Master Bioinformatique

Rapport de stage

Construction du pangénome de *Pseudogymnoascus destructans*

Autrice :
Anaïs Andre

Tutrice pédagogique :
Sèverine Berard

Année 2024 - 2025

Remerciements

Je tiens à remercier chaleureusement ma tutrice, Sèverine Berard, qui m'a accompagnée avec patience et bienveillance tout au long de ce stage. Son soutien a été précieux, tout en me laissant la liberté de prendre des initiatives et de développer mon autonomie, ce qui a grandement enrichi mon expérience.

Je tiens aussi à remercier l'étudiant Malo Lorazo, dont le travail a constitué une base solide pour mes recherches. Sa rigueur et son engagement m'ont grandement aidé à avancer efficacement et à approfondir mon travail.

Table des matières

1 Introduction

Dans le cadre de ma formation en Master 1 Bioinformatique à l'univeristé de Montpellier, j'ai eu l'opportunité d'effectuer un stage de deux mois au sein de l'Institut des Sciences de l'Évolution de Montpellier (ISEM). Ce stage m'a permis de découvrir une nouvelle manière de représenter les génomes : les pangénomes. Contrairement aux représentations classiques basées sur un génome de référence unique, les pangénomes permettent d'intégrer plusieurs séquences issues d'individus différents d'une même espèce. Ils offrent ainsi une vision plus complète de la diversité génétique et rendent visibles les variations structurales.

Afin de me familiariser avec cette nouvelle approche, le stage a été structuré en deux phases complémentaires. La première a été consacrée à l'apprentissage des outils bioinformatiques permettant de générer des pangénomes et de les visualiser. La seconde s'est focalisée sur l'application de ces différents outils dans un contexte de recherche, afin d'explorer leur potentiel dans l'analyse de la diversité génomique.

L'étude s'est portée sur l'espèce *Pseudogymnoascus destructans*, un champignon pathogène envahissant les tissus des chauves-souris et responsable du syndrome du museau blanc. Dans ce cadre, je me suis particulièrement intéressée aux inversions génomiques qui composent leurs génomes, qui sont des réarrangements structuraux pouvant jouer un rôle important dans l'évolution et la diversité génétique. Mon objectif était de repérer ces inversions à partir de la représentation pangénomique.

2 Définitions

Pour mieux comprendre les sections qui suivront, cette partie présente les définitions des termes biologiques et informatiques qui seront utilisés dans ce rapport. Les définitions sont triées par ordre lexicographique.

Fasta

Il s'agit d'un format de fichier texte couramment utilisé pour représenter une ou plusieurs séquences biologiques d'ADN, d'ARN ou de protéines. Chaque séquence est encodée sous forme d'une chaîne de lettres correspondant aux acides nucléiques ou aminés, selon la nomenclature IUPAC. La première ligne de chaque entrée, précédée du symbole ">", contient une description de la séquence incluant généralement un identifiant, le nom de l'espèce ou d'autres informations sur l'individu. Dans le cadre de la construction d'un pangénome avec des outils bioinformatiques, il est essentiel que le premier mot après ">" soit un identifiant unique car il servira à identifier et suivre la séquence tout au long du processus.

```
>red bovine
TGGCTGATCACTAATGGTTACCCAGTAGGCTGACTATAGATCATAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGT

>orange bovine
TGGCTGATCACTAATGGTTACCCAGTAGGCTGACTATAGATCATAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGT

>blue bovine
TGGCTGATCACTAATGGTTACGCAGTAGGCTGACTATAGATCATAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGT

>purple bovine
TGGCTGATCACTAATGGTTACGCAGTAGGCTGACTATAGATCATAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGT

>green bovine
TGGCTGATCACTGGATCCGACTATAGATCATAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGT

>lightBlue bovine
TGGCTGATCACTGGATCCGGATCCGACTATAGATCATAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGT

>darkGreen bovine
TGGCTGATCACTGGATCCGGATCCGGATCCGACTATAGATCATAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGTAAAAAAAAAACGT
```

FIGURE 1 – Fichier FASTA comprenant toutes les séquences de notre jeu de données de test, correspondant à celles présentées dans la figure 6.

