UNIVERSITÉ DE RENNES 1

THÈSE DE DOCTORAT DE

L'UNIVERSITÉ DE RENNES 1
COMUE UNIVERSITÉ BRETAGNE LOIRE

ÉCOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Informatique*

Par

Lolita LECOMPTE

Structural variant genotyping with long read data

Thèse présentée et soutenue à Rennes, le 4 décembre 2020
Unité de recherche : Institut de Recherche en Informatique et Systèmes Aléatoires (IRISA), UMR 6074
Thèse N° :

Rapportrices avant soutenance :

Annie Chateau	Maîtresse de conférences, LIRMM, Université de Montpellier
Hélène Touzet	Directrice de recherche, CNRS, CRIStAL Lille

Composition du Jury :

Président :	Sébastien Ferré	Professeur des Universités, IRISA, Université de Rennes 1
Examinateurs :	Sébastien Ferré	Professeur des Universités, IRISA, Université de Rennes 1
	Étienne Danchin	Directeur de recherche, INRAE, Institut Sophia Agrobiotech Sophia Antipolis
Rapportrices :	Annie Chateau	Maîtresse de conférences, LIRMM, Université de Montpellier
	Hélène Touzet	Directrice de recherche, CNRS, CRIStAL Lille
Directeur de thèse :	Dominique Lavenier	Directeur de recherche, CNRS, IRISA Rennes
Co-encadrante de thèse :	Claire Lemaitre	Chargée de recherche, Inria Rennes

ACKNOWLEDGEMENT

Tout d'abord, je remercie les membres du jury d'avoir acceptés de participer à mon jury de thèse, ainsi que de l'intérêt qu'ils portent à mon sujet de thèse. Je suis très honorée de pouvoir vous compter dans mon jury de thèse. Merci pour leur retour, tout particulièrement aux rapportrices, Annie Chateau et Hélène Touzet, pour leur rapport détaillé sur mon manuscrit de thèse.

Je souhaite aussi remercier mes encadrants de thèse, Claire Lemaitre et Dominique Lavenier qui m'ont fait confiance pour ce sujet de thèse et qui m'ont accompagné pendant ces trois ans. Je tiens à les remercier chaleureusement d'avoir pu m'aider dans les moments d'inquiétudes et de doutes. Un grand merci à Claire grâce à qui j'ai développé un intérêt scientifique pour les variants de structure. Je te suis très reconnaissante pour ta motivation, ta prise de recul ou encore ta réactivité. Merci aussi Claire de t'être rendue souvent disponible pour répondre à mes questions ou pour les nombreuses relectures du manuscrit.

Merci à Inria, l'école doctorale MathSTIC et l'Université de Rennes 1, de m'avoir donnée l'opportunité de réaliser ce travail de thèse.

Je remercie les membres de mon comité de suivi de thèse, Rayan Chikhi et Nathalie Nesi qui ont suivi l'évolution de ma thèse en fin de première année et de deuxième année. Merci d'avoir apporté un regard nouveau qui a permis d'éclaircir et de redigier les objectifs de la thèse. Ces retours ont été essentiels, merci à eux pour leur disponibilité.

Je tiens à remercier Pierre Peterlongo qui m'a d'abord accueilli en stage de Master 2 avec Camille Marchet, puis qui m'a suivi au cours de la thèse. Merci pour les conseils avisés à certains moments de ma thèse et d'avoir apporté des questions "naives" mais très pertinentes qui soulèvent les problématiques majeures.

Un grand merci à Marie Le Roic pour tout ! Les questions logistiques, le soutien, le sourire et la bonne humeur au quotidien.

Merci à tous les collègues de l'équipe Genscale, les anciens, les actuels comme les nouveaux ! J'ai énormément appris au sein de l'équipe, je suis heureuse d'avoir pu travailler à vos côtés et d'avoir pu échanger avec vous. Merci pour vos retours et vos precieux conseils sur mon travail.

Je remercie la plateforme Genouest pour son soutien essentiel, tout particulièrement

Anthony pour son aide si précieuse ! Mais aussi Matéo, Joseph, Chloé et les autres pour vos conseils et vos recommandations.

Plus généralement, merci aux collègues de l'équipe Symbiose, cette expérience aura été tellement enrichissante et divertissante grâce à vous ! Merci à Camille qui m'a guidé dans le monde de la recherche à mes débuts en stage de Master 2 à Genscale. Merci pour tout ce que tu m'as apportée sur le plan professionnel comme sur le plan personnel. Merci à Wesley et Hugo pour votre humour, votre écoute, vos conseils ou votre capacité à relativiser. Je regrette déjà de ne plus partager ces cafés du matin avec vous. Merci aussi à Méline pour ton énergie, ta motivation, ton écoute. Sans oublier tous les anciens stagiaires, les ingénieurs, les doctorants, les docteurs (vous êtes si nombreux) : Arnaud, Cervin, Clémence, Hugo, Jérémy, Lucas, Maël C, Maël K, Marie, Marine, Méziane ! Mais je tiens bien évidemment à remercier mes co-bureaux Anne, Denis, Emmanuelle, Susete, Vijay, avec qui j'ai eu le plaisir de partager des discussions instructives et enrichissantes.

Merci aux batuqueiros ! Mon seul regret est de ne pas avoir eu l'occasion de participer à une représentation à la fête de la musique de l'Inria avec vous.

À mes chers colaborateurs, Pierre Morisse, Camille Marchet et Antoine Limasset, que je remercie je suis tellement contente d'avoir participer au développement de notre petit pokémon préféré : ELECTOR. Ça a été un plaisir pour moi d'échanger et de travailler à vos côtés. Je tiens à vous remercier davantage pour votre invitation, votre aide ou encore vos conseils. J'espère qu'on sera amené à collaborer une nouvelle fois ensemble et surtout à se revoir (en vrai) en conférence le plus tôt possible !

Enfin, je tiens à remercier mon compagnon, Brewen, pour m'avoir soutenu jusqu'au bout de ce parcours initiatique ! Merci d'avoir été un soutien inconditionel dans les moments parfois longs et intenses que peuvent impliquer une thèse.

Table of Contents

Résumé de la thèse - French summary of the thesis **11**

1 Introduction **21**

1.1 Preamble . . . 22

1.2 Genomics . . . 22

1.2.1 What is a genome? . . . 22

1.2.2 Why studying genomics? . . . 23

1.2.3 Genomic polymorphism . . . 24

1.2.3.1 Structural variants . . . 25

1.2.3.2 Molecular mechanisms causing structural variants . . . 27

1.2.3.3 Impacts of structural variants . . . 28

1.3 Sequencing technologies . . . 29

1.3.1 First generation sequencing technologies . . . 30

1.3.2 Second generation sequencing technologies . . . 31

1.3.2.1 Illumina sequencing technology . . . 32

1.3.2.2 Towards longer reads . . . 32

1.3.3 Third generation sequencing technologies . . . 33

1.3.3.1 Pacific Biosciences sequencing technology . . . 34

1.3.3.2 Oxford Nanopore Technology . . . 36

1.3.4 Advances due to sequencing technologies . . . 38

1.4 Methodological approaches for genomics . . . 40

1.4.1 Bioinformatics for sequencing data . . . 40

1.4.2 Methodologies for structural variants . . . 42

2 State of the art **45**

2.1 Introduction . . . 47

2.2 Sequence alignment . . . 47

2.2.1 Short read alignment . . . 49

2.2.2 Long read alignment . . . 50

2.2.2.1 `NGMLR` . . . 51

2.2.2.2 `Minimap2` . . . 52

2.2.2.3 Long read mapping tools performances . . . 52

2.3 Structural variant discovery . . . 53

2.3.1 Structural variant discovery with short reads . . . 54

2.3.1.1 SV signature detection strategies . . . 54

2.3.1.2 A plethora of structural variant callers . . . 59

2.3.1.3 Limitations . . . 59

2.3.2 Structural variant discovery with long reads . . . 60

2.3.2.1 Three methods for SV discovery with long reads . . . 61

2.3.2.2 SV discovery methods based on other approaches . . . 67

2.3.2.3 SV callers specific to SV types . . . 69

2.3.2.4 Hybrid approaches . . . 71

2.3.2.5 Conclusion on SV discovery using long reads . . . 71

2.3.3 Conclusion on structural variant discovery . . . 72

2.4 Genotyping structural variants . . . 74

2.4.1 SV genotyping using short reads . . . 75

2.4.1.1 SV genotyping methods mapping-based against reference allele 75

2.4.1.2 SV genotyping methods representing allele variants . . . 77

2.4.1.3 Conclusion on SV genotyping methods for short reads . . . 80

2.4.2 SV genotyping with long reads . . . 81

2.4.3 Conclusion on SV genotyping . . . 82

2.5 Where the thesis stands in relation to the state of the art . . . 84

3 A SV genotyping approach using long reads **85**

3.1 Introduction . . . 87

3.2 Representative allele sequence generation . . . 88

3.3 Mapping . . . 93

3.4 Informative alignment selection . . . 94

3.4.1 Breakpoint overlap . . . 94

3.4.2 Filtering out false positives due to repeated sequences . . . 95

3.5 Genotype estimation . . . 96

3.5.1 Allele number normalization for unbalanced variants . . . 96

3.5.2 Minimum allele number threshold for genotyping . . . 97

3.5.3 Genotyping decision . . . 97

3.5.3.1 Method based on decision thresholds . . . 97

3.5.3.2 Maximum likelihood method . . . 98

3.6 Parameter estimation . . . 99

3.6.1 Material . . . 99

3.6.1.1 Simulated dataset . . . 99

3.6.1.2 Evaluation . . . 99

3.6.2 Results according to the genotyping estimation model . . . 100

3.6.2.1 Genotype estimation methods . . . 100

3.6.3 Estimation of the size of the representative allele sequences . . . 103

3.7 Implementation and availability . . . 104

3.7.1 Requirements for SV definition in input VCF file . . . 105

3.8 Conclusion . . . 109

4 Method validation **111**

4.1 Introduction . . . 113

4.2 Evaluation on simulated long read datasets . . . 113

4.2.1 Dataset presentation . . . 113

4.2.2 Genotyping results . . . 115

4.2.2.1 `SVJedi` on simulated deletions . . . 115
4.2.2.2 `SVJedi` results on simulated inversions and on simulated translocations . . . 116
4.2.3 Evaluation of the robustness of the method . . . 116
4.2.3.1 Sequencing depth . . . 117
4.2.3.2 Sequencing error rate . . . 118
4.2.3.3 Precision of the position of breakpoint position . . . 118
4.2.3.4 Evaluation with close and overlapping deletions . . . 120
4.3 Application to real datasets . . . 121
4.3.1 Material . . . 122
4.3.2 PacBio dataset . . . 123
4.3.2.1 `SVJedi` results on PacBio dataset . . . 123
4.3.2.2 Mendelian Inheritance of the Ashkenazi trio . . . 125
4.3.3 ONT dataset . . . 125
4.4 Comparison with other approaches . . . 125
4.4.1 Comparison with other genotyping tools for long reads . . . 127
4.4.1.1 Detailed genotyping results . . . 128
4.4.2 Comparison with a short read based genotyping approach . . . 129
4.4.3 Comparison with SV discovery approaches . . . 129
4.4.4 Stratified analysis . . . 130
4.4.4.1 Performance by SV size categories . . . 130
4.4.4.2 Performance by genomic context and SV type . . . 132
4.4.5 Runtime comparison . . . 133
4.5 Conclusion . . . 133

5 Conclusion **135**
5.1 Conclusion . . . 135
5.2 Perspectives . . . 136
5.2.1 Short-term perspectives . . . 136

5.2.2 Long-term perspectives 137

Appendix **139**

6.1 Appendix to Chapter 4 139

6.1.1 Data accessibility and generation 139

6.1.2 SVJedi apply to real datasets 140

6.1.2.1 SVJedi results on real PacBio dataset 140

6.1.2.2 Mendelian Inheritance of the Ashkenazi trio 141

6.1.2.3 SVJedi results on a 30x ONT dataset 141

6.1.3 Stratified analysis 142

6.1.3.1 Performance by SV size categories 142

Bibliography **143**

Scientific contributions **153**

List of Figures **156**

List of Tables **158**

Abbreviation **160**

RÉSUMÉ DE LA THÈSE

0.1 Introduction

Le génome est l'ensemble du matériel génétique présent dans chaque cellule des organismes vivants, il contient les instructions nécessaires au fonctionnement de la vie des cellules et de l'organisme. Le génome est constitué de la molécule d'acide désoxyribonucléique (ADN), composée de plusieurs nucléotides (A, T, C, G) en séquence. Les nucléotides A et T peuvent s'apparier, de même que les nucléotides C et G. La séquence du génome est sous forme stable lorsqu'elle est appariée à sa séquence complémentaire, le génome est alors composé de deux brins d'ADN complémentaires. Le génome possède une structure spécifique, c'est-à-dire une organisation précise des éléments qui le constituent, on retrouve les gènes parmi ces éléments. Un gène est une séquence d'ADN à un locus donné (une position génomique précise), cette séquence d'ADN peut coder pour une protéine par exemple qui aura une fonction spécifique.

La séquence du génome est unique pour chaque individu. Au sein de la même espèce, on observe des variations entre les séquences des génomes des individus, on parle de polymorphisme génomique. Il existe plusieurs types de variation. On peut d'abord distinguer les *Single Nucleotide Polymorphisms* (SNPs), qui sont des variations d'une base à un locus donné, c'est-à-dire à une position génomique définie. Puis, on définit d'*indels* comme des insertions ou des délétions qui peuvent faire jusqu'à 50 bp. Enfin, les *variants de structure* (SVs) sont définis comme des réarrangements génomiques d'au moins 50 bp. On distingue plusieurs types de réarrangements de SVs tels que les délétions, les insertions, les inversions, les translocations. Certains SVs combinent même plusieurs évènements de réarrangements à un même locus, on les qualifie de variants complexes. Les SVs sont des variations très hétérogènes du fait des différents types de réarrangements et de leur taille qui peuvent faire plusieurs milliers de bases. D'ailleurs, les SVs représentent environ 1.5 % des variations entre deux génomes humains et seulement que 0.1 % des SNPs (Pang et al., 2010). L'impact des SVs peut alors avoir des conséquences majeures sur le fonctionnement cellulaire. Au cours de cette thèse, nous nous sommes intéressés à l'étude des variants de structure.

0.1.1 Technologies de séquençage

Afin d'étudier les variations des séquences, ou plus largement, les séquences des génomes, on utilise des technologies de séquençage qui permettent justement d'obtenir la suite des bases de fragments de molécule d'ADN. Les fragments séquencés sont appelées des *lectures* de séquençage. Trois générations de technologies de séquençage ont été proposées au cours des quarantes dernières années et produisent ainsi des lectures avec des caractéristiques spécifiques selon les stratégies sur lesquelles elles s'appuient.

Parmi la première génération de technologie de séquençage, on retrouve la méthode de séquençage Sanger qui produit des données de séquençage de lectures de taille d'environ 1000 paires de bases (bp) avec une précision élevée de 99.999 %. La méthode de Sanger nécessite cependant une main d'oeuvre importante.

Des technologies de séquençage de seconde génération ont ensuite été developées dans les années 2000. Parmi les technologies de seconde génération, il y a la technologie Illumina qui est fréquemment utilisée à l'heure actuelle et produit des lectures de séquences d'une centaine de paires de bases (150 bp) avec un taux d'erreur supérieur à 0.1 %. Les données de séquençage obtenus par les technologies de seconde génération sont appelées les *lectures courtes*.

Ces dernières années, une troisième génération de technologies de séquençage a été deployée. Elles se caractérisent par le séquençage d'uniques longues molécules d'ADN de quelques milliers à une dizaine de bp, appelées les *lectures longues*. Deux technologies de séquençage ont été développées: la technologie de séquençage Pacific Biosciences (PacBio) et la technologie de séquençage Oxford Nanopore technologies (ONT).

— Au cours du séquençage PacBio, une longue molécule d'ADN simple brin est liée au fond d'un puit par l'enzyme d'ADN polymérase capable de synthétiser l'ADN. La molécule simple brin d'ADN est synthétisée avec des nucléotides marqués par fluorescence. Pour chaque nucléotide incorporé le fluorophore est excité au fond de chaque puit, le signal d'émission du fluorophore est enregistré pour en déduire le nucléotide incorporé.

— L'approche de séquençage ONT repose aussi sur le séquençage d'une seul molécule d'ADN mais contrairement à la technologie de PacBio, il ne s'agit pas d'un séquençage par synthèse. Chaque molécule d'ADN est apportée à la surface d'une membrane recouverte de nanopores. Chaque molécule d'ADN simple brin passe au travers d'un nanopore à l'aide de protéines. Le passage de la suite des nucléotides dans un nanopore déclenche une modification du courant électrique appliqué à la membrane. À l'aide de méthodes basées sur l'apprentissage profond, on en déduit la suite de nucléotides,

cette étape s'appelle le *basecalling*.
L'inconvénient principal des approches de séquençage de troisième génération est le fort taux d'erreur des lectures de séquençage, ce taux s'élève autour de 13 % d'erreur.

Les données longues lectures semblent être particulièrement intéressantes pour étudier les SVs. En effet, l'information longue distance contenue dans ces lectures de séquençage permet de couvrir des grands réarrangements de SVs. Les longues lectures offre un avantage pour l'analyse des SVs par rapport aux courtes lectures dont la taille est souvent inférieure à celle des SVs.

0.1.2 Problématiques autour des variants de structure

On distingue deux questions principales quand on étudie les SVs: la détection de SV et le génotypage de SV. Premièrement, on va chercher à découvrir les SVs présents chez un nouvel individu séquencé par rapport à un génome de référence, c'est l'étape de détection des SVs. Deuxièmement, on s'intéresse à identifier les variants présents et absents chez un individu séquencé pour un ensemble de SVs connus, c'est l'étape de génotypage des SVs.

Dans la prochaine section, on dresse l'état de l'art des méthodes développées pour détecter des SVs et génotyper des SVs avec des données de séquençage.

0.2 État de l'art

Les lectures de séquençage, limitées par la taille des fragments de lectures, ne contiennent que des informations partielles du génome. Pour retrouver la localisation des lectures de séquençage sur le génome, on utilise des méthodes d'alignement des séquences qui retrouvent les similarités entre la séquence du génome de référence et la séquence de lecture tout en illustrant les différences entre ces deux séquences.

0.2.1 Detection de variants de structure

La présence d'un SV chez un individu séquencé, va générer des alignements tronqués pour certaines lectures de séquençage. Les méthodes de détection de SV basées sur l'alignement identifient justement ces alignements tronqués qui suggèrent la présence de réarrangements. Puis les méthodes procèdent à une étape de filtre pour séparer les potentiels SVs des signaux de faux positifs. Enfin, les méthodes assignent à chaque SV détecté un score représentatif de

la confiance du signal de détection du SV.

Un nombre important de méthodes de détection ont été developpées pour les données de courtes lectures. Le plus souvent ces méthodes de détection sont basées sur l'alignement de courtes lectures contre le génome de réference. La présence d'un SV dans le génome séquencé par rapport au génome de référence entraine un tronquage des alignements des données de séquençage. On différencie quatres stratégies basées sur l'analyse des alignements pour découvrir des SVs: la profondeur de séquençage des lectures, l'analyse des lectures paired-end, les alignements de lectures coupées et l'assemblage local. Le plus souvent les méthodes de détection de SVs avec les lectures courtes combinent les stratégies de détection. Mais aujourd'hui aucune méthode n'a montrée qu'elle pouvait combiner une précision et un sensibilité élevées. Le taux de faux positifs peut aller jusqu'à 89 % (Mahmoud et al., 2019).

Avec l'émergence des technologies de séquençage des longues lectures, un peu plus d'une dizaine de méthodes ont été developpées pour détecter des SVs avec des données de longues lectures. En effet, les longues lectures permettent de couvrir plusieurs milliers de bases, cette information longue distance s'avère très utile pour retrouver les réarrangements des SVs qui peuvent faire plusieurs centaines à plusieurs milliers de bp. Parmi ces méthodes, `Sniffles` (Sedlazeck et al., 2018b) obtient à l'heure actuelle les meilleurs résultats d'après plusieurs études (De Coster et al., 2019; Zhou et al., 2019). Le principe de la méthode de Sniffles s'appuie sur la couverture du génome, les alignements de reads coupés et les régions d'alignements très erronnées pour détecter des SVs. Une des particularités de l'approche `Sniffles` est qu'elle s'appuie sur des méthodes d'alignement de longues lectures qui prennent en compte les lectures qui chevauchent des SVs. Parmi ces méthodes d'alignement, on trouve `NGMLR` (Sedlazeck et al., 2018b) et `Minimap2` (Li, 2018).

0.2.2 Génotypage de variants de structure

Un allèle est défini comme une version possible d'un variant, pour nous ici d'un SV. On considère l'allèle de référence (allèle 0) comme celui présent dans le génome de référence et l'allèle alternatif (allèle 1) comme le variant de structure considéré.

Le génotypage de SVs consiste à évaluer la présence et l'absence des variants pour un ensemble de variants donné. De plus selon les espèces, le génome est présent en plusieurs copies, à titre d'exemple le génome humain est diploïde, c'est-à-dire que le génome est présent en deux copies. Le génotypage repose aussi sur l'estimation du nombre de copies des allèles. Par exemple, pour un génome diploïde et pour un locus donné avec deux allèles possibles, le génotype est

homozygote si le même allèle est présent sur les deux copies du génome ou bien le génotype est hétérozygote si chaque allèle est présent sur une seule copie du génome. On distingue alors trois génotypes: homozygote pour l'allèle de référence (0/0), hétérozygote (0/1) et homozygote pour l'allèle alternatif (1/1).

La question du génotypage repose sur un ensemble de SVs déjà détectés et caractérisés pour l'espèce étudiée. Or, avant l'arrivée des longues lectures, les ensembles de SVs détectés par les méthodes de courtes lectures étaient peu fiables. Les longues lectures fournissent des ensembles de SVs plus fiables, sur lesquels on peut s'appuyer par exemple pour génotyper des SVs dans le cadre du reséquençage d'individus.

Les méthodes de génotypage de SVs utilisent aujoud'hui des données de séquençage de courtes lectures. Les premières méthodes de génotypage de courtes lectures ont été developpées dans le prolongement des méthodes de détection de SVs, ayant pour effet indésirable d'être incompatibles avec d'autres ensembles de SV que ceux obtenus par l'outil de détection (Chander et al., 2019). De plus, ces méthodes basées sur des approches d'alignement s'appuient seulement sur une représentation des allèles de réference. Cependant, ces deux dernières années, on assiste au développement de plusieurs méthodes de génotypage de SVs pour des données de courtes lectures basées sur la représentation des deux allèles, soit par une approche de k-mers (mots de taille k), soit par une approche de graphe de variations.

Toutefois, malgré l'intérêt des longues lectures déjà démontré pour l'analyse des SVs, aucune méthode n'a été décrite à l'heure actuelle dans la littérature pour le génotypage de SVs avec des données de longues lectures. Nous avons néanmoins identifié deux outils capables de réaliser cette tâche: `Sniffles -Ivcf` qui est une option de l'outil de détection de SV (Sedlazeck et al., 2018b) et `svviz2` qui est initialement un outil de visualisation (Spies et al., 2015). Après avoir identifier un manque dans l'état de l'art de méthodes dédiées au génotypage de SVs avec des données de lectures longues, nous nous sommes concentrés sur cette question.

0.3 Méthode pour le génotypage de variants de structure

Nous proposons une méthode pour génotyper des SVs avec des lectures longues qui se déclinent en quatres étapes principales (illustrées dans la Figure 1). Nous avons considéré important de représenter à la fois l'allèle de réference et l'allèle alternatif. La première étape de notre méthode consiste pour chaque SV à générer des séquences représentatives pour les deux allèles. Au

cours de la deuxième étape, les données de séquençage des lectures longues sont alignées contre ces séquences alléliques avec le mapper de longues lectures `Minimap2` (Li, 2018). La troisième étape filtre les alignements non informatifs au regard de l'estimation du génotype du variant. Plusieurs conditions sont nécessaires pour considérer une lecture comme soutenant un des allèle.

— Premièrement, pour qu'une lecture soit informative sur le génome, l'alignement de la lecture contre la séquence de l'allèle doit chevaucher d'au moins d_{over} bp de part et d'autre du point de cassure (par défaut $d_{over} = 100$ bp).

— Deuxièmement, pour éviter les alignements dûs aux séquences répétées, on sélectionne uniquement les alignements semi-globaux, c'est-à-dire les alignements qui s'étendent de chaque côté (à gauche et à droite) jusqu'à l'extrémité de la séquence de l'allèle ou jusqu'à l'extrémité de la séquence de la lecture. Afin d'apporter un peu de flexibilité à cette deuxième condition on introduit le paramètre d_{end} pour également considérer les alignements qui vont jusqu'à l'extrémité moins d_{end} bp (par défaut $d_{end} = 100$ bp).

Les deux conditions doivent être validées pour pouvoir considérer une lecture comme informative vis à vis du génotype. La quatrième et dernière étape de notre méthode estime le génotype à partir des comptages des lectures informatives soutenant chaque allèle. Le génotype est estimé avec une méthode de maximisation de la vraissemblance des génotypes (Li, 2011).

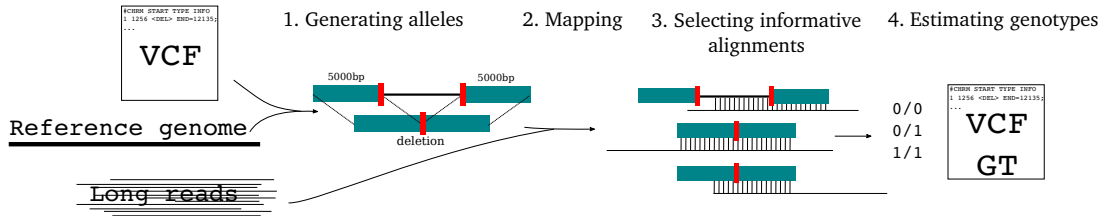

Figure 1 – Aperçu des 4 étapes de `SVJedi` pour le génotypage de délétions. 1. Pour chaque SV, deux séquences représentatives des allèles sont générées. 2. Les données de longues lectures sont alignées contre ces séquences alléliques avec `Minimap2`. 3. Les alignements informatifs sont sélectionnés. 4. Le génotype est estimé pour chaque allèle.

Notre méthode est capable de génotyper des délétions, des insertions, des inversions et des translocations (caractérisées comme des points des cassures dans les ensembles de SVs). La méthode a été décrite et acceptée dans l'article Lecompte et al. (2020).

Nous proposons une implémentation de notre méthode dans un outil nommé `SVJedi` codé en Python3 et disponible sur github et bioconda.

0.4 Validation de la méthode

Afin de valider notre méthode, nous l'avons appliqué `SVJedi` sur des données simulées et des données réelles. Nous avons également évalué notre outil par rapport aux autres outils de génotypage de SV avec des longues lectures et des courtes lectures. Ces résultats ont été présentés dans l'article Lecompte et al. (2020).

0.4.1 Données simulées

Nous avons testé `SVJedi` pour un ensemble de 1000 délétions du chromosome 1 du génome humain, échantillonnées dans la base de données de SVs dbVAR. Ces délétions sont des variations réelles qui ont été identifiées chez l'humain. Les génotypes de ces 1000 délétions ont été simulées aléatoirement de telle sorte à obtenir 333 délétions 0/0, 334 délétions 0/1 et 333 délétions 1/1. Nous avons ensuite simulé des données de séquençage longues lectures PacBio avec l'outil `SimLoRD` (Stöcker et al., 2016) à partir de deux séquences représentant les deux haplotypes avec les 1000 délétions génotypées. Les données de séquençage ont un taux d'erreur à 16 % et une profondeur de séquençage de 30x.

Les résultats de `SVJedi` sur ce jeu de données atteignent une précision de génotypage de 97.8 % (le nombre de variants génotypés correctement par rapport au nombre total de variants génotypés) et un taux de génotypage de 99.6 % (le nombre de variants de génotypés par rapport au nombre total de variants à génotyper). L'expérience a été répétées sur 9 autres ensembles de SVs échantillonés de dbVar et présentent des résultats très similaires.

Deux autres ensembles de données simulées ont été appliqués à `SVJedi` à partir de 450 inversions simulées et 450 translocations (point de cassure de translocation) simulées. `SVJedi` obtient une précision de génotypage de 97.8 % et 98.2 % pour les inversions et translocations respectivement, avec un taux de génotyapge à 100 % pour les deux types de SVs.

Enfin, nous avons réalisé une évaluation plus appronfondie afin d'évaluer la robustesse de notre outil, puis nous avons testé séparément le taux d'erreur des données de séquençage des longues lectures, la profondeur de séquençage du jeu de données, la précision des points de cassures des SVs à génotyper et le chevauchement des SVs à génotyper. Notre méthode est globalement assez robuste, cependant les performances de notre méthode semblent être impactées par le chevauchement des SVs à génotyper.

0.4.2 Données réelles

SVJedi a été testé sur des jeux de données réelles. Récemment, le Consortium Genome in a Bottle (GiaB) a réalisé une étude et constitué pour l'individu HG002 un ensemble de SV génotypé à partir de plusieurs types de données de séquençage et plusieurs méthodes de détection (Zook et al., 2020). Le résultat de cette étude est un travail essentiel qui fournit un ensemble de SV très fiables pour l'individu HG002 grâce à l'important effort de combinaison et de vérification des SVs. Cet ensemble de SV de GiaB peut alors être considéré comme un ensemble de référence. Il est composé de 12,745 SVs répartis en 5,464 délétions et 7,281 insertions.

Nous avons ainsi appliqué SVJedi à l'ensemble GiaB pour l'individu HG002. Nous avons utilisé le génome de réference GRCh37.p13 et un jeu de données PacBio pour l'individu HG002 de GiaB également, le jeu de données a été sous-échantillonné à 30x de profondeur de séquençage. `SVJedi` obtient une précision de génotypage de 92.2 % et un taux de génotypage de 90.2 %. Nous avons identifié la majorité des erreurs comme étant des SVs de petite taille et/ou localisé dans des régions de répétitions en tandem. Des résultats similaires ont été obtenus avec un jeu de données réel ONT.

Nous avons réalisé une analyse de l'hérédité Mendélienne, en évaluant la cohérence des génotypes estimés pour les 12,745 SVs entre les parents de HG002 et HG002. `SVJedi` atteint une hérédité Mendélienne de 96.9 %.

0.4.3 Comparaison à d'autres méthodes de génotypages

Nous avons ensuite comparé notre outil à autres outils de génotypage de SV pour l'ensemble de confiance élevé de GiaB pour l'individu HG002, avec le même jeu de données de lectures PacBio sous-échantillonné à 30x.

Dans un premier temps, nous avons comparé notre outil à `Sniffles -Ivcf` et `svviz2`. À l'heure actuelle, il s'agit des seuls outils de l'état de l'art qui peuvent estimer le génotype de SVs avec des données de séquençage longues lecture pour un ensemble de SV donné, même si ces deux outils n'ont pas été developpés dans ce but premier. Les performances de `SVJedi`, en précision et taux de génotypage, dépassent celles de `Sniffles -Ivcf` et de `svviz2` (Spies et al., 2015). Même si `Sniffles -Ivcf` et `svviz2` prédisent un génotype pour la plupart des SVs, la précision du génotypage de `Sniffles -Ivcf` est de 82.0 % et de `svviz2` est de 65.9 %. `SVJedi` est donc plus prudent dans ses prédictions, mais plus précis, car il atteint une précision de 92.2 %.

Nous avons aussi évalué `SVtyper` (Chiang et al., 2015), un outil de génotypage de SVs

pour des données de séquençage de courtes lectures. `SVTyper` atteint seulement une précision de génotypage de 46.5 % pour les 5,464 délétions de l'ensemble de GiaB avec un jeu de données Illumina de HG002 (30x). En effet, la plupart des méthodes de génotypage de SV de courtes lecture de séquençage ne permettent pas de génotyper les insertions, mais présentent également d'autres limitations. La comparaison des plateformes de séquençage indique l'intérêt des données de séquençage longues lectures pour le génotypage de SVs.

Enfin, nous avons évalué aussi deux outils de découverte de SVs avec des longues lectures qui disposent d'un module de génotypage: `Sniffles` (Sedlazeck et al., 2018b) et `pbsv`[1]. `Sniffles` atteint 48.8 % de précision dans ces prédictions et `pbsv` atteint 78.9 %. La comparaison des résultats des méthodes spécifiques au génotypage de SVs aux résultats des méthodes de détection de SVs montre l'intérêt de méthodes dédiées à la question du génotypage.

Nous avons aussi réalisé une analyse stratifiée où on évalue les performances des outils de génotypage précédents pour l'ensemble de GiaB en fonction de la taille des SVs et en fonction de leur contexte génomiques. L'analyse a révélée que `SVJedi` est moins conservent des performances constantes selon la taille des SVs, contrairement aux autres outils qui obtiennent une baisse de la précision du génotypage pour les longs SVs. Puis, l'analyse stratifiée en fonction du contexte génomique a aussi montré une baisse de la précision du génotypage pour tous les génotypeurs lorsque les SVs étaient situés dans des régions avec des répétitions en tandem ou des régions de duplications segmentaires.

0.5 Conclusion

Les SVs peuvent avoir d'importants impacts, ils peuvent notamment être responsables de certaines maladies ou encore être impliqués dans les évènements de spéciation. L'arrivée des données de séquençage de longues lecture offrent de nouvelles possibilités pour mieux comprendre les SVs. Nous avons constaté un manque de méthodes dédiées à la question du génotypage de SV pour ce type de donnnées de séquençage. Cette thèse propose une nouvelle méthode de génotypage de SV avec des données longues lectures. Contrairement à la plupart des méthodes de génotypage de SV pour les courtes lectures, nous avons choisi de représenter les deux allèles pour chaque variant. Cette stratégie permet d'éviter un biais de représentation de l'allèle de référence. Nous avons validé notre méthode à la fois sur des données humaines simulées et des données réelles. Notre outil, `SVJedi`, dépasse même les performances des autres

1. `https://github.com/PacificBiosciences/pbsv`

outils de génotypage de SVs sur des jeux de données réelles. Au cours de cette thèse, nous avons également démontré l'intérêt de méthode entièrement dédiées au génotypage de SVs, en nous comparant aux estimations des génotypes des méthodes de détection de SVs. Nous avons aussi démontré le bénéfice offert par les technologies de séquençage longues lectures par rapport aux courtes lectures pour la question du génotypage de SVs.

Chapter 1

Introduction

Table of contents

1.1 Preamble 22

1.2 Genomics 22

1.2.1 What is a genome? 22

1.2.2 Why studying genomics? 23

1.2.3 Genomic polymorphism 24

1.2.3.1 Structural variants 25

1.2.3.2 Molecular mechanisms causing structural variants 27

1.2.3.3 Impacts of structural variants 28

1.3 Sequencing technologies 29

1.3.1 First generation sequencing technologies 30

1.3.2 Second generation sequencing technologies 31

1.3.2.1 Illumina sequencing technology 32

1.3.2.2 Towards longer reads 32

1.3.3 Third generation sequencing technologies 33

1.3.3.1 Pacific Biosciences sequencing technology 34

1.3.3.2 Oxford Nanopore Technology 36

1.3.4 Advances due to sequencing technologies 38

1.4 Methodological approaches for genomics 40

1.4.1 Bioinformatics for sequencing data 40

1.4.2 Methodologies for structural variants 42

1.1 Preamble

Structural variants (SVs) are variations in genomes that vary widely in terms of type and size of variation, ranging from 50 base pairs to several million base pairs. SVs can, therefore, affect molecular and cellular processes and they are responsible for several human diseases, and also appear to be involved in species diversity and speciation (Ho et al., 2019; Mahmoud et al., 2019). One major objective is, therefore, to identify the entire set of structural variants in a given species and then, to genotype these variants to know the distribution of SVs within populations. Until now, sequencing data have had difficulties in detecting SVs. But recent sequencing technologies producing much longer read data seem to be an effective solution to study SVs. The arrival of this new type of sequencing data motivates the subject of this thesis that focuses on methodological bioinformatic methods for SV genotyping with long read data.

In this chapter, we present the notions aimed at defining these variations and exposed their complexity, while presenting the characteristics of the genomic sequencing data on which we rely to study structural variants and introducing the issues of SVs in bioinformatics.

1.2 Genomics

1.2.1 What is a genome?

The genome is the genetic material present in every living cell and unique to every living organism. The support material of the genome is the deoxyribonucleic acid (DNA) molecule, that is composed of four different types of nucleotides: adenine (A), cytosine (C), guanine (G) and thymine (T). Several nucleotides in a sequence form a polynucleotide chain or a single strand of DNA. The latter is directed with two distinct ends: the 5' end with a phosphate group and the 3' end with a ribose group. During DNA synthesis, nucleotides are added in the direction from the 5' end to the 3' end. A 3' end can only pair to a 5' end (Figure 1.1).

Nucleotides have the property of forming bonds between specific nucleotides. A and T are paired together, as are C and G. Thus, for example, the `ATCG` sequence matches the `TAGC` sequence. A single chain of polynucleotides, also called the *forward* strand, will then pair with a chain of complementary polynucleotides in the opposite direction, also called the *reverse* strand. The two paired strands of DNA form a double helix structure.

The genome can be represented as a text of nucleotide bases. The genome size varies

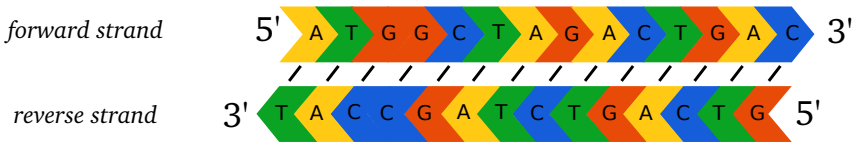

Figure 1.1 – **Double-stranded DNA molecule** composed of A, T, G, and C nucleotides (illustrated by different colors) from 5' end to 3' end with forward and reverse strands pairing.

across species, for instance among the most studied model organisms: *Escherichia coli* genome is 4.6 Megabases (Mb), *Caenorhabditis elegans* is 100 Mb, *Homo sapiens* is 3.1 Gigabases (Gb) and the *Triticum aestivum* (wheat) genome size is estimated approximately 17 Gb.

1.2.2 Why studying genomics?

The genome carries the genetic instructions for the biological functions or structures, development, and reproduction of living organisms. The genome is distributed into one or several chromosomes depending on the species. These chromosomes contain genes that encode for ribonucleic acid (RNA) molecules and eventually, RNAs are translated into proteins with specific biological functions at specific regions. Genes and their regulatory elements are therefore the genetic material of the biological functions necessary to sustain the life of a living organism.

The main interest when studying genomics is then to identify all the information contained in genomes in order to understand the biological functions of the living organisms. Knowing the genomic functions can be very useful in a medical context, with many applications in human health. The study of genomes is also particularly useful in ecology in order to know which organisms are present and how they interact with each other. It is useful in phylogeny as well, to know the evolutionary relationships between the genomes of different species sharing a common ancestor and to understand how the genomes of different species evolved.

During the life of an organism, the genome undergoes biological events (such as the replication of genetic material) or events related to physical or chemical mutagens that can lead to nucleotides changes in the genome. These mutations in the genome can potentially have a huge impact on cellular functions. Thus, identifying and characterizing variations in different individuals, species, cell types, etc. is particularly important when studying genomics.

In the following section 1.2.3, we discuss genomic variations in more detail.

1.2.3 Genomic polymorphism

The genomes of different individuals have differences between them, called variations. We speak of *polymorphism* when certain individuals of the same species show variations at a given *locus*, that corresponds to a fixed position in the genome. At a given locus, each possible version is called an *allele*.

The human genome is diploid, *i.e.* in each cell most chromosomes are present in two copies and homologous chromosomes refer to both copies of each chromosome. Therefore, each locus is present twice. The zygosity indicates whether the alleles are identical between homologous chromosomes at a particular locus. If the alleles are identical for a given locus, the genotype is homozygous, while the genotype is heterozygous if the alleles of the locus are different. The allele present on the reference genome is referred to as the *reference allele* (noted allele 0). The allele different from the reference allele is defined as the *alternative allele* (noted allele 1). For a diploid genome for example, there are three possible genotypes for a locus: either the genotype is homozygous for the reference allele (noted 0/0), or the genotype is heterozygous (noted 0/1), or the genotype is homozygous for the alternative allele (noted 1/1). In addition, if a locus has two possible alleles, as in the previous example, we speak of a biallelic variant. Or if the locus has more than three possible alleles, we speak of a multiallelic variant. A haplotype corresponds a set of allele of several loci located on a single chromosome.

The size of the variants can be very heterogeneous. Indeed, the variations can be limited to one base pair, in which case we speak of a *single nucleotide polymorphism* (SNP) or expand from a few tens to more than thousands of base pairs (bp) such as *structural variants* (SVs). At a given locus, a SNP implies the substitution of a single nucleotide. *Indels* refer to either deletions or insertions of less than 50 bp long, as opposed to SVs defined as genomic rearrangements of at least 50 bp long. SVs include multiple types of variations such as deletions and insertions but also other types of variations.

