D, Sipos B, Lloyd JH, Bruce M, Pantic N, Admassu T, James P, Warland A, *et al* (2018) Highly parallel direct RNA sequencing on an array of nanopores. *Nat Methods* 15: 201–206

He PC & He C (2021) m^6 A RNA methylation: from mechanisms to therapeutic potential. *EMBO J* 40: e105977

Herbert C, Valesyan S, Kist J & Limbach PA (2024) Analysis of RNA and Its Modifications. *Annu Rev Anal Chem* 17: 47–68

Hu L, Liu S, Peng Y, Ge R, Su R, Senevirathne C, Harada BT, Dai Q, Wei J, Zhang L, *et al* (2022) m6A RNA modifications are measured at single-base resolution across the mammalian transcriptome. *Nat Biotechnol* 40: 1210–1219

Hussain S, Sajini AA, Blanco S, Dietmann S, Lombard P, Sugimoto Y, Paramor M, Gleeson JG, Odom DT, Ule J, *et al* (2013) NSun2-Mediated Cytosine-5 Methylation of Vault Noncoding RNA Determines Its Processing into Regulatory Small RNAs. *Cell Rep* 4: 255–261

Li H (2018) Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics* 34: 3094–3100

Linder B, Grozhik AV, Olarerin-George AO, Meydan C, Mason CE & Jaffrey SR (2015) Single-nucleotide-resolution mapping of m6A and m6Am throughout the transcriptome. *Nat Methods* 12: 767–772

Meyer KD, Saletore Y, Zumbo P, Elemento O, Mason CE & Jaffrey SR (2012) Comprehensive Analysis of mRNA Methylation Reveals Enrichment in 3′ UTRs and near Stop Codons. *Cell* 149: 1635–1646

Midha MK, Wu M & Chiu K-P (2019) Long-read sequencing in deciphering human genetics to a greater depth. *Hum Genet* 138: 1201–1215

Pagès-Gallego M & de Ridder J (2023) Comprehensive benchmark and architectural analysis of deep learning models for nanopore sequencing basecalling. *Genome Biol* 24: 71

Saletore Y, Meyer K, Korlach J, Vilfan ID, Jaffrey S & Mason CE (2012) The birth of the Epitranscriptome: deciphering the function of RNA modifications. *Genome Biol* 13: 175

Samarakoon H, Ferguson JM, Jenner SP, Amos TG, Parameswaran S, Gamaarachchi H & Deveson IW (2023) Flexible and efficient handling of nanopore sequencing signal data with slow5tools. *Genome Biol* 24: 69

Samarakoon H, Liyanage K, Ferguson JM, Parameswaran S, Gamaarachchi H & Deveson IW (2024) Interactive visualization of nanopore sequencing signal data with *Squigualiser*. *Bioinformatics* 40: btae501

Scalfani VF (2021) Using NCBI Entrez Direct (EDirect) for Small Molecule Chemical Information Searching in a Unix Terminal. *J Chem Educ* 98: 3904–3914

Schwartz S, Bernstein DA, Mumbach MR, Jovanovic M, Herbst RH, León-Ricardo BX, Engreitz JM, Guttman M, Satija R, Lander ES, *et al* (2014) Transcriptome-wide Mapping Reveals Widespread Dynamic-Regulated Pseudouridylation of ncRNA and mRNA. *Cell* 159: 148–162

Thorvaldsdottir H, Robinson JT & Mesirov JP (2013) Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration. *Brief Bioinform* 14: 178–192

Van Dijk EL, Jaszczyszyn Y, Naquin D & Thermes C (2018) The Third Revolution in Sequencing Technology. *Trends Genet* 34: 666–681

Wang Y, Zhao Y, Bollas A, Wang Y & Au KF (2021) Nanopore sequencing technology, bioinformatics and applications. *Nat Biotechnol* 39: 1348–1365

Wick RR, Judd LM & Holt KE (2019) Performance of neural network basecalling tools for Oxford Nanopore sequencing. *Genome Biol* 20: 129

Yu B, Nagae G, Midorikawa Y, Tatsuno K, Dasgupta B, Aburatani H & Ueda H (2024) m6ATM: a deep learning framework for demystifying the m6A epitranscriptome with Nanopore long-read RNA-seq data. *Brief Bioinform* 25: bbae529