Graphe

Un graphe est une structure de données composée d'objets appelés sommets ou nœuds, et de relations entre ces objets appelées arêtes. Dans un graphe non orienté, les arêtes relient deux sommets de manière symétrique, tandis que dans un graphe orienté, les arêtes, appelées arcs, relient deux sommets de manière asymétrique. Chaque sommet ou arête peut être associé à une étiquette, une valeur ou un ensemble d'attributs, permettant de représenter des informations supplémentaires telles qu'un nom, un poids, une catégorie ou toute autre donnée utile à l'analyse du graphe.

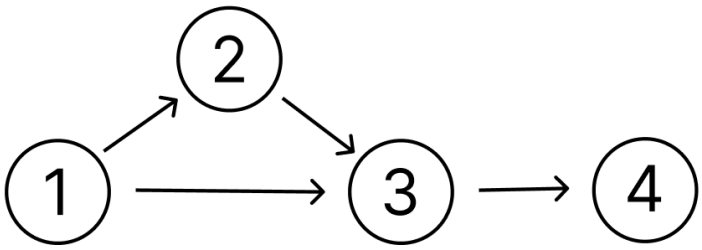


FIGURE 2 – Représentation d'un graphe orienté à 4 sommets {1,2,3,4} et 4 arcs {(1,2), (1,3), (2,3), (3,4)}.

Dans ce rapport, nous traitons des graphes pangénomiques. Pour rester cohérents avec la terminologie des fichiers GFA, qui servent à stocker ces graphes, les nœuds seront appelés segments et les arcs liens. Les différents composants d'un fichier GFA seront présentés plus en détail à la figure 8.

Inversion

Une inversion est un type de variation structurale du génome où un segment d'ADN double brin est coupé puis réinséré à l'envers sur le même chromosome. Cette inversion concerne les deux brins de la molécule, inversant ainsi l'ordre des bases sans perte ni gain de matériel génétique.

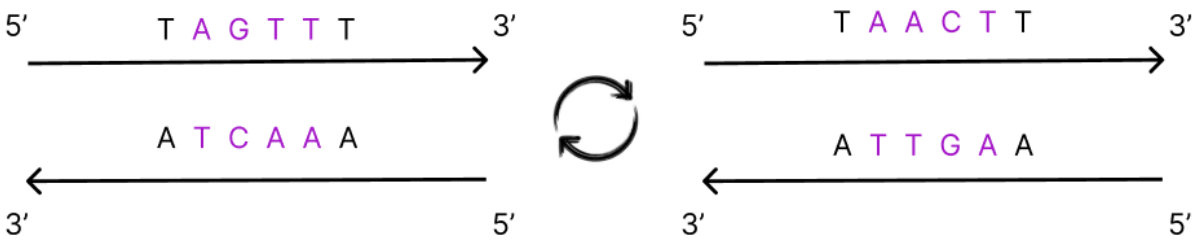


FIGURE 3 – Schéma d'une inversion entre deux brins d'un même chromosome.

k-mer

En génomique, un k-mer est un sous-mot de longueur k extrait d'une séquence biologique, comme l'ADN ou l'ARN. Par exemple, la séquence « AGCT » donne les 3-mers suivants : « AGC », « GCT » et « CTA ». [6]

Minimizer

Un minimizer est un représentant sélectionné parmi les k-mers contenus dans une fenêtre glissante d'une séquence. Il est de même taille ou plus petit qu'un k-mer classique. Ce représentant correspond au sous-k-mer ayant la plus petite valeur selon un critère donné, comme l'ordre lexicographique ou une fonction de hachage. L'utilisation des minimizers

permet de réduire la redondance, d'accélérer les calculs et d'optimiser le stockage, en ne conservant qu'une partie significative de l'information. [6]