As can be assumed, depending on the size of the variations the impact on the functionality of the genome will vary. SVs affect more bp than the other types of variants, as a consequence, SVs can have important consequences on biological functions (Mills et al., 2011). For the human genome precisely, SVs represent 1.5 % of the variations of genome base pairs between two individuals, compared to 0.1 % for SNPs (Pang et al., 2010).

We focus on SVs in this thesis. In the next section, we will first attempt to define and characterize SVs, and then to understand the mechanisms responsible for these rearrangements. We will also see why it is important to describe SVs.

1.2.3.1 Structural variants

A structural variant (SV) corresponds to a genomic segment of at least 50 base pairs (bp) that has been rearranged regarding a reference genome. A rearrangement is delineated by its breakpoints (Quinlan and Hall, 2012). A breakpoint is the junction of genomic segments that are adjacent in the reference allele, but are not adjacent in the alternative allele. SVs are very heterogeneous variants (Alkan et al., 2011). They can be 50 bp long but also reach several megabases (Mb) long. Although the size of the SVs has great variability, in the human genome, most of them are less than 10 kilobases (kb) (Chaisson et al., 2019). But SVs also vary in the large number of possibilities of rearrangement. So there are several classes of SVs. First, we distinguish the *unbalanced* variants from the *balanced* ones, whether there is a change in the amount of DNA material between the reference allele and the alternative allele.

Rearrangements can be *unbalanced*, with either a loss or a gain of genetic material between the alleles for deletion and insertion, respectively. *Unbalanced* variants, also called copy number variants (CNVs), include deletions, insertions of novel sequences, and duplications. In case of a deletion, as shown in Figure 1.2, the reference allele contains the deletion sequence, while in the alternative allele, the deletion sequence has been removed. As opposed to an insertion, where the alternative allele contains a gain of copy number compared to the reference allele. Finally, a duplication can either be a tandem duplication, in this case, a genomic segment has been copied and inserted right after the initial segment copy. Or the duplication can be an interspersed duplication, where the duplicated segment is inserted throughout the genome, resulting in non-adjacent duplicated segments.

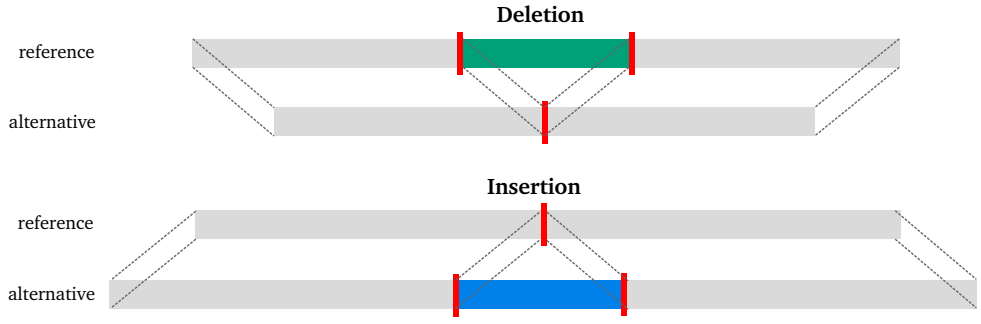

Figure 1.2 – Unbalanced SVs (deletions and insertions). Red thick marks represent breakpoint positions. The deletion sequence (green) is present only in the reference allele. The insertion sequence (blue) is present only in the alternative allele.

Genomic rearrangements can also be *balanced*. Balanced SVs imply that a genomic segment has been moved. It is the case of inversions, for which the genomic segment orientation has

been inverted between the breakpoint positions (see Figure 1.3). Reciprocal translocations are also *balanced* rearrangements. They are interchromosomal rearrangements involving two non-homologous chromosomes where the end of one chromosome has been swapped with the end of another chromosome as illustrated in Figure 1.4.

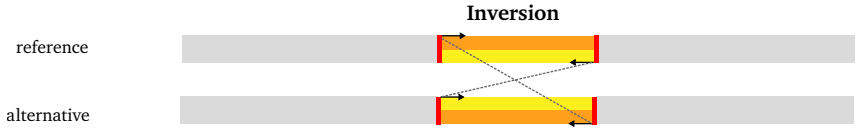

Figure 1.3 – Inversion. Red thick marks represent breakpoint positions, the orange and yellow segments correspond to the forward and reverse strand, respectively and the 5' ends are indicated by black arrows. The genomic segment between the breakpoint positions have been inverted in the alternative allele compared to the reference allele.

But SVs can be more complex by involving a multitude of combinations of the rearrangements seen previously in a single mutational event (Quinlan and Hall, 2012; Weckselblatt and Rudd, 2015). Complex SVs then encompass a wide variety of rearrangements with several breakpoints and loci, making them more difficult to identify. Two examples of complex SVs are represented in Figure 1.5 with a deletion-duplication (A), deletion-inversion-deletion rearrangement (B) and a duplication-inversion-duplication (C) (Quinlan and Hall, 2012). We notice that the same complex SV can be obtained through several intermediate rearrangement steps.

When complex rearrangements occur in an extreme way in the same region, we speak of *chromothripsis* when several chromosomes are involved and of *chromoanasynthesis* when only one chromosome is involved (Weckselblatt and Rudd, 2015). These variations involve massive rearrangements with a large number of breakpoints and have drastic impacts on cell function. They have been identified mainly in cancer cells and have been observed also in germ cells (Sanchis-Juan et al., 2018).

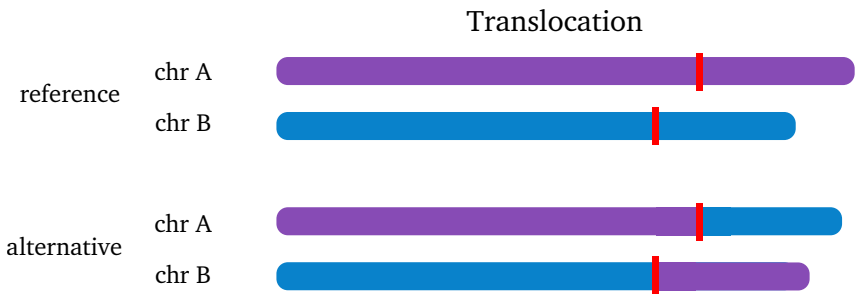

Figure 1.4 – Translocation. Chromosome A (purple) and chromosome B (blue) ends have been exchanged. Red thick marks represent breakpoint position.

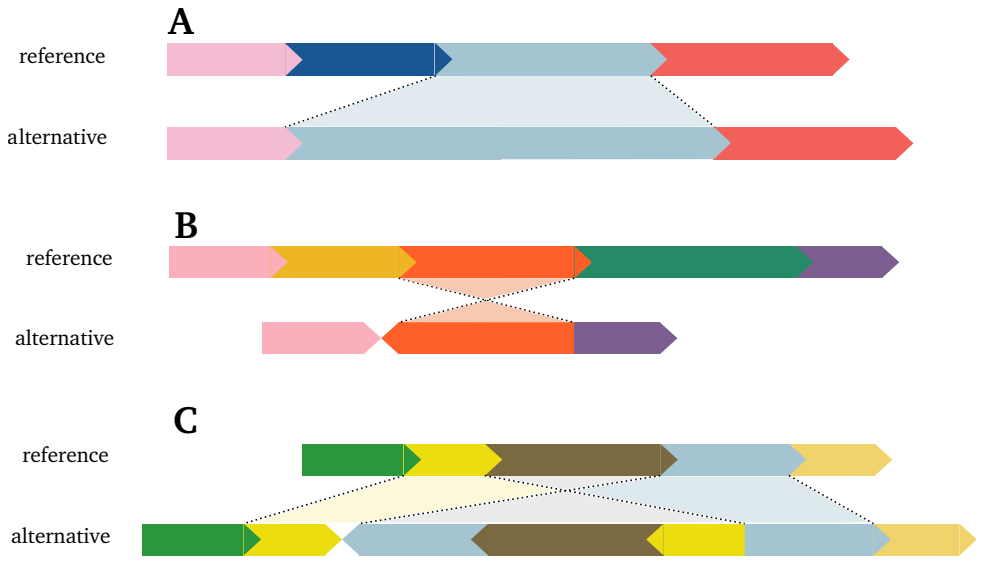

Figure 1.5 – Complex SV **A deletion-duplication**: dark blue segment has been deleted in the alternative allele and the light blue segment has been duplicated. **B deletion-inversion-deletion**: yellow and green segments have been deleted and orange segment has been inverted. **C duplication-inversion-duplication**: yellow and grey segments have been duplicated and then yellow-brown-grey block has been inverted.

There are other types of SVs as well, such as *tandem repeats*, which are adjacent repeated units that can either be expanted or contracted. Another type of SV include mobile elements when their sequences are greater than 50 bp. A mobile element is a DNA sequence that can be moved around in the genome. There are several types of mobile element with a structure specific to each type and repeated sequences that allow genomic rearrangements , such as duplications, mobile element insertions or mobile element deletions.

1.2.3.2 Molecular mechanisms causing structural variants

The molecular mechanisms responsible for genomic rearrangements occur during improper repair of double-strand DNA breaks. Double-strand breaks are not uncommon and occur randomly even though some regions are more susceptible to double-strand breaks (Currall et al., 2013). Double-strand breaks can even be biologically induced to generate antibody diversity through V(D)J recombination.

There are two main types of repair mechanisms for double-stranded breaks: the Non-Homologous End-Joining (NHEJ) and the Homologous Recombination (HR) repair. The HR repair mechanism is referred as a conservative mechanism because it relies on homologous

sequences (see Figure 1.6). It uses either the sister chromatid or the homologous chromosome as a template for synthesizing the broken double-stranded DNA. Unlike the HR repair mechanism, the NHEJ repair mechanism is not conservative. The NHEJ mechanism recruits several proteins to bind exposed DNA ends (Figure 1.6). This mechanism can result in small loss of genetic material (a few bp) and is therefore prone to errors.

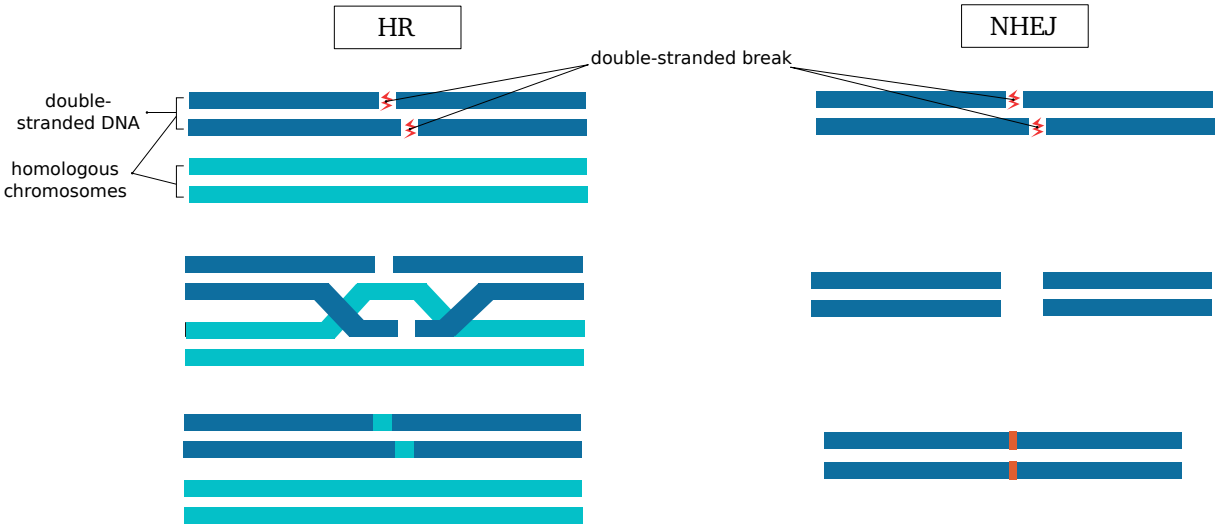

Figure 1.6 – Homologous Recombination (HR) and Non-Homologous End-Joining (NHEJ) repair mechanisms. In the HR mechanism, the double-strand break is repaired by homology with the homologous chromosomes as template. In the NHEJ mechanism, the exposed DNA ends of the double-strand break are ligated, resulting in sequence loss.

In most cases, the repair mechanisms are faithful in their reconstruction and do not alter the initial sequence of the genome at the breakpoints. However, it sometimes happens that these repair mechanisms generate gains, losses or rearrangements of genetic material. Some repair mechanisms are more prone to certain types of errors.

For instance, the Non-allelic Homologous Recombination (NAHR) mechanism is a homologous recombination from non-allelic sequences, but which share high similarity, resulting in inversions (see Figure 1.7), deletions, duplications or more complex SVs in the genome.

1.2.3.3 Impacts of structural variants

Studying SVs is particularly important since SVs can have major impacts on biological functions. Some SVs can be responsible of diseases (Mills et al., 2011), therefore, identifying and characterizing these variations is crucial for disease diagnosis or for finding a cure for the disease.

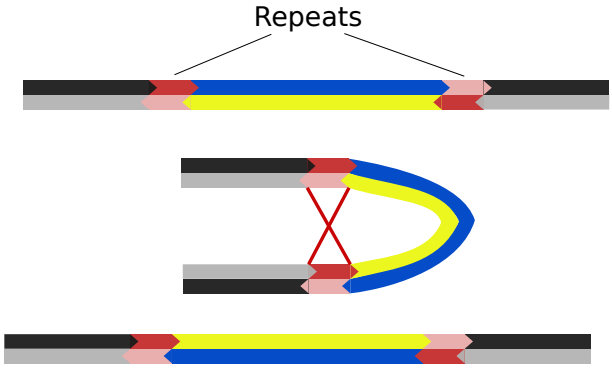

Figure 1.7 – Inversion resulting from Non-Allelic Homologous recombination (NAHR).

SVs have also have important ecological and evolutionary impacts (Alkan et al., 2011). This type of polymorphism has greatly contributed to the genetic diversity within populations and combined with a selection mechanism it can lead to the speciation of new species.

Because of their heterogeneity in size and type, SVs play an important role in genome diversity and can have significant impacts. The study of SVs is therefore essential to characterize this diversity and to understand the role they play in genetic diseases and ecosystems.

1.3 Sequencing technologies

In order to access genomic sequence information, sequencing technologies are used. DNA sequencing is the determination of the base pair sequence of a DNA fragment. The sequencing of several individuals is then an approach to study and characterize the genomic sequences within populations. Unfortunately, the sequencing technologies cannot obtain the genome sequence in a single piece. They require the DNA to be fragmented upstream. The size of the fragments varies according to the sequencing technologies.

Most often genomes are sequenced using the shotgun strategy approach which was proposed in 1979. The whole-genome shotgun approach consists first of fragmenting the genome randomly and several times (see Figure 1.8). The fragments are then sequenced. A *read* corresponds to a DNA fragment once it has been sequenced and, from the overlapping reads, the genome sequence is retrieved. Following the sequencing of a genome, we obtain a set of reads that corresponds to the entire sequenced DNA in multiple copies.

Thus, each position of the genome is represented by n reads, it corresponds to the sequencing depth at that position. The genome *sequencing depth* is defined as the average

number of reads representing each base of the genome sequence. For instance, in a sequencing sample with a depth of 10x, each position will be sequenced in average 10 times. Lastly, the genome *coverage* represents the percentage of the genome sequence spanned by at least one read.

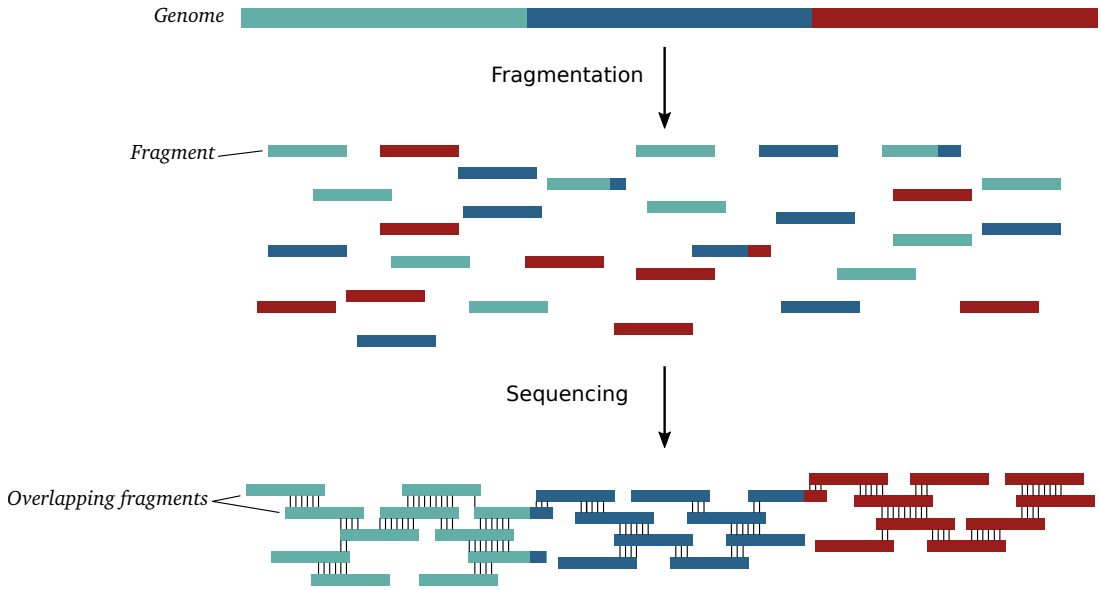

Figure 1.8 – **Whole-genome shotgun sequencing.** A first step of fragmentation of the genetic material results in random fragments. This step is performed several times in order to obtain overlapping fragments. The fragments are then sequenced. The initial sequence can be obtained by overlapping fragments.

1.3.1 First generation sequencing technologies

In 1977, the first generation of sequencing technology method was developed. It is referred to as the Sanger sequencing method and relies on the chain termination technique (Sanger et al., 1977). The Sanger sequencing method is based on the DNA synthesis reaction where deoxyribonucleotides triphosphates (dNTPs) are successively incorporated into the complementary DNA strand by the DNA polymerase enzyme. This sequencing relies on the use of modified dNTPs, the dideoxynucleotides (ddNTPs), which prevent the incorporation of the next nucleotide and thus stop the elongation process. The Sanger sequencing uses both dNTP and a small percentage of labeled ddNTP, either labeled by radioactivity or by fluorescence. During DNA polymerization, ddNTPs are then randomly incorporated into sequences and stops further elongation sequence. As a result, the fragments obtained are of all sizes because the sequences are stopped at each position of the genome. The nucleotide can be deduced at the end of each fragment according

to the label. Then by making the fragments of different sizes migrate by electrophoresis on a polyacrylamide gel, the bases of the sequence are retrieved (Heather and Chain, 2016).

This sequencing technique is therefore very expensive and labor-intensive, but it is also very accurate and robust. Sanger approach sequences reads of approximately 1 kb long and with 99.999 % accuracy (Shendure and Ji, 2008). Other sequencing technologies from the first generation have also been developed, based on sequence reconstruction from the successive sizes of fragments cut at specific positions as well.

Following this first generation of sequencing technology, two successive generations of approaches were developed and offered major contributions in the field of genomics. First, the second generation of sequencing technologies was developed and has completely revolutionized our understanding of genomes, especially by providing high throughput sequencing. Then, more recently, the third generation of sequencing technologies was developed, bringing innovations to complement the second generation. We develop the principles and interests of these generations in the following sections.

1.3.2 Second generation sequencing technologies

The second generation of sequencing technologies, or more commonly named Next Generation Sequencing (NGS) technologies, have been developed in the 2000s. NGS include several sequencing platforms such as 454 sequencing, Illumina technology, SOLiD platform, Ion Torrent. Data from NGS technologies, known as *short read* data, are characterized by a few hundred bp sizes and are produced at a high throughput scale (up to billions of reads per run) (Ardui et al., 2018). The throughput of NGS technology is significantly better than Sanger technology at a lower relative cost. However, the size of the NGS reads is smaller and at a slightly higher error rate than Sanger technology (>0.1 %). NGS sequencing errors are mainly substitution errors.

Despite the diversity of approaches, most technologies share similar steps. The first step is the preparation of the library, where the DNA is fragmented and then ligated at each end to adapter sequences. This is followed by a step where the fragments are amplified by a Polymerase Chain Reaction (PCR) to generate clonal amplification clusters so that a sufficient signal can be measured afterward. Next, there is a step of sequencing by synthesis occurring in parallel, in which a base is incorporated in each cycle. The sequence of bases is then deduced from the labeled nucleotides.

1.3.2.1 Illumina sequencing technology

Among NGS approaches the most widely used, even today, is the Illumina sequencing technology. We detail the principle of this sequencing technology which is essential today when we are interested in genome sequencing.

The principle of the Illumina sequencing technology consists first of all in fragmenting the DNA molecules and adding an adapter sequence tag at each end of the fragments. In addition, complementary sequences to the adapters are attached on 8 lanes of a flow cell, resulting in pairing of the complementary sequences. The Illumina sequencing technology then exploits a bridge PCR amplification technique to produce clonal amplification of all fragments on the flow cell in parallel.

Once the fragment clones have been produced on the surface of the support, the sequencing step is then carried out simultaneously in each cluster. Sequencing relies on a polymerization reaction of the fragment clones using labeled nucleotides. As opposed to Sanger, the Illumina technology relies on four fluorescent labeled nucleotides with reversible terminators, so that the termination of the nucleotide preventing the incorporation of the following base can be removed. Sequencing is carried out per cycle. At each cycle, the different types of labeled nucleotides are added successively and only one type is incorporated in each fragment since the nucleotide termination prevents further elongation (Goodwin et al., 2016). Then after laser excitation, the fluorescence emission of the clusters is recorded and image processing enables us to infer each newly incorporated base. Finally, the termination is released and a new cycle can begin to incorporate the next base. Several cycles are repeated to obtain the sequence of the fragments.

In many applications the size of Illumina's reads proved to be a limit, for instance due to repetitive sequences. Illumina data can be improved by sequencing *paired-end* reads rather than *single-end* reads. Instead of sequencing one end of 150 bp, two ends of 150 bp on opposing strands are sequenced (see Figure 1.9). For each end fragment sequence, the paired-end reads can either overlap or be a given size apart, depending on the size of the fragments. The main advantage of paired-end reads compared to the single-end reads, is that they provide additional information on the relative distance of paired reads.

1.3.2.2 Towards longer reads

Based on Illumina sequencing technology a new type of sequencing data has recently emerged in 2016 (Ho et al., 2019) and provides long range information using NGS data. This

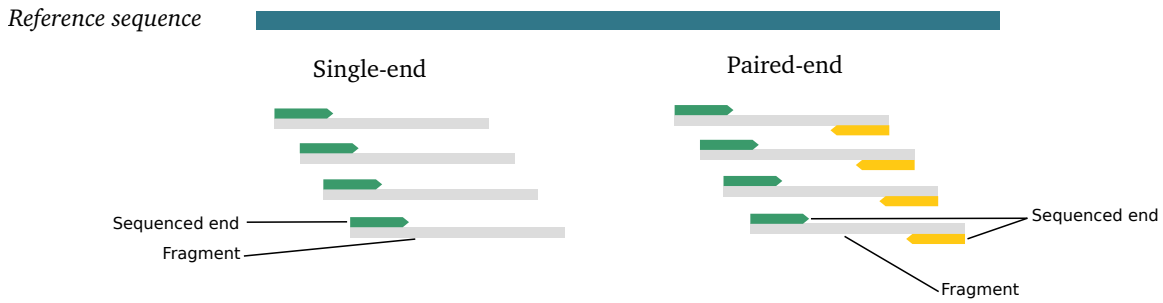

Figure 1.9 – **Single-end and paired-end sequencing.** Only one end fragment (green segment) is sequenced in single-end sequencing and both ends fragments are sequenced on opposing strands (green and yellow segments) in paired-end sequencing.

type of reads are produced by Moleculo technology and called *synthetic long reads* (Voskoboynik et al., 2013), and produced as well by 10X Genomics technology and called *linked-reads* (Zheng et al., 2016). The innovation of these technologies is based on a step upstream of conventional NGS sequencing to identify reads coming from the same long molecule of DNA and thus improve the bioinformatics processing step (van Dijk et al., 2018).

First of all, the principle of these technologies is to partition the genome into long DNA molecules of 10-100 kb for the 10X genomics technology (Goodwin et al., 2016; Meleshko et al., 2019). These long molecules are then split into fragments for sequencing and linked to a specific barcode for all fragments of the same long DNA molecule. The fragments are sequenced and the barcode sequences identify the partition from which the reads come.

1.3.3 Third generation sequencing technologies

In the last decade, a new generation of sequencing technologies has emerged: the third generation of sequencing technologies. The main features of the technologies of this generation are that they perform a direct single DNA molecule sequencing and in real-time. In contrast, the NGS approaches use an amplification step (Bleidorn, 2016; van Dijk et al., 2018). These improvements have increased the size of the reads compared to previous sequencing generations. Third generation approaches produce *long reads*, at least several kb in length, which is a distinctive feature of these sequencing technologies.

The third generation of sequencing technologies includes two sequencing approaches, that we will describe in more details in the next sections: Pacific Biosciences (PacBio) and Oxford Nanopore Technology (ONT). These recent technologies are still being improved to obtain better quality sequencing data. Raw data from long reads has a significantly higher error rate than

previous sequencing technologies, averaging about 13 % (Dohm et al., 2020). In contrast to NGS data, long reads have more gap errors than substitution errors, especially the PacBio technology.

1.3.3.1 Pacific Biosciences sequencing technology

The first third-generation sequencing technology developed is Pacific Biosciences (PacBio) and produces single molecule real-time sequencing (SMRT) (Eid et al., 2009). Like the Illumina technology, PacBio sequencing relies on sequencing by synthesis. But what makes the PacBio technology unique is that it sequences a single DNA molecule and exploits the zero-mode waveguide (ZMW) technology (Levene et al., 2003). ZMWs are nanowells at the bottom of which a DNA polymerase enzyme is fixed and where a DNA template molecule can be placed (Figure 1.10). The template molecule, illustrated in Figure 1.11, is a double-stranded molecule with a hairpin adapter ligated at both ends (green sequences). The single stranded sequence connecting sequences of adapters is called a *subread* (Figure 1.11). Then, in each ZMW unit simultaneously, the polymerase enhances replication with an adapter primer sequence and elongation processed with four fluorescent labeled dNTP for each nucleotide (Rhoads and Au, 2015; Logsdon et al., 2020). From the recorded emissions of dNTPs in ZMW units, the sequence of bases can be deduced.

PacBio technology has developed several sequencing protocols which produce sequencing data of different characteristics with varying error rates. One of the limiting factors of PacBio technology is the lifetime of the polymerase. Thus, two strategies are possible, either one chooses long DNA template molecules to sequence with a high error rate *Long Continuous Reads* (CLR), either short DNA template molecule to sequence *Circular Consensus Sequence* (CCS) reads with a lower error rate. With a lower error rate, the CLR corresponds to a replicated molecule without the adapter sequences, in other words, a subread sequence. Besides, because of the hairpin adapter sequences, the polymerase produces a circular DNA template. Therefore, both strands can be sequenced multiple times. A CCS read results from the computed consensus of all subreads in the replicated molecule (see Figure 1.11).

CLRs have an average size of 10 kb and can approach 100 kb (Sedlazeck et al., 2018a). According to the PacBio company, CLR have an accuracy of 89 %. Publications have reported PacBio data error rates between 11 and 15 % (Rhoads and Au, 2015; Ardui et al., 2018).

More recently in 2019, PacBio proposes a new type of data based on the CCS protocol, namely *High-Fidelity* (HiFi) reads with high accuracy ($\geq$ 99 %) and an average length of 13.5 kb (Wenger et al., 2019) (range between 10-20 kb). But HiFi reads are more expensive than

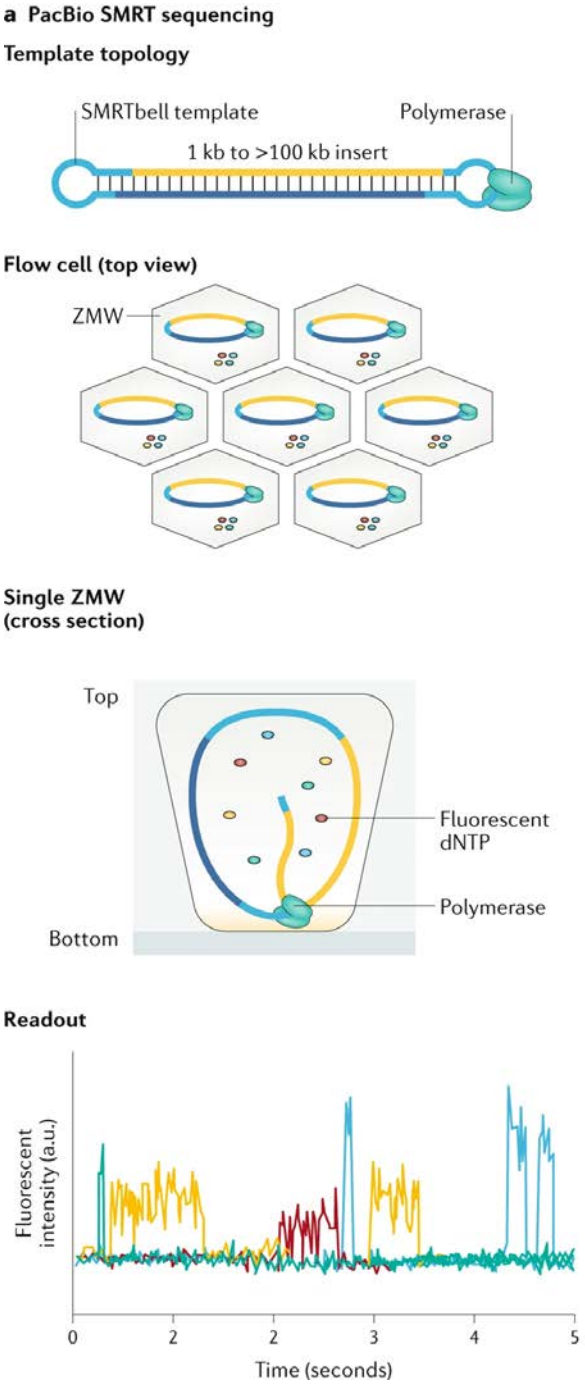

Figure 1.10 – Overview of the PacBio sequencing technology from Logsdon et al. (2020) (Figure 2a). DNA is fragmented and ligated to hairpin adapters to form a circular molecule known as a SMRTbell. Each SMRTbell template with a DNA polymerase is bound at the bottom of ZMW unit. Each SMRTbell molecule is then synthesized with fluorescently labeled dNTP. At each incorporated dNTP the fluorophore is excited and a camera records the fluorophore emission to deduces the base of the nucleotide.

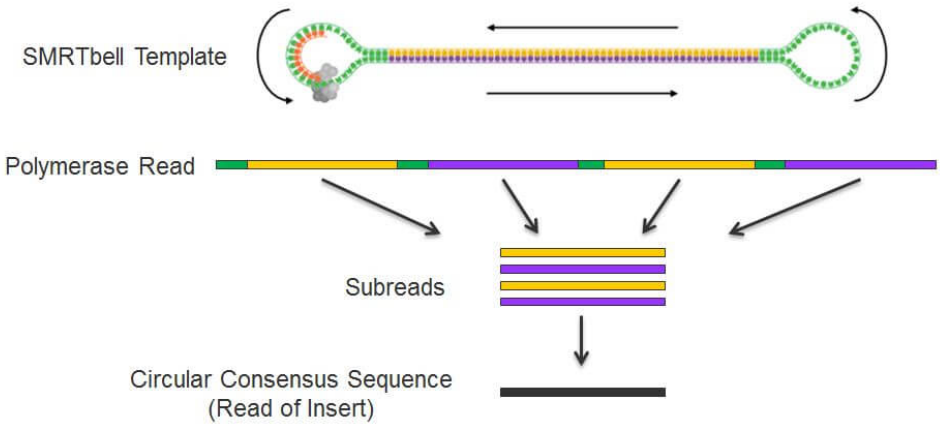

Figure 1.11 – Cicular Consensus Sequence (CCS) read diagram from the Pacific Biosciences Glossary of Terms. In this illustration the polymerase has synthesized twice the complete molecule, resulting in a total of 4 versions of the subread. The CCS read is obtained from the consensus of these 4 subreads.

other PacBio protocols. The use of HiFi reads is preferred for *de novo* sequencing, *i.e.* when no reference genome exists for the studied species.

So the main disadvantage of the PacBio data compared to the NGS data is the error rate, which is around 13 % (CLR). PacBio produces data with nonsystematic errors (Carneiro et al., 2012) and therefore PacBio read errors can then be corrected later with dedicated correction methods to improve data quality. Unlike the Illumina technology which mostly produce substitution error, PacBio is more sensitive to gap errors (insertions and deletions). But gap errors are more difficult to manage than substitutions.

1.3.3.2 Oxford Nanopore Technology

In 2014, a new third generation sequencing technology producing long reads was commercialized by Oxford Nanopore Technologies. The Oxford Nanopore sequencing Technology (ONT) is based on a nanopore arranged on the surface of a membrane, through which a single DNA molecule can pass (Lu et al., 2016; Jain et al., 2015). The passage of the nucleotides of a DNA molecule through the nanopore causes a change in the electrical current of the membrane. A motor protein, illustrated in yellow in Figure 1.12, is used to carry the double-stranded DNA molecule to the pore and then to pass one DNA strand through the pore base by base. Unlike the PacBio technology, ONT does not use a synthetic sequencing approach, but rather sequences by recording electrical current changes along a sequence molecule.

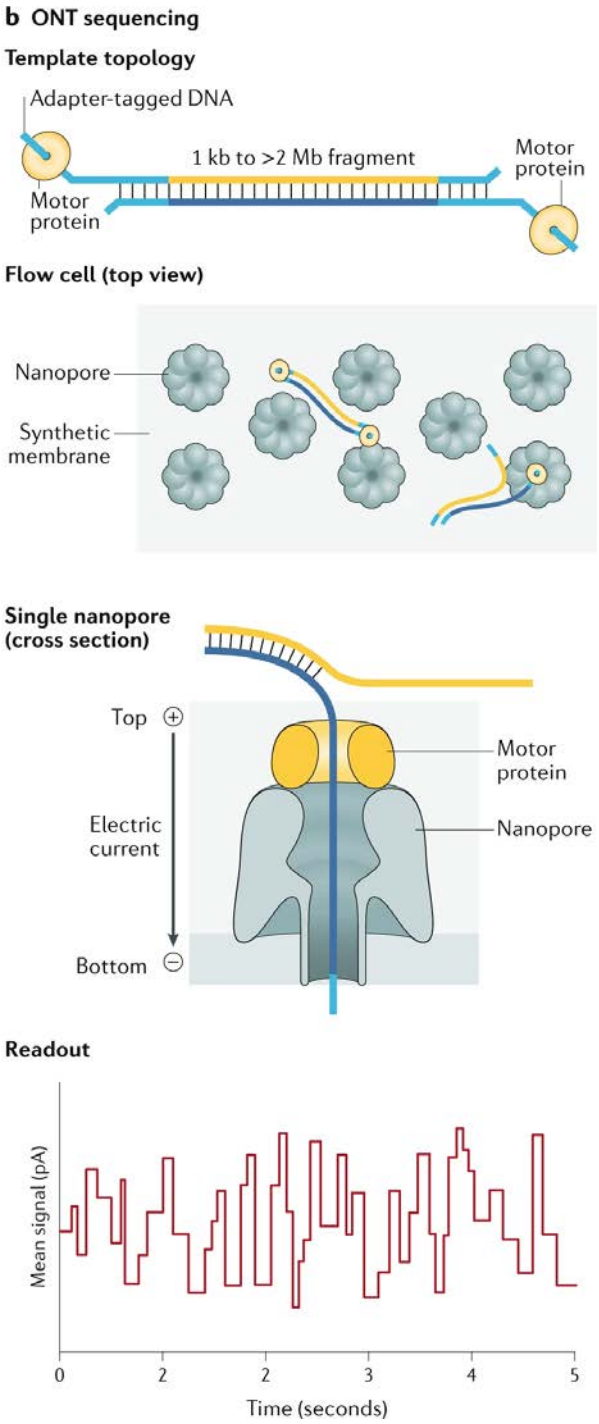

Figure 1.12 – Overview of the ONT from Logsdon et al. (2020) (Figure 2 b). Long DNA is tagged with sequencing adapters (light blue) preloaded with a motor protein on one or both ends. Binding proteins bring each template to a nanopore that crosses a membrane. The motor protein separates the double-stranded DNA template and a single stranded DNA passes through the nanopore. An electric current is applied to the membrane and thus each passage of a nucleotide creates a change in the electric current. Finally, basecalling tools interpret the current change measurements into a string of DNA bases.

The company ONT has developed several types of pore. The pore model R9 is the most common. In current R9 pores, the signal sensor is not measured on one base at a time but on approximately five bases (Wick et al., 2019). As a result, a direct related limitation of the ONT measurement system is the absence of signal change when the homopolymer size exceeds 5 bases, *i.e.* when more than five identical bases follow one another. The technology fails to distinguish the length of the homopolymers. However, ONT has tracks under development to overcome this limitation, based in particular on the use of a new type pore with two signal sensors (R10 pores).

In parallel with an improvement in reagent chemistry, several protocols (1D, 2D, $1D^2$) have been developed to increase data accuracy. The most recent protocol is the $1D^2$ protocol that uses specific adapters sequence, such as both forward and reverse strands pass successively through the nanopore in order to improve accuracy (de Lannoy et al., 2017).

The ONT technology requires the development of new bioinformatics tools to translate this raw signal data into a DNA sequence, this step is referred to as *basecalling*. Basecalling methods were initially based on a Hidden Markov Model (HMM) and were quickly replaced by machine learning approaches, more particularly deep learning approaches. Initially, the error rates of the data from the ONT were very high, around 40 %. Today, with improvements in sequencing protocols and basecalling methods, it is possible to obtain reads with an error rate of less than 5 % (Wick et al., 2019). Currently, ONT produces reads of 10 to 20 kb long, with an error rate ranging from 5 to 15 % depending on the sequenced genome and basecaller.

Besides, ONT developed a protocol to produce ultra-long reads (Jain et al., 2018), with a medium read size of 100 kb and reads up to nearly 1 Mb long. Ultra-long reads are particularly useful for genome assembly. One of the main advantages of ONT compared to any sequencing technology is the portability it offers. The smallest sequencing tool offered by ONT technology is the size of a USB flash drive.

1.3.4 Advances due to sequencing technologies

Sanger sequencing technology has enabled the sequencing of the very first genomes. Due to the high labor intensity of Sanger sequencing, the first organisms to be sequenced were small genomes, such as phages, viruses, bacteria, yeasts. A major effort by the Human Genome Project resulted in sequencing the very first human genome (draft sequence published in 2001) using the Sanger technology.

Over the past two decades, the emergence of the second generation sequencing technologies has revolutionized the field of genomics by improving our knowledge of the genome sequences and structure through increasingly effective approaches over the years. NGS technologies made it possible to sequence multiple living organisms at a large scale because of their high yield, high accuracy, and relatively affordable cost.

The short read data acquired over the years, provided access to information on the genomic diversity of the tree of life. Many projects aim at reconstructing genome sequences, and to build a reference genome sequence for model and non-model organisms. But NGS technologies not only enable to build reference genomes, they also have had major impact on genomics studies by allowing access at the base level of sequences with short reads and at an affordable cost. Indeed, there are many applications for NGS, such as *de novo* genome assembly, genomics variations, transcriptomics, epigenomics, metagenomics, population genomics, epidemiology. In 2008, a new project was launched, the 1000 Genomes Project, with the objective of sequencing 1000 human genomes using, in particular the NGS technologies, in order to establish the most complete catalog of variations of the human genome. However the size of the short reads does not allow to overlap complex regions, especially with repeated sequences longer than the size of the reads. As a result, many regions of the genome cannot be resolved with short reads, and are characterized by gaps in the genome assemblies.

More recently, the long range information provided with linked-reads, improves precisely downstream bioinformatics processing. With a size of up to 100 kb and the accuracy of NGS data (99.9 %), linked-read data has shown its usefulness in the context of assembly, haplotyping (Weisenfeld et al., 2017) and SVs study as well (Spies et al., 2017).

But for more than five years now, the third generation sequencing technologies have made it possible to access another level of genomic information by sequencing direct single DNA molecules, known as the long read data. These sequencing technologies have enabled a significant improvement of the assembly contiguity, as well as the haplotyping and the gap filling tasks. Studies achieve gap-free human chromosome assemblies (Miga et al., 2019). The long read sequencing technologies have also made it possible to access long distance information to study the structure of large rearrangements that were inaccessible with NGS technologies. However, both third generation sequencing technologies, PacBio and ONT, are producing data with a much higher error rate than NGS data. Long read data also require new methodological development to deal with such features. In parallel, these technologies are still being improved

and the continuous evolution of data quality brings an additional difficulty for long read methods that have to constantly adapt to these technologies.

The features of short read and long read data complement each other. On the one hand, the long read data provide information on the structure and on the other hand, the short read data provide high accuracy at the bp level. Several studies have precisely combined the two types of data and obtained more advanced biological results than with the use of only one of the two types of sequence data. In fact, following the emergence of the third generation of sequencing technologies, several so-called hybrid methods have been developed. These hybrid methods use both short read and long read data to exploit the benefits of both types of data. Several hybrid correction methods use the quality of the short reads to correct the long reads. It is the case as well for hybrid assembly methods that can for example use the long reads to assist the short read assembly. However, these experiments have an additional cost. So, genome sequencing by two complementary approaches cannot be chosen for everyday use.