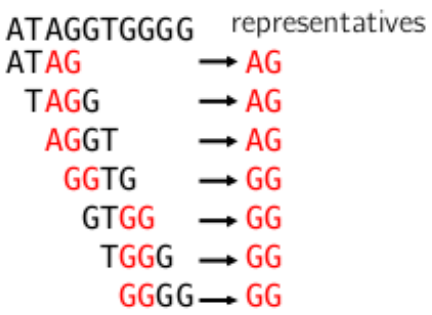


FIGURE 4 – Schéma illustrant un ensemble de minimizers, représentés en rouge. Pour chaque k-mer (de taille 4), le minimizer est défini comme la plus petite sous-séquence d'une taille définie (ici 2), selon l'ordre lexicographique, parmi toutes celles qu'il contient. Source : [6]

Polymorphisme nucléotidique simple (SNP)

Un SNP est une variation d'un seul nucléotide à une position précise de l'ADN qui diffère entre les individus d'une population.

Variant structural (SV)

Un variant structural est une modification de grande taille (généralement >50 paires de bases) dans la structure d'un génome. Il peut s'agir d'inversions, de délétions ou de duplications.

3 Création de pangénome

Un pangénome désigne l'ensemble des séquences génétiques issues de plusieurs individus appartenant à une même espèce, ou à des espèces très proches. Il reflète la diversité génomique au sein de cette population et permet de mettre en évidence les variations présentes dans les séquences d'ADN. Il peut être représenté de différentes façons selon l'aspect que l'on souhaite mettre en avant. Par exemple, il peut prendre la forme d'un diagramme de Venn pour illustrer la composition du génome, en distinguant le génome core du reste comme le montre la figure 5.

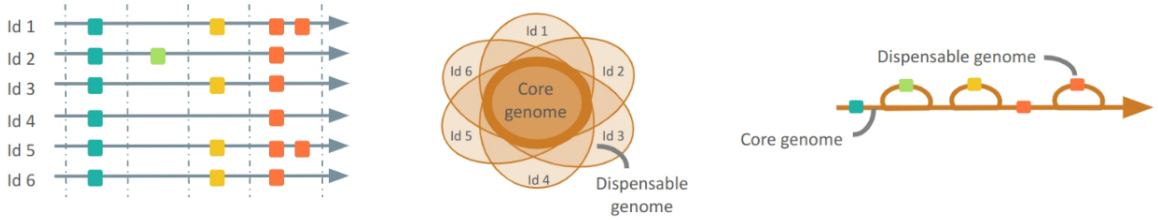


FIGURE 5 – Trois représentations du pangénome. Le pangénome est représenté (à gauche) comme un inventaire d'éléments génomiques partagés ou non au sein d'un groupe d'individus ; (au centre) sous la forme d'un diagramme de Venn dans lequel le génome central est l'ensemble commun de séquences partagées par tous les individus du groupe, le reste appartient au génome dispensable ; (à droite) sous la forme d'un graphe orienté, dans lequel des chemins alternatifs représentent les variantes structurales. Source [1]

Dans le cadre de ce rapport, il ne sera pas question de travailler seulement avec des ensembles de gènes. C'est pourquoi le pangénome sera représenté sous forme de graphe orienté, dans lequel les nœuds correspondent aux séquences communes aux individus, et les arcs représentent les liens entre ces séquences. Sous cette forme, il sera possible d'observer les variations génétiques entre les différents individus. Cette partie présentera les différents outils utilisés pour créer un pangénome et les afficher, en s'appuyant sur un jeu de données de test reposant sur les individus présentés à la figure 6.

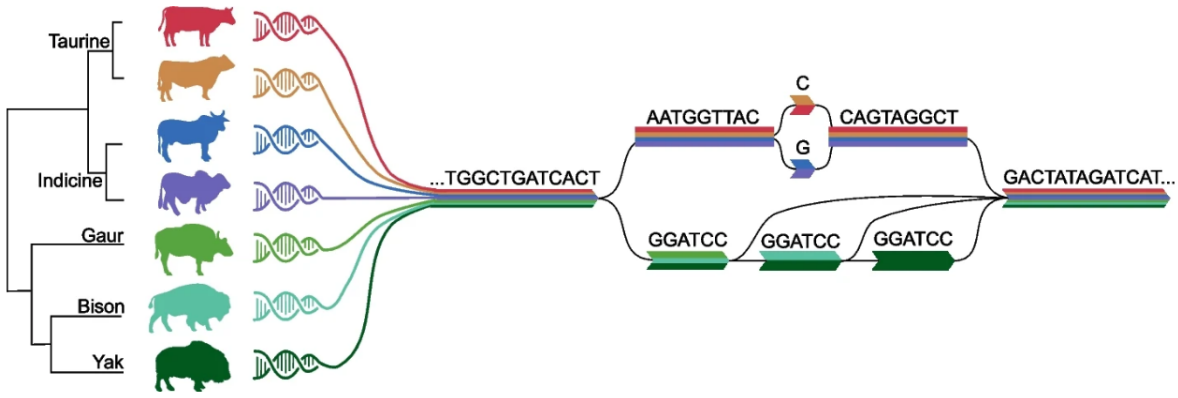


FIGURE 6 – Relations entre espèces bovines et conceptualisation du pangénome. Les variations nucléotidiques simples (SNP) et structurelles entre espèces, sous-espèces ou races sont représentées comme différents chemins à travers un pangénome multi-espèces. Afin de faciliter leur identification dans la suite de cette section, les individus sont nommés, de haut en bas : red, orange, blue, purple, green, lightBlue et darkGreen. Source : [7]

La création de pangénomes nécessite des séquences au format FASTA. Des fichiers ont donc été générés pour chaque individu représenté dans la figure 6, comme illustré dans la figure 7.

```
(base) anais@OptiPlex-7460:~/Documents/pangenome/bovin$ ls
allBovine.fasta  darkGreen.fasta  lightBlue.fasta  purple.fasta
blue.fasta       green.fasta      orange.fasta     red.fasta
```

FIGURE 7 – Fichiers FASTA contenant les séquences bovines de l'exemple que nous suivrons tout au long de cette section, affichés grâce de la commande bash ls.

Une fois le graphe de pangénome construit, les informations du graphe sont stockées au format GFA, dont deux versions coexistent : GFA1 et GFA2. Le premier format est plus simple et ne gère pas directement les insertions ou les délétions de manière explicite, il est utilisé pour des assemblages de fragments de base. GFA2 offre plus de flexibilité, permettant de gérer des graphes plus complexes et d'ajouter des annotations supplémentaires. [3] Un fichier GFA1 a été construit manuellement, présenté dans la figure 8, à partir des séquences bovines illustrées dans la figure 6. Ce format permet de représenter un graphe sous forme textuelle, en décrivant les segments de séquences, leurs connexions, et les chemins suivis par les différents individus utilisés lors de la construction du graphe.

```
1 H     VN:Z:1.0
2 S     1       TGGCTGATCACT
3 S     2       AATGGTTAC
4 S     3       C
5 S     4       CAGTAGGCT
6 S     5       GACTATAGATCAT
7 S     6       G
8 S     7       GGATCC
9 S     8       GGATCC
10 S    9       GGATCC
11 L    1       +       2       +       0M
12 L    2       +       3       +       0M
13 L    3       +       4       +       0M
14 L    4       +       5       +       0M
15 L    2       +       6       +       0M
16 L    6       +       4       +       0M
17 L    1       +       7       +       0M
18 L    7       +       5       +       0M
19 L    7       +       8       +       0M
20 L    8       +       5       +       0M
21 L    8       +       9       +       0M
22 L    9       +       5       +       0M
23 P    red     1+,2+,3+,4+,5+  *
24 P    orange  1+,2+,3+,4+,5+  *
25 P    blue    1+,2+,6+,4+,5+  *
26 P    purple  1+,2+,6+,4+,5+  *
27 P    green   1+,7+,5+        *
28 P    lightBlue       1+,7+,8+,5+     *
29 P    darkGreen       1+,7+,8+,9+,5+  *
```