To conclude, the sequencing technologies have contributed greatly to our current knowledge of genomic sequences. However, these advances are not only based on sequencing data, but have required the development of new bioinformatics methods capable of extracting information from the reads and providing interpretable analysis. In the next and last section of this chapter we develop the main methodologies needed for genomics.

1.4 Methodological approaches for genomics

1.4.1 Bioinformatics for sequencing data

Genome sequencing data represents only partial information. These data, therefore, require bioinformatics processing to extract the information of interest. We present in this section the main bioinformatic approaches needed when sequencing a given organism for the first time with genomic data.

Genome assembly. When studying a new species, usually the primary objective is to retrieve the full genome sequence to build a reference genome for this species. Since the datasets are sequenced at a high sequencing depth so that each position is seen at least n times in the

reads, the assembly methods reconstruct sequences based on read overlaps. But genomes are made up of repetitive sequences, whose repeated pattern sometimes exceeds the size of the read (short reads in particular). Assembly approaches must confront the genome complexity and methods are not always able to assemble all regions, leaving gaps in the assembly.

Genome annotation. Once the genome has been assembled, the next goal is to characterize the sequences of the genome in order to know the elements that make up the genome. The genome is made up of several elements, such as genes, intergenic regions, and regions of regulatory sequences, whose have specific features. For instance, the ends of the chromosomes are referred to as telomeres and are characterized by highly repeated sequences. All these elements encode for specific functions. So genome annotation aims at determining the location of the genes and at predicting their function.

Genome polymorphism. The first question of interest when studying genome polymorphism is to identify the variations between the different sequenced individuals, for instance. A naive approach to identify variations between two individuals of the same species with sequencing data first is to build the assembly of the two studied sequence genomes and then to compare them to each other at the same genomic positions. However, this approach is not very cost-effective in terms of time and memory for bioinformatics processing, especially if we want to detect variations from a new dataset of the same species. One solution is, therefore, to use a reference genome for the studied species. But the reads of the sequencing datasets represent only partial information of the entire sequenced genome. Thus, before identifying variations, we require the use of mapping approaches to find the location of the reads on the genome. Read mapping consists of identifying highly similar regions between the reference genome and the reads, but also highlights the differences between the two sequences. Hence, variations can be identified by comparing at each position the sequence of the genome and the sequence of the reads.

However, the identification of SNP or SV variants are two very different problems. In the case of SNPs, the variations are contained in the sequence reads, so the methods identify SNPs from the mismatches in the mappings. In contrast, the size of SVs is much larger than the size of the SNPs, so most SVs overlap several reads. As a result, bioinformatic methods have to use other strategies to address the problem of SV detection.

We detail, in the following section, the main issues that SV discovery methods have to face to identify these variations with sequencing data.

1.4.2 Methodologies for structural variants

The study of SVs using sequencing data raises several methodological problems in bioinformatics. Firstly, the aim is to identify SVs against a reference genome. Secondly, the aim is to genotype the SVs of an individual or on a population scale.

The first objective is, therefore, to detect and characterize the SVs in terms of structure, content, and copies (for CNV). The presence of SVs in a resequenced dataset will create truncated mappings for some reads. Thus, methods for discovering SVs regarding a reference genome will rely on these truncated mappings to identify SVs.

But discovery approaches must face several difficulties. First of all, SVs are much more difficult to study than short variants such as SNP, due to their diversity. SVs are more heterogeneous in terms of size and types than SNPs. Besides, the methods rely on reads to discover SVs *i.e.* partial information of the genome. We can already assume that the size of the reads will have an impact on the ability to detect SVs. Indeed the SVs are at least 50 bp and can be up to several hundred kb. On the one hand, the short reads of 150 bp still do not allow to cover the rearranged segments. One the other hand, because long reads have a high error rate (approximately 15 %), the mappings will potentially be more difficult to interpret, and structural variations may be difficult to distinguish from sequencing errors.

Finally, the genome is made up of very complex sequences, such as repeated sequences, regions of low mappability which consist of high repetitive sequences producing ambiguous mappings instead of unique mappings (Derrien et al., 2012), or highly polymorphic regions. Genomes even contain long genomic segments of at least 1 kb that are highly similar ($>$ 90 % identity). These very similar genomic regions are called *segmental duplication*s (Seq Dups) and can be either tandem or interspersed. Overall repeated sequences represent 50 % of the human genome (Treangen and Salzberg, 2012). The genome complexity of the genome makes it difficult to analyse the mappings.

Despite all these difficulties, methods based on read mapping are much less expensive from a methodological point of view than assembly-based approaches. Assembly-based approaches require much greater sequencing depth. Multiple methods have been developed based on a mapping approach to discover SVs in a new sequencing sample compared to a reference genome,

to catalog the diversity of SVs present in populations. Most of the SV methods focus on the human genome, for instance, for medical research applications or for knowing the distribution of the structural variations within human populations. However, the SV databases remain incomplete today, due to the difficulty of the SV discovering problem. Moreover, the databases are often redundant, *i.e.* the same SVs are described several times. Indeed, the predictions of the coordinates of the SV breakpoints differ between the methods of SV discovery. Thus, another issue is to combine predictions describing the same variants while avoiding merging distinct SVs.

Another question of interest when studying SVs is to assess the distribution of SV alleles in the population. The genotyping task consists of identifying genotypes for a set of known variants in newly sequenced individuals. This makes it possible to study the distribution of allele genotypes, for example, according to healthy or diseased cases. Methods specifically dedicated to the task of genotyping are more efficient than methods dedicated to the discovery of SVs because genotyping methods rely on prior knowledge of SVs. With the arrival of third generation sequencing technologies, several methods have been developed to achieve SV discovery. The field of study of SVs is expanding. Databases of SVs are updated, completed, and accurate SV catalogs are created as well. These new SV catalogs open the way to SV genotyping, which has many applications in research, medicine, agrology, and ecology.

This thesis focuses on SV genotyping with long read data. In the following chapter we first study the state of the art of methods dedicated to SVs. In particular, we detail the principle and limitations of the methods developed to answer the two main problems: the discovery of SVs and the genotyping of SVs. Based on the state of the art, we propose and present in Chapter 3 a new method dedicated to the genotyping of several types of SV from long read data. Chapter 4 is devoted to the validation of the proposed method, we provide a comparative analysis with other genotyping tools as well. Finally, in Chapter 5 we discuss the contributions and perspectives on the issue of SV genotyping with long read data.

Chapter 2

State of the art

Table of contents

2.1 Introduction 47

2.2 Sequence alignment 47

2.2.1 Short read alignment 49

2.2.2 Long read alignment 50

2.2.2.1 `NGMLR` 51

2.2.2.2 `Minimap2` 52

2.2.2.3 Long read mapping tools performances 52

2.3 Structural variant discovery 53

2.3.1 Structural variant discovery with short reads 54

2.3.1.1 SV signature detection strategies 54

2.3.1.2 A plethora of structural variant callers 59

2.3.1.3 Limitations 59

2.3.2 Structural variant discovery with long reads 60

2.3.2.1 Three methods for SV discovery with long reads 61

2.3.2.2 SV discovery methods based on other approaches 67

2.3.2.3 SV callers specific to SV types 69

2.3.2.4 Hybrid approaches 71

2.3.2.5 Conclusion on SV discovery using long reads 71

2.3.3 Conclusion on structural variant discovery 72

2.4 Genotyping structural variants 74

2.4.1 SV genotyping using short reads 75

2.4.1.1 SV genotyping methods mapping-based against reference allele 75

2.4.1.2 SV genotyping methods representing allele variants 77

2.4.1.3 Conclusion on SV genotyping methods for short reads . . 80

2.4.2 SV genotyping with long reads 81
2.4.3 Conclusion on SV genotyping 82
2.5 Where the thesis stands in relation to the state of the art . . . 84

2.1 Introduction

When studying SVs there are two main problems: the discovery of SVs and the genotyping of SVs. As we introduced in the previous chapter, detecting SVs consists in identifying SVs present in a sequenced dataset compared to a given reference genome. To identify variations between the sequenced dataset against the reference genome, the SV discovery methods look for specific signatures resulting from an SV in the sequenced dataset. While, in the second problem, SV genotyping aims at assigning a genotype to each SV in a set for a given sequencing dataset. The set of SVs required for genotyping contains known SVs which are therefore defined in terms of SV type, coordinates, and other relevant features. Thus, the discovery and the genotyping of SVs do not address the same difficulties, since genotyping relies on a defined set of variants, whereas SV detection does not allow prior information about the potential SVs between the reference genome and a given individual.

This chapter is divided into three main parts. In the first part, we will present the principle of alignment of two sequences, a concept on which the majority of SV discovery and genotyping methods rely on. So, we will define the alignment problem and briefly explain the tools developed for short read data, then we will detail the alignment problem applied to long read data. In the second part of the chapter, we will move on to describe the SV discovery problem by presenting the general principle of SV detection. First, we will briefly explain the SV discovery methods based on short reads to understand their difficulties. Then, we will detail the approaches for SV discovery dedicated to long reads. Finally, in the last part of this chapter we will analyze the state of the art of the genotyping methods for both short reads and long reads.

2.2 Sequence alignment

The alignment of two sequences is a representation to identify regions of similarity between the two sequences. The alignment of two sequences consists in indicating if, at each position of the two sequences their bases are identical or not. At a given position, there are three possible operations: if the bases are identical between the two sequences, there is a match, if not, it is a mismatch, and lastly there is a gap when a base is present in one sequence and absent in the other. The number of possible alignments between two sequences is very large. We define then a score function to evaluate each alignment. Each match, mismatch, and gap operation has a specific cost and the score of a pairwise alignment is the sum of the operation's cost. The goal is to retrieve the sequence alignment that shares the most similarity, *i.e.* the most matches

or the best alignment score.

In 1970, Needleman-Wunsch algorithm was proposed to retrieve the exact solution of the *global alignment* problem (Needleman and Wunsch, 1970). The global alignment aims at obtaining the best alignment score of the alignment of two sequences over their whole lengths. The Needleman-Wunsch algorithm relies on dynamic programming and guarantees to find a solution of the best alignment score within a quadratic time ($O(mn)$ for two sequences of length m, n). Then, in 1981 the Smith-Waterman algorithm was proposed to retrieve the exact solution of the *local alignment* problem (Smith and Waterman, 1981). Unlike global alignment, the problem with local alignment aims to identify similar regions between the two sequences. The Smith-Waterman algorithm also relies on dynamic programming to retrieve the best alignment score of the local alignment problem in quadratic time ($O(mn)$). For instance, Figure 2.1 represents a global alignment of two sequences that align along their entire length. While Figure 2.2 shows a local alignment of two sequences, where only the most similar regions are represented in the alignment.

```
sequence 1 :    ATCGATTTATG-TCTTAGTCGATATATTATGCTAGTTAGATCGATA
                ||||.||||||.||||| ||||||||||||  |||.|||||||||
sequence 2 :    ATCGCTTTATGATCTTACTCGATATATTAT--TAGATAGATCGATC
```

Figure 2.1 – **Global alignment**: Match (`|`), mismatch (`.`) and gap (`-`) operations illustrated the similarities and differences between sequence 1 and sequence 2.

```
sequence 3 :    ATCGATTTATGATC-TAGTCGATATATTATGCTTAGATAGATC
                        |||||| |||||||||.||||
sequence 4 :        GCAGATGATCTTAGTCGATAGATTATGGACTCCC
```

Figure 2.2 – **Local alignment** of sequence 3 and sequence 4.

When aligning a read against a reference, we want to align the entire sequence of the read, and if possible in a single piece. This step is also referred to as the mapping task, where the aim is to locate the sequence fragments on the genome and to identify differences between the reference and the read. Differences could result from sequencing errors or sequence polymorphism.

Because of the size of the genomes (several Gb long) and the number of sequences (millions to hundreds of millions), the problem required the use of heuristics to solve the alignment problems within a reasonable runtime. `BLAST` (Altschul et al., 1990) is the very first tool to use a heuristic approach to find sequence similarities. The heuristic proposes a *seed-and-extend*

strategy and has since been widely adopted by other methods. The seed-and-extend approach first looks for highly similar regions, called seeds, between the genome sequence and the read sequence, usually based on an efficient indexing approach. Then from the seeds, the alignment is extended using dynamic programming.

For sequencing data, the alignment problem aims to find the location of a read on the reference genome. However, the genome consists of many repeated sequences, for example, the repetitive DNA sequences represent approximatively 50 % of the human genome (Treangen and Salzberg, 2012). The main difficulty of mapping will be to find the right location of the reads on the genome. However, a read can map to more than a single location, according to the size of the repeated motifs and to the size of the reads. The *mappability* is a measure of the ability to align the reads to a unique location in the reference genome (Derrien et al., 2012).

So, since the aim is to align the entire read sequence to a specific region of the genome, we can already anticipate that the characteristics of the sequenced reads will have an impact on the achievement runtime of the alignment. Indeed, the problem of aligning a short read of 150 bp with a low error rate is not the same problem as aligning a long read of a few tens kb with a high error rate. Distinct methods have therefore been developed to deal with short reads and long reads. In the following sections, we will first briefly explain the methods developed to map the short reads against a reference genome. Then, in a second section, we will develop in more details the recent methods developed to map long reads against a reference genome.

2.2.1 Short read alignment

Short reads are characterized by a size of a few hundred base pairs (100-200 bp) and a low error rate (approximately 0.1 %). Thus, when mapping short reads against a reference genome, the alignment is expected to align to a region of the reference genome and along the entire read length. It is also referred as the semi-global alignment problem. Therefore, the short read mapping methods rely either on local alignment or on semi-global alignment algorithms.

In addition, NGS technologies produce high-throughput sequencing with millions of sequence reads. Meanwhile, the reference genome represents several Gb. Short reads mapping methods require the use of heuristics such as seed-and-extend strategy. Contrary to the `BLAST` algorithm which looks for local similarities (> 70 %), the objective of short read mapping is to find the unique (if possible) location on the reference genome, so we want specific alignments with a very high similarity. Mapping methods often require the use of sequence indexing to provides access to the number of occurrences and occurrence positions of the indexed sequences.

Increasing the size of the indexed seeds allows to be more specific, but raises a problem of indexing. To overcome the indexing problem, short read mapping methods rely on efficient data structures such as the Burrows-Wheeler transform or the FM-index. In addition, the mapping methods must be fast to index large volumes of data: the full genome sequence and millions of reads. There are two major consensus tools for short read mapping: `BWA` (Li and Durbin, 2009), and `BOWTIE2` (Langmead and Salzberg, 2012).

In general, the Sequence Alignment/Map (SAM) output format is used to describe pairwise alignments per line through multiple fields. These fields include mapping information useful later for SV calling, such as the `MAPQ` score for describing the mapping quality and the `CIGAR` string that describes at each position if the two sequences match, mismatch, or gap. The SAM format compresses into Binary Sequence Alignment/Map (BAM) format, which is its binary version.

2.2.2 Long read alignment

Even if the majority of long read mapping methods also follow the seed-and-extend principle, the problem of long read alignment is not quite the same as the short reads alignment problem. Because of the different data characteristics between short reads and long reads, long read alignment methods must adapt to map longer reads (a few tens of kb) at a higher error rate (approximately 15 %) with a majority of gap error rather than substitution. The long read mapping methods had then to find other solutions for the search for seeds, by proposing new approaches than short read indexation methods, to avoid having a number of seeds too important per long read.

The very first long read mapper to be developed is `BLASR` (Chaisson and Tesler, 2012). `BLASR` uses an FM-index to locate candidate intervals and then calculates the pairwise alignment for the best candidates. Because of the method's choice of data structure, `BLASR` is not very time-efficient. Among the very first long read mappers is `BWA-MEM` (Li and Durbin, 2009). The original short read aligner has been adapted to perform PacBio and ONT data mapping. But later the same author has developed another tool, `Minimap` (Li, 2016), which responds in a more relevant way to the characteristics (read's length and error profile) of the long read data (PacBio and ONT). Subsequently, a new version of `Minimap`, `Minimap2` (Li, 2018) was then developed and presents important improvements, especially to adapt to SVs. Other long read mappers have also been developed such as `GraphMap` (Sović et al., 2016). However `GraphMap` has a high time and memory consumption compared to other long read mappers. Or the `LAST` (Kiełbasa et al., 2011) long read aligner has also gained interest in the context of SV disocvery for its ability to compute several alignments for the same read.

However, two long read mappers stand out because of their specificity to consider SV: `NGMLR` (Sedlazeck et al., 2018b) and `Minimap2`. `NGMLR`, like `Minimap2`, differs from other methods precisely by taking into account the particularity that long reads can span at least partially the SVs sequences. We will, therefore, detail in the next two paragraphs both of these two long read mapping methods.

2.2.2.1 `NGMLR`

`NGMLR` (Sedlazeck et al., 2018b) is a long read mapper that aligns both PacBio and ONT data against a reference genome. `NGMLR` was developed to precede a SV calling tool, `Sniffles`. Thus, it was designed to directly address the problem of aligning long read data for the subsequent discovery of SVs. The main innovation brought by `NGMLR` is the choice of the concave gap cost which is quite relevant for long read data. `NGMLR` was the first long read mapper to choose a gap score model different from the affine gap model. `NGMLR` is implemented in `C++` and is available in Bioconda.

The principle of `NGMLR` mapping is declined in four main steps. The first step is the *search for anchors*, obtained by a seed-and-vote approach. First of all, sub-reads are defined as 256 bp non-overlapping segments of the read. For a given sequence of size N, k-mers are defined as all overlapping words of size k and it consists of a set of $N-k+1$ k-mers. Then, local pairwise alignment is performed between the regions of the genome and the sub-reads that share common k-mers (by default $k=13$). The anchors are then considered to be the genomic regions with the highest alignment scores while filtering out highly repetitive regions. From the anchors, the second step is to look for the *linear mapping segments* of sub-reads to the reference genome. The largest set of sub-reads that aligns co-linearly to the reference genome is then identified and this set of sub-reads is joined.

The third step performs a *Smith-Waterman alignment* using a dynamic programming algorithm with a concave gap cost scoring model to compute a pairwise sequence alignment between the linear mapping segments and the reference genome. Long reads have two distinctive features for the alignment of sequences: i) they can span over deletions or insertions, *i.e.* large gaps in the alignment, and ii) their error rate is very high compared to short reads. Linear gap cost and affine gap cost are ill-adapted to this type of data and generate the alignments spanning SVs to be falsely split into several smaller indels. Conversely, with a concave gap cost, the cost of an additional gap depends on the total size of the gap. Thus, from a given total gap size, the cost of an additional gap declines. The concave gap cost is less penalizing for the

alignments spanning SVs than linear or affine gap costs.

Lastly, in the fourth step, `NGMLR` *selects the best combination of linear alignments* that does not overlap on read coordinates. A mapping quality score is also computed for each selected linear alignment.

2.2.2.2 `Minimap2`

`Minimap2` (Li, 2018) is a versatile mapper for long read data, including both PacBio and ONT data. It offers many applications by processing both genomic and transcriptomic data taking into account spliced regions, it can also find overlaps for long read data as well as perform genome to genome mapping. `Minimap2` succeeds `Minimap` (Li, 2016), this improved implementation now makes it possible to align to the base accurately, while also considering SVs.

The `Minimap2` relies on the seed-and-extend strategy. One of the main features of `Minimap2` is the use of minimizers. A minimizer corresponds to the smallest k-mer, according to a certain ordering, for each sliding window w of k-mers. The advantage of minimizers is that they reduce data representation. `Minimap2` therefore relies on the fact that two similar sequences will tend to share the same minimizers. So the first step of `Minimap2` is to look for seeds between the query sequence and the target sequence that share collinear minimizers in a width band. Then in a second step, `Minimap2` identifies sets of collinear seeds, called chains. Finally, alignments are extended between adjacent seeds using dynamic programming. Like `NGMLR`, `Minimap2` does not use a linear gap score model either, but it uses a concave gap cost to facilitate the identification of SVs, such as long deletions and long insertions.

`Minimap2` uses an efficient implementation and data structures to achieve both accurate and fast alignments compared to other long read mappers (De Coster et al., 2019). The tool is implemented in `C` and it is available in Bioconda. `Minimap2` returns the alignments either in the usual SAM format or in the pairing mapping format (PAF), which is another tab-separated mapping format proposed by the author.

2.2.2.3 Long read mapping tools performances

Long read mapping tools performances were assessed for the purpose of SV calling (De Coster et al., 2019; Zhou et al., 2019). `NGMLR`, `Minimap2`, and `LAST` performances were compared. Both studies indicate that `Minimap2` is the fastest long read aligner. According to De Coster et al. (2019) `Minimap2` has an average runtime of 178 sec/100,000 reads, followed by

NGMLR with a runtime of 1,289 sec/100,000 reads. LAST is the slowest with 3,392 sec/100,000 reads. The results for the SV discod ery according to the different long read mappers is discussed later in the manuscript (see section 2.3.2.5).

2.3 Structural variant discovery

The SV discovery task consists in defining in terms of content and structure, and possibly copy number, the set of SVs of at least 50 bp present in an individual (Alkan et al., 2011). The SV discovery process is often referred to as SV detection or SV calling, as well. Over the past decade, many approaches and methods have been developed to address this issue. Most of the methods identify SVs for a given sample regarding a reference genome.

Before the arrival of the NGS data, other methods were already being used to identify SVs. Probes and hybridization-based approaches, such as CGH arrays or FISH, were used to detect Copy Number Variations (CNV) (Alkan et al., 2011). But these low throughput approaches to discover SVs were replaced as soon as high throughput and low cost sequencing data became available, with the NGS technologies. Sequencing data offer many advantages regarding the SV discovery problem, as the SV content can be retrieved as well as the precise identification of breakpoint positions in order to resolve the SV structure. This, then, allows a better understanding of the impacts caused by these variations.

The SV detection problem attempts to identify SVs in a given sequence dataset relative to a reference genome. Thus, it implies on one hand to have a reference genome of the sequenced species, and on the other hand to have a sequenced sample for a given individual, where each read represents partially the genome of this individual. In this manuscript, we focus only on the SV discovery methods based on a mapping approach, which represents the majority of the methods developed to address the SV discovery problem. There are other approaches based on *de novo* assembly but which require significant sequencing depths (50x) compared to the mapping-based methods (15x) (Mahmoud et al., 2019). Moreover, most often when studies aim to discover the set of SVs present in an individual or within a population, there already exists a reference genome published for the studied species. This explains why the SV detection methods based on a mapping approach are more widely developed and used.

The SV discovery pipeline is divided into two main steps, illustrated in Figure 2.3. This pipeline is valid for short reads as well as for long reads. The first step consists in aligning the sequencing reads against the reference genome to identify their localization. The second step is

the actual SV calling process. Alignments are analyzed to identify signatures likely to indicate the presence of an SV. Then, the methods apply a clustering approach, to merge SV signatures indicating the same SV. Thus, the resulting clusters correspond to a single SV candidate. The clustering is followed by a filtering step to remove spurious SV candidates. Finally, the discovery methods calculate a score for each remaining SV candidates, representative of the reliability of the SV calls.

2.3.1 Structural variant discovery with short reads

Many methods have been proposed for SV discovery using short read data. But the major challenge of these methods is to succeed in detecting SVs of several hundreds of bp to thousands bp with read fragments of only 100 to 150 bp. Most of the time, the size of the reads does not exceed the size of the variants, and therefore, the interpretation of the signals is not straight forward.

In this section, we review the strategies exploited by SV detecting tools and highlight their limitations.

2.3.1.1 SV signature detection strategies

Once the short read data have been aligned to the reference genome, SV calling methods first analyze the alignments and try to identify signatures that could result from SVs in the sequenced sample. Thus, SV discovery directly depends on the quality of the alignments (Guan and Sung, 2016). Alignment spanning the entire read length and mapping to a unique region of the genome, are high quality alignments. Paired-end sequencing data are often used for the discovery of SV, the alignments are referred to as *concordant*, if paired-end reads are consistent in terms of relative alignment position and in the opposite orientation. Thus, if a sequenced sample has no variation in regards to the reference genome for a given a genomic region, the alignments are expected to be *concordant*. Whereas, in the case of an SV present in a sequenced sample, the SV will generate truncated alignments.

However, genomes are not random sequences: sometimes genomes have so-called complex sequence regions, with repeated sequences, highly polymorphic regions, and sequences of low complexity. The alignment methods fail to correctly and uniquely align the short reads to these complex sequences, as the sequences are not sufficiently discriminating with respect to the size of the short reads. This results can also result in *discordant* alignments. Therefore, the difficulty of the SV discovery problem is precisely to detect SV potentially indicated by

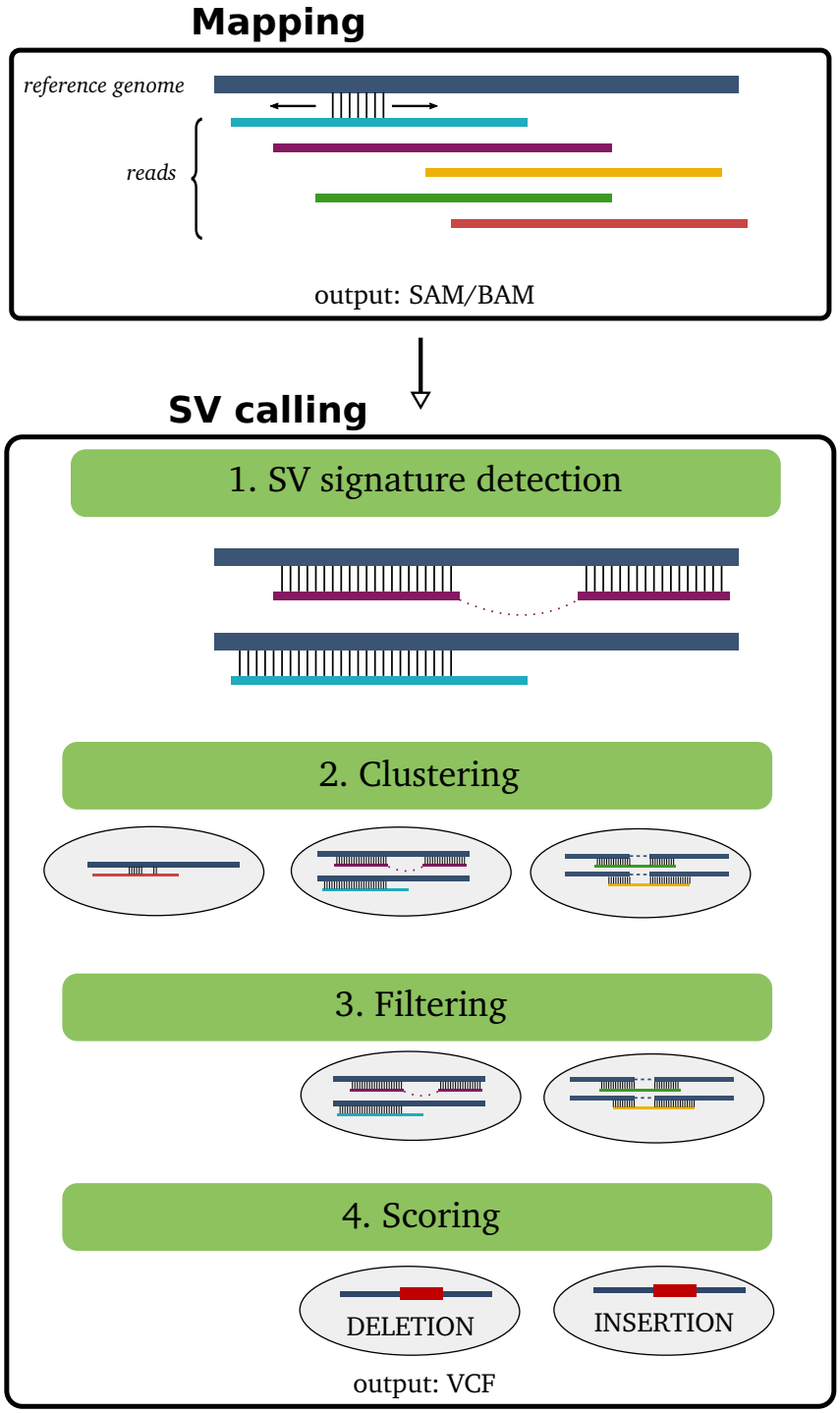

Figure 2.3 – SV discovery pipeline based on alignment. The SV calling process is divided into 4 main steps: 1. Identification of the SV signatures in the sequence alignments. 2. Clustering of SV signatures, most of the time each cluster represents a candidate SV. 3. Filtering to keep the clusters of the most likely SV candidates. 4. Scoring SV candidates for SV calling.

truncated alignments without detecting false SVs resulting from non-informative aligned reads.

Most SV discovery methods share the same detection process, illustrated in Figure 2.3. Although most SV callers share the same general pipeline, the methods rely on different strategies and signals to discover SVs and resolve their breakpoints (Medvedev et al., 2009; Alkan et al., 2011). According to the signals, only certain types of SVs can be detected. Four main strategies are exploited to identify SV signatures and are described in the following sections.

a. Read depth strategy

Read depth represents the number of times a given position of the genome has been sequenced. This information can be used as a strategy to identify gains or losses of genomic regions (Figure 2.4). For instance, a deletion in a sequenced individual will result in a decrease of the sequencing depth of the deletion sequence on the reference genome. A duplication in sequenced individuals results in an increase of the sequencing depth for the duplicated sequence. Thus, variations of sequencing depth can be used as pieces of evidence of SVs. However, this SV detection strategy must deal with the heterogeneity of the sequencing depth along the genome. Therefore, variations in sequencing depth may be due to artifacts and not deletions or duplications. To face this difficulty, methods use a sliding window to compare sequencing depth variations within a precise region. The choice of the size of the sliding window is essential, because if the window is too small then we will be too sensitive to detect depth variation due to other biological variations. If, however, the window is too large then we will not have enough resolution to detect small variations in depth, resulting in missing small CNV. Read depth detection methods rely on statistical approaches to deal with the heterogeneity of the sequencing depth, where the mapping read depth is modeled by either a Poisson distribution or a Negative Binomial distribution. Read depth approaches are particularly efficient when comparing sequencing depth between the case and the control data because, in the absence of variation, the sequencing depth should remain identical at the same positions.

b. Read pair strategy

Read pair based approaches rely on paired-end data to detect SVs, by assessing the orientation, and distance of mapping between the paired-end reads. As we've seen in the previous Chapter 1, paired-end reads are produced by sequencing both ends of a fragment separated of a specific size. So, paired-end reads are expected to map at a specific distance from each other, corresponding to the insert size and in opposite orientation. However, the presence

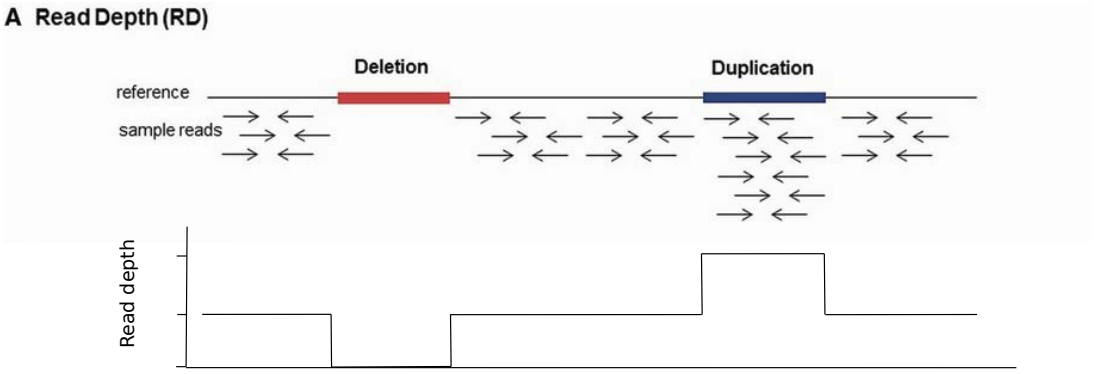

Figure 2.4 – **Read depth strategy**, figure adapted from Figure 2-A from (Escaramís et al., 2015). Read depth changes indicate either a deletion (loss) or a duplication (gain)

of SVs alters this type of expected signature and produces *discordant* read pair alignments, as illustrated in Figure 2.5. For instance, a deletion will result in the paired-end reads to map further apart from each other, rather than the expected insert size. Also an inversion will produce paired-end reads mapping at a discordant relative distance on the same chromosome and in an inconsistent orientation. So, read pair signatures can be used to detect the main types of SVs (Figure 2.5). The difficulty of the detection methods based on read pair, is that a given SV type can generate multiple signatures, and a single signature can also result from multiple types of SV. In addition, nested SVs produce signatures of discordant read alignments that are difficult to interpret. Additionally, discordant read alignments can result from artifacts, due to repeated sequences, leading to the detection of false positive SVs. Therefore, methods need to combine and cluster the different SV signatures that support the presence of the same variant. Then, methods assign to each SV candidate a score representing its reliability.

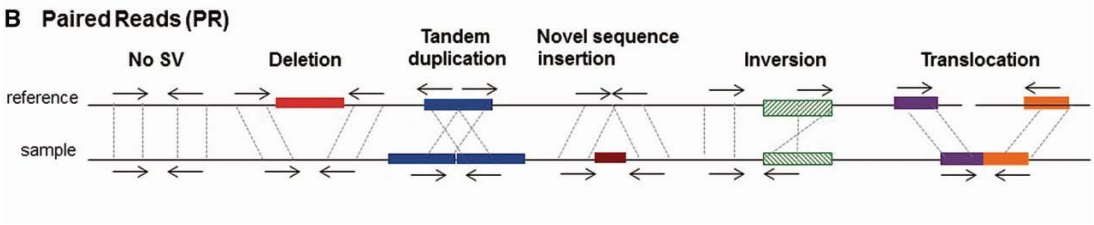

Figure 2.5 – **Read pair strategy**, Figure 2-B from (Escaramís et al., 2015). Based on paired-end reads relative distance, orientation and strand, we can detect deletions, tandem duplications, insertions, inversions, translocation.

c. Split reads strategy

Split reads are the reads whose alignments have been interrupted. Approaches based on this strategy also make it possible to detect the majority of the types of SVs, such as deletions, novel insertions, inversions, duplications, and translocations. A distinction is made between *split reads*, which are reads aligned in two parts with segments non-adjacent on the reference and *soft-clipped* reads, which are reads that partially aligns to the reference. These two types of interrupted alignments are illustrated in Figure 2.6. For instance, a deletion in a sequenced sample results in reads splitting to align to two non-adjacent portions of the reference. Thus, the absence of complete alignment along the read suggests the presence of a rearrangement. A drawback of this strategy is that it requires to align shorter sequences, and thus it decreases the specificity and increases the runtime. Analyzing split reads for all reads is too expensive. So most often methods apply the split reads strategy after using approaches to select potential truncated reads.

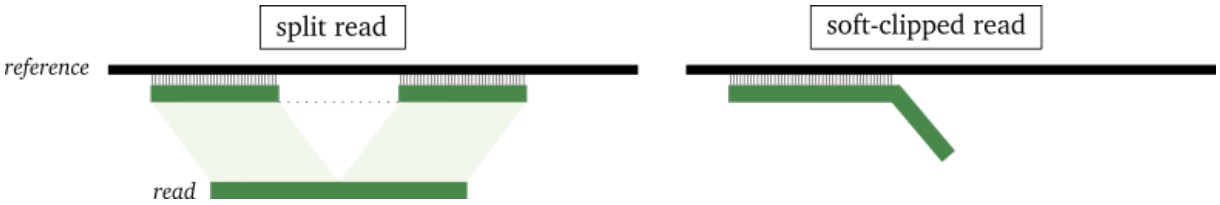

Figure 2.6 – Split read alignment and soft-clipped read alignment. The split read alignment indicates the presence of a deletion and the soft-clipped read alignment indicates the presence of a breakpoint at the unmapped end.

d. Local assembly strategy

Mapping-based SV discovery methods can rely on local assembly as well. The *de novo* assembly of all the reads of the genome is very costly and requires a significant sequencing depth compared to mapping-based methods. However, some variants are difficult to discover with mapping, such as insertions. One possible strategy is to perform a local assembly from a subset of reads. For instance, the unmapped reads can be assembled locally to detect insertions. The local assembly strategy can also be used to refine breakpoints of complex SVs, as well as all types of SVs. It is the most resolving strategy for SV discovery, as long as the sequencing depth is sufficient.

Figure 2.7 shows an example of a novel insertion detection, the short reads are assembled in a contig (brown segment). Then, the contig fails to be aligned against the reference, so we can infer the presence of a novel insertion since the sequence is not included in the reference.

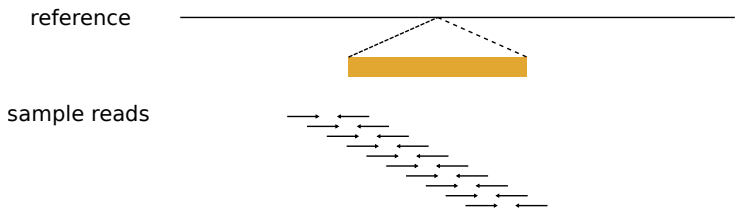

Figure 2.7 – **Local assembly based strategy**, figure adapted from Figure 2-D from (Escaramís et al., 2015). Short read local assembly and contig mapping to the reference.

2.3.1.2 A plethora of structural variant callers

Today, there exists a plethora of software for detecting SVs using short read data (more than 70) (Mahmoud et al., 2019; Kosugi et al., 2019), and these SV callers rely on the four detection strategies, that we detailed above. The first SV callers developed are usually based on a single signature detection strategy, whereas most methods developed today exploit several if not all strategies. Combining multiple strategies increase SV detection sensitivity and accuracy.

Some of the most recommended tools today include `DELLY` (Rausch et al., 2012), `LUMPY` (Layer et al., 2014) or `Manta` (Chen et al., 2016). `DELLY` relies on both paired-end reads and split reads strategies. For their part, `LUMPY` and `Manta` use read depth, paired-end read, and split read strategies to perform SV calling. Most methods combine different signals to detect SVs but differ by using different clustering and scoring of SV calls approaches.

Combining several SV detection tools, also based on several strategies, increase SV calling performances, improving both accuracy and sensitivity (De Coster and Van Broeckhoven, 2019). Some tools have even been developed for this purpose. These tools include `MetaSV` (Mohiyuddin et al., 2015) or `Parliament2` (Zarate et al., 2018), `SURVIVOR` (Jeffares et al., 2017), which combine different SV callers for short read data.

2.3.1.3 Limitations

Unfortunately, today no tool can solve the problem of SV discovery with short reads. Short read SV callers achieve low sensitivity, which can vary widely from 10 % to 70 %, and at the same time low accuracy, with false positive rates of up to 89 % (Mahmoud et al., 2019). Besides, there is a lack of consistency between the prediction tools (Guan and Sung, 2016). SV detection is a difficult problem to solve and is based on the analysis of signature alignment which can be obtained by several types of SV. Such limitation is crucial to understand when interpreting SV calls according to the studied SVs and context.

On the one hand, SV callers achieve low specificity, illustrated by many false positive calls detected. False positives can result for example from repeated sequences that also produce alignment signatures as SV. Therefore, SV detection signals are similar to complex sequence signals, resulting in misinterpretation of the signatures that do not result from an SV event.

On the other hand, SV callers achieve low sensitivity, and it varies depending on the type of SV. For instance, deletions can be more easily detected than novel insertion sequences (Kosugi et al., 2019; Delage et al., 2020). Chaisson et al. (2019) also showed that SV calling tools for short reads only allow the discovery of 52 % of deletions and 18 % of insertions for a human genome. SV callers struggle to retrieve true positive SVs because SVs are often located close to repetitive sequences (Zook et al., 2020). These complex sequences can be characterized as low mappability regions, which makes it difficult to interpret the signal. Detection methods then apply stringent parameters to avoid false positive calls.

The SV detection task is already difficult, due to the diversity of types, sizes and complexity of SVs, but the discovery problem is even more tricky because of the proximity of SV to complex regions of the genome, such as repeated regions or segmental duplications. Such complex sequences make the SV discovery more complex due to short read mapping failing (Alkan et al., 2011). Since the size of the short read data does not span along the repeated regions close to the SVs. Thus, the information contained in short read data cannot not always resolve the rearrangement breakpoints and it, therefore, shows that short read data have significant weaknesses regarding the SV detection problem.

2.3.2 Structural variant discovery with long reads

Since 2012 and 2014, respectively, the sequencing technologies PacBio and ONT produce a new type of sequencing data: long reads, characterized as long single molecule sequenced in real-time. This new type of sequencing data offers new features compared to the short read data, such as a longer read length but at the expense of a higher error rate. Therefore, long read data create new methodological and computational problems. The long read data are particularly applicable to the study of SVs and allowed to expand our knowledge of these variants. Indeed, long read data are very useful in the context of SVs, because the size of the reads allows spanning the entire rearrangement or at least one of the breakpoints of the rearrangement, as opposed to short read data which cannot span such a length. Furthermore, the long read data also allows us to cover repeated regions. As SV breakpoints are often close to repeated regions, long read data are particularly interesting for the study of SVs.

Long read methods for SV detection share similar steps and strategies with methods dedicated to short read data (Figure 2.3), as the majority of SV callers rely on a step of sequence data alignment before interpreting its alignments to detect the presence of SVs. Once the long reads have been aligned against the reference genome, we can proceed to the actual SV detection step. As with the SV detection methods for short reads, the long read SV detection methods look for signatures in the resulting alignments that would indicate the presence of SVs. The long read SV detection methods are inspired by previous short read methods and rely on similar criteria as the previous methods, such as orientation or span of the alignments. Even with the long reads, the SVs will also produce split read alignments and soft-clipped read alignments (see Figure 2.6), indicating the presence of a breakpoint. Since the size of the long reads can potentially overlap the SVs, the presence of SVs will then cause very erroneous regions (mismatch or gap) in the alignments.

Some SV calling tools include directly the first mapping step of the long read against the reference genome. While some SV calling tools require the long read alignments file, usually in sorted BAM format, the binary version of the pairwise alignment SAM format. SV callers output simply the set of discovered SV in a VCF file, as short reads SV callers.

In this section, we present in more detail three SV callers that differ from other methods. First, we will present `PBHoney` (English et al., 2014), which was the very first method developed for SV detection with long reads and which influenced the following methods. Next, we will detail a method that is unanimously acclaimed by the community, `Sniffles` (Sedlazeck et al., 2018b), followed by another SV caller, `SVIM` (Heller and Vingron, 2019), which also gets very good feedback. Then we will cite other tools that can be used to respond to specific SV discovery problems.

2.3.2.1 Three methods for SV discovery with long reads

a. `PBHoney`

`PBHoney` (English et al., 2014) is the very first SV caller dedicated to long read data to be developed in 2014. `PBHoney` proposes two distinct methods to identify SVs with PacBio data. One method looks for mapping interruption (`PBHoney-Tails`) and the other method analyzes the genome coverage (`PBHoney-Spots`). To detect SVs, `PBHoney` relies on alignments made upstream with `BLASR` (Chaisson and Tesler, 2012), a long read mapper for PacBio data commonly used at the time. `PBHoney` detects deletions, insertions, inversions or translocations.

One of `PBHoney`'s algorithm, **`PBHoney-Tails`**, identifies interrupted mapping. For the best alignment of each read, `PBHoney` looks to see if the alignment is interrupted before the

ends of the read. If the ends of the read do not align over several hundred bases, soft-clipped alignments (Figure 2.6) are identified. These unaligned ends are mapped to the reference genome with `BLASR`. Then, `PBHoney` performs a clustering step and group other reads supporting the same breakpoint according to both identical genomic position and orientation. For each cluster, the different types of variants are called according to the observed alignments.

The other `PBHoney`'s algorithm, **`PBHoney-Spots`**, identifies from the alignments very erroneous genomic regions that suggest the presence of SV. These discordant "spots" are characterized by an increase in the error rate, followed by a decrease in the error rate. Thanks to the stochastic nature of the PacBio data, `PBHoney` can analyze the error rate and coverage to identify breakpoints and detect SVs.

However, no approach for integrating the two detection methods was proposed in `PBHoney` publication.

b. `Sniffles`

`Sniffles` (Sedlazeck et al., 2018b) published in 2018, has quickly become the essential tool for SV calling with long reads. To discover SVs, `Sniffles` relies on genome coverage, split read alignment, and high-mismatch regions. `Sniffles` has been proposed to discover SVs from the alignments obtained with `NGMLR`, a long read mapper that takes into account the particularity of long read data to span the SVs. Although alignments obtained with `NGMLR` are particularly suitable for `Sniffles`, other aligners such as `Minimap2` (including the MD tag, `-MD` option), can also be used to detect SVs with `Sniffles`. Unlike `PBHoney`, `Sniffles` detects SVs with both PacBio and ONT aligned reads. `Sniffles` can call several types of SVs, such as deletions, insertions, duplications, inversions, translocations or more complex SVs.

The principle of the `Sniffles` method includes four steps, which are described below. First, the method estimates several parameters to best fit the sequencing dataset (species genome) and the sequencing technology (PacBio or ONT, that do not have the same error profiles). These estimated parameters include the sequencing error rate observed of the reads, the average distance between differences (indels or mismatches) in the read alignments, and the 95th percentile of the number of mismatches and indels in a 100bp window (Sedlazeck et al., 2018b). Before searching for SV signatures, the alignments are obviously filtered to focus only on qualitative alignments.

Then the second step of the method consists of *identifying signatures of SVs* using two detection approaches carried out in parallel. On the one hand, the method analyzes alignments to identify regions enriched in mismatches and indels that would indicate the presence of SVs.

On the other hand, the method searches for split read alignments that also suggest the presence of SVs. So first, `Sniffles` method scans alignments with a PlaneSweep algorithm to identify regions of mismatches or indels of the read (of at least 30 bp by default). In parallel, `Sniffles` also searches for SV signatures by *analyzing split reads*, *i.e.* reads with segments mapping on several non-adjacent portions of the reference genome. Split reads resulting from only two alignments are analyzed to determine SV type according to the mapping chromosome, positions, and orientation. So, if the read alignments share the same chromosome and orientation, then depending on the distances on the read or the reference genome, a deletion or an insertion or a duplication can be inferred. Whereas if the read alignments share the same chromosome but align in opposite orientations, an inversion or an inverted duplication can be inferred according to the overlap. Finally, if the two alignments of the read align on different chromosomes, then a translocation is inferred.

In the third step, the `Sniffles` method tries to reconcile the two approaches to discover SV signatures, unlike the `PBHoney` method. In an attempt to combine the two approaches, the `Sniffles` method uses a self-balanced binary tree structure where SV candidates are stored as they are discovered. A node of the tree corresponds to a single SV, and nodes are sorted according to the start coordinates of each SV. At each new detected SV signature, the tree is traversed and if the SV candidate does not appear in the tree then a new leaf node is created for this SV. Otherwise, the SV candidate is merged with the existing SV in the tree if the SV type and breakpoint positions are consistent between the two calls.

Finally, SV calls are *filtered and summarized* to output only high confidence calls. Each node of the tree represents a given SV and includes several supporting reads. Nodes indicating fuzzy SV breakpoint positions are removed, and nodes gathering several SVs are split to represent a single SV. Lastly, `Sniffles` outputs the SV calls supported by a certain number of reads (by default 10). This parameter gives users the flexibility to control the trustworthiness of the output SV calls or to enable SV discovery for low sequencing depth datasets. This parameter has been shown to affect the accuracy or the recall (De Coster et al., 2019).

Once the discovery of SVs has been made, `Sniffles` proposes two optional modules for SV genotyping and SV phasing.

c. `SVIM`

`SVIM` (Heller and Vingron, 2019) is another SV caller for long reads (PacBio and ONT) that share similarities with `Sniffles`, because `SVIM`'s method also relies on searching for alignment indel regions and split read alignments. Nevertheless, `SVIM` offers an advantage over

other SV callers, by distinguishing different types of insertions, such as novel insertions, tandem duplications, and interspersed duplications. Overall, `SVIM` detects five different types of SVs, including deletions, inversions, novel element insertions, tandem duplications, and interspersed duplications. The `SVIM`'s discovery method can rely on `NGMLR` alignments as well as `Minimap2`. The `SVIM` method consists of three main steps, which are described below.

The first step of the `SVIM` method identifies signatures that would indicate the presence of SVs based on two approaches. On the one hand, the SV discovery method looks for each read for *intra-alignment signatures* such as gaps that would indicate the presence of deletions or insertions. On the other hand, the method looks for *inter-alignment signatures*, also known as split read alignments, that show inconsistencies in the relative alignment positions or orientation between the different alignments of the read. Based on this alignment information, the method captures six different signatures (see Figure 2.8): deleted regions, inserted regions, inserted regions with a detected region of origin, tandem duplicated regions, and lastly translocation breakpoints.

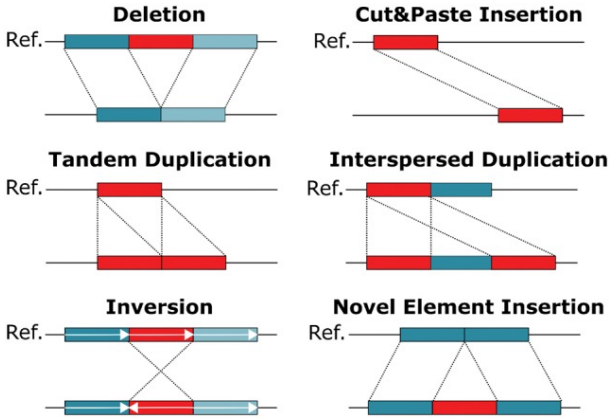

Figure 2.8 – `SVIM` signatures, figure from (Heller and Vingron, 2019)

The second step of the `SVIM` method clusters SV signatures that indicate the presence of the same variant based on a novel distance that takes into account the genomic positions and the span of the signatures. To do so, the method builds a non-oriented graph where a signature is defined as a node and an edge connects two nodes if the distance between two signatures is less than a defined threshold. Then, SV signature clusters are identified by extracting the maximum cliques, *i.e.* the maximum set of nodes that are all connected to each other. Finally, each extracted cluster, representing the same SV, is assigned a score that represents its reliability according to several criteria that support the presence of an SV.

Finally, the third step of the method combines and classifies signature clusters into types

of SV: deletion, inversion, novel insertion, tandem duplication, and interspersed duplication. In order to distinguish novel insertions from interspersed duplications, characterized by two genomic regions, the method combines several signature clusters. For instance, if an insertion signature cluster is close to a breakpoint signature cluster, then the method calls an interspersed duplication, otherwise, the method calls a novel insertion.

Once the SVs have been called, the `SVIM` method also offers an optional module to genotype the discovered SVs.

d. Main differences between `PBHoney`, `Sniffles`, and `SVIM`

SV discovery methods for long reads share several common features. `PBHoney`, `Sniffles` and `SVIM` use two main approaches to detect SV: very noisy regions within alignments and split read alignments. However, there are many differences between these methods. First of all, `PBHoney`, the oldest method that has been proposed, does not include a step of reconciliation of candidate SVs between the two detection approaches, unlike `Sniffles` and `SVIM`.

The methods of reconciliation differ between `Sniffles` and `SVIM`. `Sniffles`, on its side, relies on a self-balanced binary tree data structure that is updated as SV candidates are detected. While `SVIM` uses a graph-based clustering approach, introducing a novel metric. Other differences between these two methods are the choice of filtering or merging SV call or the calculation of the SV call score.

`Sniffles` distinguishes itself by its ability to detect nested SVs, while `SVIM` is the only SV caller to distinguish several types of insertions (novel insertions, tandem duplications, and interspersed duplications).

SV caller	SV type						Strategy	Mapper	Genotype	Sequencing tech.	
	DEL	INS	DUP	INV	TRANS	BND				PacBio	ONT
PBHoney	✓	✓		✓	✓		SPLIT, COV	BLASR	✗	✓	✗
SMRT-SV	✓	✓	✓	✓			local ASM	BLASR	SR	✓	✗
SMRT-SV2	✓	✓		✓			local ASM	BLASR	SR	✓	✗
NanoSV						✓	SPLIT	LAST	✓	✓	✓
Sniffles	✓	✓	✓	✓	✓	✓	SPLIT, COV	NGMLR, Minimap2	✓	✓	✓
SVIM	✓	✓	✓	✓			SPLIT, GAP	NGMLR, Minimap2	✓	✓	✓
pbsv [1]	✓	✓	✓	✓	✓		flanking seq. CONS.	pbmm2	✓	✓	✗
Picky	✓	✓	✓	✓	✓		SPLIT	LAST	✗	✓	✓
npINV				✓			SPLIT	Minimap2 NGMLR	✓	✗	✓
invDet				✓			COV	BLASR	✗	✓	✗
NanoVar	✓	✓		✓	✓	✓	SPLIT, COV, ANN	HS-BLASTN, Minimap2	✓	✗	✓

Table 2.1 – **SV calling tools for long read data** Several strategies are used by the SV callers: SPLIT: split read alignment, COV: coverage analysis (increase in error), local ASM: local assembly, GAP: large gap alignments, flanking seq. CONS.: a consensus sequence is generated at the flanking sequences of the breakpoints, ANN: Artificial Neural Network. Both SMRT-SV and SMRT-SV2 perform genotyping only with short read data (indicated as SR). All these tools except pbsv have been published.

2.3.2.2 SV discovery methods based on other approaches

Other methods for SV discovery with long reads have also been proposed. In this section, we present other approaches chosen by these SV discovery methods.

a. SV caller based on breakpoint aware alignment

More recently, the Pacific Biosciences company developed `pbsv` [2] to detect SVs for several types of PacBio data. The principle of the `pbsv` method differs from other SV discovery methods, with an additional step for predicting accurate breakpoint sequences which are then used for a re-alignment of the long reads against the reference. For the moment a thorough description of `pbsv` is absent from the literature.

The `pbsv` method shares several common steps with other SV discovery methods for long reads. The first step of the method is precisely to analyze the alignments of the long reads to detect signatures indicating the presence of SVs. Then, the reads pointing to the same SV signatures are clustered. But unlike other SV callers, the `pbsv` method first generates a consensus of the breakpoint sequences for each cluster and then re-aligns the long reads against the reference using a breakpoint aware aligner. Finally, the method reports SV calls for several types of SVs, such as insertions, deletions, inversions, duplications, and translocations, and can estimate the genotype of the SV calls as well as.

The `pbsv` SV caller was developed for use with a specific mapper for PacBio data, `pbmm2` [3] which consists in a `Minimap2` wrapper.

2. `https://github.com/PacificBiosciences/pbsv`
3. `https://github.com/PacificBiosciences/pbmm2`

b. SV callers specific to split read alignments

Other methods of SV discovery, such as `NanoSV` (Stancu et al., 2017) and `Picky` (Gong et al., 2018) rely only on the split read alignment identification approach. These two methods of SV discovery recommend the use of the `LAST` (Kiełbasa et al., 2011) long read alignment, which favors the generation of several short alignments for a read, rather than a single long alignment covering regions of gaps or indels. The detection methods, based on the alignments obtained with `LAST`, are more sensitive for SV discovery than with other long read mappers, such as `NGMLR` or `Minimap2`. However, `LAST` is slower and requires more computing resources than other long read mappers.

`NanoSV` (Stancu et al., 2017) uses `LAST` alignments where reads can be split in overlapping segments. The first step of the `NanoSV` method is to retrieve all mapped segments of each split read and defines a candidate breakpoint junction as two consecutive alignment segments in a read. Then, the method clusters consecutive alignment segments according to their genomic position and orientation. Clusters supported by at least T reads ($T=2$ by default) are finally reported as breakpoint junctions.

Hence, a unique feature of `NanoSV` among other SV callers is reporting only breakpoints (`BND`) rather than the location and type of SVs, in order to avoid misinterpretation according to the authors. As a consequence, `NanoSV` then leaves the interpretation of the breakpoints of the SV call, to the user, which is not an easy step. `NanoSV` was the very first SV caller developed for ONT data, but can now be adapted to PacBio data.

`Picky` (Gong et al., 2018) is much more time efficient (Zhou et al., 2019) than `NanoSV`, thanks to a prior step of selecting the best alignment segments (*'picking alignments'*). Thus, the `Picky` method first identifies high scoring segment pairs of the long reads against the reference using `LAST`. In the second step, the method generates a read alignment using a seed-and-extend approach, where the high scoring segment pairs are considered as the seeds and then linked to the other seeds with close read coordinates. Seeds can be overlapping or separated by a gap. Finally, the third step of the `Picky` method infers split read alignment from linked segments and classifies the split read into seven different types of SV, based on the alignment features such as chromosomes, orientation, and relative positions.

So, each split read reports an SV call. We notice that the authors do not describe any approach to combine the SV calls of the different reads. According to the tool's github, this

step is under development [4].

c. SV callers based on assembly

Initially, `SMRT-SV` (Chaisson et al., 2015; Huddleston et al., 2017), which was later replaced by `SMRT-SV2` (Audano et al., 2019), use a reference-guided assembly approach to detect SVs from PacBio data. First of all, their method aligns the long reads against the reference and searches for signatures of SV. Then, the reads mapping to the regions of SV signatures are assembled and polished. Finally, the assembled contigs are aligned against the reference allowing to identify breakpoints and SVs.

d. SV callers using a neural network model for SV scoring

More recently `NanoVar` (Tham et al., 2020) has been proposed to address SV detection for low depth (8x) whole-genome ONT sequencing data. Briefly, the method relies on incomplete alignments obtained with `HS-BLASTN` (Chen et al., 2015) and `Minimap2`, which are evaluated to identify SVs with a characterization algorithm, and to classify into six different types of SV. Then, the method determines a confidence score for each SV using a neural network classifier based on the read depth of SV and other SV characteristics. Only SV calls with a sufficient score are reported.

The neural network classifier was trained on simulated long-reads generated from a genome with simulated SVs. Thus, the main disadvantage of `Nanovar` is that it applies to datasets with the same characteristics as the one it was trained on. We don't know to what extent it can detect SVs in datasets with other characteristics. In case the user wants to apply it to other types of dataset, he will have to re-train the neural network model.

2.3.2.3 SV callers specific to SV types

Other methods for SV discovery have been developed that detect a single type of SV. These methods are therefore based on characteristics specific to the type of detected SV. We first describe the discovery method for inversions, and then the discovery methods for complex SV.

a. SV callers specific to inversion detection

Two methods have been designed to detect specifically inversions using long read data.

4. `https://github.com/TheJacksonLaboratory/Picky/wiki`

npInv (nanopore Inversion) (Shao et al., 2018) aims to detect non-allelic homologous recombination (NAHR) inversions for ONT data. Although there is no contraindication to apply `npInv` on PacBio data. Inversions obtained through the NAHR mechanism are characterized by two sequences that are inverted repeats on either side of the inversion (see Figure 1.7 in Chapter 1). `npInv` first retrieves read that produce multiple alignments and then identifies the inversions according to the chromosomes and the orientations of the read alignment pairs. Lastly, the inversions detected are genotyped. Initially `npInv` was designed to process `BWA` alignments, but it can call inversions from `Minimap2` and `NGMLR` alignments as well.

invDet (Zhu et al., 2018) is another SV caller that focus on detecting inversions using PacBio data. The principle of the `invDet` method consists first of all in identifying validated segments based on concordant read alignments. Next, the method builds a graph, where vertices correspond to validated segments, and edges correspond to discordant read alignments that are located in two different validated segments. The method then removes conflicts and detects inversions from one of the two sets of a bipartite graph. Today, the `invDet` tool is not the most time efficient, since it uses `BLASR` to align PacBio data against the reference.

b. SV callers specific to complex SV detection

Complex SVs are "multiple combinations of SV types nested or clustered with one another" (Ho et al., 2019), and are still a limitation that most SV callers face. Thus, SV callers have been developed to precisely detect complex SVs.

`TSD` (Meng et al., 2019) is an SV discovery tool to study complex SVs for a genomic region of interest, that use targeted PacBio sequencing data and generate highly accurate long reads. `TSD` relies on split read alignments and an assembly step to find the structural organization of the SVs.

Among the complex regions of the genome, there are segmental duplications (Seg Dup), which are long and highly similar genomic segments. By performing a haplotype assembly first, `SDip` (preprint Heller et al., 2020) is the very first SV caller to propose haplotype-aware SV calling in segmental duplication regions using PacBio HiFi reads and eventually ONT ultra-long reads. So, the `SDip` method first generates phased contigs, which are then aligned to the reference genome to detect SVs. `SDip` is actually an extension of `SVIM` (Heller and Vingron, 2019).

Note that other methods exist to study complex SV with long reads. Indeed, methods have been developed to specifically address tandems repeat variants or mobile element insertions. However, these methods do not perform discovery of variants strictly speaking, since they rely

on prior known information of these types of variations. Therefore, we do not present them in this section dedicated to SV discovery methods with long reads.

2.3.2.4 Hybrid approaches

Another strategy for SV discovery is to combine both short read and long read data. On the one hand, the long read data provide long distance information about the structure of the rearrangements, and one the other hand the short read data counterbalance the error rate of the long reads. Thus, hybrid methods for SV calling have been proposed to improve SV detection in terms of accuracy and recall (Sedlazeck et al., 2018a).

`Multibreak-SV` (Ritz et al., 2014) enables to combine multiple sequencing platforms, such as PacBio and/or paired-end data to discover SVs. Based on split read alignments, the `Multibreak-SV` method detects breakpoints and then uses a probabilistic approach to score the SV calls.

In 2017, an actual hybrid SV caller that is assembly-based was developed using both short reads and long reads: `HySA` which is Hybrid Structural variant Assembly (Fan et al., 2017). `HySA` first uses short read data to locate SV regions, then these regions are assembled using long read data and mapped to the reference to resolve the SVs. This assembly-based approach makes it easy to identify insertions but it is time-consuming (Zhu et al., 2018).

Although hybrid methods attempt to solve SVs by profiling complementary sequencing data, they require an additional cost for the user who has to perform two sequencings from different platforms.

2.3.2.5 Conclusion on SV discovery using long reads

SV discovery methods using long read data have greatly benefited from the long-distance information provided by this type of sequencing data. The main advantage of long read data is that they offer a higher read mappability than the short read data, especially for complex regions of the genome (like repeated regions or highly polymorphic regions). Thus, the majority of the SV calling methods with long reads are relying on a mapping approach. Most often the methods use at least one detection strategy, by analyzing the split read alignments, if not two, by analyzing alignments with very erroneous regions.

The difficulty of SV discovery methods for long reads lies in the method of combining the different SV signatures that indicate the same SV. Indeed, the methods seek to merge identical

SV candidates, while relying on alignments that may contain sequencing or alignment errors. It is precisely this step which consists in merging the same SV candidates, which differ between the methods of discovering SV with long reads. Also, the methodological choices for scoring and classifying SV calls differ between the methods.

Two benchmarks (De Coster et al., 2019; Zhou et al., 2019) carried out to compare the performance of SV callers using ONT data, indicate that the best performance is provided by combining the `Minimap2` mapper and then the SV caller `Sniffles`. However, higher precision is obtained when coupling the `NGMLR` mapper and `Sniffles` but at the expense of a lower recall and a slower runtime (1289 sec/100,000 reads for `NGMLR` vs. 178 sec/100,000 reads for `Minimap2`). In contrast, `SVIM` does not seem to be impacted by the choice of mapper between `NGMLR` and `Minimap2` and has a high recall. `SVIM` and `Sniffles` are the fastest and have comparable runtimes: on the human chromosome 21 `SVIM` takes 1.33-1.86 min and `Sniffles` takes 3.73 min using 12 threads (De Coster et al., 2019). However, `NanoSV`, even if it only reports breakpoints, is very slow compared to the other SV callers ($>$ 100 min).

As for short reads, studies show that combining multiple methods to discover SV can either increase accuracy or sensitivity (de Lannoy et al., 2017). Thus, a few approaches have been proposed to integrate SV calls from different approaches (Fang et al., 2018; Zhou et al., 2019). However, merging SV calls between different SV calling methods is often a very complex task.

The performance of SV detection tools varies according to the type of SV and for instance insertions, are still difficult to solve with both long reads and short reads technology (Sedlazeck et al., 2018a). In addition, although the methods can detect SVs despite their high error rate, the error rate of long reads makes it difficult to precisely resolve the breakpoints of the rearrangements. Nevertheless, an increase in the data quality of the third generation sequencing technologies can be expected in the next years, as they still have room for improvement. One disadvantage is that the recent third generation sequencing technologies are still evolving and do not ensure data stability, so bioinformatics methods must constantly adapt.

2.3.3 Conclusion on structural variant discovery

To conclude, SV discovery using short read is a complex task due to the short size of the reads and the genome complexity. Many methods of SV discovery have been developed so far, there are more than 70 of them (Kosugi et al., 2019). The precision of these methods can be controlled, but resulting in a very low recall (Mahmoud et al., 2019). To date, no method offers a reliable solution to discover SV using short reads for all SV types and all SV sizes (Mahmoud et al.,

2019). The short read sequencing data may have reached their limit for the SV discovery problem.

In recent years, several studies have observed the interest of long read data for SV discovery compared to short read data. Long reads provide a higher mappability and facilitate the resolution of SVs (De Coster and Van Broeckhoven, 2019; Mahmoud et al., 2019), by spanning over complex regions, such as repeated regions, segmental duplications, and highly polymorphic regions. Despite a high error rate, long reads provide more direct information, compared to short reads for which methods need to assemble multiple pieces of evidence to infer a single rearrangement.

Moreover, the short read and long read sequencing technologies have a different cost. For the same sequencing depth, the cost of long reads exceeds the cost of short reads (Mahmoud et al., 2019). Long read SV discovery methods, such as `Sniffles`, require a sequencing depth of approximately 15x (Sedlazeck et al., 2018a). While short read SV discovery methods required an optional sequencing depth of 30x for most types of SV with the exception of insertions. To detect and resolve insertion sequences, the optional sequencing depth required is 60x with short read data. As a result, because of their high throughput and more accessible cost, short read data remains the technology of choice in routine medical applications.

The number of SV studies has exploded with the arrival of long read data, and therefore long read data help expand the human SV catalog (Audano et al., 2019; Jain et al., 2018). Two comprehensive studies were conducted in particular to build up gold standard datasets and to evaluate methods dedicated to SVs. By combining short read and long read data as well as other types of sequencing data such as Bionano data or linked-read data, two gold-standard call sets of human genomes have been published in 2019 (Chaisson et al., 2019; Zook et al., 2020). These reliable SV calls sets obtained from multiplying data sources and detection approaches, required a massive effort of material and resources and cannot be applied regularly. But these recent studies provide essential knowledge to understand SV, and at the same time, they provide the material needed to accurately evaluate the SV discovery methods developed to date.

Therefore from increasing SV databases and from these reference datasets for the main types of SV (deletions and insertions), the next question is to estimate the genotype of these known human variants for new individuals. In the next section, we will see exactly what SV genotyping consists of, as well as SV genotyping approaches based on the second and on the third sequencing technologies.

2.4 Genotyping structural variants

The SV genotyping task consists in identifying which variants are present and which variants are absent in a given individual from a set of characterized and validated SVs. As opposed to SV discovery, SV genotyping implies to already have the description of the SV alleles to evaluate their presence. Thus the genomic position of the SV, the SV type, and the chromosome(s) are information known upstream.

Like all variants, SVs are represented by at least two alleles. Most of the time SVs are bi-allelic variants: there is the reference allele which is the reference sequence at the locus and there is the alternative allele which is the sequence rearranged at the same locus. The SV genotyping problem includes as well the estimation of which alleles, are present and in what quantity. For example, the human genome is diploid, meaning that the genome is present in two copies. Given a bi-allelic locus, consider *allele A* as the reference allele and *allele B* as the alternative allele. There are thus three configurations: i) *allele A* is present on both copies of the genome, so the genotype is homozygous for the reference allele (noted 0/0) ii) *allele A* and *allele B* are both presents in a single copy each, so the genotype is heterozygous (noted 0/1), iii) finally *allele A* is absent, it is *allele B* which is present on both copies, so the genotype is homozygous for the alternative allele (noted 1/1).

Thus, SV genotyping is a different problem from the SV discovery task. The genotyping of SV is also applied at the scale of multiple individuals, to see the distribution of alleles within populations. Several methodologies have been proposed to address this problem. There are two main strategies for the genotyping of SV. One strategy is based on the analysis of alignments obtained only against the reference genome. As for SV discovery methods, SV genotyping methods then rely on SV signatures at the breakpoints to count the number of reads that support the presence of one of the two alleles. The other strategy for SV genotyping consists of analyzing alignments obtained against both the alternative and reference allele sequences. This strategy is based on a representation of both allele sequences. Finally, the SV genotype is estimated from the relative counts of the reads supporting the presence of one of the alleles. In this section, we will see the approaches that tackle the SV genotyping problem, in a first instance using short read data, and then in a second instance using long read data.

In general, genotyping, when performed in parallel with the SV discovery, requires a reference genome and the sample of sequenced reads. The set of SV to genotyped SV is obviously obtained through the detection process. Whereas when genotyping is performed independently, the analysis requires a reference genome, then the sample of reads or the alignment of reads

(BAM) and finally the set of annotated SV (VCF) to be genotyped.

2.4.1 SV genotyping using short reads

SV genotyping methods have adopted both strategies mentioned above. Initially, the SV genotyping tools were part of a global pipeline that performs SV discovery followed by SV genotyping. Firstly, these methods rely on the interpretation of the alignments of the short reads against the reference genome, so only whether the reads align against the reference allele is evaluated. In other words, the sequence of the alternative allele is not represented in the reference genome. Secondly, these SV genotyping tools integrated to SV discovery tools, evaluate only the variants present within the set of detected SVs. So, only the heterozygous genotype (0/1) and the homozygous for the alternative allele genotype (1/1) can be estimated. Then, these SV genotyping approaches use the same detection strategies as for the SV discovery methods, such as the read depth, the paired reads and the split reads strategies. Nevertheless, some methods can take as input a set of SV (most often in VCF format), so even the homozygous genotype for the reference allele variants can be estimated. But recently, new methods have been developed based on the strategy of representing both alleles of the SVs.

In this section, we give an overview of the SV genotyping methods for short read data. First, we present the approaches that use the representation of only the reference allele, by analyzing alignments of the short reads against the reference. Second, we detail the different methodologies developed to estimate the SV genotype based on the representation of both allele sequences for each SV.

2.4.1.1 SV genotyping methods mapping-based against reference allele

The first methods to perform SV genotyping with short read sequencing data were developed along with methods dedicated to SV discovery for the same data.

DELLY. Then the SV discovery methods for short reads also integrated a module into their tool to estimate the genotype of detected SV. `DELLY` (Rausch et al., 2012) is precisely an SV caller that can perform SV genotyping. Based on split read and discordant paired-end alignments, `DELLY` compute the genotype of several types of SVs, *i.e.* deletions, duplications, inversions and translocations. But the integration of SV genotypers within the SV calling pipelines has important limitations. Firstly, genotyping is limited to the set of SVs detected

by the specific SV discovery method. Therefore, only homozygous variants for the alternative allele and heterozygous variants can be genotyped. Secondly, the genotyping module is often incompatible with standard VCF files. For instance, `DELLY` only genotypes deletions from a standardized VCF, which was obtained by another SV caller than `DELLY` (Chander et al., 2019). This is the major limitation of SV genotyping tools embedded in SV callers, the user must be limited to a particular SV caller. In addition, the genotyping method was not described in the original `DELLY` publication, as this feature was added at a later date.

`SVtyper` genotyper is part of the SpeedSeq pipeline (Chiang et al., 2015), which performs SV detection with `LUMPY` (Layer et al., 2014) prior to SV genotyping. `SVtyper` is a full-fledged SV genotyping method for short reads. To perform genotyping, `SVtyper` requires a VCF file of the set of SVs to be genotyped, as well as alignments of the short reads against the reference genome. Then, on the one hand, `SVtyper` counts discordant read alignments at the breakpoints of the SVs, such as split read alignments or inconsistent paired-end alignments. On the other hand, `SVtyper` also counts concordant read alignments, *i.e.* paired-end read alignments mapping at an expected distance and in an opposite orientation. `SVtyper` considers that concordant alignments near breakpoints support the reference allelic sequence, while discordant alignments near breakpoints support the alternative allelic sequence. Then `SVtyper` calculates the likelihood of genotypes from counts of alignments supporting each allele and finally the genotyping that maximizes the likelihood is reported. `SVtyper` can genotype most types of SVs with the exception of insertions, despite the large proportion of this type of SV in genomes. Although `SVtyper` is a method in its own right, it is part of the Speedseq pipeline and therefore requires specific information in the VCF that SpeedSeq's SV caller, `LUMPY`, provides. As for `DELLY`, the main drawback of `SVtyper` is that it requires to use specifically `LUMPY`, without this SV caller `SVtyper` is more constraining to use.

`SV2` (Antaki et al., 2018) is also a genotyping method strictly speaking for small variants and SVs using short read data. `SV2` is a machine learning based genotyping method. First, the `SV2` method preprocesses the alignment file and the set of variants, to record statistical information, such as median coverage, insert size, and read length. Then, the method extracts for each SV four informative features for genotyping, which are the depth of coverage, discordant paired-end reads, split-reads, and heterozygous allele ratio (HAR). Finally, the `SV2` method uses an ensemble of support vector machine classifiers, to genotype each SV according to its category (SV type, SV size, male sex chromosome). Classifiers were trained on short read data

from the 1000 Genome Project. SV2 estimates genotypes only for deletions and duplications.

2.4.1.2 SV genotyping methods representing allele variants

SV genotyping methods that rely on previously obtained alignments of sequence data against the reference genome are biased towards the reference allele. This is because the reads are only aligned to the reference allele sequences. The alternative allele sequences are not represented in the reference genome. Thus methods relying solely on the sequences of the reference alleles are exposed to a bias of representation in favor of the reference allele. In this section we describe the SV genotyping methods that represents both allele sequences.

a. Linear representation of the alternative allele

`Genome StRiP` (Genome STRucture In Populations) proposes a framework that includes SV genotyping at the population level (Handsaker et al., 2011). `Genome StRiP` was developed along with the 1000 Genomes Project, it enables SV genotyping of cohorts simultaneously. `Genome StRiP` genotyping module aligns the unmapped reads and the already aligned reads at breakpoints against a library containing the alternative allele sequences. Then based on three different sources of evidence, which are the read depth, discordant read pair, and breakpoint-spanning reads, `Genome StRiP` estimates the likelihood for each possible genotype. Finally, the estimated genotype is the one that maximizes the combination of these likelihoods.

`Genome StRiP` is limited to SV genotyping of deletions and duplications.

`SMRT-SV2` (Audano et al., 2019) not only includes a method for SV discovery using long read, but also a method to genotype SV with short reads. `SMRT-SV genotyper` is a machine learning based method that represents both sequence alleles for each SV, therefore it avoids bias towards the reference allele.

First, the `SMRT-SV genotyper` method builds a reference genome representing the alternative allele. To do this, the genotyping method needs to process a set of SVs obtained by the `SMRT-SV2` caller or from a similar VCF file, which must contain information about the contigs where the SVs are located (primary contigs). The alternative sequence of SV is added and locally assembled to the primary contigs. Then, the genotyping method maps all short reads against the reference genome containing the alternative sequences, using `BWA-MEM` (Li and Durbin, 2009). Reads mapping to the reference alleles are extracted from the reference

genome alignment file. Thus, the reads associated with SVs, either map to the reference allele or to the alternative allele. Finally, the genotyping method uses a machine learning model to estimate the genotype for each SV based on 15 features extracted, such as the SV type, the SV size, and alignment features like paired-read, split read, read depth features.

`SMRT-SV genotyper` can estimate the genotype for all types of genotypes, *i.e.* homozygous variants for the reference allele, homozygous variants for the alternative allele, and heterozygous variants. However, remapping all the reads to the alternative reference is very expensive for the method of `SMRT-SV genotyper`.

b. Mapping-free method

`Nebula` (Khorsand and Hormozdiari, 2019) (preprint) is a mapping-free method for genotyping SV using short read data. The principle of the method is based on the changes in the k-mers profile between the two alleles of a variant. Firstly, the `Nebula` method selects from the set of SVs, k-mers with different frequencies between the two alleles of an SV. Secondly, from the set of selected k-mers, the genotyping method counts k-mers in the short read datasets. Then, the method models the SV genotyping as an optimization problem, to find the genotypes of the set of SVs that minimize the total number of differences between the observed k-mer counts from the sequence datasets and the expected k-mer counts for the alleles of the set of SVs. The method solves the optimization problem with a linear program, then applies a rounding to estimate the SV genotypes.

c. Graph representation of the alleles

Very recently, novel SV genotyping methods for short read data have been proposed based on variation graphs. "Variation graphs are bidirected DNA sequence graphs that compactly represent genetic variation across a population, including large-scale structural variation [...]" (Garrison et al., 2018). The main advantage of this representation is that it more accurately reflects the variability of the sequences compared to a linear representation of the sequences of the reference genome. Variation graphs require specific methods dedicated to this representation to build the graph, to align the sequences on the graph, or to genotype the variations on the graph.

`BayesTyper` (Sibbesen et al., 2018) is a genotyping method based on exact k-mer matching and uses a graph representation of the variants. The method can genotype SNPs, indels and SVs. First, the method clusters close variants (less than k - 1 nucleotides apart, $k = 55$ by

default), and constructs a graph for each cluster. The graph nodes represent the allelic sequences, and the edges correspond to connections between the node sequences. Then, the `BayesTyper` method measures the k-mer counts in the short read dataset. Paths in the graph correspond to the possible haplotypes. The method then, compares the k-mer profile of the most likely haplotypes, to the k-mer profile of the sequencing dataset. Finally, the variant genotypes are inferred using a probabilistic model.

`Paragraph` (Chen et al., 2019) is a variation graph based method to genotype SV with short reads at a population scale. The `Paragraph` method, first builds a graph for each SV breakpoint, where a node represents a sequence and an edge corresponds to a breakpoint connecting two sequences. Therefore each graph has a path for the reference allele and a path corresponding to the alternative allele. Then, the method extracts from BAM alignment files the reads mapping near SV regions and realign these extracted reads on the graphs. The `Paragraph` method uses filtering to retain only the best graph alignment reads that discriminate between the two alleles. To estimate the genotype of each breakpoint, the method uses a genotype likelihood maximization approach based on read counts of each allele. Finally, the `Paragraph` method assess the reliability and then the consistency of each combined breakpoint genotypes, and finally reports the SV genotypes.

`GraphTyper2` (Eggertsson et al., 2019) is a genome graph based approach for SV and small variants genotyping using short reads at a population scale. The method first builds a directed acyclic graph from the reference genome, the SNP and indel sites and the set of SVs, representing the genome graph (also known as the pangenome graph). Then, the principle of the `GraphTyper2` method is relatively similar to `Paragraph`. The method realigns all reads to the genome graph. The method uses two models based on the read realignements to the SV breakpoints and on the alignment coverage, to estimate the SV genotype. Finally, to report the SV genotype for each sequencing sample, `GraphTyper2` also relies on a likelihood maximizing approach of the read counts aligned on sequence alleles.

`GraphTyper2` and `Paragraph` uses two different representations of variations. On one side `GraphTyper2` uses a pangenome graph representation, and on the other side `Paragraph` uses a variation graph representation. We can then assume that `Paragraph` will be faster than `GraphTyper2` which builds graphs and aligns reads only on specific regions of the genome.

`VG-toolkit` (Hickey et al., 2020) also relies on a pangenome graph representation to genotype

SV. First, `VG-toolkit` builds a bidirected genome graph and map all reads against it. The read support is computed for each node and edge from the alignments. `VG-toolkit` identifies variation sites corresponding to intervals along the reference paths and are referred to as snarls. For each snarl, based on the average read support of each path, the two most supported paths are selected, so each selected path represents a haplotype. Finally, using the read support of each haplotype, the genotype is estimated according to the following rules. `VG-toolkit` calls a homozygous genotype if the most supported path exceeds the minimum support threshold and whose read support exceeds by at least n the second most supported path. Otherwise, a heterozygous genotype is called if the second most supported path exceeds the minimum support threshold.

2.4.1.3 Conclusion on SV genotyping methods for short reads

Unfortunately, most SV genotypers are not entirely satisfactory and show significant drawbacks. On the one hand, some tools do not allow the genotyping of the majority of SVs and on the other hand, some tools are limited only to the detection of heterozygous and homozygous variants for the alternative allele in the sample. Finally, some implementation choices restrict the applications of SV genotypers.

Most SV genotyping methods based solely on the representation of the reference allele are not capable of estimating the genotype of insertions. However, these insertions represent a significant proportion of all SV in the genome. A benchmark study (Chander et al., 2019) indicates that `SVtyper` shows the highest rate of correctly genotyped SVs on a simulated dataset.

Methods that take into account the representation of both the reference allele and the alternative allele can genotype the majority of SVs, especially insertions. However, these methods are very recent and the graph-based representation is of particular interest since 2018. However, to date no comparative analysis has been published, we, therefore, have no hindsight on these graph-based genotyping methods for short read data.

Knowing the genomic positions of the sequences of the SVs, allows the SV genotyping to be more specific than for the SV discovery, using more stringent parameters. However, short read genotyping methods rely on read alignments and are therefore also subject to the problem of short read mappability to repetitive sequences, which often surround the SVs. Since long reads are less subject to this problem, we detail in the following section the genotyping methods dedicated to long reads.

2.4.2 SV genotyping with long reads

Just as short read SV calling methods offer a module for genotyping, long read SV calling methods also offer a module for genotyping. Long read SV callers such as `Sniffles`, `SVIM` or `pbsv` can estimate genotype of the previously called SVs. Thus, SV genotyping is again limited to heterozygous SVs and homozygous SVs for the alternative allele, and fail to assess the genotype of the homozygous for the reference allele variants. Furthermore, the SV genotyping is also specific to each SV caller and therefore limits their use. In addition, De Coster et al. (2019) disclosed SV callers often fail to assign correct genotypes. For instance, Sniffles misclassified heterozygous SVs as homozygous for the alternative allele. To date, no SV genotyping method independent of SV discovery methods, has been published. Nevertheless there are two tools on github, to our knowledge, capable of performing SV genotyping with long read data from a standardized VCF. We detail them in the following paragraphs.