FIGURE 8 – Fichier GFA1 créée manuellement à partir des séquences bovines introduites dans la figure 6.

La première ligne du fichier marquée par le préfixe H (pour Header) est optionnelle, elle spécifie la version du format utilisée, ici 1.0. Les lignes commençant par S (pour Segment) correspondent aux nœuds du graphe. Chaque segment est identifié par un identifiant unique et contient une séquence nucléotidique. Les lignes débutant par L (pour Link) décrivent les connexions entre deux segments. Elles précisent les identifiants des segments reliés, leur orientation (+ pour le sens direct, - pour une inversion), ainsi que la taille du chevauchement entre eux. Dans notre exemple, la valeur 0M indique qu'il n'y a aucun chevauchement de bases, chaque liaison représentant une simple jonction. Enfin, les lignes P (pour Path) décrivent des chemins linéaires parcourant le graphe, correspondant chacun à la séquence d'un individu.

Après cette phase exploratoire sur un jeu de données simplifié, les outils suivants seront utilisés pour la construction du pangénome de l'espèce *Pseudogymnoascus destructans*.

3.1 Outil de création de pangénome

Dans cette section seront présentés les outils utilisés pour la construction de graphe pangénomiques. Afin de générer un graphe à partir de notre jeu de données de test — constitué de séquences bovines — il a été nécessaire d'augmenter la taille de ces séquences. En effet, leurs longueurs initiales (entre 30 et 50 pb) étaient trop courtes pour être directement exploitées par les outils.

3.1.1 PGGB

PGGB (`https://github.com/pangenome/pggb`) permet de construire un graphe de pangénome en réalisant un alignement all-to-all des séquences renseignées en entrée, c'est-à-dire qu'il compare chaque séquence à toutes les autres de manière pair-à-pair, sans privilégier une séquence de référence unique pour guider l'alignement.

Son utilisation requiert un seul fichier au format FASTA regroupant toutes les séquences, préalablement compressé avec Bgzip et indexé à l'aide d'une commande de la suite d'outils Samtools.

```
bgzip data.fasta
samtools faidx data.fasta.gz
```

Commandes à réaliser avant l'utilisation de pggb.

Pour réaliser un pangénome, PGGB utilise plusieurs outils open-source qui sont les suivants :

- **wfmash** : réalise un alignement pair à pair entre les séquences d'entrée.
- **seqwish** : induit un graphe à partir des alignements générés.
- **smoothxg et gfaffix** : réalise la normalisation progressive du graphe pour le simplifier et le rendre plus utilisable.
- **odgi** : permet de visualiser et analyser le graphe généré par pggb. Il peut aussi servir à créer des visualisations diagnostiques.

```
pggb -i data.fasta.gz       # fichier FASTA compressé
     -o out                 # dossier de sortie
     -n 7                   # nb de séquences
     -t 16                  # nb de threads (opt.)
     -p 80                  # identité min. (opt.)
     -s 100                 # taille des segments (opt.)
```

Lancement de pggb avec ses principales options.

Après avoir lancé la commande, plusieurs fichiers sont générés (comme présentés dans la figure 9) pour faciliter l'analyse et la visualisation du pangénome obtenu. Ces fichiers incluent :

- **Fichier .log** : Ce fichier contient un récapitulatif de la commande exécutée, incluant les paramètres utilisés et les étapes réalisées.
- **Fichier .yml** : Un fichier de configuration qui détaille les paramètres choisis pour l'exécution de PGGB.
- **Fichier .png** : Des images générées par **odgi** qui offrent une représentation visuelle.
- **Fichier .gfa** : Le fichier contenant le graphe de pangénome au format GFA1.
- **Fichier .paf** : Résultat d'alignement entre deux séquences par l'outil wfmash.

```
(base) anais@OptiPlex-7460:~/Documents/pangenome/pggb$ ls out
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.05-26-2025_140542.log
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.05-26-2025_140542.params.yml
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.gfa
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og.lay
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og.lay.draw_multiqc.png
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og.lay.draw.png
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og.lay.tsv
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og.viz_depth_multiqc.png
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og.viz_inv_multiqc.png
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og.viz_multiqc.png
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og.viz_O_multiqc.png
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og.viz_pos_multiqc.png
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.og.viz_uncalled_multiqc.png
allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.fix.affixes.tsv.gz
allBovine.fasta.gz.d1e0da2.alignments.wfmash.paf
```

FIGURE 9 – Fichiers générés par la commande pggb, affichés grâce à la commande bash ls. Le fichier qui nous intéresse le plus est le fichier nommé `allBovine.fasta.gz.d1e0da2.11fba48.5e7ace0.smooth.final.gfa` contenant les informations du graphe.

Le pangénome des différentes espèces bovines obtenu grâce à PGGB est disponible en annexe A.1. On observe que les deux segments « ancres » (présents chez tous les individus) ont été retrouvés, à un nucléotide près pour le premier. Cependant, les chemins apparaissent bien plus complexes et moins lisibles que dans le schéma présenté en figure 8. On peut observer que le SNP illustré dans cette figure se trouve sur les segments 10 et 11 du pangénome en annexe.

3.1.2 Minigraph

Minigraph (`https://github.com/lh3/minigraph`) permet de construire un graphe de pangénome en suivant une approche itérative, dans laquelle chaque nouvelle séquence

est alignée au graphe déjà construit, puis intégrée en ajoutant uniquement les régions qui divergent de la structure existante. Son utilisation nécessite une séquence de référence sur laquelle les séquences suivantes seront alignées, afin d'identifier les variations et d'enrichir progressivement le graphe. Chaque séquence d'entrée doit être placée dans un fichier FASTA distinct, la première séquence mentionnée dans la commande sera utilisée comme référence de départ pour la construction du graphe.

```
./minigraph -cxggs
            -t16                                  # nb de threads (opt.)
            ref.fasta data1.fasta data2.fasta     # fichiers FASTA
            > pangenome.gfa                       # fichier de sortie
```

Lancement de minigraph avec ses principales options.

Il est important de placer chaque séquence correspondant à un individu dans un fichier FASTA différent. En effet, si plusieurs séquences sont regroupées dans un seul fichier, Minigraph filtrera la majorité des alignements (voir figure 10) car celles-ci couvriront plusieurs fois les mêmes régions du graphe et cela va à l'encontre de son principe d'alignement 1-à-1. Ce fonctionnement peut vite devenir une contrainte en ligne de commande lorsqu'on travaille avec un grand nombre d'individus : il faut alors gérer de nombreux fichiers séparés, voire des répertoires entiers à renseigner manuellement.