`Sniffles` proposes a specific module `-Ivcf` [5] that performs actual SV genotyping when provided with a set of SVs to assess their genotype. However, this genotype module was not described in Sedlazeck et al. (2018b), only the principle of genotyping from the set discovered by the SV caller Sniffles.

During alignment scanning, `Sniffles` also records reads that do not support SV at the same location. Then, according to the number of reads that support or that do not support an SV at the same location, `Sniffles` computes the allele frequency of the SVs and finally estimate the SV genotype. To summarize, the `-Ivcf` module represents only the reference alleles; the `Sniffles -Ivcf` genotyper is thus exposed to a bias of representation in favor of the reference alleles.

`svviz2` (Spies et al., 2015) is another implemented tool that performs SV genotype estimation from long read data as well. Although `svviz2` [6] is primarily intended to provide a tool for evaluating and visualizing SVs, but it includes a genotype estimation step.

First `svviz2` retrieve all read mapping near the breakpoints of the studied SVs. Then, the principle of `svviz2` is to realign these selected long reads to both the reference and the alternative alleles, so that each read either support one allele or the other for each SV or is labelled as ambiguous. Finally, `svviz2` uses a likelihood model to estimate genotype, from

5. `https://github.com/fritzsedlazeck/Sniffles/wiki/SV-calling-for-a-population`
6. `https://svviz2.readthedocs.io/en/latest/`

the read counts supporting each allele.

To reduce the set of reads to be aligned and thus be faster, svviz2 estimates the SV genotyping from reads previously aligned on the reference genome and therefore also exposes itself to a bias in favor of the reference alleles.

2.4.3 Conclusion on SV genotyping

The field of research around the question of SV genotyping is very recent. The very first publications devoted specifically to SV genotyping methods were published only from 2018 onwards. It can be seen that there are far fewer approaches developed today for the problem of SV genotyping than for the problem of SV discovery. Whereas SV genotyping has a major interest especially in clinical diagnosis, but also to know the allele distribution within the population as much for applications in human health, ecology, or evolution. For example, the SV genotype is very useful for diagnosis, it is preferred that genotyping tools can respond to it accurately and rapidly.

Table 2.2 shows the main characteristics of state of the art SV genotyping tools. Following the description of the methods, there are two possible strategies for estimating the genotype of SVs: either the methods are based on the genomic representation of only one of the alleles (reference allele), or the methods are based on the genomic representation of both alleles (reference allele and alternative allele). The second strategy appears to be much more effective in genotyping all types of SV, especially insertions. In fact, no SV genotyping method that relies solely on the representation of the sequences of the reference alleles can be used to estimate the genotype of insertions.

Even though several methods have been developed to address this problem with short reads, it has been seen that they have several drawbacks. They lack versatility to perform SV genotyping. But the problem of SV genotyping remains a problem of interest, illustrated by the recent number of publications proposing methods to address it. On the contrary, very few methods exist to perform SV genotyping with long read data, despite the fact that this type of data has proven to be effective for the study of SV.

Genotyper	Data	SV types					Strategy	Mapping	All. repres.	Desc. Lit.
DELLY	SR	DEL	-	INV	DUP	TRANS/BND	Linear Ref	All reads	REF	No
SVtyper	SR	DEL	-	INV	DUP	TRANS/BND	Linear Ref	All reads	REF	2015
SV2	SR	DEL	-	-	DUP	-	SVM classifiers	All reads	REF	2018
Genome StRiP	SR	DEL	-	-	DUP	-	Remapping	Target&Unmap	REF&ALT	2011
SMRT-SV2	SR	DEL	INS	-	-	-	SVM classifier	All reads	REF&ALT	2019
Nebula	SR	DEL	INS	INV	-	-	*k*mer counts	All reads	REF|ALT	Preprint
BayesTyper	SR	DEL	INS	INV	DUP	-	VG for close SVs	All reads	REF|ALT	2018
Paragraph	SR	DEL	INS	INV	DUP	-	VG for each SV	Target reads	REF|ALT	2019
Graphtyper2	SR	DEL	INS	INV	DUP	TRANS/BND	Genome graph	Target reads	REF|ALT	2019
VG-toolkit	SR	DEL	INS	INV	-	-	Genome graph	All reads	REF|ALT	2020
svviz2	LR	DEL	INS	INV	DUP	TRANS/BND	Linear Ref	All reads	REF|ALT	No
Sniffles -Ivcf	LR	DEL	INS	INV	DUP	TRANS/BND	Linear Ref	All reads	REF	No

Table 2.2 – SV genotyping methods for short read (SR) and long read (LR) sequencing data. The types of genotyped SV are indicated in columns, "-" means that the current tool does not allow genotyping this type of SV. *VG*: variation graph. *Target reads*: reads selected from a genomic region. *All. repres.*: allele representation, this column indicates if reads are mapped only on the reference allele (REF), or on the reference allele and then on the alternative allele (REF&ALT), or mapped considering both alleles simultaneously (REF|ALT). *Desc. Lit.*: the method has been described in a publication, if so the year of publication is indicated.

2.5 Where the thesis stands in relation to the state of the art

SVs have been attracting a lot of interest since the development of NGS technologies. However, as shown by the numerous methods of SV discovery, none of them are unanimously recognized in terms of accuracy and recall.

But since the advent of long read technologies, the field of SVs is growing rapidly, thanks to the long distance information of the sequencing data which is very useful in the context of genomic rearrangements. During the last few years several methods of SV discovery have been developed, such as `Sniffles`, `SVIM`, `pbsv`, etc., illustrating the competitiveness of this domain. With the development of all these methods, Studies of SVs with long read data follow one another today.

At the end of 2019, two major studies (Chaisson et al., 2019; Zook et al., 2020), to propose each a set of high-confidence set of SV for human individuals. These studies build from multiple sequencing technologies and multiple discovery approaches can be referred to as gold-standard call sets, which are priceless for assessing the performance of SV tools. These brand new catalogs also represent valuable resources to address the problem of genotyping.

Long reads have already proven their efficiency for the detection of SVs by offering a better mappability than the short reads. Despite the interest of long reads in the context of SV, to date no method has been described and developed for genotyping SV with long read data. Nevertheless, two tools have been found that has this functionality, but have not been described specifically for this context: `Sniffles -Ivcf` and `svviz2`.

During this thesis, I therefore focused on the question of genotyping SVs with long reads. I developped a brand new method, described and validated in the next chapters, to estimate the SV genotype with long read data.

Chapter 3

A SV genotyping approach using long reads

Table of contents

3.1 Introduction 87

3.2 Representative allele sequence generation 88

3.3 Mapping 93

3.4 Informative alignment selection 94

3.4.1 Breakpoint overlap 94

3.4.2 Filtering out false positives due to repeated sequences 95

3.5 Genotype estimation 96

3.5.1 Allele number normalization for unbalanced variants 96

3.5.2 Minimum allele number threshold for genotyping 97

3.5.3 Genotyping decision 97

3.5.3.1 Method based on decision thresholds 97

3.5.3.2 Maximum likelihood method 98

3.6 Parameter estimation 99

3.6.1 Material 99

3.6.1.1 Simulated dataset 99

3.6.1.2 Evaluation 99

3.6.2 Results according to the genotyping estimation model 100

3.6.2.1 Genotype estimation methods 100

a. Method based on thresholds: Ranges estimation 100

b. Maximum likelihood method: Estimation of the sequencing error value 101

c. Comparison of `SVJedi` genotyping methods 102

3.6.3 Estimation of the size of the representative allele sequences 103

3.7 Implementation and availability 104
3.7.1 Requirements for SV definition in input VCF file 105
3.8 Conclusion 109

3.1 Introduction

Given a set of SVs, the genotyping task seeks to evaluate for each SV, whether or not the alleles present in a sequenced individual and in the case of a diploid organism, if a particular allele is present on both haplotypes or a single haplotype only. The state of the art, in the previous chapter, describes a lack of dedicated SV genotyping method for long reads. Indeed most methods are designed for short reads. However, long read data, despite their error rate, have been shown to be particularly useful for SV discovery. In the previous chapter we noted that there are two main approaches for genotyping SVs. One possible strategy is to analyze breakpoint alignments of the sequencing data against the reference genome to estimate the genotype, so the alternative allele is not explicitly represented in the reference genome. The other strategy is to represent the reference allele and the alternative allele, and derive the genotype from the alignments against these alleles.

Therefore, we propose a novel method that assigns a genotype for a set of already known SVs in a given individual sample sequenced with long read data, that relies on representation of both the reference and the alternative alleles. For each SV, our method evaluates if it is present in the given individual, and if so, how many variant alleles it holds, *i.e.* whether the individual is heterozygous or homozygous for the reference or the alternative allele.

Our method processes the major SV types, which are deletions, insertions, inversions, and translocations. Notice that deletions and insertions are very similar variations and can be considered symmetrical (Chapter 2 Figure 1.2). Whereas inversions and translocations (Chapter 2 Figure 1.3 and Figure 1.4) differ from unbalanced SVs since they involve no material gain or loss, they are balanced SVs, and also differ in the number of breakpoints to be examined.

The method takes as input a variant file with SV coordinates in VCF format, a reference genome, and the sample of long read sequences. It outputs a variant file complemented with the individual genotype information for each input variant in VCF format. The method consists of four different steps, that are illustrated in Figure 3.1. The fundamentals of the method lie in its first step, which generates *representative allele sequences* that represent the two alleles of each SV. Long reads are then aligned on the whole set of representative allele sequences. An important step consists of selecting and counting only informative alignments to finally estimate the genotype for each input variant.

In this chapter, we will first detail the steps of our method, justifying in parallel the taken methodological choices. Then, we will present the results of simulation experiments on which we have set the parameters of our method. Finally, we describe SVJedi, an implementation of our SV genotyping method with long reads, coded in Python.

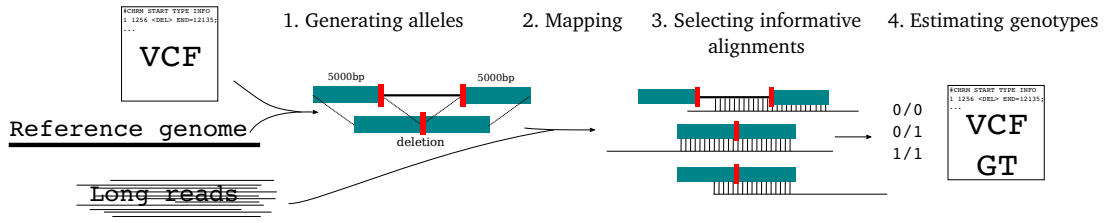

Figure 3.1 – Illustration of the 4 `SVJedi` steps for genotyping deletions. 1. Two corresponding representative allele sequences are generated for each selected SV, one corresponds to the original sequence and the other to the sequence with the deletion. 2. Long read sequenced data are aligned on these allele sequences using `Minimap2`. 3. Informative alignments are selected. 4. Genotypes are estimated.

3.2 Representative allele sequence generation

The first step of our method is to represent the sequences of the alleles. We chose to represent each variant by partial genomic sequences. These partial genomic sequences include the sequence of the variant as well as its adjacent sequences. So each SV is treated independently from the other SVs. Moreover, this avoids having to align all the reads of the given sequencing sample on the whole genome. Instead, we choose to focus only on the sequences of interest, *i.e.* the SVs and their breakpoints. In addition, we aim to represent both sequences of the biallelic variants in order to avoid a bias in favor of the reference allele. However, if we choose to map all the reads against a linear reference genome, so some long reads can overlap closely located variants. The difficulty would then be to represent the sequenced alleles for closed variants at the same time. But the possible number of allele sequences for different variants is exponential. One solution would be to reconstruct the haplotypes, but this task requires bioinformatics processing that is more expensive than genotype estimation. We have therefore chosen to represent each variant by partial genome sequences and independently of the other SVs.

Starting from a known variant file in VCF format and the corresponding reference genome, the first step consists of generating two sequences for each SV, corresponding to the two possible allele sequences. These representative allele sequences are hereafter simply called *allele sequences*. We describe each SV type with a reference allele sequence (allele 0) and an alternative allele sequence (allele 1). These representative allele sequences also contain sequences adjacent to the SV, in order to use the genomic context of the variant to retrieve the allele sequence reads from the sequencing dataset.

For all types of SV, the representative allele sequences contain the sequence of one of the two alleles of the SV or its breakpoint (for unbalanced SVs) as well as L_{adj} bp of adjacent sequence on each side of the variant or the breakpoint. The L_{adj} variable is set by default to

5,000 bp, and corresponds approximately to half the average size of a long read. Thus, the size of the representative allele sequences are $2\times L_{adj}$ to $2\times L_{adj}$ plus the size of the variation. To avoid a bias toward the largest SVs, two representative allele sequences of $2\times L_{adj}$ bp, are generated for each breakpoint of the allele, if the SV size is greater than $2\times L_{adj}$ bp.

We then detail the representative allele sequences generated for each type of SV.

A deletion corresponds to a genomic segment that may be absent in a given individual compared to the reference genome. In the VCF file, deletions are characterized by a starting position on the reference genome and a length. We define the reference allele sequence (allele 0) as the sequence of the deletion with adjacent sequences at each side, and the alternative allele sequence (allele 1) consists of the joining of the two previous adjacent sequences.

Depending on the size of the deletion there are two ways to represent the allele sequences. In the case the deletion is less than or equal to $2\times L_{adj}$, each allele is represented by one sequence, illustrated in Figure 3.2 (left box). Given the size of the adjacent sequences, L_{adj}, allele 1 is a $2\times L_{adj}$ bp sequence size and allele 0 has a sequence size of $2\times L_{adj}$ bp plus the deletion sequence size. Conversely, in the case the deletion is larger than $2\times L_{adj}$, then two representative sequences are generated for allele 0, one for each breakpoint and a single sequence represents allele 1 Figure 3.2 (right box).

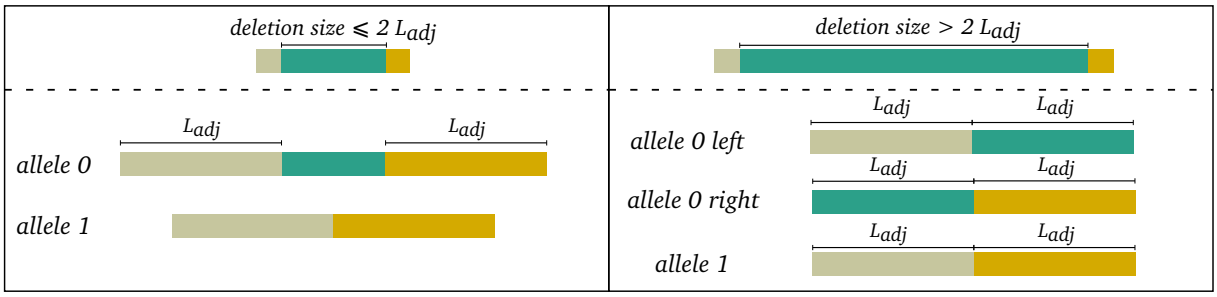

Figure 3.2 – **Generation of representative allele sequences of deletions.** If the deletion is smaller or equal to $2\times L_{adj}$ (left box), one sequence per allele are generated: *allele 0*, the reference allele, contains the deletion sequence (green segment) with L_{adj} bp of adjacent sequences at each side of the deletion and for the *allele 1*, the alternative allele, the deletion sequence is removed such that the two segments adjacent to the deletion are joined. If the deletion is greater than $2\times L_{adj}$ (right box), three sequences are generated, including one sequence for *allele 1* and two sequences for allele 0 representing each end of the deletion, *allele 0 left* and *allele 0 right*. Only L_{adj} bp of each end of the deletion sequence are represented in *allele 0 left* and *allele 0 right*.

An insertion consists of an inserted segment in a given individual compared to the reference genome. Insertions are variations which are symmetrical to deletions, with one breakpoint on the reference allele, this is the anchor point of the insertion sequence, and two breakpoints on the alternative allele corresponding to the ends of the insertion sequence. In VCF files, insertions are defined by a sequence and a start position that corresponds to the breakpoint on the reference genome. We defined allele 0, the reference allele, as the $2 \times L_{adj}$ bp adjacent sequences centered on the breakpoint position of the reference genome. Similarly to allele sequences of deletions, one or two sequences are generated to represent allele 1, whether the insertion size is greater or not than $2 \times L_{adj}$, see Figure 3.3.

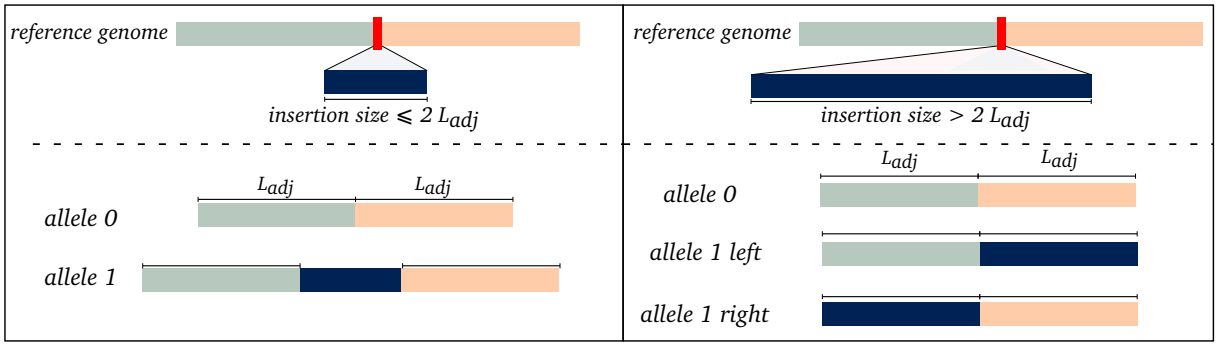

Figure 3.3 – **Generation of representative allele sequences of insertions**. If the insertion size is less or equal to $2 \times L_{adj}$ bp (left box), two sequences representing each allele are defined. Allele 0 is defined as L_{adj} bp adjacent sequence preceding and L_{adj} bp adjacent sequence following the breakpoint (red thick mark) on the reference genome. Allele 1 is defined as L_{adj} bp adjacent sequence preceding the breakpoint, followed by the insertion sequence and L_{adj} bp adjacent sequence following the breakpoint. If the insertion size is greater than $2 \times L_{adj}$ bp (right box), three sequences are defined including one sequence of allele 0 and two sequences for allele 1 representing both insertion breakpoints. Allele 0 is identically defined as if the insertion is smaller or equal to $2 \times L_{adj}$ bp. Allele 1 left breakpoint is defined as L_{adj} bp of adjacent sequence preceding the breakpoint and L_{adj} bp of the insertion left end. Allele 1 right breakpoint is defined as L_{adj} bp of the insertion right end and L_{adj} bp of the adjacent sequence following the breakpoint.

z

Inversions are balanced SVs where a genomic segment has been inverted in the same position. Inversions have then two breakpoints in both the reference allele and the alternative allele, considering the forward strand as a reference, we differentiate the breakpoints as left and right. Because of the 5' end and 3' end of the sequence at the breakpoints, it is the reverse complement sequence of the inversion that is juxtaposed to the forward strand of the sequences adjacent to the inversion. In VCF files, inversions are defined by a start position and an end position.

As for deletions and insertions, each allele is represented by one sequence or two sequences depending on the inversion size. If the size of the inversion is less than or equal to $2 \times L_{adj}$ bp, Figure 3.4 (left box), then each allele is defined by its sequence which consists of the inversion sequence plus L_{adj} bp of adjacent sequence at each side of the inversion. Thus, allele 0 is defined by the forward strand of the inversion sequence, bounded by the forward strand of the two sequences adjacent to the inversion. Conversely, allele 1 is defined by the reverse complement of the inversion sequence, bounded by the forward strand of the two sequences adjacent to the inversion sequence.

If, however, the inversion size is greater than $2 \times L_{adj}$ bp, then the two alleles are represented each by two sequences at both breakpoints, resulting in four representative allele sequences of $2 \times L_{adj}$ bp long (see Figure 3.4 left box)).

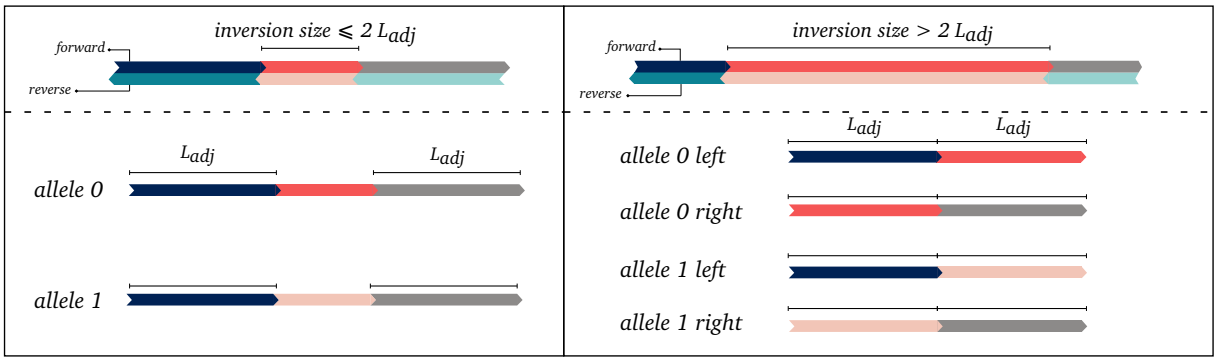

Figure 3.4 – **Generation of representative allele sequences of inversions.** The forward and reverse strands of genomics are differentiated by colors. The inversion sequence is represented in red (*forward* strand) and light pink (*reverse* strand).
Left box: If the inversion size is smaller than or equal to $2 \times L_{adj}$ bp, the two alleles are represented by a sequence each. Allele 0 is defined by the forward strand of the inversion sequence with L_{adj} bp sequence at both sides of the inversion sequence. The sequence of allele 1 is identical to the one of allele 0 with however the forward sequence of the inversion (red) replaced by the reverse complement sequence of the inversion (light pink).
Right box: If the inversion size is greater than $2 \times L_{adj}$ bp, four alleles are represented, illustrating the two alleles at the two breakpoints. Allele 0 left breakpoint is defined by L_{adj} bp of forward strand adjacent sequence preceding the left breakpoint and L_{adj} bp of forward strand inversion sequence following the left breakpoint. Allele 0 right breakpoint is defined by L_{adj} bp of forward strand inversion sequence preceding the right breakpoint and L_{adj} bp of forward strand adjacent sequence following the right breakpoint. Allele 1 left breakpoint is defined by L_{adj} bp of forward strand adjacent sequence preceding the left breakpoint and L_{adj} bp reverse complement inversion sequence preceding the right breakpoint. Allele 1 right breakpoint is defined by L_{adj} bp of reverse complement inversion sequence following the left breakpoint and L_{adj} bp of forward strand adjacent sequence following the right breakpoint.

Translocations are rearrangements between two non-homologous chromosomes. We consider in our method, a reciprocal translocation as two distinct events which are described separately in the VCF file. Depending on the approach, translocations can have different definitions and VCF representations. However, they can be represented more generally by breakpoints where rearrangements take place. Precisely, the VCF format, represents a reciprocal translocation as two separate breakpoints, also referred as `BND` type. Each breakpoint specifies the two joined regions and the type of the junction, so only one locus is considered for the representative allele sequences.

We define *chr A* as the chromosome of the observed locus and *pos A* the breakpoint position on *chr A*, while *chr B* and *pos B* are defined as the second chromosome and breakpoint position respectively, that recombines with *chr A*. Because of the structure of the DNA only a 3' end can be linked to a 5' end. Thus, there are only two possible translocations between two chromosomes and thus four possible recombination points, which are illustrated in Figure 3.5 as `BND 1`, `BND 2`, `BND 3`, `BND 4`. Each of the two chromosomes in the translocation has a breakpoint, illustrated in Figure 3.5 by different colors (dark blue/light blue and orange/yellow). The forward and reverse strands are also illustrated in different colors. The four possible `BND` are as follows:

- `BND 1`: *chr A* left end is joined with the right end of *chr B*.
- `BND 2`: *chr B* left end is joined with the right end of *chr A*.
- `BND 3`: *chr A* left end is joined to the reverse complement left end of *chr B*.
- `BND 4`: The reverse complement of the right end of *chr B* is joined with the right end of *chr A*.

Unlike other types of SVs, there are always two representative allelic sequences for a breakpoint, *i.e.* one sequence for each allele. Both sequences have a size of $2 \times L_{adj}$ bp.

Hence, as illustrated in Figure 3.5, allele 0 is defined by L_{adj} bp of forward sequence preceding the breakpoint on *chr A* and L_{adj} bp of forward sequence following the breakpoint on *chr A*. Then, allele 1 of `BND 1` is defined by the concatenation of L_{adj} bp of forward sequence preceding the breakpoint on *chr A* and of L_{adj} bp of forward sequence following the breakpoint on *chr B*. Allele 1 representing `BND 2`, is defined by L_{adj} bp of forward sequence preceding the breakpoint on *chr B* and L_{adj} bp of forward sequence following the breakpoint on *chr A*. The other two BNDs are described in the Figure 3.5.

Variants of type BND describe a new adjacency to a given locus. This type of variant can be generalized to describe other types of SVs such as intra-chromosomal translocations of very distant regions or transpositions, the movement of a genomic segment.

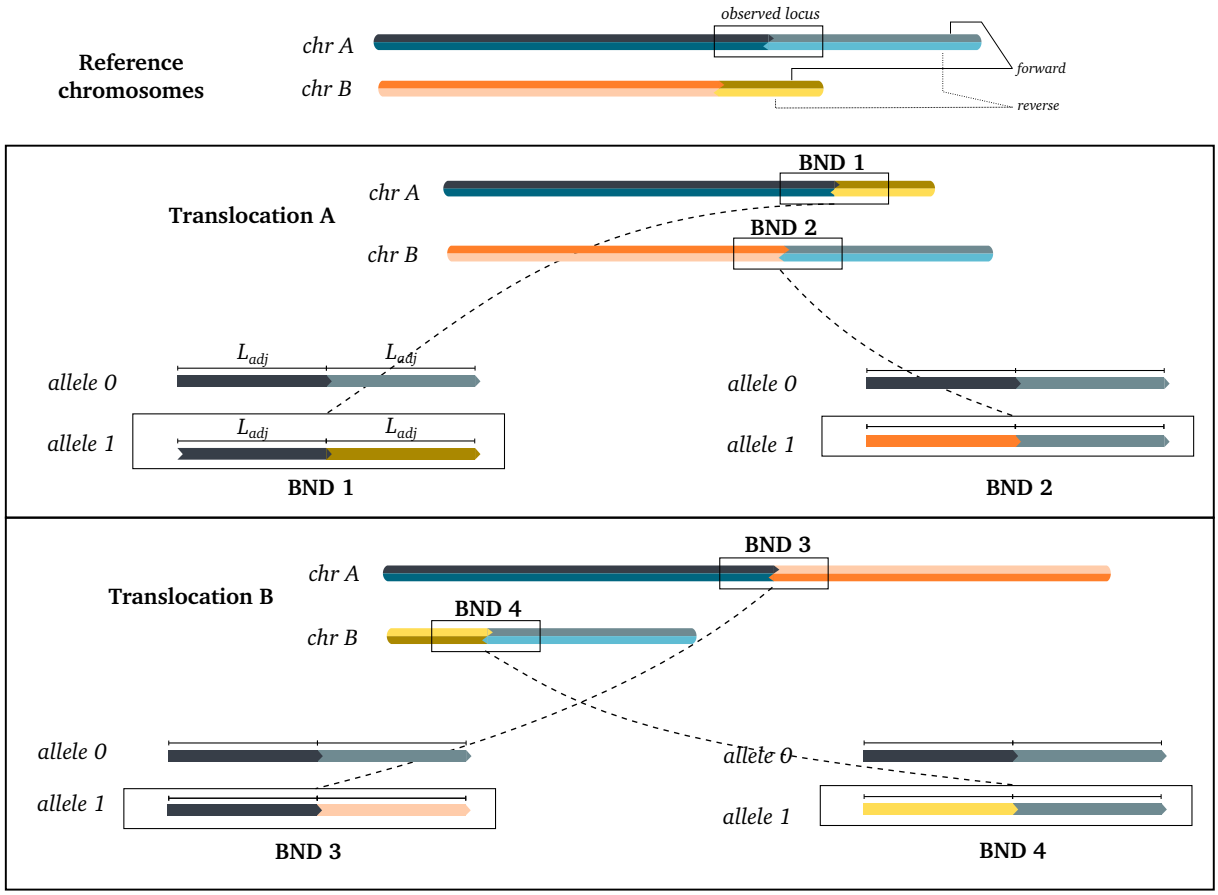

Figure 3.5 – **Generation of representative allele sequences of translocations**. The complementary strands of the two chromosomes A (blue segments) and B (orange-yellow segments) are represented by distinct colors, as well as the two colors on each strand that identify the breakpoints where the chromosomes recombine. Only the allelic sequences of the observed locus are represented, in this example, it is the framed locus on *chr A*. There are two types of translocations possible between *chr A* and *chr B*, translocations A and B. In all cases, the sequence of the reference allele, allele 0, remains identical and is represented by a $2 \times L_{adj}$ bp sequence centered on the breakpoint of *chr A* forward strand. The alternative allele, allele 1, is represented by four sequences corresponding to four possible `BND`, each sequence with a length of $2 \times L_{adj}$ centered on the breakpoint as well. In `BND 1`, the left segment of *chr A* (dark blue segment) is rearranged with right segment *chr B* (yellow segment). In `BND 2`, the left segment of *chr A* is rearranged with the reverse complement of the left segment of *chr B* (orange segment). In `BND 3`, the left segment of *chr B* is rearranged with the right segment of *chr A* (light blue segment). In `BND 4`, the reverse complete of the right segment of *chr B* is rearranged with the right segment of *chr A*.

3.3 Mapping

In the second step of the method, sequenced long reads are aligned on all previously generated allele sequences, using `Minimap2` (Li, 2018) (version 2.17-r941). All alignments are

done in one run. The main interest of `Minimap2` compared to other long read mapping tools is that it outperforms them in speed by an order of magnitude, while reaching a similar accuracy. In addition, the principle of the `Minimap2` method takes into account that some long reads can overlap SVs, as its local alignment uses a concave gap score model. For a given long read which shares similarities at several locations in the genome, several alignments of the reads are output by `Minimap2`. Besides, `Minimap2` proposes another output alignment format than SAM, it is the PAF alignment format which is more intuitive and easy to parse. Option `-c` is specified to generate a `CIGAR` string for each alignment, this option performs a base-level alignment, rather than a approximate mapping.

3.4 Informative alignment selection

Once the long reads have been aligned on the allele sequences, `Minimap2` raw alignment results have to be carefully filtered out to remove i) uninformative alignments, which are those not discriminating between the two possible alleles, and ii) spurious false positive alignments, that are mainly due to repeated sequences in the genome. The following two sections describe the conditions necessary to consider an alignment as informative.

3.4.1 Breakpoint overlap

Regarding the genotyping problem, informative alignments are those that overlap the SV breakpoints, which are the sequence adjacencies that are specific to one or the other allele. In the case of a deletion, the reference allele contains two such breakpoints, the start and end positions of the deletion sequence; the alternative sequence, the shortest one, contains one such breakpoint at the junction of the two adjacent sequences (see the red thick marks of Figure 3.1). This condition is also tested for the other types of SV.

To be considered as overlapping a breakpoint, an alignment must cover at least d_{over} bp from each side of the breakpoint (d_{over} is set by default to 100 bp). In other words, if x and y are the sizes of the aligned parts on the allele sequence at the respectively left and right sides of the breakpoint (see Figure 3.6), they must satisfy the following condition in Equation (3.1) for the alignment to be kept :

$$(x > d_{over}) \ \& \ (y > d_{over}) \tag{3.1}$$

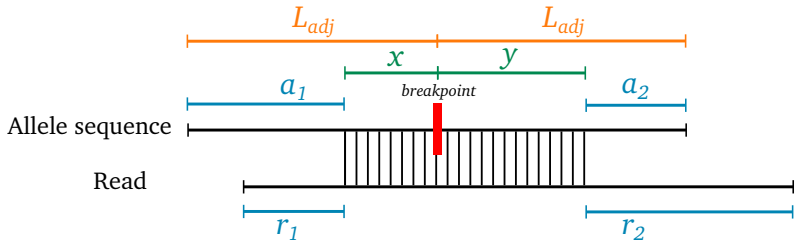

Figure 3.6 – Definition of the different distances used to select informative alignments between an allele sequence and a read. The aligned parts of the sequences are illustrated by vertical bars. The allele sequence is composed of two adjacent sequences of size L_{adj} on either side of the breakpoint, which is represented by a red vertical thick bar. x and y are the distances of the breakpoint to respectively the start and end coordinates of the alignment on the allele sequence. a_1 and a_2 (resp. r_1 and r_2) are the distances of the alignment left and right extremities to the, respectively, left and right extremities of the allele sequence (resp. read sequence). It follows here, that $a_1+x=L_{adj}$ and $a_2+y=L_{adj}$.

3.4.2 Filtering out false positives due to repeated sequences

Concerning the filtering of spurious false positive alignments, `Minimap2` alignments are first filtered based on the mapping quality (`MAPQ`) score. To focus uniquely on mapped reads, the `MAPQ` score of the alignments must be greater than 10. However, this is not sufficient to filter out alignments due to repetitive sequences since mapping is performed on a small subset of the reference genome and these alignments may appear as uniquely mapped on this subset.

As `Minimap2` is a sensitive local aligner, many of the spurious alignments only cover subsequences of both the allele and the read sequences. To maximize the probability that the aligned read originates from the genomic region holding the SV, we, therefore, require the read to be aligned with the allele sequence in a semi-global manner. Each alignment extremity must correspond to an extremity of at least one of the two aligned sequences. This criterion gathers four types of situations, namely the read is included in the allele sequence, or *vice-versa*, or the read left end aligns on the allele sequence right end or *vice-versa*.

Indeed this criterion is not strictly applied and a distance of d_{end} of the alignment to an extremity of at least one of the two aligned sequences is tolerated (d_{end} is set by default to 100 bp). More formally, if a_1 and a_2 (resp. r_1 and r_2) are the sizes of the unaligned parts at the, respectively, left and right sides of the alignment on the allele sequence (resp. read sequence) (see Figure 3.6), then the alignment must fulfill the following condition in Equation (3.2) to be kept:

$$(a_1 < d_{end} \parallel r_1 < d_{end}) \;\&\; (a_2 < d_{end} \parallel r_2 < d_{end}) \tag{3.2}$$

The left member of Equation (3.2) imposes that the unaligned part at the left of the alignment is small in at least one of the two aligned sequences; the right member imposes the same condition at the right side of the alignment.

3.5 Genotype estimation

In this last step of the genotyping method, we carry out the estimation of the genotype for each variant from the informative alignments retained in the previous step. As an input we have read counts that support the presence of each allele of the variant. At the end of this last step, each variant is assigned a genotype (0/0, 0/1, 1/1 or ./.), results are output in a VCF format file.

3.5.1 Allele number normalization for unbalanced variants

For each variant, the genotype is estimated based on the ratio of amounts of reads informatively aligned to each allele sequence. In the case of deletions and insertions, the allele sequences of a given variant are of different sizes and contain a different number of breakpoints (for deletion, for instance, the reference allele contains 2 breakpoints, whereas the alternative allele contains only 1), so even if both alleles are covered with the same read depth, there would be fewer reads that align on the shortest allele sequence and that overlap at least one breakpoint. To prevent a bias towards the longest allele, reported read counts for the longest allele are normalized according to the allele sequence length ratio, assuming that read count is proportional to the sequence length. More precisely, in the case of a deletion, the reference allele is the longest allele. If the deletion is smaller than $2 \times L_{adj}$, its allele sequence size is the deletion size plus $2 \times L_{adj}$ (cumulative size of the adjacent sequences). Otherwise, it is composed of two sequences of size $2 \times L_{adj}$ each centered on each breakpoint. We, therefore, apply the Equation 3.3 to compute the normalized read count for the reference allele, c_0^*, as a function of the observed read count for the reference allele, c_0, and the deletion size, del_{size}:

$$c_0^* = \begin{cases} c_0 \times \frac{2 \times L_{adj}}{(2 \times L_{adj} + del_{size})} & \text{if} \quad del_{size} < 2 \times L_{adj} \\ c_0 \times \frac{1}{2} & \text{otherwise} \end{cases} \tag{3.3}$$

The same is true for insertions, for which the longest allele, the alternative allele, is normalized. c_1 is the read count for the alternative allele and ins_{size} is the insertion size, to

which the normalization Equation 3.4 are applied.

$$c_1^* = \begin{cases} c_1 \times \frac{2 \times L_{adj}}{(2 \times L_{adj} + ins_{size})} & \text{if} \quad ins_{size} < 2 \times L_{adj} \\ c_1 \times \frac{1}{2} & \text{otherwise} \end{cases} \tag{3.4}$$

3.5.2 Minimum allele number threshold for genotyping

Finally, a genotype is estimated if the variant presence or absence is supported by at least *min_cov* different reads after normalization (sum of the read counts for each allele). By default, this parameter is set to 3. We consider the number of alignments obtained as not informative if it is less than the *min_cov*, the genotype of the variant is not estimated and noted `./.`.

3.5.3 Genotyping decision

We decided to compare two genotyping approaches. First, we propose to assign a genotype based on fixed allele frequency thresholds. Then, we consider a genotyping model based on maximum likelihood.

3.5.3.1 Method based on decision thresholds

First we designed the method with a genotype estimation approach based on allele frequency thresholds. The allele frequency is the presence rate of an allele for a specific locus, either in an individual or in a population and is expressed as a proportion (see Equation 3.5 where allele 1 corresponds to the alternative allele). Given a locus if the allele is homozygous for the reference allele (0/0), then it is most likely to observe only reads supporting the presence of the reference allele. According to the Equation 3.5, we then obtain a frequence of the alternative allele (allele 1) close to 0. Conversely, if the individual is homozygous for the alternative allele (1/1), the frequency of allele 1 will be close to 1. Finally, if the individual is heterozygous at the given locus (0/1), then there should be as many reads that support the presence of the reference allele as the alternative allele, resulting in a frequency of allele 1 close to 0.5. Thus, we can use the frequency of alleles for a given locus to deduce the genotype of the variants, for a sequenced individual for instance.

$$\textit{Allelic frequency (allele 1)} = \frac{\text{\# of reads allele 1}}{\text{\# of reads allele 1 + \#reads allele 0}} \tag{3.5}$$

But the observed allele frequencies do not correspond exactly to those expected. Indeed,

because of random sampling and sequencing or alignment errors, read counts that support the presence of one of the alleles are noisy. The difficulty will then be to find decision thresholds to conclude on a genotype given the observed allele frequencies. However, some frequency values are in shadowy areas.

After testing several decision range values (see Section 3.6.2.1 a.), the following ranges were determined: [0;0.15[, [0.15;0.85],]0.85;1]. Thus, genotypes are estimated according to a decision threshold method, so if the allele frequency of allele 1 is smaller than 0.15, then the genotype of the variant is estimated to be homozygous for the reference allele. While, if the allele frequency of allele 1 is greater than 0.85, then the genotype of the variant is estimated to be homozygous for the alternative allele. Thus, if the allele frequency of allele 1 is between 0.15 and 0.85 (inclusive), then the genotype of the variant is estimated to be heterozygous.

3.5.3.2 Maximum likelihood method

Another possible method to estimate the genotype is to use a maximum likelihood approach. The likelihoods of the three possible genotypes given the observed normalized read counts (c_0^* and c_1) are computed based on a simple binomial model, assuming a diploid individual, as described in (Nielsen et al., 2011) (see also (Li, 2011)):

$$\mathcal{L}(0/0) = (1-err)^{c_0} \times err^{c_1} \times C^{c_0}_{c_0+c_1} \tag{3.6}$$

$$\mathcal{L}(1/1) = err^{c_0} \times (1-err)^{c_1} \times C^{c_0}_{c_0+c_1} \tag{3.7}$$

$$\mathcal{L}(0/1) = \left(\frac{1}{2}\right)^{c_0+c_1} \times C^{c_0}_{c_0+c_1} \tag{3.8}$$

where err is the probability that a read maps to a given allele erroneously, assuming it is constant and independent between all observations. err was fixed to 5.10^{-5}, after empirical experiments on a simulated dataset (see Figure 3.7 in Section b.). In the case of unbalanced SV, the normalized read counts are used in Equations 3.6, 3.7 and 3.8, *i.e.* c_0^* and c_1^* for deletions and insertions, respectively.

Finally, the genotype with the maximum likelihood is reported and all three likelihoods are also output (in Phred-scaled score *i.e.* $-log_{10}$ transformation) as additional information in the VCF file.

It turns out that for estimating the genotype of SVs, the maximum likelihood method ob-

tains better results than the method based on decision thresholds. Therefore our SV genotyping method for long read, currently uses the maximization likelihood approach to estimate genotypes.

3.6 Parameter estimation

3.6.1 Material

3.6.1.1 Simulated dataset

Parameters were determined empirically on a simulated dataset. To do this, we simulated a dataset on the human chromosome 1 (assembly GRCh37.p13) based on real characterized deletions for the human genome. 1,000 existing deletions on chromosome 1 were selected from the dbVar database (Phan et al., 2016), these sampled deletions are separated by at least 10,000 bp. The sizes of the deletions vary from 50 bp to 10 kb (with median and average sizes of 950 bp and 2,044 bp respectively).

For this dataset, the genotypes of the deletions are simulated in such a way that the genotypes are evenly distributed, *i.e.* 333 homozygous deletions (0/0), 334 heterozygous deletions (0/1) and 333 homozygous deletions for the alternative allele (1/1). Two different sequences are simulated containing each overlapping sets of deletions, representing the two haplotypes of the simulated individual. 1/1 genotype deletions are simulated on both haplotype sequences, whereas deletions of 0/1 genotype are simulated each on one randomly chosen of the two haplotype sequences. Then PacBio CLR reads are simulated on both haplotypes, using `SimLoRD` (Stöcker et al., 2016) (version v1.0.2). The dataset was simulated at a sequencing depth of 30x and an error rate of 16 % (11 % insertions, 4 % deletions and 1 % substitution, as the distribution of LRCstats (La et al., 2017)). Ten such datasets were simulated to assess the reproducibility of the results.

3.6.1.2 Evaluation

To evaluate the accuracy of the method, a contingency table between the estimated genotypes and the true (simulated) ones is computed, providing a clear view of the number and type of correctly and incorrectly estimated genotypes. The genotyping accuracy of the method is then assessed as the number of correctly estimated genotypes overall all estimated genotypes, as shown in equation (3.9). The percentage of SVs for which a genotype could be

estimated is also measured, and hereafter called the genotyping rate (Equation (3.10)).

$$Genotyping\ accuracy = \frac{\#\text{ of correctly estimated genotypes}}{\#\text{ of estimated genotypes}} \tag{3.9}$$

$$Genotyping\ rate = \frac{\#\text{ of estimated genotypes}}{\#\text{ of known SVs}} \tag{3.10}$$

3.6.2 Results according to the genotyping estimation model

First, we applied the simulated data to the two genotype estimation models proposed in section 3.5.3. We first present the results of our method with a model based on decision thresholds, and secondly, we present the results obtained with a model based on maximum likelihood.

3.6.2.1 Genotype estimation methods

We tested two methods to estimate the SV genotype from informative read counts and normalized for unbalanced variants, namely deletions and insertions. The genotypes were estimated using a decision threshold method based on allele frequencies. We opted for a method based on the maximum likelihood genotypes of the read counts. We present below the results obtained with each method as well as the values of the default parameters chosen for each approach. First, we compare the SV genotyping results on three different decision thresholds on simulated dataset. Then, we estimate the likelihood parameters and finally we compare the two methods to estimate the SV genotype.

a. Method based on thresholds: Ranges estimation

Three different ranges of decision thresholds were tested on simulated datasets:

— `SVJedi` [0.15;0.85]

— `SVJedi` [0.2;0.8]

— `SVJedi` [0.25;0.75]

where the ranges correspond to the allele frequencies for a given variant between which `SVJedi` attributes a heterozygous genotype (0/1). `SVJedi` attributes a homozygous genotype for the reference allele (0/0), if the allele frequencies are below the left bound. Finally `SVJedi` assigns a homozygous genotype for the alternative allele (1/1), if the allelic frequencies are higher than the right bound.

Table 3.1 details the `SVJedi` genotype estimates in relation to the expected simulated genotypes for the three ranges of decision thresholds. The thresholds of 0.15-0.85 achieve an accuracy of 97.1 %, false positive estimations are heterozygous deletions estimated as homozygous for the reference allele and homozygous for the alternative allele deletions estimated by `SVJedi` as heterozygous deletions. `SVJedi` obtains a genotyping accuracy of 96.3 % for the threshold model of 0.2-0.8 and a genotypaging accuracy of 94.5 % for the threshold model of 0.25-0.75. The estimation errors of these two models concern the same categories as for the 0.15-0.85 threshold model. However, there is a higher number of false positives of heterozygous deletions estimated by `SVJedi` as homozygous deletions for the reference allele. Only 13 heterozygous deletions are estimated 0/0 by `SVJedi` for the 0.15-0.875 threshold model, compared to 22 for the 0.2-0.8 threshold model and 38 for the 0.25-0.75 threshold model.

		SVJedi [0.15-0.85]				**SVJedi** [0.2-0.8]				**SVJedi** [0.25-0.75]			
		0/0	0/1	1/1	./.	0/0	0/1	1/1	./.	0/0	0/1	1/1	./.
Truth	0/0	332	1	0	0	332	1	0	0	332	1	0	0
	0/1	13	321	0	0	22	311	1	0	38	293	3	0
	1/1	0	15	317	1	1	12	319	1	2	11	319	1
		GT acc: 97.1%				GT acc: 96.3%				GT acc: 94.5%			

Table 3.1 – `SVJedi` genotyping results for three decision threshold models on 1,000 simulated deletions on the human chromosome 1. GT acc indicate the genotyping accuracy of each decision threshold model.

The decision model with thresholds 0.15-0.85 is the least impacted by false positives and thus achieves the best genotyping accuracy, so we choose the 0.15-0.85 thresholds for estimating genotypes with `SVJedi`.

b. Maximum likelihood method: Estimation of the sequencing error value

The decision model based on maximum likelihood is also tested. We performed empirical tests to correctly estimate the *err* value of the Equations 3.6, 3.7, 3.8. This parameter corresponds to the probability that a read maps to a given allele erroneously. The tests are performed on the same dataset as the models based on decision thresholds. The Figure 3.7 shows `SVJedi` genotyping results with the maximum likelihood model as a function of *err* values.

The Figure 3.7 forms a bell, centered approximately on 5.10^{-5}, we then consider the default value of $err = 5.10^{-5}$. Table 3.2 details `SVJedi` predictions per type of genotype for the *err* value set to 5.10^{-5}.

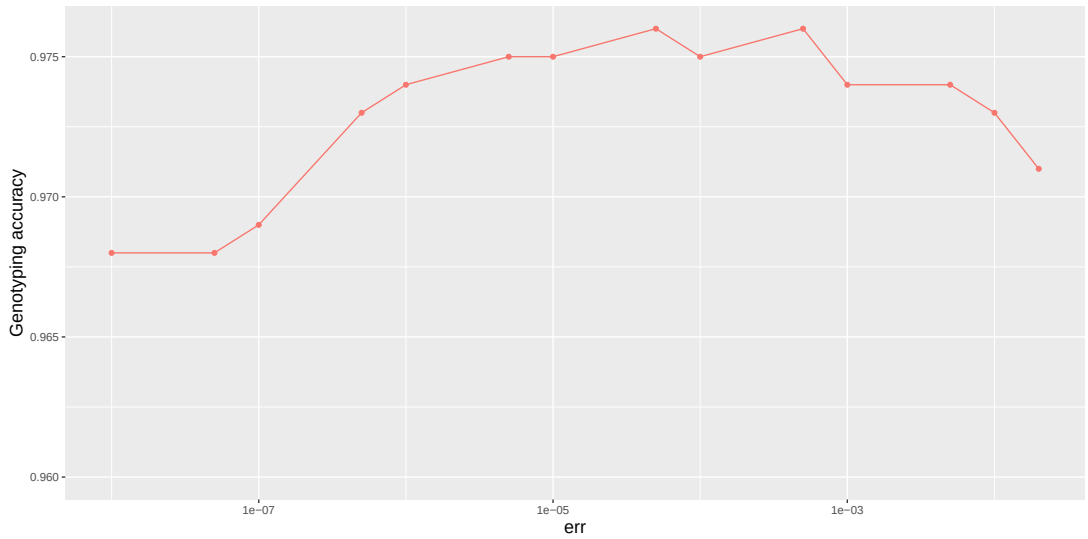

Figure 3.7 – Genotyping accuracy according to the *err* parameter for the estimation of genotype likelihoods on a simulated dataset, with 1,000 existing (in dbVar) deletions on human chromosome 1. *err* is the probability that a read maps to a given allele erroneously, it is used to compute each genotype likelihood. The default value for *err* was fixed to 5.10^{-5}.

		SVJedi			
		0/0	0/1	1/1	./.
Truth	0/0	331	1	0	1
	0/1	3	330	0	1
	1/1	0	18	313	2

Genotyping acc: 97.8 %

Table 3.2 – SVJedi genotyping results for the maximum likelihood model on 1,000 simulated deletion of the human chromosome 1.

`SVJedi` with the maximum likelihood model achieves a genotyping accuracy of 97.8 %. The majority of false positives are homozygous deletions for the alternative allele that are estimated as heterozygous by `SVJedi`. There are also 3 heterozygous deletions estimated by `SVJedi` as 0/0 and conversely 1 deletion homozygous for the reference allele estimated by `SVJedi` as 0/1.

c. Comparison of `SVJedi` genotyping methods

According to Table 3.1 and Table 3.2, the maximum likelihood model achieves a greater genotyping accuracy than the threshold decision model 0.15-0.85. Not only is the number of false

positives lower for the maximum likelihood model, but the type of false positives is essentially the same. We thus choose the maximum likelihood model for genotype estimation in `SVJedi`.

3.6.3 Estimation of the size of the representative allele sequences

In order to assign a default value to the sizes of the representative allele sequences, we have tested several values for the L_{adj} parameter. Tests were perform on a 30x simulated dataset of 1,000 deletions of the human chromosome 1, sampled from dbVar. L_{adj} was varied and tested for the following values: 1,000; 2,000; 3,000; 4,000; 5,000; 6,000; 7,000; 8,000; 9,000; 10,000 bp. The results of SVJedi are presented in the Figure 3.8.

The genotyping accuracy tends to increase with the L_{adj} value between 1,000 and 6,000 bp. For $Ladj = 5000$, the genotyping accuracy reaches 97.8 %. Peak accuracy is reached at 6,000 bp with a precision of 98.1 %. However for $L_{adj} = 6000$ bp there is a slight decrease in the genotyping rate. At the same time `SVJedi`'s runtime is proportional to the L_{adj} value. This can be explained by the fact that there are more sequences to align, the alignment step is then longer. Note that this is a small set of variants with only 1,000 deletions to genotype, so we can expect higher runtimes for larger sets of variants.

So to choose the optimal trade-off that maximizes the genotyping accuracy and the genotyping rate while keeping the runtime relatively low, we choose $L_{adj} = 5{,}000$ by default.

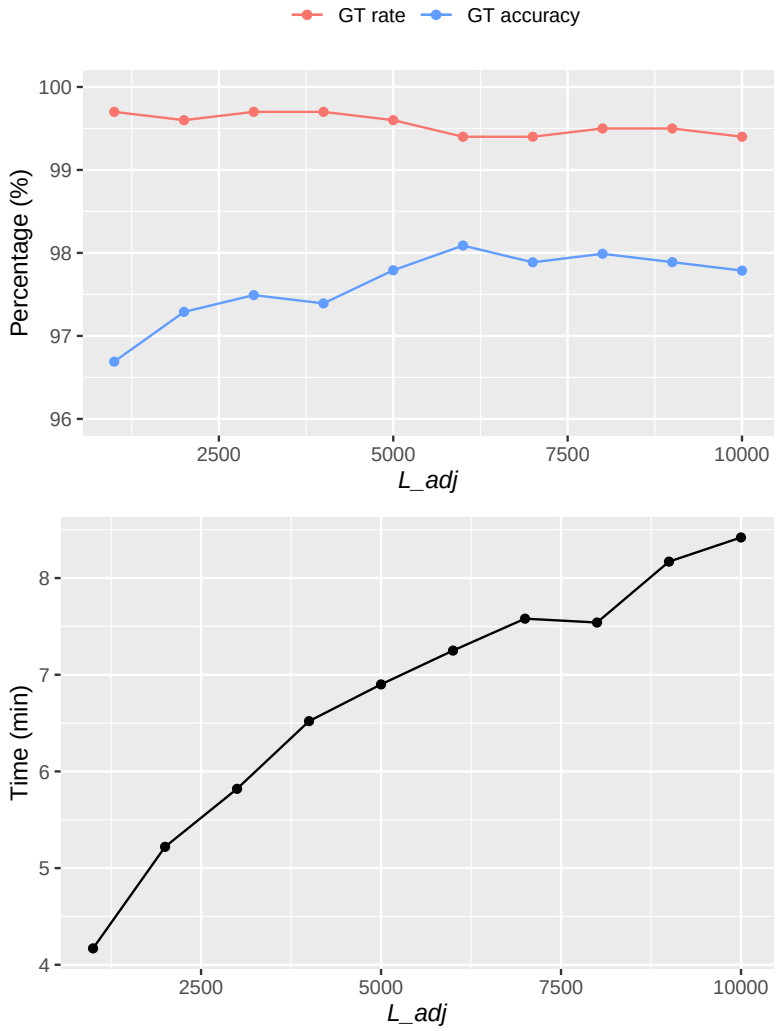

Figure 3.8 – `SVJedi` genotyping results according to L_{adj} size on a 30x simulated dataset of 1,000 deletions sampled from dbVar. GT: genotyping. The top graph shows the genotyping accuracy (blue) and genotyping rate (red) according to L_{adj}'s values. The bottom graph represents `SVJedi` runtimes according to L_{adj}'s values. Runtimes were measured on a 24-CPU computing node.

3.7 Implementation and availability

We provide an implementation of this method named `SVJedi`, freely available on github [1], under the GNU Affero GPL license. `SVJedi` can also be installed from Bioconda. `SVJedi` is written in Python 3, it requires as input a set of SVs (VCF format), a reference genome (fasta format) and a sequencing read file (fastq or fasta format), see `SVJedi` command line bellow.

1. `https://github.com/llecompte/SVJedi`

```
python3 svjedi.py -v <set_of_sv.vcf> -r <refgenome.fasta> -i <long_reads.fastq>
```

Notably, the main steps are implemented in a modular way, allowing the user to start or re-run the program from previous intermediate results. As an example, the first step that generates representative allele sequences, is not to be repeated if there are several long read datasets to be genotyped on the same SV set.

In the first module of `SVJedi`, to generate the representative sequences of the alleles, we first retrieve the sequence of the reference genome. The input VCF is then scanned to extract SV-related information such as genomic positions, insertion sequences or `BND` type. For each SV, representative sequences of the two alleles are generated and saved in a fasta file. Then during the second module, all the reads are aligned at once by `Minimap2` against the newly obtained fasta file. The third and last module consists in genotyping the SVs from the alignments obtained. This last module is divided into two main steps. During the first step, the alignment file is browsed to save only the read IDs of the so-called informative alignments (`MAPQ` ≥ 10, breakpoint overlap, semi-global alignment). For the second step, the initial VCF file is browsed. Insertion and deletion read counts are normalized and then rounded. For each SV, the genotype likelihoods are calculated from the read counts, and the genotype having the maximum likelihood is then assigned. For each SV browsed from the initial VCF file, we save in a new VCF file the SV and its estimated genotype, along with read count and likelihood informations.

3.7.1 Requirements for SV definition in input VCF file

A VCF file is a tabulated file where variations, including SVs, are described in lines. Most often, the top of the file contains metadata information and a header, differentiated from variations by respectively `##` and `#` at the beginning of the line. Then at least 8 fields, separated by tabs, describe each variation and which can be broken down as follows: `CHROM` field corresponding to the chromosome identifier, `POS` field corresponds to the position of the first base of the variation, `ID` field is an identifier, `REF` field is a reference base at the position, `ALT` is the alternate base at the position, `QUAL` field corresponds to a quality score of the called variant (Phred-scale quality score), `FILTER` field is a filter status and lastly a `INFO` field that contains additional information.

The VCF file is more appropriate for small variants such as SNPs or small indels. As suggested by the required fields in the VCF a lot of information is missing when describing SVs. Thus, the `INFO` tag contains additional information necessary for the characterization

of SVs. Although there are VCF specifications [2], they are not always respected by SV discovery tools. Besides, there are multiple ways to describe the same SV which further complicates the standardization of the VCF format for SV tools. Describing SVs in VCF format remains a challenge, the tools have to juggle and adapt to the many ways of describing SV features.

We describe in the following paragraphs the information required to genotype SV by our method.

Deletions in order to be considered, must be defined by a deletion SV type, have a start position on the reference and a deletion length.

- SV type: either the `ALT` column is `<DEL>` or `INFO` column contains `SVTYPE=DEL`.
- The chromosome of the deletion is retrieved in the `CHROM` column.
- The deletion start position is directly given by the `POS` column.
- The deletion size is retrieved from the `INFO` column with the `SVLEN=x` tag, where x is a relative integer.

Example of deletions in VCF:

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO
1	1285401	del1	A	<DEL>	10	PASS	SVLEN=-1399;SVTYPE=DEL;END=1286800
1	41024252	del2	C	<DEL>	20	PASS	SVLEN=-119;SVTYPE=DEL;END=41024371
2	35034437	del3	C	<DEL>	20	PASS	SVLEN=-245;SVTYPE=DEL;END=35034682
3	27430321	del4	A	<DEL>	20	PASS	SVLEN=-83;SVTYPE=DEL;END=27430404
5	115465267	del5	A	<DEL>	20	PASS	SVLEN=-214;SVTYPE=DEL;END=115465481

Insertions to be considered, must refer an insertion SV type, the breakpoint position of the insertion on the reference and the sequence of the insertion.

- SV type: `INFO` column must have the following tag, `SVTYPE=INS`.
- The chromosome of the insertion is retrieved in the `CHROM` column.
- The insertion breakpoint corresponds to the integer in the `POS` column.
- The insertion sequence must be precised in the `ALT` column.

Example of insertions in VCF:

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO
1	931634	ins1	A	AGGTCATTATGCG	10	PASS	SVLEN=-13; SVTYPE=INS;END=931647
1	97717628	ins2	T	TTCCCAGACAAAT	10	PASS	SVLEN=13;SVTYPE=INS;END=97717641

2. http://samtools.github.io/hts-specs/

Note that for reasons of sequence representation in the `ALT` field, the insertions given in the example are less than 50 bp.

Inversions to be considered, they must precise the inversion SV type, the start and end positions of the variant.

— SV type: either the `ALT` column is `<INV>` or `INFO` column contains `SVTYPE=INV`.

— The chromosome of the inversion is retrieved in the `CHROM` column.

— The inversion start position is the integer in `POS` column.

— The inversion end position is the x integer in the `END=x` tag of the `INFO` column.

Example of inversions in VCF:

```
#CHROM  POS     ID      REF     ALT     QUAL    FILTER      INFO
1       891646  id1     .       <INV>   .       .           SVTYPE=INV;END=897179;SVLEN=0; GT 0/0
1       940516  id2     .       <INV>   .       .           SVTYPE=INV;END=944353;SVLEN=0; GT 0/1
1       1452973 id3     .       <INV>   .       .           SVTYPE=INV;END=1454492;SVLEN=0; GT 1/1
1       1867826 id4     .       <INV>   .       .           SVTYPE=INV;END=1875854;SVLEN=0; GT 0/1
1       1934321 id5     .       <INV>   .       .           SVTYPE=INV;END=1935416;SVLEN=0; GT 0/0
```

A translocation is defined as two separate breakpoints, so one VCF line for each breakpoint. Thus, a breakpoint to be considered by our method, must have a chromosome A and a chromosome B, a breakpoint position on each chromosome, and a type of junction among the four possible ones.

— SV type: `INFO` column contains `SVTYPE=BND`.

— Chromosome A is the `CHROM` column.

— Breakpoint position on chromosome A is the `POS` column.

— Chromosome B can either be retrieved in the `CHR2=chromosome B` tag in the `INFO` column or in the `ALT` column.

— Breakpoint position on chromosome B can either be retrieved from the `END` tag in the `INFO` column or in the `ALT` column.

— Type of segment junction is given in the `ALT` column as well. As we've seen there are four possible `BND` which are listed in Figure 3.9.

Type of BND	Encoding in VCF in <ALT> field	Rearrangement	Novel adjacency
BND 1	nt[chrB:posB[		
BND 2	]chrB:posB]nt		
BND 3	nt]chrB:posB]		
BND 4	[chrB:posB[nt		

Figure 3.9 – The four types of BND in the <ALT> field of the VCFs. nt is the nucleotide at the breakpoint on chromosome A.

Example of BND in VCF (from VCF version 4.2 Specification [3]).

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO
2	321681	bnd_W	G	G]17:198982]	6	PASS	SVTYPE=BND
2	321682	bnd_V	T	]13:123456]T	6	PASS	SVTYPE=BND
13	123456	bnd_U	C	C[2:321682[	6	PASS	SVTYPE=BND
13	123457	bnd_X	A	[17:198983[A	6	PASS	SVTYPE=BND
17	178982	bnd_Y	A	A]2:324681]	6	PASS	SVTYPE=BND
17	198983	bnd_Z	C	[13:123457[C	6	PASS	SVTYPE=BND

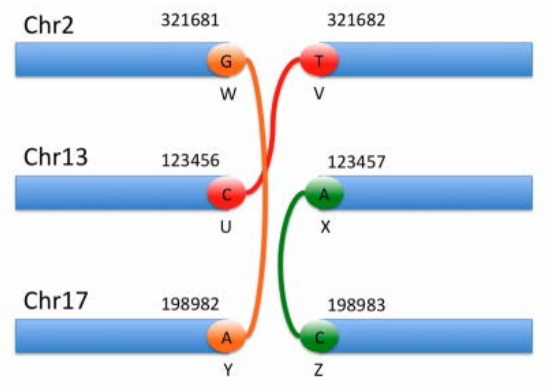

Figure 3.10 – All types of BND from VCF version 4.2 Specification

3. https://github.com/samtools/hts-specs

3.8 Conclusion

In conclusion, we propose a new method to estimate SV genotype with long reads data for the main types of SVs, including deletions, insertions, inversions and translocations. Our method uses for each variant a representation of the two alleles. To reduce the represented sequence, we thus choose to represent only the sequences of the variants rather than the whole genome. Once the long PacBio or ONT reads are aligned against the representative allele sequences, the alignments are filtered to retain only the informative alignments regarding the presence of one of the two alleles. Finally, our method uses a maximum likelihood model to estimate the SV genotype from the counts of informative reads of each allele.

We have implemented the long read genotyping method in a tool called `SVJedi`, available on github and conda. We also tested some methodological choices and some parameters on simulated data. We continue the validation of our method in the next chapter by evaluating our method on several types of data and comparing our method to other approaches.

Chapter 4

Method validation

Table of contents

4.1 Introduction 113

4.2 Evaluation on simulated long read datasets 113

4.2.1 Dataset presentation 113

4.2.2 Genotyping results 115

4.2.2.1 `SVJedi` on simulated deletions 115

4.2.2.2 `SVJedi` results on simulated inversions and on simulated translocations 116

4.2.3 Evaluation of the robustness of the method 116

4.2.3.1 Sequencing depth 117

4.2.3.2 Sequencing error rate 118

4.2.3.3 Precision of the position of breakpoint position 118

4.2.3.4 Evaluation with close and overlapping deletions 120

4.3 Application to real datasets 121

4.3.1 Material 122

4.3.2 PacBio dataset 123

4.3.2.1 `SVJedi` results on PacBio dataset 123

4.3.2.2 Mendelian Inheritance of the Ashkenazi trio 125

4.3.3 ONT dataset 125

4.4 Comparison with other approaches 125

4.4.1 Comparison with other genotyping tools for long reads 127

4.4.1.1 Detailed genotyping results 128

4.4.2 Comparison with a short read based genotyping approach 129

4.4.3 Comparison with SV discovery approaches 129

4.4.4 Stratified analysis 130

4.4.4.1 Performance by SV size categories 130

4.4.4.2 Performance by genomic context and SV type 132
4.4.5 Runtime comparison 133
4.5 Conclusion 133

4.1 Introduction

In order to validate our new SV genotyping method for long reads, we assess `SVJedi` on several datasets. The majority of the results presented in this chapter have been submitted and accepted in an article, (Lecompte et al., 2020). In this chapter, we present the results of the SV genotyping method for long read data, first on a simulated data set of SVs in order to evaluate the performance and limitations of our method. Then we apply our tool on a real human dataset with a high confidence set of genotyped SVs. Finally, we will compare the performance of `SVJedi` on the same real dataset, to other genotyping tools.

4.2 Evaluation on simulated long read datasets

We first applied our genotyping tool, `SVJedi`, on simulated datasets to evaluate its performance. `SVJedi` results shown in this chapter were obtained with release version 1.1.0. In this section, we first present the generation of the simulated datasets, and then comment on the `SVJedi` results obtained on these datasets. Finally, we evaluate the robustness of our tool by varying the characteristics of the simulated datasets.

4.2.1 Dataset presentation

Separate simulated datasets were generated for each type of SVs. One dataset contains only deletions have as already been described in the previous chapter in section 3.6.1.1, another contains only inversions, and the last contains only translocations. The datasets were simulated from the chromosome 1 of the human genome (reference assembly GRCh37.p13). In this experiment, real characterized variants were added to chromosome 1, based on the human SV database, dbVar (Phan et al., 2016). Thus, the SVs were not randomly placed, but were placed at specific positions according to the observations in dbVar. Far from being random, the structure of the genome is complex and can cause difficulties in signal analysis in bioinformatics. In order to get as close as possible to reality, we choose to generate SVs that have already been observed in chromosome 1 of the human genome. However, the dbVar database is far from complete and the majority of the observed insertions were detected by short reads. The detection methods for short reads have difficulties to retrieve the sequence of insertions. For most insertions calls, the insertions sequence is not provided in dbVar, so we could not create a dataset for the insertions. Nevertheless, insertions are simply a counterpart to deletions, for which a data set is simulated..

Simulated dataset with deletions. The simulated dataset contains 1,000 deletions on chromosome 1 of the human genome, described in dbVar. Deletions have been selected with lengths ranging from 50 bp to 10 kb. Very few deletions referenced in dbVar are larger than 10,000 bp and we prefer to focus on most human deletion sizes. Moreover, we chose to represent simple cases of variants, in other words, not nested. We have selected non overlapping deletions, which are at least 10,000 bp apart. Then, deletion were equality distributed between the three possible genotypes: 333 deletions are simulated with 0/0 genotype, 334 deletions with 0/1 genotype and the 333 remaining deletions with 1/1 genotype. In order to estimate the genotypes, two sequences corresponding to the two haplotypes are generated. 1/1 genotype deletions were simulated on both haplotype sequences, whereas deletions of 0/1 genotype were simulated each on one randomly chosen of the two haplotype sequences. From the two haplotype sequences, a long read PacBio dataset was simulated with `SimLoRD` (Stöcker et al., 2016) (version v1.0.2) with a 30x sequencing depth and a sequencing error rate of 16 % (11 % insertions, 4 % deletions and 1 % substitutions). We also simulated datasets by varying the sequencing depth (from 6x to 60x) and other datasets by varying the the percentage of error (from 6 % error to 20 %).

Simulated dataset with inversions. Unlike the simulated deletion dataset, we simulated random inversions because they are less frequent than deletions in dbVar. We have simulated 450 random inversions on chromosome 1 of the human genome, by sampling the left breakpoint in a uniform distribution and choosing the inversion size uniformly between 50 bp and 15 kb. Variants were only simulated outside the gap regions according to the UCSC track and at a distance of at least 20 kb.

Simulated dataset with translocations. Similarly as the simulation inversion dataset, we simulated 450 random non-reciprocal translocations, or in other words 450 breakpoints. Only few reciprocal translocations are described in dbVar and to facilitate the generation of simulated breakpoints, we generated 450 breakpoints successively between chromosome 1 and chromosome 2 of the human genome. Breakpoints were sampled uniformly on both chromosomes. In the same way as for inversions, breakpoints are simulated outside the gap regions according to the UCSC track. Additionally, breakpoints on the same chromosome are separated by at least 50 kb. Only breakpoints on chromosome 1 were reported with the same junction type (see Translocation type A, Figure 3.5).

Evaluation. In order to evaluate the performance of our genotyping method, we use two metrics presented in the previous chapter: the *genotyping accuracy* (Equation 3.9), that informs about the number of correctly estimated genotypes relative to the total number of estimated genotypes, and the *genotyping rate* (Equation 3.10), that indicates the proportion of variants with an estimated genotype relative to the set of variants to be genotyped.

4.2.2 Genotyping results

4.2.2.1 `SVJedi` on simulated deletions

Table 4.1 shows the obtained genotypes compared with expected ones for a simulated dataset at 30x read depth on 1000 deletions sampled from dbVar. The sizes of the deletions vary from 50 bp to 10 kb (with median and average sizes of 950 bp and 2,044 bp respectively).

		`SVJedi` predictions			
		0/0	0/1	1/1	./.
Truth	0/0	331	1	0	1
	0/1	3	330	0	1
	1/1	0	18	313	2

Genotyping accuracy : 97.8 %

Table 4.1 – Contingency table of `SVJedi` results on PacBio simulated data (30x) of human chromosome 1 with 1,000 deletions from dbVar. `SVJedi` genotype predictions are indicated by column and the expected genotypes are shown by row. The genotype "./." column corresponds to deletions for which `SVJedi` could not assess the genotype.

On this dataset, `SVJedi` achieves 97.8 % genotyping accuracy, with 974 deletions correctly predicted over 996 with an assigned genotype. Among the 1,000 assessed deletions, only 4 could not be assigned a genotype due to insufficient coverage of informative reads, the genotyping rate being thus 99.6 %. Among the few genotyping errors, most concern 1/1 genotypes that were incorrectly predicted as 0/1.

The experiment was replicated on 10 other sampled sets of deletions, and similar results were obtained.

4.2.2.2 `SVJedi` results on simulated inversions and on simulated translocations

`SVJedi` was also applied to a dataset of 450 randomly simulated inversions and another dataset of 450 randomly simulated translocations (non-reciprocal). Table 4.2 describes `SVJedi` estimation results according to the expected genotypes for the inversion dataset (left) and the translocation dataset (right).

Inversions

		`SVJedi` estimations			
		0/0	0/1	1/1	./.
Truth	0/0	150	0	0	0
	0/1	0	150	0	0
	1/1	0	10	140	0

Genotyping accuracy: 97.8 %

Genotyping rate: 100 %

Translocations

		`SVJedi` estimations			
		0/0	0/1	1/1	./.
Truth	0/0	150	0	0	0
	0/1	0	150	0	0
	1/1	1	7	142	0

Genotyping accuracy: 98.2 %

Genotyping rate: 100 %

Table 4.2 – Contingency tables of `SVJedi` genotyping results on two 30x PacBio simulated dataset: one dataset with 450 inversions (left) and one dataset with 450 translocations (right). Grey labelled boxes correspond to identical predictions between the two methods. The number of genotypes that `SVJedi` fails to assess is indicated by the "./." column.

We see in Table 4.2, that `SVJedi` assigns a genotype to all 450 inversions and all 450 translocations. `SVJedi` achieves a genotyping accuracy of 97.8 % and 98.2 % for the inversion dataset and the translocation dataset, respectively. We notice that `SVJedi` tends to wrongly estimate 0/1 genotypes instead of 1/1 genotypes for both inversion and translocation datasets. We notice that the same type of false positive results were observed in the simulated deletion dataset.

4.2.3 Evaluation of the robustness of the method

We assessed the accuracy and robustness of our SV genotyping method on simulated datasets of deletions. To do this, we have modified the characteristics of the simulated datasets to see if `SVJedi` has limits depending on the characteristics of the dataset. The sequencing depth was, therefore, varied to see if it has an impact on `SVJedi`'s ability to correctly genotype SVs. The second step was to evaluate the impact of the error rate on the genotyping results. Finally, we evaluated the results of the genotyping of `SVJedi` when the coordinates of the breakpoints are not correctly predicted in the VCF of the set of SVs.

4.2.3.1 Sequencing depth

First of all, we are interested in the impact of the sequencing depth on the genotyping results of `SVJedi`. It can indeed be hypothesized that the greater the sequencing depth, the easier the signal will be to analyze for genotyping. Furthermore, what would be the necessary and sufficient sequencing depth for `SVJedi` to achieve high performance?

The same set of 1,000 selected deletions from dbVar on human chromosome 1, is used for simulating eight datasets with varying sequencing depths: 6x, 10x, 16x, 20x, 30x, 40x, 50x, 60x. Like the previous long read datasets, `SimLoRD` simulates PacBio reads with an error rate fixed at 16 %. We added to our analysis 9 replicates of different sets of sampled deletions in dbVar following the same protocol as in the section 4.2.1. We then have a total of 10 replicates for each sequencing depth.

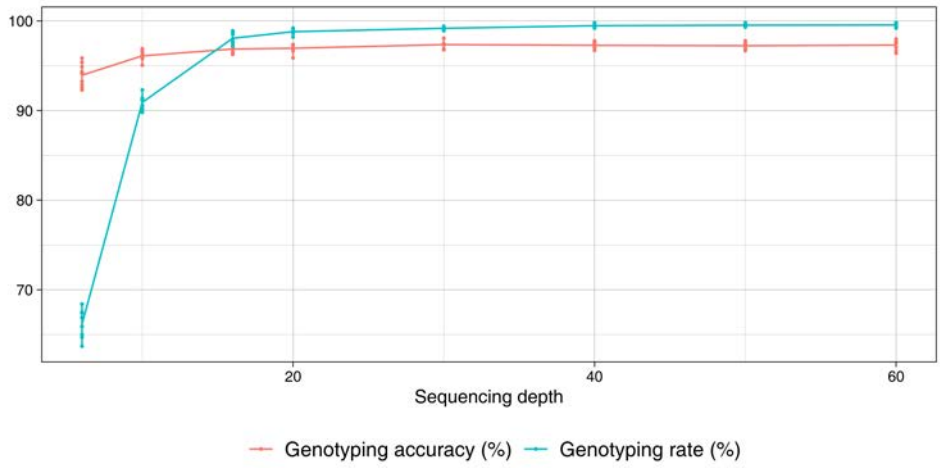

Figure 4.1 – `SVJedi` genotyping accuracy results as a function of the sequencing depth for nine simulated PacBio datasets of human chromosome 1, containing 1,000 deletions from the dbVar database. The red dots correspond to the average genotyping accuracy and the red segments represent the standard deviations, at each sequencing depth.

`SVJedi` genotyping accuracy results are illustrated in Figure 4.1, in terms of varying sequencing depths, ranging from 6x to 60x. As expected, the accuracy of `SVJedi` increases with the read depth. But interestingly, even at low coverage (6x) the accuracy is on average above 94 % and a plateau is quickly reached between 20x and 30x, with already 97 % of genotyping accuracy at 20x. The genotyping rate reaches its plateau at a sequencing depth of 16x.

4.2.3.2 Sequencing error rate

Similarly, `SVJedi` results were evaluated in terms of varying sequencing error rates. In this case, both genotyping accuracy and genotyping rate were not impacted by a lower or higher sequencing error rate as long as it stays realistic. Once again, we rely on the set of 1,000 deletions sampled from dbVar. `SimLoRD` is also used to simulate PacBio datasets at a fixed 30x sequencing depth and an error rate varying from 6 %, 10 %, 16 % and 20 %. `SimLoRD` can be used to specify the distribution of error types. Since the majority of PacBio sequencing errors are mainly due to indels, especially insertions, error type distribution is detail in Table 4.3. `SVJedi` was applied on 10 replicates of sets of 1,000 deletions, for each sequencing error value. Genotyping results can potentially be expected to be inversely proportional to sequencing error rates.

Total error rate	Insertion (%)	Deletion (%)	Substitution (%)
6 %	3	3	1
10 %	6	3	1
16 %	11	4	1
20 %	12	6	2

Table 4.3 – Distribution of sequencing error types of PacBio simulated data

Figure 4.2 describes the genotyping accuracy and the genotyping rate results according to the sequencing error rate. Although we notice a slight decrease in the genotyping rate between the error rate of 16 % and 20 %, the accuracy of the genotyping remains stable even for high error datasets, and it does not seem to be affected by the sequencing error rate on simulated datasets. The performance of `SVJedi` between replicates of the sequencing errors is consistent.

This result can be explained by the fact that the genotyping method is based the mapping of long sequences over the breakpoint of one of the allele sequences, and these alignments are, therefore, not impacted by the sequencing error rate.

4.2.3.3 Precision of the position of breakpoint position

The breakpoint coordinates of SVs detected by SV discovery methods are not always defined at the base pair resolution. To assess to what extent this potential imprecision can impact the genotyping accuracy of `SVJedi`, we performed experiments with altered breakpoint positions in the input variant VCF file. All breakpoint positions have been randomly shifted

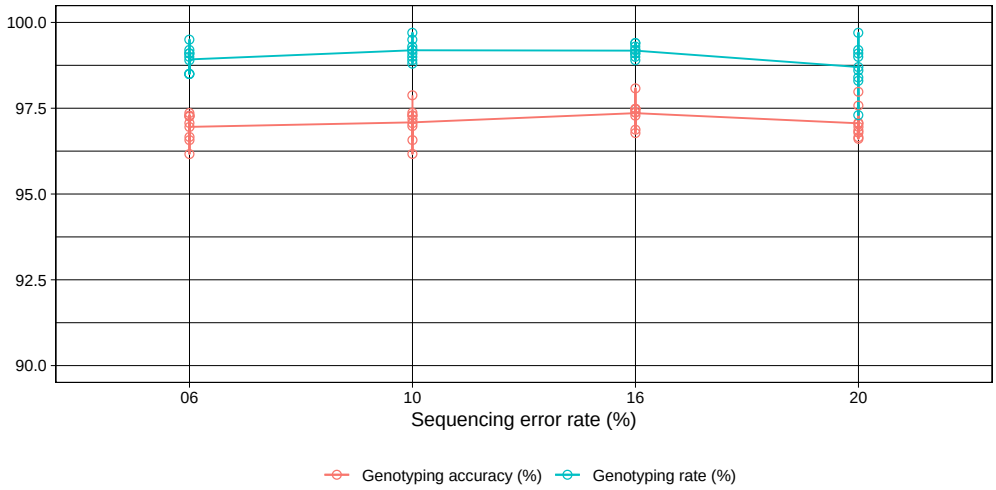

Figure 4.2 – `SVJedi` results on simulated 30x datasets at several sequencing error rates : 6 %, 10 %, 16 %, 20 %.

according to a Normal distribution centered on the exact breakpoint position with several standard deviations (σ) values ranging from 10 to 100 bp.

We show that the genotyping accuracy with σ equals 50 bp does not fall below 94 % (see Figure 4.3), indicating that `SVJedi` is not much impacted by the exact definition of the positions of the reference breakpoints.

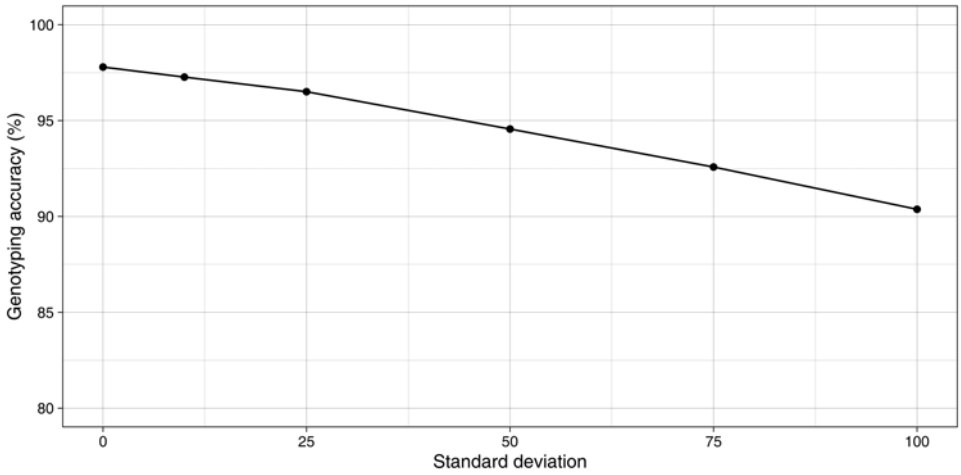

Figure 4.3 – `SVJedi` genotyping accuracy on a simulated 30x dataset with respect to the precision of breakpoint positions in the input deletion file. All breakpoint positions have been randomly shifted according to a Normal distribution centered on the exact breakpoint position with several standard deviations values ranging from 10 to 100 bp.

4.2.3.4 Evaluation with close and overlapping deletions

In our method, we represent allele sequences of each SV independently of the other SVs in the set, and long reads are mapped to the whole set of alleles for all SVs. We try to evaluate here the impact of the proximity on the genome of the SVs to be genotyped on `SVJedi`'s results. From the set of 1,000 deletions of human chromosome 1 that were sampled from the dbVar database, we added 889 additional deletions of size ranging from 50 to 2000 bp in the set of SVs to be genotyped (input vcf file). The input individual long read sample is not modified, so these deletions have a genotype 0/0.

We performed several experiments where initially we added 889 new deletions to at least 10 kb on either side of the deletion and then we progressively bring the additional deletions closer to the deletion on their left. So, each of the 889 simulated deletions is placed between 5 kb and 10 kb of the left deletion, then between 1 kb and 5 kb, between 500 bp and 1 kb, between 100 bp and 500 bp, between 50 bp and 100 bp, and finally between 0 and 50 bp of the left deletion. Then in a second time, we carried out other experiments where the 889 simulated deletions overlap the deletion that precedes on the left. We simulated successively an overlap ranging from 0 to 50 bp, then from 50 bp to 100 bp, from 100 bp to 200 bp, from 200 bp to 300 bp, from 300 bp to 400 bp and finally from 400 bp to 500 bp. The maximum overlap did not exceed the size of the left deletion.

The results of the genotyping accuracy and rate of `SVJedi` on a 30x PacBio simulated dataset (16 % error rate) are reported in Figure 4.4, according to the proximity of the deletions (left graph), and according to the overlapping of the deletions (right graph). The genotyping rate of `SVJedi` declines as the distance between deletions decreases. 80 % of the 1,889 deletions are genotyped when the distance of the 889 deletions from their previous deletion is between 50 bp and 100 bp. The accuracy of genotyping remains relatively constant at around 98 % despite the proximity of the deletions' coordinates.

For deletion overlap experiments, the rate of genotyping decreases as the number of overlapping base pairs increases. 77.3 % of deletions are genotyped when deletions overlap from 0 to 50 bp, and only 67.7 % of deletions are genotyped when deletions overlap from 400 bp to 500 bp. As with the results according to the proximity of the deletions, the genotyping accuracy remains relatively constant, however, we observe a decrease to less than 95 % from an overlap of 200 to 300 bp.

So to summarize, as deletions are close to overlapping, `SVJedi` tends to not estimate the genotype of these variants to maintain high accuracy. So even if the input SV set contains

close or nested variants, the estimated genotypes by `SVJedi` can be trusted.

The decrease of the genotyping rate when the proximity of the SVs increases can possibly be explained by our methodological choice to filter variants with mapping quality (MAPQ) lower than 10. With close variants we represent several times the same genomic sequences in our allele sequences, which therefore, decreases the uniqueness (the mappability) of the sequences to be aligned. Consequently, the alignments of close SVs are potentially filtered out, so the genotype of these variants is not estimated by `SVJedi`.

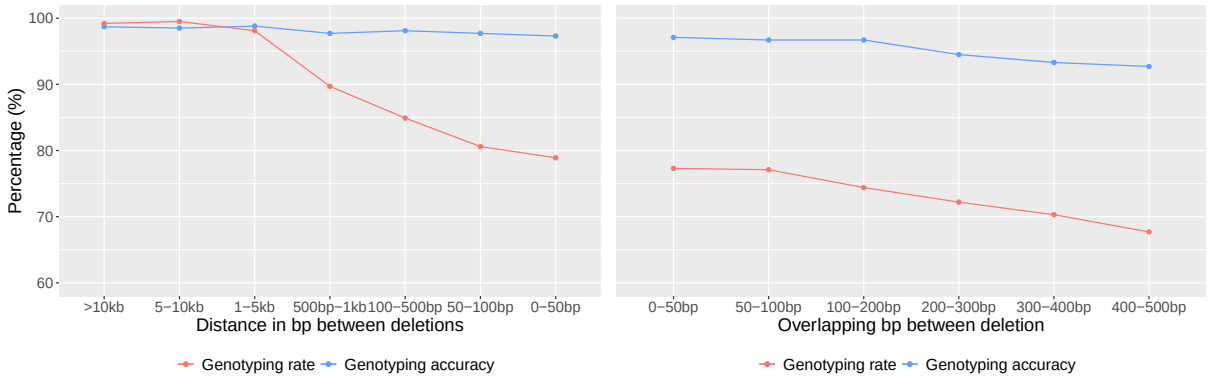

Figure 4.4 – `SVJedi` genotyping accuracy (blue) and rate (red) results for 1,889 deletions on 30x PacBio simulated dataset (16 % error), according to distance in bp between simulated deletions (left) and according to overlapping bp between simulated deletions (right).

4.3 Application to real datasets

To get closer to the reality of biological data, we applied our tool to a real human dataset, the HG002 individual, son of the so-called *Ashkenazi trio* dataset, which has been highly sequenced and analyzed in various benchmarks and especially by the GiaB Consortium (Zook et al., 2016, 2020). The latter, precisely, provides a set of high confidence SV calls together with their genotype in the individual HG002. SV discovery and genotyping were based on several sequencing technologies, SV callers, and careful call set merging. Their work estimated genotypes for 5,464 deletions and 7,281 insertions, which can then be considered as the ground truth. It should be noted that we can focus only on heterozygous (0/1) and homozygous for the alternative allele (1/1) genotypes. Indeed, the SV call set was obtained from SV discovery methods, which can only detect variations between the individual and the reference genome.

`SVJedi` was applied on multiple sequencing datasets, we first present the datasets in the

Material section, followed by `SVJedi` results on datasets in the following sections.

4.3.1 Material

We considered the assembly GRCh37.p13 as the human genome reference and we used the SV benchmark set (v0.6) of HG002 individual provided by the GiaB Consortium (Zook et al., 2020) as a gold standard SV call set. This set contains 5,464 high confidence deletions and 7,281 insertions (with `PASS` in the filter tag), whose sizes range from 50 bp to 125 kb (median sizes of 149 bp and 215 bp for deletions and insertions, respectively). We used the `TRgt100=TRUE` tags present in the GiaB VCF file to identify SVs located in Tandem Repeats greater than 100 bp (denoted as TRs, n= 6,469). Also, 48 SVs were found located inside large (> 10 kb) segmental duplications, using the UCSC Segmental Dups feature track (Bailey et al., 2002).

`SVJedi` was applied on a PacBio Continuous Long Read (CLR) sequencing dataset for HG002 which was downloaded from the FTP server of GiaB and down-sampled to 30x read depth (FTP links for all real sequencing datasets are given in Appendix Section 6.1.1).

These SVs were also genotyped in PacBio sequencing datasets of the two parents (HG003 and HG004, 30x and 27x, respectively) to assess the level of Mendelian inheritance consistency of the son predicted genotypes.

`SVJedi` was also applied on a real human ONT PromethION 44x dataset for the individual HG002 as well.

Evaluation. The GiaB SV call set was obtained from multiple sequencing data and multiple detection methods, in parallel with a huge effort to merge and cure the calls. Knowing that the GiaB set is a curated SV call set, the performance of the tools was evaluated against GiaB's predictions. Thus the genotyping accuracy equation is slightly modified for the real data (see Equation 4.1).

$$\textit{Genotyping accuracy} = \frac{\#\text{ estimated genotypes as GiaB}}{\#\text{ of estimated genotypes}} \tag{4.1}$$

The genotyping rate equation remains unchanged (see Equation 3.10).

4.3.2 PacBio dataset

4.3.2.1 SVJedi results on PacBio dataset

SVJedi genotyping results of 5,464 deletions and 7,281 insertions with a 30x PacBio dataset are shown in Table 4.4. The SVJedi predictions (shown in column) are compared to the GiaB predictions (shown in line), so no real genotyping accuracy can be calculated. We then calculate the overlap of SVJedi's predictions compared to GiaB's predictions whose confidence is high. The overlap is measured similarly as the genotyping accuracy, where the expected genotypes now correspond to the GiaB genotypes.

We observe a good overlap of 92.2 % between the estimated genotypes of SVJedi and the GiaB call set. More precisely, among the assigned genotypes, there are 91.7 % of deletions and 92.5 % of insertions that are genotyped by SVJedi identically as the GiaB call set.

DELETIONS

		SVJedi predictions			
		0/0	0/1	1/1	./.
GiaB	0/1	227	2,773	38	395
	1/1	2	124	1,522	383

Genotyping accuracy: 91.7 %

Genotyping rate: 85.8 %

INSERTIONS

		SVJedi predictions			
		0/0	0/1	1/1	./.
GiaB	0/1	18	2,870	290	327
	1/1	1	199	3,436	140

Genotyping accuracy: 92.5 %

Genotyping rate: 93.6 %

Table 4.4 – Contingency tables of SVJedi genotyping results on the real 30x PacBio dataset of human individual HG002 with respect to the high confidence GiaB call set. Results for the 5,464 deletions (left) and 7,281 insertions (right) are indicated in two separated tables, where columns indicate SVJedi genotypes and rows GiaB ones. Grey labelled boxes correspond to identical predictions between the two methods. The number of genotypes that SVJedi fails to assess is indicated by the "./." column.

To identify the cause of SVJedi's false predictions, we analyzed the characteristics that distinguish true positives from false positives. Among the SVs differently genotyped between SVJedi and GiaB, a large part is represented by small variants (57 % are smaller than 100 bp). The genomic context of the SVs seems also to impact the genotyping accuracy: 75 % of the differently genotyped variants are located in Tandem Repeats greater than 100 bp (TRs), compared to 51 % for the whole SV set. Both features, size and location in TR, have similar impacts on the genotyping accuracy, with a difference of 9 and 11 % for small-vs-large and TR-vs-nonTR located SVs, respectively. Combining the two factors leads to a larger difference, with the

SV count

	len(SV) < 100bp	len(SV) > 100bp	Total
Inside TR	2,873	3,596	6,469
Outside TR	1,399	4,877	6,276
Total	4,272	8,473	12,745

Genotyping accuracy (%)

	len(SV) < 100bp	len(SV) > 100bp	Total
Inside TR	81.3	91.2	87.6
Outside TR	89.7	97.9	96.3
Total	84.5	95.1	92.2

Genotyping rate (%)

	len(SV) < 100bp	len(SV) > 100bp	Total
Inside TR	68.2	96.4	83.9
Outside TR	87.8	99.4	96.8
Total	74.6	98.1	90.2

Table 4.5 – Comparison of SVJedi results on the HG002 real PacBio dataset between several classes of SVs. SVs are classified according to two binary variables: the SV size is larger or not than 100 bp and the SV is located inside a Tandem Repeat of size greater than 100 bp (TR) or not. The top, middle and bottom tables indicate respectively the SV count, the average genotyping accuracy and the average genotyping rate in each class. Both features, SV size and location in TR, are not independently distributed among the SVs, as confirmed by a Chi-squared test applied on the top table (SV counts) (Chi-squared statistics of 698.5, p-value $<2.10^{-16}$).

small SVs that are located in TRs having the lowest genotyping accuracy of 81.3 % compared to near perfect accuracy of 97.9 % for the larger ones outside TRs (see the cross table in Table 4.5).

Compared to previous results on simulated data, SVJedi shows a lower genotyping rate on this real dataset, for both deletions and insertions (85.8 % and 93.6 %, respectively). As in the case of accuracy, we notice that the great majority of the non-genotyped variants are either small or located in TRs: 87 % are of size less than 100 bp and 84 % are located in TRs. The factor impacting most the genotyping rate is the SV size (genotyping rates of 74.6 % and 98.1 % for small and large SVs respectively, see Table 4.5). The presence of a TR at the breakpoint of small SVs worsens the genotyping task, with only 68.2 % of such SVs that could be genotyped. Notably, the GiaB deletion set contains more in proportion of such small SVs (39 vs 29 % for deletions and insertions respectively), explaining the observed difference in genotyping rate between the two SV types.

Interestingly, these kinds of variants seem to be more impacted by the heterogeneity of PacBio sequencing depth since when using the full 63x dataset, the overall genotyping rate increases to 96.6 %. Detailed genotyping results are given in Appendix in Table 6.1.

4.3.2.2 Mendelian Inheritance of the Ashkenazi trio

Since sequencing data are available for the parents of the studied individual (HG003 for the father and HG004 for the mother), we can check, as an alternative validation approach, if the predicted genotypes for the son are consistent with his parent genotypes, assuming perfect Mendelian inheritance and a very low *de novo* mutation rate. To do so, from the same set of deletions and insertions, which is the GiaB call set, `SVJedi` was applied to three PacBio sequence datasets, one per individual, with a sequencing depth of about 30x for each dataset. Overall, the Mendelian inheritance consistency of `SVJedi` on this trio dataset is high, with 96.9 % of the son genotypes that are consistent with his parent genotypes (see Appendix Table 6.2 for detailed results). As expected, most inconsistent genotypes concern SVs that were genotyped differently between `SVJedi` and GiaB (48.7 %, n = 154), confirming for those that they are probably wrongly assessed by `SVJedi`. However, these confirmed errors represent only 1.2 % of the dataset.

4.3.3 ONT dataset

`SVJedi` was applied on the same SV call set and for the same HG002 individual, but with sequencing data obtained by a different long read technology, namely the Oxford Nanopore sequencing technology. With a 44x PromethION dataset, `SVJedi` shows very similar genotyping performances as with the PacBio dataset (90.7 % accuracy and 86.2 % rate, see Table 4.6), highlighting its versatility with respect to long read sequencing technologies.

The performance of SVJedi was also evaluated on the ONT dataset sub-sampled at 30x for a fairer comparison with the PacBio sequencing data. The detailed results in Table 6.3 indicate very similar performance between the two sequencing technologies.

4.4 Comparison with other approaches

To find out where `SVJedi` stands in the state of the art, `SVJedi` has been compared to other methods that also perform SV genotyping.

DELETIONS

GiaB \ SVJedi predictions	0/0	0/1	1/1	./.
0/1	401	2,483	57	492
1/1	1	42	1,431	557

Genotyping accuracy: 88.7 %

Genotyping rate: 80.8 %

INSERTIONS

GiaB \ SVJedi predictions	0/0	0/1	1/1	./.
0/1	43	2,704	306	452
1/1	4	167	3,351	254

Genotyping accuracy: 92.1 %

Genotyping rate: 90.3 %

Table 4.6 – Contingency tables of `SVJedi` genotyping results on the real 44x Oxford Nanopore (PromethION) dataset of human individual HG002 with respect to the high confidence GiaB call set. Results for the 5,464 deletions (left) and 7,281 insertions (right) are indicated in two separated tables, where columns indicate `SVJedi` genotypes and rows GiaB ones. Grey labelled boxes correspond to identical predictions between the two methods. The number of genotypes that `SVJedi` fails to assess is indicated by the "./." column.

We consider in our benchmark two tools in addition to `SVJedi` that perform genotyping from a set of SVs with long reads: `Sniffles` genotyper (option `-Ivcf`) (Sedlazeck et al., 2018b) and `svviz2` (Spies et al., 2015). We added to our benchmark the `SVtyper` (Chiang et al., 2015) short read genotyping tool to compare the contribution of the two generations of sequencing technologies. Finally, we added to the benchmark two SV calling tools which include a genotyping module, `Sniffles` and `pbsv`[1], to highlight the differences between tools dedicated to SV discovery or SV genotyping. All the tools in the benchmark were evaluated on the 12,745 deletions and insertions of the GiaB call set in the HG002 individual. Except for the short read genotyping tool (`SVtyper`) that uses a 2 X 250 bp Illumina sequencing dataset for HG002 individual, that was down-sampled to 30x read depth, all other tools were run with a 30x PacBio CLR dataset for HG002 individual. FTP links are given in Appendix Section 6.1.1.

`SVJedi` was first compared to two tools, `Sniffles` and `svviz2`. Both tools, although not dedicated to genotyping, have options that allow them to also do SV genotyping from a set of SVs and with a long read sequencing dataset. Following the recommendations of `Sniffles`[2], reads were first aligned with `NGMLR` (version 0.2.7) on the human reference genome. Then, we used `Sniffles` (version 1.0.11) with the `-Ivcf` option to genotype the GiaB call set. For `svviz2`, reads were aligned on the human reference genome using `Minimap2` (version 2.17-r941). Genotyping was then performed from the sorted `Minimap2` alignments using `svviz2`

1. `https://github.com/PacificBiosciences/pbsv`
2. `https://github.com/fritzsedlazeck/Sniffles/wiki/SV-calling-for-a-population`

(version 2.0a3) with default parameters.

We also compared our approach with two SV discovery tools, `Sniffles` again but in its default mode and Pacific Biosciences SV caller, `pbsv`[3]. `Sniffles` was run with the `-genotype` parameter with the previously obtained `NGMLR` read alignments. For `pbsv`, reads were aligned to the reference genome using its own mapper `pbmm2` (version 1.1.0) with the `-sort`, `-median-filter` and `-sample` parameters. SVs were then discovered and called with `pbsv` (version 2.2.2) using default parameters. Both `Sniffles` and `pbsv` analyses do not always predict SVs at the exact simulated coordinates, so a predicted SV is considered identical as the expected SV if both SVs overlap by at least 70 %.

Finally, `SVJedi` was also compared to a SV genotyping approach based on short read data. To do this, the short reads are first aligned with SpeedSeq (Chiang et al., 2015) (version 0.1.2), then the known variants are genotyped with `SVtyper` (version 0.7.0) with the default settings.

All tools were run on a Linux 40-CPU node running at 2.60 GHz.

The genotyping results of the tools compared to the GiaB gold-standard call set are shown in the Table 4.7. We will first detail the results of the long read genotyping tools, then the results of the short read genotyping tools, followed by the genotyping results of the SV calling tools. Finally, we will present a stratified analysis to understand the limitations of the tools.

4.4.1 Comparison with other genotyping tools for long reads

`SVJedi` was compared on the PacBio HG002 dataset to two other tools that can genotype a set of SVs with long read sequencing data, `Sniffles` (Sedlazeck et al., 2018b) and `svviz2` (Spies et al., 2015).

As shown in Table 4.7, both `Sniffles -Ivcf` and `svviz2` have genotyping rates close to 100 % at the expense of lower genotyping accuracies (detailed results are given in Table 4.8). `Sniffles -Ivcf` is 10 % less accurate than `SVJedi` (82.0 % vs 92.2 %). `svviz2` obtained the lowest genotyping accuracy (65.9 %, with a 10 % difference between deletions and insertions).

3. `https://github.com/PacificBiosciences/pbsv`

	Deletions		Insertions		
Tool	Genotyping accuracy	Genotyping rate	Genotyping accuracy	Genotyping rate	Time
`SVJedi`	**91.7**	85.8	**92.5**	93.6	**2h25m**
`Sniffles -Ivcf`	82.5	**99.9**	81.7	**99.8**	17h16m
`svviz2`	72.5	**100**	61.0	**100**	5days*
`SVtyper` (Illumina dataset)	46.5	99.2	-	-	5h32m
`Sniffles` (discovery mode)	48.7	52.4	39.8	44.8	18h04m
`pbsv`	90.1	72.7	68.8	59.8	5h29m

Table 4.7 – Comparison of tools for genotyping the 12,745 deletions and insertions of the GiaB call set in the HG002 individual on 30x PacBio long read dataset. Three approaches are compared: using long read genotyping tools (first three tools), using a short read genotyping tool (`SVTyper`), and using long read discovery tools (last two tools). Runtimes were measured on a 40-CPU computing node. * `svviz2` is not parallelized.

4.4.1.1 Detailed genotyping results

The results of the predictions by type of genotype in relation to the GiaB set are detailed in the Table 4.8.

`Sniffles -Ivcf` has a very high genotyping rate compare to `SVJedi`, but on the other hand there are many false positives (a total of 2,286 variants are wrongly genotyped). We notice that `Sniffles` tends to wrongly predict mainly 1/1 genotypes instead of 0/1 genotypes and conversely 0/1 genotypes instead of 1/1 genotypes.

`svviz2` also has an excellent genotyping rate, however it has a large number of false positives. The main false genotype estimates are predicted to be 0/1 instead of 1/1, and `svviz2` predicts even 1,446 SV genotyped as 0/0, whereas they are predicted 1/1 in the GiaB set.

		SVJedi				**Sniffles –Ivcf**				**svviz2**			
		0/0	0/1	1/1	./.	0/0	0/1	1/1	./.	0/0	0/1	1/1	./.
GiaB	0/1	245	5,643	328	722	404	5,447	1,078	9	301	5,947	690	0
	1/1	3	323	4,958	523	47	757	4,996	7	1,446	1,903	2,458	0
		GT accuracy: 92.2 %				GT accuracy: 82.0 %				GT accuracy: 65.9 %			

Table 4.8 – Results on three long reads SV genotypers (`SVJedi`, `Sniffles -Ivcf`, `svviz2`) for genotyping the 12,745 deletions and insertions of the GiaB call set in the HG002 individual. For each tool, a contingency table of the estimated genotypes with respect to the high confidence GiaB call set is given, where columns indicate the tool estimated genotypes and rows GiaB ones. Grey labeled boxes correspond to identical predictions between the tool and the call set. The number of genotypes that each tool fails to assess is indicated by the "./." column. GT: genotyping.

4.4.2 Comparison with a short read based genotyping approach

For this same individual (HG002), some short read datasets are also available. We, therefore, can compare SV genotyping performances between two approaches and data types, namely long versus short reads. `SVJedi` predictions were compared to a SV genotyping tool for short reads, `SVtyper`, known as a reference tool in the state of the art (Chiang et al., 2015; Chander et al., 2019). Since `SVtyper` does not support insertion variants, we focus here only on deletions, and the 5,464 deletions from the GiaB call set were genotyped with `SVtyper` using a 2x250 bp 30x Illumina read dataset of HG002.

Table 4.7 shows that more than half of the deletions are genotyped differently by `SVtyper` than in the high confidence GiaB call set, resulting in a genotyping accuracy of only 46.5 %, while this percentage rises to 91.7 % for `SVJedi` with long reads. Remarkably, many of the discrepancies of `SVtyper` with GiaB are totally contradictory with 0/0 genotypes instead of 1/1 ones (see Table 4.9).

		`SVtyper` (del only)			
		0/0	0/1	1/1	./.
GiaB	0/1	1,852	1,561	12	8
	1/1	800	233	961	37

Genotyping accuracy: 46.5 %

Table 4.9 – `SVtyper` genotyping results for the 12,745 deletions and insertions of the GiaB call set on a real 30x Illumina dataset of the HG002 individual. The contingency table of the estimated genotypes with respect to the high confidence GiaB call set is given, where columns indicate the tool estimated genotypes and rows GiaB ones. Grey labeled boxes correspond to identical predictions between the tool and the call set. The number of genotypes that each tool fails to assess is indicated by the "./." column

4.4.3 Comparison with SV discovery approaches

One can wonder if these SVs could be easily detected and genotyped by long read SV discovery tools. We applied here two such tools, among the bests to date, `Sniffles` and the Pacific Biosciences SV caller, `pbsv` (Sedlazeck et al., 2018b; De Coster et al., 2019).

As a result, both tools obtained the lowest genotyping rates over all genotyping approaches: among the 12,745 SVs, only 6,127 were discovered by `Sniffles`, and 8,326 by `pbsv` (genotyping rates of 48.1 % and 65.3 % respectively, see Table 4.7). Detailed results are given in

Table 4.10, as expected, most of the missed SVs have an heterozygous genotype in the GiaB call set. More surprisingly, for the discovered SVs, their genotyping accuracy is overall smaller than with other approaches, with 43.9 % and 78.9 % for `Sniffles` and `pbsv`, respectively. In particular, `Sniffles` misassigns 85 % of the discovered SVs with a 1/1 genotype in GiaB as heterozygous. `pbsv` shows the same type of errors but mainly for insertions, resulting in an important difference of genotyping accuracy between deletions and insertions (90.1 vs 68.8 %). These results highlight the fact that SV discovery tools, are much less precise for the genotyping task than a dedicated genotyping tool.

		`Sniffles` (discovery)				**`pbsv`**			
		0/0	0/1	1/1	./.	0/0	0/1	1/1	./.
GiaB	0/1	349	2,189	146	4,254	0	4,110	149	2,679
	1/1	5	2,936	502	2,364	0	1,606	2,461	1,740
		GT accuracy: 48.8 %				GT accuracy: 78.9 %			

Table 4.10 – Genotype predictions for `Sniffles` and `pbsv` SV callers, for the 12,745 deletions and insertions of the GiaB call set on a real 30x PacBio dataset of the HG002 individual. The contingency table of the estimated genotypes with respect to the high confidence GiaB call set is given, where columns indicate the tool estimated genotypes and rows GiaB ones. Grey labeled boxes correspond to identical predictions between the tool and the call set. The number of genotypes that each tool fails to assess is indicated by the "./." column. GT: genotyping.

4.4.4 Stratified analysis

4.4.4.1 Performance by SV size categories

A stratified analysis of the genotyping performances of all three tools with respect to the SV size is presented in Figure 4.5.

We can observe that `SVJedi` has a better accuracy than `Sniffles -Ivcf` for all SV size classes. As mentioned previously, the lowest accuracy of `SVJedi` is observed for small SVs (<100 bp), but, apart from this size class, its accuracy is quite robust with respect to the size of SVs. On the opposite, `svviz2` obtained its best genotyping accuracy for the smallest SVs (<100 bp) and it rapidly drops for SVs larger than 250 bp, falling below 30 % for SV sizes between 1 kb and 10 kb.

When comparing between SV types, `svviz2` genotyping accuracy is significantly lower for insertions than deletions, with, in particular, less than 10 % of the insertions larger than

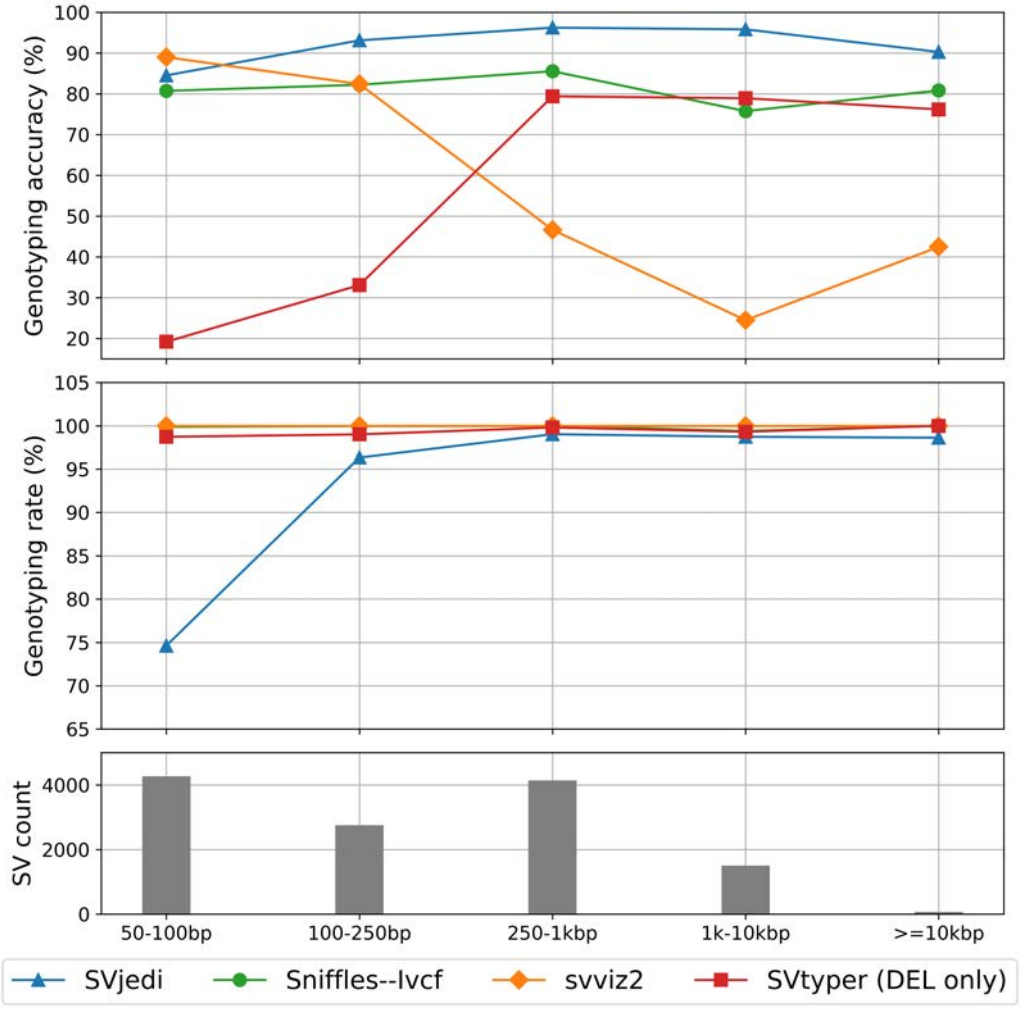

Figure 4.5 – Results of genotyping tools for the 12,745 deletions and insertions from the GiaB call set in the HG002 individual according to different SV size classes: 50 to 100 bp, 100 to 250 bp, 250 bp to 1 kb, 1 to 10 kb and $\geq$ to 10 kb. The two figures on top represent the genotyping accuracies and the genotyping rates of `SVJedi`, `Sniffles -Ivcf` and `svviz2` on a 30x PacBio dataset, and of `SVtyper` for deletions only on a 30x Illumina dataset. The bottom figure represents the SV count of each SV size class.

1 kb that are correctly genotyped (see Appendix Figure 6.1). This inability for genotyping large SVs can probably be explained by the way `svviz2` identifies informative reads for a given SV: only the reads mapped initially to the reference genome are selected before re-aligning them against both the reference and alternative alleles. Consequently, most reads coming from large insertion alternative alleles could probably not be used for estimating these genotypes. To a lesser extend, `Sniffles -Ivcf` genotyping accuracy is also lower for large insertions than large deletions (69.6 % for insertions vs 85.7 % for deletions, $\geq$ 1 kb), whereas `SVJedi`

genotyping accuracy is unaffected by SV type for all size classes.

4.4.4.2 Performance by genomic context and SV type

We then compared the genotyping performances with respect to the genomic context of the SVs (Figure 4.6).

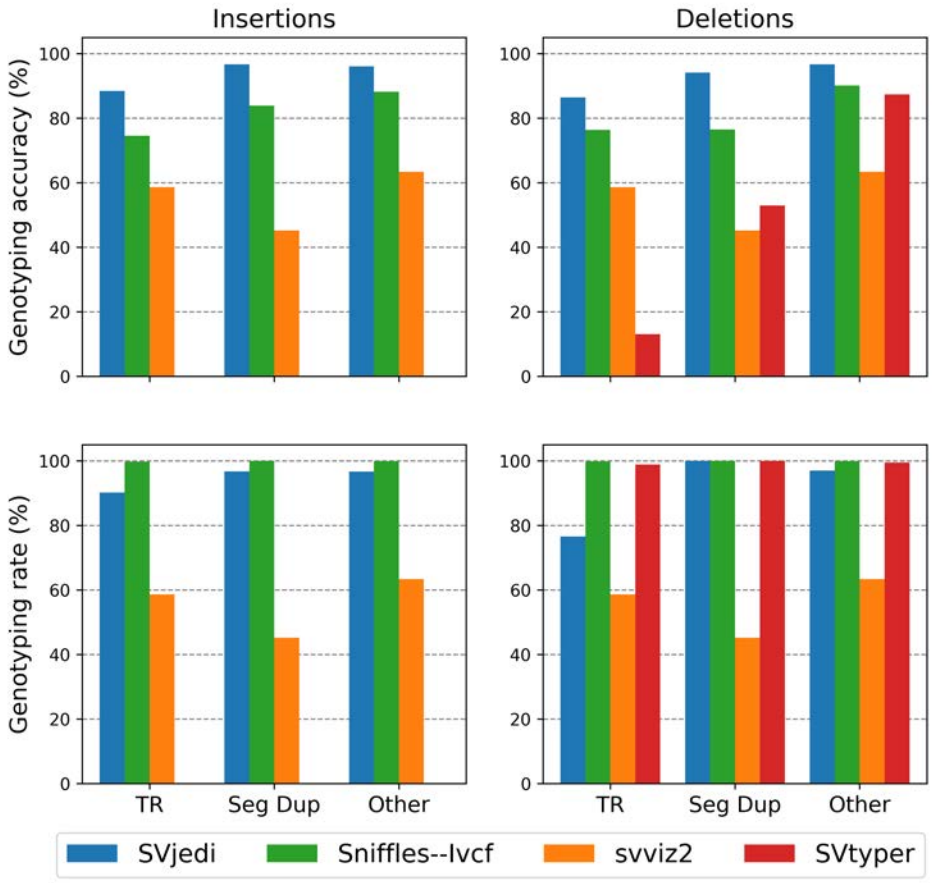

Figure 4.6 – Stratified analysis of genotyping accuracy and rate for several genotyping tools with respect to the genomic context of the SVs. Three categories of genomic context are considered: tandem repeats greater than 100 bp (TR), segmental duplications larger than 10 kb (SeqDup) and all other regions (Other).

As shown previously, SVs falling in a Tandem Repeat (TR) greater than 100 bp are harder to genotype with `SVJedi` with a 9 % decrease of accuracy for these SVs, compared to those outside TRs. A larger decrease of genotyping accuracy (14 %) is observed with `Sniffles`

`-Ivcf` in these regions. Although less frequent (here, only 48 concerned SVs), large segmental duplications, typically larger than 10 kb, are also likely to affect long read mapping accuracy and thus genotyping accuracy. `SVJedi` accuracy seemed not to be affected by these duplications, contrary to `Sniffles -Ivcf` (Figure 4.6).

Lastly, we can clearly observe, in Figure 4.6, that short reads based genotyping is much more impacted by the presence of TRs at the breakpoint. As expected, mapping reads in these regions is much more challenging for short than long reads. This demonstrates the higher benefit of using long reads and a dedicated genotyping tool such as `SVJedi` rather than short reads.

4.4.5 Runtime comparison

Importantly, `SVJedi` does not come with a high computational cost. On a 40-CPU computing node, genotyping the 12,745 SVs with the 30x PacBio HG002 dataset took only 2h25m (Table 4.7). The alignment step is actually the most time-consuming step and took 2h15m. Compared to other tools, `SVJedi` was the fastest among the tested ones (Table 4.7). Among the long read genotypers, `SVJedi` was 7 times faster than Sniffles`-Ivcf` and 50 times faster than `svviz2`. The large runtime of `svviz2` (more than 5 days) can be explained by the fact that it is not natively parallelized, when manually parallelized on 20 CPU (only 20 due to memory limits), it took roughly 11h.

4.5 Conclusion

In conclusion, we provide a novel SV genotyping approach for long read data, that showed good results on simulated and real datasets. The approach is implemented in the `SVJedi` software for most SV types (insertions, deletions, inversions and translocations). The robustness of our tool, `SVJedi`, was highlighted in this work, for several sequencing depths and error rates, but also related to the precision of the breakpoint positions. On a real human dataset with more than 12,000 insertions and deletions, `SVJedi` obtained a better genotyping accuracy than other tested genotyping and discovery tools. `SVJedi`, like the other tools, had more difficulties to accurately genotype small variants (<100 bp) and those located in large tandem repeat regions. However, `SVJedi` showed a more conservative behavior than other tools, with a lower genotyping rate for these most difficult SVs: instead of estimating an incorrect genotype, it favored not assigning any genotype at all.

This work also demonstrated that this is crucial to develop dedicated SV genotyping methods, as well as SV discovery methods. Firstly, because this is the only way to get evidence for the absence of SVs in a given individual. Secondly, and more surprisingly, because SV discovery tools are not as efficient and precise to genotype variants once they have been discovered, at least with long read data as was shown here. Indeed, without a priori knowledge of the variants, SV discovery is a much harder task than genotyping. Because the alternative allele is not known in discovery, discovery methods rely on fewer or noisier signals to identify the SVs than genotyping methods. Consequently, both approaches would likely benefit from different optimal parameter settings, with for instance discovery methods requiring a more stringent set of parameters to limit the false discovery rate. When the aim is strictly to genotype or compare individuals on a set of already known variants, we have shown that using as much as possible the known features of variants is much more efficient. Discovery methods and genotyping methods are complementary and are intended to different purposes, and therefore require different parameter settings.

Also, on real human data, we were able to quantify the impact of the sequencing technology on SV genotyping. Although this was expected that long read data would perform better than short read ones, the observed difference is considerable with a twofold increase of the genotyping accuracy with long reads. This is in particular due to the very poor performances obtained with short reads, that are ill-adapted to deal with the complex and repeat-rich regions often present at SV junctions. On the opposite, we show that the long-distance information contained in long reads can be efficiently used to discriminate between breakpoints, despite relatively high sequencing error rates and variability in sequencing coverage. This result underlines the relevance of such a method dedicated to genotyping with long read data.

Chapter 5

Conclusion

5.1 Conclusion

Structural variant polymorphism is widespread in genomes and is very heterogeneous in terms of type of variation and size of variation. The study of SVs is a major problem in research because SVs can impact species diversity and disrupt biological functions and even cause disease. For the problem of SV discovery, short read data faces an almost impossible problem to solve as some studies have shown. The limitations of short reads are directly related to the characteristics of the sequencing data, *i.e.* their size. Long read data solve this problem because they contain long-distance information and can cover the SVs. In the course of the thesis, several new tools have addressed the problem of SV discovery with long read data. The development of this type of tool has made it possible to build more reliable SV catalogs. It is only last year that two major studies have made it possible to build very high-confidence SV call sets, based on multiple detection methods and multiple types of data. We underline that the recency of methods and studies dedicated to SVs illustrates the importance of this field of research today. The creation of accurate catalogs of SVs with the long reads SV discovery tools then allows to address the issue of SV genotyping. After a state of the art study, we found that there is no dedicated SV genotyping method for long reads, despite the fact that long reads have proven to be of interest in the context of SVs. Once I identified this lack of a method entirely dedicated to SV genotyping with long reads, I addressed this issue.

During the thesis, I developed a new SV genotyping method for long read data. This method is an orginal work, different from the previously developped methods for short reads. We have designed our method so that the two allele sequences of each variant are represented, to avoid a bias of representation of the reference allele. I implemented this method in a tool called `SVJedi` which allows the genotyping of several types of SVs with long reads. Our tool is able to genotype deletions, insertions, inversions and translocations (breakpoints).

I then validated the method by applying `SVJedi` on several simulated datasets, the results on simulated data confirm the interest of our method. We also tested the robustness of `SVJedi` according to the depth of the sequencing dataset, according to the error rate and finally according to the precision of the breakpoint positions. Then I applied `SVJedi` on a real sequencing dataset to genotype 12,745 deletions and insertions of a high-confidence SV call set of GiaB. `SVJedi`'s genotype predictions are consistent with the GiaB's reliable genotype calls for both PacBio and ONT data.

Once our long read SV genotyping method was validated on both simulated and real datasets, we compared it to other genotyping tools for the same set of high-confidence SVs. Our tool proved to be much more powerful than two other tools capable of genotyping with long reads for a given set of SVs. Among these tools are `Sniffles -Ivcf` which does not represent alternative allele sequences, and `svviz2` which is a visualization tool rather than a genotyping tool, so it is very slow and it also wrongly predicts large SVs. The results of the benchmark highlighted the advantage of long reads over short reads for the problem of SV genotyping. The benchmark also indicates a benefit for the genotyping methods from a given set of SV compared to SV calling methods that predict genotype. We performed a stratified analysis showing that SV genotyping tools have more difficulty genotyping SVs localized in tandem repeat regions, especially for short reads. `SVJedi` is sensitive to the SVs present in tandem repeats as well, and we also showed that it is sensitive to the size of the SVs. `SVJedi` performs better at estimating genotypes of large variants than small ones. Finally, we have identified the main limitation of `SVJedi` which is the difficulty of genotyping overlapping SVs.

5.2 Perspectives

5.2.1 Short-term perspectives

A first short-term perspective is to apply `SVJedi` on a real dataset for homozygous SV for the reference allele. The GiaB call set on which `SVJedi` was evaluated, does not contain any variant with a genotype homozygous for the reference allele. However, the genotyping task can also estimate homozygous for the reference allele genotypes (0/0). I would, therefore, like to complete the evaluation with a set of SVs with homozygous reference genotypes (0/0). To do this, the study of Zook et al. (2020) did also identifies regions in which they are relatively sure that they do not contain any SV. I will then be able to evaluate the performance of `SVJedi`

for genotyping homozygous SVs for the reference allele by augmenting these regions with the dbVar database for example or with SVs detected in other studied individuals (e.g. individual NA19240 (Chaisson et al., 2019)).

Another short-term perspective is to apply `SVJedi` to more complex SV sets to study the versatility of our tool. For the moment, `SVJedi` has been applied to a real SV set obtained from multiple sequencing data and multiple methods, but with huge efforts of merging and curation. However, we can expect that a raw or uncurated set of SVs is more likely to contain redundant SV calls, as shown by the current SV databases. However, the first preliminary experiments on the robustness of `SVJedi` concerning the proximity, or even the overlapping of the SVs to be genotyped, have shown that our tool is sensitive to the proximity of SV calls. The redundancy of SV calls should have the same effect.

The limit of `SVJedi` for close SVs can be explained by two reasons. A first explanation is that in our method we choose to represent sequences for each variant. Thus, in case of overlapping close variants, the same sequences will be represented several times, which will result in a decrease in mappability because the allele sequences will no longer be so discriminating from each other. A second explanation is that our method considers each SV independently, so in the case where the variants are overlapping or nested, then potentially no allele sequence represents the studied sample.

A short-term solution to overcome the limit of close SVs would be to adapt the `SVJedi` parameters, more precisely the parameter of the size of the allele sequences (L_{adj}). By modifying this parameter, we could then partially solve the problem of mapping close variants. Also, to simplify the users' task of having to set the parameter of the size of the allele sequence, we could consider to add a preliminary step to `SVJedi` to estimate this parameter according to the input VCF and maybe even according to the size of the sequencing data.

5.2.2 Long-term perspectives

Other longer-term solutions can be considered to address the limitations of `SVJedi` concerning the issue of the proximity of SVs or overlapping SVs in the input VCF. We develop here three avenues that could be explored to address this problem.

A first solution to address the redundancy of SVs in the input VCF is to bypass the "problematic" SVs. We could then propose a user module upstream of `SVJedi` to filter the input VCF so that it contains no more redundant SVs. But this filter could prove to be a

disadvantage for users for whom the tool modifies their VCF arbitrarily.

Another solution to be considered to respond to close or nested SVs could be to build the allele sequences by genomic regions rather than by SVs. The problem with this representation is that we do not know the genotypes (it is the purpose of our method), nor the phasing of the variants (which variants are present on each haplotype). To answer this problem, we could consider doing for each region of close SVs a targeted assembly to reconstruct each haplotype. The rest of the method could remain unchanged: the method would align the reads against these haplotypes of the studied regions to then deduce the genotype of each SV from the alignments. This solution, even if it relies on an input VCF, may be close to the discovery of SVs for each region reconstructed by assembly.

Finally, a third solution to solve the problem of overlapping SVs is to develop a completely different approach based on variation graphs. As discussed in Chapter 2, variation graphs are highly interesting since they can represent the whole variability of the studied sequences, since all possible haplotypes are represented. Variation graphs are a beneficial solution to manage close to nested variants, such as complex SVs (Hickey et al., 2020). In the same way as for the short read data, such a method would first rely on the construction of the variation graph(s) and then align the long sequencing data on the graph paths corresponding to the haplotypes. The variation graph can for instance correspond to a specific region and includes one or more SVs, so we will calculate several variation graphs for each region of interest. This solution offers a gain in terms of computing resources and runtime by targeting regions of interest (but potentially exposes itself to biases in the selection of reads). Another stategy is to consider the variation graph as the whole genome. It is then a pangenome graph but this solution can be more expensive in terms of calculation and runtime. Finally, the genotypes for each SV would be estimated from the alignments on specific paths of the graph correponding to the haplotypes. This strategy also allows to phase the variants, this additional knowledge can be very useful in several applications. Moreover, the use of pan-genomic computational approaches may prove to be a relevant solution in the future for managing ever-increasing sequencing data while managing data variability (Consortium, 2018).

APPENDIX

6.1 Appendix to Chapter 4

6.1.1 Data accessibility and generation

The gold standard call set for individual HG002, provided by Genome in a Bottle (GiaB) Consortium, is available at the following link:

— `ftp://ftp-trace.ncbi.nlm.nih.gov/giab/ftp/data/AshkenazimTrio/analysis/NIST_SVs_Integration_v0.6/HG002_SVs_Tier1_v0.6.vcf.gz`

In our study, we focus only on variants of the Tier 1 call set which are isolated and sequence-resolved SVs, corresponding to the `PASS` filter of this VCF file. This call set represents 5,464 deletions and 7,281 insertions.

The Pacific Biosciences (PacBio) sequence datasets for Ashkenazi trio individuals provided by GiaB are available at the following links:

— `ftp://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data/AshkenazimTrio/HG002_NA24385_son/PacBio_MtSinai_NIST/`

— `ftp://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data/AshkenazimTrio/HG003_NA24149_father/PacBio_MtSinai_NIST/`

— `ftp://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data/AshkenazimTrio/HG004_NA24143_mother/PacBio_MtSinai_NIST/`

The PacBio data of the individual HG002 has a sequencing depth of 63x. The sequence data were sub-sampled using SAMtools **?** to a depth of 30x.

The ONT Promethion dataset for HG002 is available at the following link:

— `ftp://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data/AshkenazimTrio/HG002_NA24385_son/UCSC_Ultralong_OxfordNanopore_Promethion/`

The Illumina dataset for HG002 is available at the following link:

— `ftp://ftp-trace.ncbi.nlm.nih.gov/ReferenceSamples/giab/data/AshkenazimTrio/HG002_NA24385_son/NIST_Illumina_2x250bps/`

As with the PacBio data sample of the individual HG002, the Illumina data sample was sub-sampled to a sequencing depth of 30x as well.

6.1.2 SVJedi apply to real datasets

6.1.2.1 SVJedi results on real PacBio dataset

DELETIONS

		SVJedi predictions			
		0/0	0/1	1/1	./.
GiaB	0/1	188	3,065	44	136
	1/1	5	176	1,673	177

Genotyping accuracy: 92.0 %

Genotyping rate: 94.3 %

INSERTIONS

		SVJedi predictions			
		0/0	0/1	1/1	./.
GiaB	0/1	8	3,095	316	86
	1/1	1	142	3,599	34

Genotyping accuracy: 93.5 %

Genotyping rate: 98.4 %

Table 6.1 – Contingency tables of `SVJedi` genotyping results on the real 63x PacBio dataset of human individual HG002 with respect to the high confidence GiaB call set. Results for the 5,464 deletions (left) and 7,281 insertions (right) are indicated in two separated tables, where columns indicate `SVJedi` genotypes and rows GiaB ones. Grey labelled boxes correspond to identical predictions between the two methods. The number of genotypes that `SVJedi` fails to assess is indicated by the "./." column.

6.1.2.2 Mendelian Inheritance of the Ashkenazi trio

		Mother (HG004)								
		0/0			0/1			1/1		
Father (HG003)	0/0	0/0	0/1	1/1	0/0	0/1	1/1	0/0	0/1	1/1
		36	66	0	26	1106	45	6	387	36
	0/1	0/0	0/1	1/1	0/0	0/1	1/1	0/0	0/1	1/1
		41	1096	23	19	1196	473	5	507	512
	1/1	0/0	0/1	1/1	0/0	0/1	1/1	0/0	0/1	1/1
		10	352	13	4	530	485	0	108	3170

Mendelian Inheritance: 96.9 %

Table 6.2 – `SVJedi` genotyping results on 30x PacBio dataset of the Ashkenazi trio. The table shows for each genotype type predicted in the father (HG003) and mother (HG004), the number of SV predicted for each genotype in the son. Consistent predictions between son and parents are shown in green and inconsistent predictions are shown in red.

6.1.2.3 SVJedi results on a 30x ONT dataset

DELETIONS

		SVJedi predictions			
		0/0	0/1	1/1	./.
GiaB	0/1	386	2,326	51	670
	1/1	1	31	1,303	696

Genotyping accuracy: 88.6 %

Genotyping rate: 75.0 %

INSERTIONS

		SVJedi predictions			
		0/0	0/1	1/1	./.
GiaB	0/1	34	2,523	312	636
	1/1	4	153	3,247	372

Genotyping accuracy: 92.0 %

Genotyping rate: 86.2 %

Table 6.3 – Contingency tables of `SVJedi` genotyping results on the real 30x Oxford Nanopore (PromethION) dataset of human individual HG002 with respect to the high confidence GiaB call set. Results for the 5,464 deletions (left) and 7,281 insertions (right) are indicated in two separated tables, where columns indicate `SVJedi` genotypes and rows GiaB ones. Grey labelled boxes correspond to identical predictions between the two methods. The number of genotypes that `SVJedi` fails to assess is indicated by the "./." column.

6.1.3 Stratified analysis

6.1.3.1 Performance by SV size categories

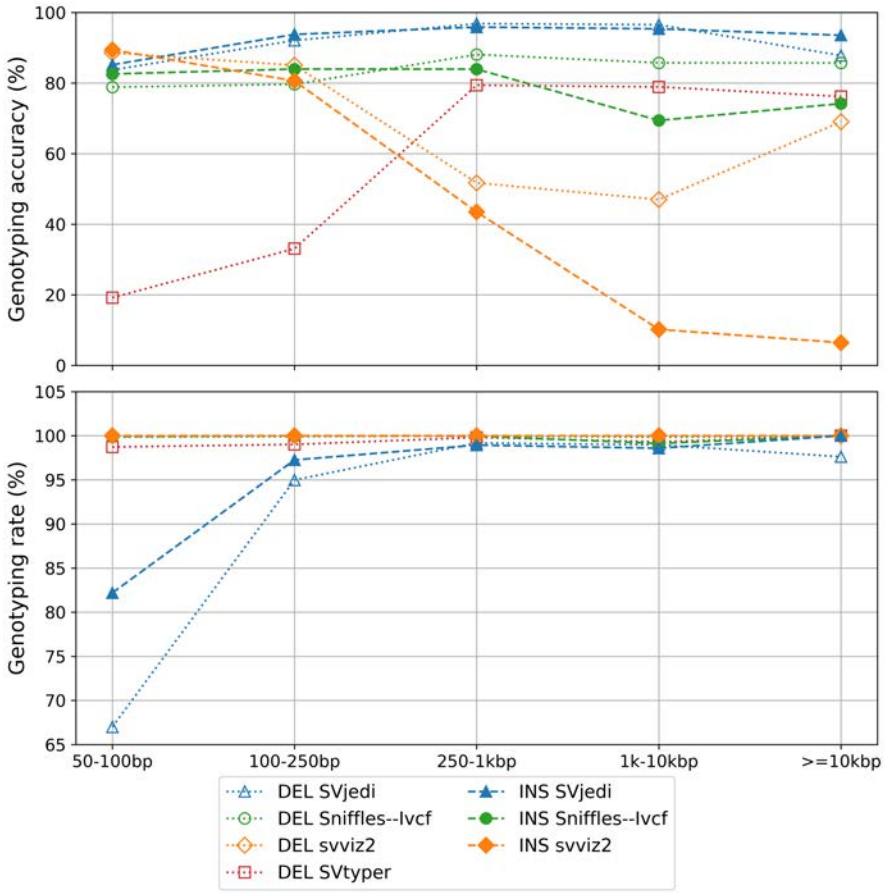

Figure 6.1 – Results of genotyping tools for the 5,464 deletions (dotted lines) and the 7,182 insertions (dashed lines) of the GiaB call set in the HG002 individual according to different SV size classes: 50 to 100 bp, 100 to 250 bp, 250 bp to 1 kb, 1 to 10 kb and $\geq$ to 10 kb. The two figures on top represent the genotyping accuracies and the genotyping rates of SVJedi, Sniffles`-Ivcf` and svviz2 on a 30x PacBio dataset, and of SVtyper for deletions only on a 30x Illumina dataset.

BIBLIOGRAPHY

Alkan, C., Coe, B. P., and Eichler, E. E. (2011). Genome structural variation discovery and genotyping. *Nature Reviews Genetics*, 12(5):363–376.

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. (1990). Basic local alignment search tool. *Journal of molecular biology*, 215(3):403–410.

Antaki, D., Brandler, W. M., and Sebat, J. (2018). SV2: accurate structural variation genotyping and de novo mutation detection from whole genomes. *Bioinformatics*, 34(10):1774–1777.

Ardui, S., Ameur, A., Vermeesch, J. R., and Hestand, M. S. (2018). Single molecule real-time (SMRT) sequencing comes of age: applications and utilities for medical diagnostics. *Nucleic acids research*, 46(5):2159–2168.

Audano, P. A., Sulovari, A., Graves-Lindsay, T. A., Cantsilieris, S., Sorensen, M., Welch, A. E., Dougherty, M. L., Nelson, B. J., Shah, A., Dutcher, S. K., et al. (2019). Characterizing the major structural variant alleles of the human genome. *Cell*, 176(3):663–675.

Bailey, J. A., Gu, Z., et al. (2002). Recent segmental duplications in the human genome. *Science*, 297(5583):1003–1007.

Bleidorn, C. (2016). Third generation sequencing: technology and its potential impact on evolutionary biodiversity research. *Systematics and biodiversity*, 14(1):1–8.

Carneiro, M. O., Russ, C., Ross, M. G., Gabriel, S. B., Nusbaum, C., and DePristo, M. A. (2012). Pacific biosciences sequencing technology for genotyping and variation discovery in human data. *BMC genomics*, 13(1):375.

Chaisson, M. J., Huddleston, J., Dennis, M. Y., Sudmant, P. H., Malig, M., Hormozdiari, F., Antonacci, F., Surti, U., Sandstrom, R., Boitano, M., et al. (2015). Resolving the complexity of the human genome using single-molecule sequencing. *Nature*, 517(7536):608–611.

Chaisson, M. J., Sanders, A. D., Zhao, X., Malhotra, A., Porubsky, D., Rausch, T., Gardner, E. J., Rodriguez, O. L., Guo, L., Collins, R. L., et al. (2019). Multi-platform discovery of haplotype-resolved structural variation in human genomes. *Nature communications*, 10(1):1–16.

Chaisson, M. J. and Tesler, G. (2012). Mapping single molecule sequencing reads using basic local alignment with successive refinement (BLASR): application and theory. *BMC bioinformatics*, 13(1):238.

Chander, V., Gibbs, R. A., and Sedlazeck, F. J. (2019). Evaluation of computational genotyping of structural variation for clinical diagnoses. *GigaScience*, 8(9):giz110.

Chen, S., Krusche, P., Dolzhenko, E., Sherman, R. M., Petrovski, R., Schlesinger, F., Kirsche, M., Bentley, D. R., Schatz, M. C., Sedlazeck, F. J., et al. (2019). Paragraph: a graph-based structural variant genotyper for short-read sequence data. *Genome biology*, 20(1):1–13.

Chen, X., Schulz-Trieglaff, O., Shaw, R., Barnes, B., Schlesinger, F., Källberg, M., Cox, A. J., Kruglyak, S., and Saunders, C. T. (2016). Manta: rapid detection of structural variants and indels for germline and cancer sequencing applications. *Bioinformatics*, 32(8):1220–1222.

Chen, Y., Ye, W., Zhang, Y., and Xu, Y. (2015). High speed BLASTN: an accelerated MegaBLAST search tool. *Nucleic acids research*, 43(16):7762–7768.

Chiang, C., Layer, R. M., Faust, G. G., Lindberg, M. R., Rose, D. B., Garrison, E. P., Marth, G. T., Quinlan, A. R., and Hall, I. M. (2015). SpeedSeq: ultra-fast personal genome analysis and interpretation. *Nature methods*, 12(10):966–968.

Consortium, T. C. P.-G. (2018). Computational pan-genomics: status, promises and challenges. *Briefings in Bioinformatics*, 19(1):118–135.

Currall, B. B., Chiangmai, C., Talkowski, M. E., and Morton, C. C. (2013). Mechanisms for structural variation in the human genome. *Current genetic medicine reports*, 1(2):81–90.

De Coster, W., De Rijk, P., De Roeck, A., De Pooter, T., D'Hert, S., Strazisar, M., Sleegers, K., and Van Broeckhoven, C. (2019). Structural variants identified by Oxford Nanopore PromethION sequencing of the human genome. *Genome research*, 29(7):1178–1187.

De Coster, W. and Van Broeckhoven, C. (2019). Newest methods for detecting structural variations. *Trends in biotechnology*.

de Lannoy, C., de Ridder, D., and Risse, J. (2017). The long reads ahead: de novo genome assembly using the MinION. *F1000Research*, 6.

Delage, W., Thevenon, J., and Lemaitre, C. (2020). Towards a better understanding of the low recall of insertion variants with short-read based variant callers. *bioRxiv*.

Derrien, T., Estellé, J., Sola, S. M., Knowles, D. G., Raineri, E., Guigó, R., and Ribeca, P. (2012). Fast computation and applications of genome mappability. *PloS one*, 7(1):e30377.

Dohm, J. C., Peters, P., Stralis-Pavese, N., and Himmelbauer, H. (2020). Benchmarking of long-read correction methods. *NAR Genomics and Bioinformatics*, 2(2):lqaa037.

Eggertsson, H. P., Kristmundsdottir, S., Beyter, D., Jonsson, H., Skuladottir, A., Hardarson, M. T., Gudbjartsson, D. F., Stefansson, K., Halldorsson, B. V., and Melsted, P. (2019). GraphTyper2 enables population-scale genotyping of structural variation using pangenome graphs. *Nature communications*, 10(1):1–8.

Eid, J., Fehr, A., Gray, J., Luong, K., Lyle, J., Otto, G., Peluso, P., Rank, D., Baybayan, P., Bettman, B., et al. (2009). Real-time DNA sequencing from single polymerase molecules. *Science*, 323(5910):133–138.

English, A. C., Salerno, W. J., and Reid, J. G. (2014). PBHoney: identifying genomic variants via long-read discordance and interrupted mapping. *BMC bioinformatics*, 15(1):180.

Escaramís, G., Docampo, E., and Rabionet, R. (2015). A decade of structural variants: description, history and methods to detect structural variation. *Briefings in functional genomics*, 14(5):305–314.

Fan, X., Chaisson, M., Nakhleh, L., and Chen, K. (2017). HySA: a Hybrid Structural variant Assembly approach using next-generation and single-molecule sequencing technologies. *Genome research*, 27(5):793–800.

Fang, L., Hu, J., Wang, D., and Wang, K. (2018). NextSV: a meta-caller for structural variants from low-coverage long-read sequencing data. *BMC bioinformatics*, 19(1):1–11.

Garrison, E., Sirén, J., Novak, A. M., Hickey, G., Eizenga, J. M., Dawson, E. T., Jones, W., Garg, S., Markello, C., Lin, M. F., et al. (2018). Variation graph toolkit improves read mapping by representing genetic variation in the reference. *Nature biotechnology*, 36(9):875–879.

Gong, L., Wong, C.-H., Cheng, W.-C., Tjong, H., Menghi, F., Ngan, C. Y., Liu, E. T., and Wei, C.-L. (2018). Picky comprehensively detects high-resolution structural variants in nanopore long reads. *Nature methods*, 15(6):455–460.

Goodwin, S., McPherson, J. D., and McCombie, W. R. (2016). Coming of age: ten years of next-generation sequencing technologies. *Nature Reviews Genetics*, 17(6):333.

Guan, P. and Sung, W.-K. (2016). Structural variation detection using next-generation sequencing data: a comparative technical review. *Methods*, 102:36–49.

Handsaker, R. E., Korn, J. M., Nemesh, J., and McCarroll, S. A. (2011). Discovery and genotyping of genome structural polymorphism by sequencing on a population scale. *Nature genetics*, 43(3):269–276.

Heather, J. M. and Chain, B. (2016). The sequence of sequencers: The history of sequencing DNA. *Genomics*, 107(1):1–8.

Heller, D. and Vingron, M. (2019). SVIM: structural variant identification using mapped long reads. *Bioinformatics*, 35(17):2907–2915.

Heller, D., Vingron, M., Church, G., Li, H., and Garg, S. (2020). SDip: A novel graph-based approach to haplotype-aware assembly based structural variant calling in targeted segmental duplications sequencing. *bioRxiv*.

Hickey, G., Heller, D., Monlong, J., Sibbesen, J. A., Sirén, J., Eizenga, J., Dawson, E. T., Garrison, E., Novak, A. M., and Paten, B. (2020). Genotyping structural variants in pangenome graphs using the vg toolkit. *Genome biology*, 21(1):1–17.

Ho, S. S., Urban, A. E., and Mills, R. E. (2019). Structural variation in the sequencing era. *Nature Reviews Genetics*, pages 1–19.

Huddleston, J., Chaisson, M. J., Steinberg, K. M., Warren, W., Hoekzema, K., Gordon, D., Graves-Lindsay, T. A., Munson, K. M., Kronenberg, Z. N., Vives, L., et al. (2017). Discovery and genotyping of structural variation from long-read haploid genome sequence data. *Genome research*, 27(5):677–685.

Jain, M., Fiddes, I. T., Miga, K. H., Olsen, H. E., Paten, B., and Akeson, M. (2015). Improved data analysis for the MinION nanopore sequencer. *Nature methods*, 12(4):351–356.

Jain, M., Koren, S., Miga, K. H., Quick, J., Rand, A. C., Sasani, T. A., Tyson, J. R., Beggs, A. D., Dilthey, A. T., Fiddes, I. T., et al. (2018). Nanopore sequencing and assembly of a human genome with ultra-long reads. *Nature biotechnology*, 36(4):338.

Jeffares, D. C., Jolly, C., Hoti, M., Speed, D., Shaw, L., Rallis, C., Balloux, F., Dessimoz, C., Bähler, J., and Sedlazeck, F. J. (2017). Transient structural variations have strong effects on quantitative traits and reproductive isolation in fission yeast. *Nature communications*, 8(1):1–11.

Khorsand, P. and Hormozdiari, F. (2019). Nebula: Ultra-efficient mapping-free structural variant genotyper. *bioRxiv*, page 566620.

Kiełbasa, S. M., Wan, R., Sato, K., Horton, P., and Frith, M. C. (2011). Adaptive seeds tame genomic sequence comparison. *Genome research*, 21(3):487–493.

Kosugi, S., Momozawa, Y., Liu, X., Terao, C., Kubo, M., and Kamatani, Y. (2019). Comprehensive evaluation of structural variation detection algorithms for whole genome sequencing. *Genome biology*, 20(1):117.

La, S., Haghshenas, E., and Chauve, C. (2017). LRCstats, a tool for evaluating long reads correction methods. *Bioinformatics*, 33(22):3652–3654.

Langmead, B. and Salzberg, S. L. (2012). Fast gapped-read alignment with Bowtie 2. *Nature methods*, 9(4):357.

Layer, R. M., Chiang, C., Quinlan, A. R., and Hall, I. M. (2014). LUMPY: a probabilistic framework for structural variant discovery. *Genome biology*, 15(6):R84.

Lecompte, L., Peterlongo, P., Lavenier, D., and Lemaitre, C. (2020). SVJedi: genotyping structural variations with long reads. *Bioinformatics*. btaa527.

Levene, M. J., Korlach, J., Turner, S. W., Foquet, M., Craighead, H. G., and Webb, W. W. (2003). Zero-mode waveguides for single-molecule analysis at high concentrations. *science*, 299(5607):682–686.

Li, H. (2011). A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. *Bioinformatics*, 27(21):2987–2993.

Li, H. (2016). Minimap and miniasm: fast mapping and de novo assembly for noisy long sequences. *Bioinformatics*, 32(14):2103–2110.

Li, H. (2018). Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, 34(18):3094–3100.

Li, H. and Durbin, R. (2009). Fast and accurate short read alignment with Burrows–Wheeler transform. *bioinformatics*, 25(14):1754–1760.

Logsdon, G. A., Vollger, M. R., and Eichler, E. E. (2020). Long-read human genome sequencing and its applications. *Nature Reviews Genetics*, pages 1–18.

Lu, H., Giordano, F., and Ning, Z. (2016). Oxford Nanopore MinION sequencing and genome assembly. *Genomics, proteomics & bioinformatics*, 14(5):265–279.

Mahmoud, M., Gobet, N., Cruz-Dávalos, D. I., Mounier, N., Dessimoz, C., and Sedlazeck, F. J. (2019). Structural variant calling: the long and the short of it. *Genome biology*, 20(1):246.

Medvedev, P., Stanciu, M., and Brudno, M. (2009). Computational methods for discovering structural variation with next-generation sequencing. *Nature methods*, 6(11):S13–S20.

Meleshko, D., Marks, P., Williams, S., and Hajirasouliha, I. (2019). Detection and assembly of novel sequence insertions using Linked-Read technology. *bioRxiv*, page 551028.

Meng, G., Tan, Y., Fan, Y., Wang, Y., Yang, G., Fanning, G., and Qiu, Y. (2019). TSD: A computational tool to study the complex structural variants using PacBio targeted sequencing data. *G3: Genes, Genomes, Genetics*, 9(5):1371–1376.

Miga, K. H., Koren, S., Rhie, A., Vollger, M. R., Gershman, A., Bzikadze, A., Brooks, S., Howe, E., Porubsky, D., Logsdon, G. A., et al. (2019). Telomere-to-telomere assembly of a complete human X chromosome. *BioRxiv*, page 735928.

Mills, R. E., Walter, K., Stewart, C., Handsaker, R. E., Chen, K., Alkan, C., Abyzov, A., Yoon, S. C., Ye, K., Cheetham, R. K., et al. (2011). Mapping copy number variation by population-scale genome sequencing. *Nature*, 470(7332):59–65.

Mohiyuddin, M., Mu, J. C., Li, J., Bani Asadi, N., Gerstein, M. B., Abyzov, A., Wong, W. H., and Lam, H. Y. (2015). MetaSV: an accurate and integrative structural-variant caller for next generation sequencing. *Bioinformatics*, 31(16):2741–2744.

Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453.

Nielsen, R., Paul, J. S., Albrechtsen, A., and Song, Y. S. (2011). Genotype and SNP calling from next-generation sequencing data. *Nature Reviews Genetics*, 12(6):443–451.

Pang, A. W., MacDonald, J. R., Pinto, D., Wei, J., Rafiq, M. A., Conrad, D. F., Park, H., Hurles, M. E., Lee, C., Venter, J. C., et al. (2010). Towards a comprehensive structural variation map of an individual human genome. *Genome biology*, 11(5):R52.

Phan, L., Hsu, J., Le Quang Minh Tri, M. W., Mansour, T., Kai, Y., Garner, J., Lopez, J., and Busby, B. (2016). dbVar structural variant cluster set for data analysis and variant comparison. *F1000Research*, 5.

Quinlan, A. R. and Hall, I. M. (2012). Characterizing complex structural variation in germline and somatic genomes. *Trends in Genetics*, 28(1):43–53.

Rausch, T., Zichner, T., Schlattl, A., Stütz, A. M., Benes, V., and Korbel, J. O. (2012). DELLY: structural variant discovery by integrated paired-end and split-read analysis. *Bioinformatics*, 28(18):i333–i339.

Rhoads, A. and Au, K. F. (2015). PacBio sequencing and its applications. *Genomics, proteomics & bioinformatics*, 13(5):278–289.

Ritz, A., Bashir, A., Sindi, S., Hsu, D., Hajirasouliha, I., and Raphael, B. J. (2014). Characterization of structural variants with single molecule and hybrid sequencing approaches. *Bioinformatics*, 30(24):3458–3466.

Sanchis-Juan, A., Stephens, J., French, C. E., Gleadall, N., Mégy, K., Penkett, C., Shamardina, O., Stirrups, K., Delon, I., Dewhurst, E., et al. (2018). Complex structural variants in Mendelian disorders: identification and breakpoint resolution using short-and long-read genome sequencing. *Genome medicine*, 10(1):95.

Sanger, F., Nicklen, S., and Coulson, A. R. (1977). DNA sequencing with chain-terminating inhibitors. *Proceedings of the national academy of sciences*, 74(12):5463–5467.

Sedlazeck, F. J., Lee, H., Darby, C. A., and Schatz, M. C. (2018a). Piercing the dark matter: bioinformatics of long-range sequencing and mapping. *Nature Reviews Genetics*, 19(6):329–346.

Sedlazeck, F. J., Rescheneder, P., Smolka, M., Fang, H., Nattestad, M., von Haeseler, A., and Schatz, M. C. (2018b). Accurate detection of complex structural variations using single-molecule sequencing. *Nature methods*, 15(6):461–468.

Shao, H., Ganesamoorthy, D., Duarte, T., Cao, M. D., Hoggart, C. J., and Coin, L. J. (2018). npInv: accurate detection and genotyping of inversions using long read sub-alignment. *BMC bioinformatics*, 19(1):261.

Shendure, J. and Ji, H. (2008). Next-generation DNA sequencing. *Nature biotechnology*, 26(10):1135.

Sibbesen, J. A., Maretty, L., and Krogh, A. (2018). Accurate genotyping across variant classes and lengths using variant graphs. *Nature genetics*, 50(7):1054–1059.

Smith, T. F. and Waterman, M. S. (1981). Comparison of biosequences. *Advances in applied mathematics*, 2(4):482–489.

Sović, I., Šikić, M., Wilm, A., Fenlon, S. N., Chen, S., and Nagarajan, N. (2016). Fast and sensitive mapping of nanopore sequencing reads with GraphMap. *Nature communications*, 7(1):1–11.

Spies, N., Weng, Z., Bishara, A., McDaniel, J., Catoe, D., Zook, J. M., Salit, M., West, R. B., Batzoglou, S., and Sidow, A. (2017). Genome-wide reconstruction of complex structural variants using read clouds. *Nature methods*, 14(9):915.

Spies, N., Zook, J. M., Salit, M., and Sidow, A. (2015). Svviz: A read viewer for validating structural variants. *Bioinformatics*, 31(24):3994–3996.

Stancu, M. C., Van Roosmalen, M. J., Renkens, I., Nieboer, M. M., Middelkamp, S., De Ligt, J., Pregno, G., Giachino, D., Mandrile, G., Valle-Inclan, J. E., et al. (2017). Mapping and phasing of structural variation in patient genomes using nanopore sequencing. *Nature communications*, 8(1):1–13.

Stöcker, B. K., Köster, J., and Rahmann, S. (2016). SimLoRD: simulation of long read data. *Bioinformatics*, 32(17):2704–2706.

Tham, C. Y., Tirado-Magallanes, R., Goh, Y., Fullwood, M. J., Koh, B. T., Wang, W., Ng, C. H., Chng, W. J., Thiery, A., Tenen, D. G., et al. (2020). NanoVar: accurate characterization of patients' genomic structural variants using low-depth nanopore sequencing. *Genome Biology*, 21(1):1–15.

Treangen, T. J. and Salzberg, S. L. (2012). Repetitive DNA and next-generation sequencing: computational challenges and solutions. *Nature Reviews Genetics*, 13(1):36–46.

van Dijk, E. L., Jaszczyszyn, Y., Naquin, D., and Thermes, C. (2018). The third revolution in sequencing technology. *Trends in Genetics*, 34(9):666–681.

Voskoboynik, A., Neff, N. F., Sahoo, D., Newman, A. M., Pushkarev, D., Koh, W., Passarelli, B., Fan, H. C., Mantalas, G. L., Palmeri, K. J., et al. (2013). The genome sequence of the colonial chordate, Botryllus schlosseri. *elife*, 2:e00569.

Weckselblatt, B. and Rudd, M. K. (2015). Human structural variation: mechanisms of chromosome rearrangements. *Trends in Genetics*, 31(10):587–599.

Weisenfeld, N. I., Kumar, V., Shah, P., Church, D. M., and Jaffe, D. B. (2017). Direct determination of diploid genome sequences. *Genome research*, 27(5):757–767.

Wenger, A. M., Peluso, P., Rowell, W. J., Chang, P.-C., Hall, R. J., Concepcion, G. T., Ebler, J., Fungtammasan, A., Kolesnikov, A., Olson, N. D., et al. (2019). Accurate circular consensus long-read sequencing improves variant detection and assembly of a human genome. *Nature biotechnology*, 37(10):1155–1162.

Wick, R. R., Judd, L. M., and Holt, K. E. (2019). Performance of neural network basecalling tools for Oxford Nanopore sequencing. *Genome biology*, 20(1):129.

Zarate, S., Carroll, A., Krasheninina, O., Sedlazeck, F. J., Jun, G., Salerno, W., Boerwinkle, E., and Gibbs, R. (2018). Parliament2: fast structural variant calling using optimized combinations of callers. *BioRxiv*, page 424267.

Zheng, G. X., Lau, B. T., Schnall-Levin, M., Jarosz, M., Bell, J. M., Hindson, C. M., Kyriazopoulou-Panagiotopoulou, S., Masquelier, D. A., Merrill, L., Terry, J. M., et al. (2016). Haplotyping germline and cancer genomes with high-throughput linked-read sequencing. *Nature biotechnology*, 34(3):303.

Zhou, A., Lin, T., and Xing, J. (2019). Evaluating nanopore sequencing data processing pipelines for structural variation identification. *Genome biology*, 20(1):237.

Zhu, S., Emrich, S. J., and Chen, D. Z. (2018). Inversion detection using PacBio long reads. *International Journal of Data Mining and Bioinformatics*, 20(3):230–246.

Zook, J. M., Catoe, D., McDaniel, J., Vang, L., Spies, N., Sidow, A., Weng, Z., Liu, Y., Mason, C. E., Alexander, N., et al. (2016). Extensive sequencing of seven human genomes to characterize benchmark reference materials. *Scientific data*, 3(1):1–26.

Zook, J. M., Hansen, N. F., Olson, N. D., Chapman, L., Mullikin, J. C., Xiao, C., Sherry, S., Koren, S., Phillippy, A. M., Boutros, P. C., et al. (2020). A robust benchmark for detection of germline large deletions and insertions. *Nature biotechnology*, pages 1–9.

SCIENTIFIC CONTRIBUTIONS

Publications

Lolita Lecompte, Pierre Peterlongo, Dominique Lavenier, Claire Lemaitre, "SVJedi: Genotyping structural variations with long reads", accepted in *Bioinformatics*, [btaa527]

Camille Marchet, Pierre Morisse, Lolita Lecompte, Arnaud Lefebvre, Thierry Lecroq, Pierre Peterlongo, Antoine Limasset, "ELECTOR: evaluator for long reads correction methods", *NAR Genomics and Bioinformatics*, Volume 2, Issue 1, March 2020, [lqz015]

Camille Marchet, Lolita Lecompte, Corinne Da Silva, Corinne Cruaud, Jean-Marc Aury, Jacques Nicolas, Pierre Peterlongo, "De novo clustering of long reads by gene from transcriptomics data", *Nucleic Acids Research*, Volume 47, Issue 1, 10 January 2019, Page e2, [gky834]

Camille Marchet, Lolita Lecompte, Antoine Limasset, Lucie Bittner, Pierre Peterlongo, "A resource-frugal probabilistic dictionary and applications in bioinformatics." *Discrete Applied Mathematics*, Volume 274, March 2020, Page 92-102, [doi]

Communications

Talk

SeqBIM 2019 - SVJedi: Structural variations genotyping with long reads
December 16th-17th, Marne-la-Vallée, France (virtual)

HitSeq 2019 - Genotyping structural variations using long reads data
July 22nd-23rd, Basel, Switzerland

JOBIM 2019 - Genotyping Structural Variations using Long Reads data
July 2nd-5th, Nantes, France

Poster

HitSeq 2019 Genotyping structural variations using long reads data
July 22nd-23rd, Basel, Switzerland

JOBIM 2018 - ELECTOR: EvaLuation of Error Correction Tools for lOng Reads
July 3rd-6th, Marseille, France

List of Figures

1 Aperçu des étapes de `SVJedi` pour le génotypage de délétion 16

1.1 Double-stranded DNA molecule 23
1.2 Unbalanced SVs: deletion and insertion 25
1.3 Inversion 26
1.4 Translocation 26
1.5 Complex SVs: deletion-duplication, deletion-inversion-deletion, duplication-inversion-duplication 27
1.6 Homologous Recombination and Non-Homologous End-Joining repair mechanisms 28
1.7 Inversion resulting from NAHR mechanism 29
1.8 Whole-genome shotgun sequencing 30
1.9 Single-end and paired-end sequencing 33
1.10 Overview of the PacBio sequencing technology 35
1.11 Circular Consensus Sequence read 36
1.12 Overview of the ONT sequencing 37

2.1 Global alignment of two sequences 48
2.2 Local alignment of two sequences 48
2.3 Main steps of mapping-based SV discovery methods 55
2.4 Read depth strategy 57
2.5 Read pair strategy 57
2.6 Split read alignment and soft-clipped read alignment 58
2.7 Local assembly based strategy 59
2.8 SV signatures detected by `SVIM` 64

3.1 Overview of `SVJedi` for deletion genotyping 88

3.2 Representative allele sequences for deletions 89
3.3 Representative allele sequences for insertions 90
3.4 Representative allele sequences for inversions 91
3.5 Representative allele sequences for translocations 93
3.6 Definition of the different distances used to select informative alignments between an allele sequence and a read 95
3.7 Genotyping accuracy according to the *err* parameter for the estimation of genotype likelihoods on a simulated dataset 102
3.8 `SVJedi` genotyping results according to L_{adj} size on a simulated dataset . . . 104
3.9 The four types of `BND` in the `<ALT>` field of the VCFs 108
3.10 Example for all possible `BND` types 108

4.1 `SVJedi` genotyping accuracy results according to different sequencing depth . 117
4.2 `SVJedi` results according to severa sequencing error rates 119
4.3 `SVJedi` genotyping accuracy results according to the precision of breakpoint position in the input deletion file 119
4.4 `SVJedi` results according to the proximity and overlap of the deletions 121
4.5 Benchmark of SV genotyping tools for the GiaB gold standard SV call set . . 131
4.6 Stratified analysis of genotyping accuracy and rate for several genotyping tools with respect to the genomic context of the SVs 132

6.1 Benchmark of SV genotyping tools for the 5,464 deletions and the 7,182 insertions of GiaB gold standard SV call set 142

List of Tables

2.1 List of SV calling tools for long read data 66

2.2 List of SV genotypers 83

3.1 `SVJedi` genotyping results on simulated dataset with genotype estimation based on decision thresholds 101

3.2 `SVJedi` genotyping results on simulated dataset with genotype estimation based maximum likelihood 102

4.1 `SVJedi` results on PacBio 30x simulated data of human chromosome 1 with 1,000 deletions from dbVar 115

4.2 textttSVJedi results on two 30x PacBio simulated data of human chromosome 1 with 450 inversions and 450 translocations from dbVar 116

4.3 Distribution of sequencing error types of PacBio simulated data 118

4.4 `SVJedi` genotyping results on real 30x PacBio dataset of human individual HG002 for the GiaB gold standard SV call set 123

4.5 `SVJedi` results on the HG002 real 30x PacBio dataset between several classes of SVs 124

4.6 `SVJedi` genotyping results on the real 44x Oxford Nanopore (PromethION) dataset of human individual HG002 for the GiaB gold standard SV call set . 126

4.7 Comparison of tools for genotyping the GiaB call set in the HG002 individual on 30x PacBio long read dataset 128

4.8 Detailed genoytyping results of three long reads SV genotypes for the GiaB gold standard call set: `SVJedi`, `Sniffles -Ivcf`, `svviz2` 128

4.9 Detailed results of the SV genotyping tool for short reads, `SVtyper`, on real 30x Illumina dataset of the human individual HG002 for the GiaB gold standard call set 129

4.10 Detailed genotyping results for SV callers `Sniffles` and `pbsv` on real 30x Pacbio dataset of the human individual HG002 for the GiaB gold standard call set 130

6.1 `SVJedi` genotyping results on the real 63x PacBio dataset of human individual HG002 with respect for the GiaB gold standard call set 140

6.2 `SVJedi` genotyping results on 30x PacBio dataset of the Ashkenazi trio for the GiaB gold standard call set 141

6.3 `SVJedi` genotyping results on real 30x ONT dataset of human individual HG002 for the GiaB gold standard call set 141

Abbreviations

BAM Binary Sequence Alignment/Map

BND breakpoint

bp base pairs

CCS Circular Consensus Sequence

CLR Long Continuous Reads

CNV copy number variant

ddNTP dideoxynucleotide

DEL deletion

DNA deoxyribonucleic acid

dNTP deoxyribonucleotides triphosphate

DUP duplication

Gb Gigabases

HiFi High-Fidelity

HMM Hidden Markov Model

HR Homologous Recombination

INS insertion

INV inversion

kb kilobases

Mb Megabases

NAHR Non-allelic Homologous Recombination

NGS Next Generation Sequencing

NHEJ Non-Homologous End-Joining

ONT Oxford Nanopore Technology

PacBio Pacific Biosciences

PAF pairing mapping format

PCR Polymerase Chain Reaction

RNA ribonucleic acid

SAM Sequence Alignment/Map

Seq Dup segmental duplication

SNP single nucleotide polymorphism

SV structural variant

TRANS translocation

ZMW zero-mode waveguide

UNIVERSITÉ DE RENNES 1

Titre : Génotypage de variations de structure avec des données de séquençage longues lectures

Mot clés : Bioinformatique, génomique, variants de structure, génotypage

Résumé : Les variants de structure (SVs) sont des réarrangements génomiques de plus de 50 paires de base et restent encore aujourd'hui peu étudiés malgré les impacts importants qu'ils peuvent avoir sur le fonctionnement des génomes. Récemment, les technologies de séquençage de troisième génération ont été développées et produisent des données de longues lectures qui s'avèrent très utiles car elles peuvent chevaucher les réarrangements. À l'heure actuelle, les méthodes bioinformatiques se sont concentrées sur le problème de la découverte de SVs avec des données de longues lectures. Aucune méthode n'a cependant été proposée pour répondre spécifiquement à la question du génotypage de SVs avec ce même type de données. L'objectif du génotypage de SVs vise pour un ensemble de SVs donné à évaluer les allèles présents dans un nouvel échantillon séquencé. Cette thèse propose une nouvelle méthode pour génotyper des SVs avec des longues lectures et repose sur la représentation des séquences des allèles. Notre méthode a été implémentée dans l'outil SVJedi. Nous avons testé notre outil à la fois sur des données simulées et réelles afin de valider notre méthode. SVJedi obtient une précision élevée qui dépasse les performances des autres outils de génotypage de SVs, notamment des outils de détection de SVs et des outils de génotypage de SVs de lectures courtes.

Title: Structural variant genotyping with long read data

Keywords: Bioinformatics, genomics, structural variants, genotyping

Abstract: Structural Variants (SVs) are genomic rearrangements of more than 50 base pairs. Since SVs can reach several thousand base pairs, they can have huge impacts on genome functions, studying SVs is, therefore, of great interest. Recently, a new generation of sequencing technologies has been developed and produce long read data of tens of thousand of base pairs which are particularly useful for spanning over SV breakpoints. So far, bioinformatics methods have focused on the SV discovery problem with long read data. However, no method has been proposed to specifically address the issue of genotyping SVs with long read data. The purpose of SV genotyping is to assess for each variant of a given input set which alleles are present in a newly sequenced sample. This thesis proposes a new method for genotyping SVs with long read data, based on the representation of each allele sequences. We also defined a set of conditions to consider a read as supporting an allele. Our method has been implemented in a tool called SVJedi. Our tool has been validated on both simulated and real human data and achieves high genotyping accuracy. We show that SVJedi obtains better performances than other existing long read genotyping tools and we also demonstrate that SV genotyping is considerably improved with SVJedi compared to other approaches, namely SV discovery and short read SV genotyping approaches.