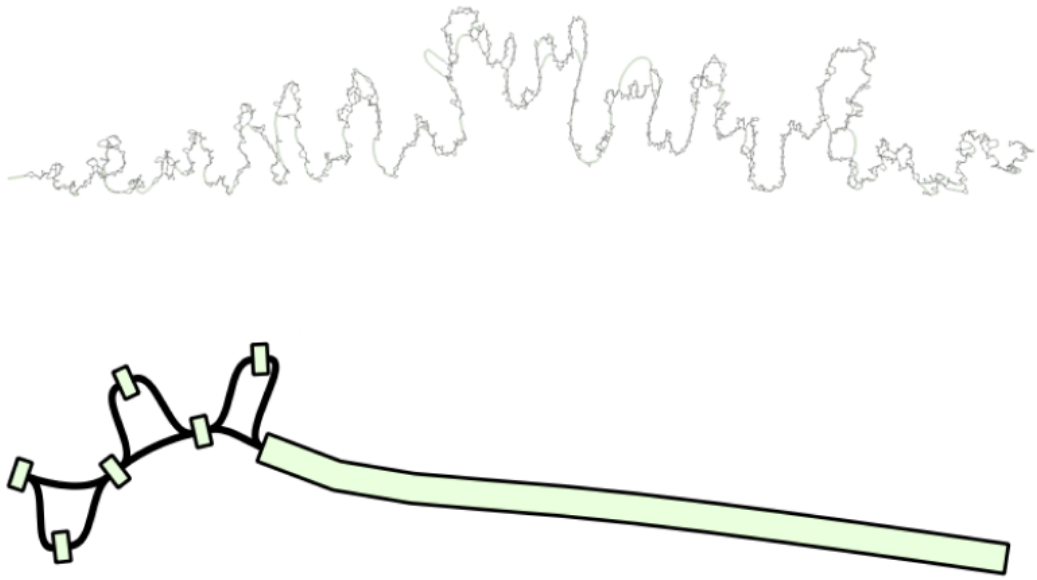


FIGURE 10 – Comparaison de deux pangénomes créés par Minigraph à partir des mêmes données du jeu de données `gd_chr10` et affichés dans Gfaviz. Le graphe du haut a été construit à partir de fichiers FASTA contenant chacun une seule séquence. Celui du bas, structurellement incohérent et de qualité discutable, a été obtenu en regroupant toutes les séquences (hors séquence de référence) dans un seul fichier FASTA. Cette différence met en évidence l'importance de séparer chaque séquence dans un fichier distinct lors de la construction du graphe.

Minigraph construit un graphe à partir d'une séquence de référence puis il aligne ensuite les séquences d'entrée sur ce graphe grâce à ces minimizers. Cela permet d'identifier

les régions similaires et celles qui diffèrent. Les parties divergentes sont ajoutées au graphe pour représenter les variations structurales entre les génomes.

Une fois le graphe construit, il est possible d'y ajouter de nouvelles données séquentielles à l'aide de la même commande afin d'intégrer d'autres individus sans avoir à tout reconstruire depuis zéro.

```
./minigraph -cxggs
            pangenome.gfa        # graphe GFA préablement construit
            data3.fasta          # fichier FASTA
            > pangenome2.gfa     # fichier de sortie
```

Lancement de minigraph à partir d'un graphe déjà construit.

Minigraph est limité à la détection de variants structuraux entre séquences, il ne peut donc pas gérer efficacement les petites variations ponctuelles ou les séquences très proches. Par conséquent, nos données bovines de test ne conviennent pas pour tester cet outil. Ce sont donc des données de l'espèce *Pseudogymnoascus destructans* sur le chromosome 10 de 4 individus différents qui seront utilisées. Ce jeu de données, que nous réutiliserons par la suite, sera désigné sous le nom `gd_chr10`.

Le fichier GFA généré par Minigraph ne contient pas de ligne P (Path), ce qui empêche d'extraire le parcours spécifique de chaque individu dans le pangénome. Il est aussi important de savoir que le graphe final dépend de l'ordre dans lequel les fichiers FASTA sont fournis en entrée. Comme chaque nouvelle séquence est ajoutée au graphe en se basant sur la structure actuelle, les variations introduites par les premiers individus influencent la manière dont les suivants seront intégrés. Ainsi, l'architecture du graphe peut varier significativement selon l'ordre des fichiers comme le montre la figure 11, ce qui peut avoir un impact sur l'interprétation du pangénome.

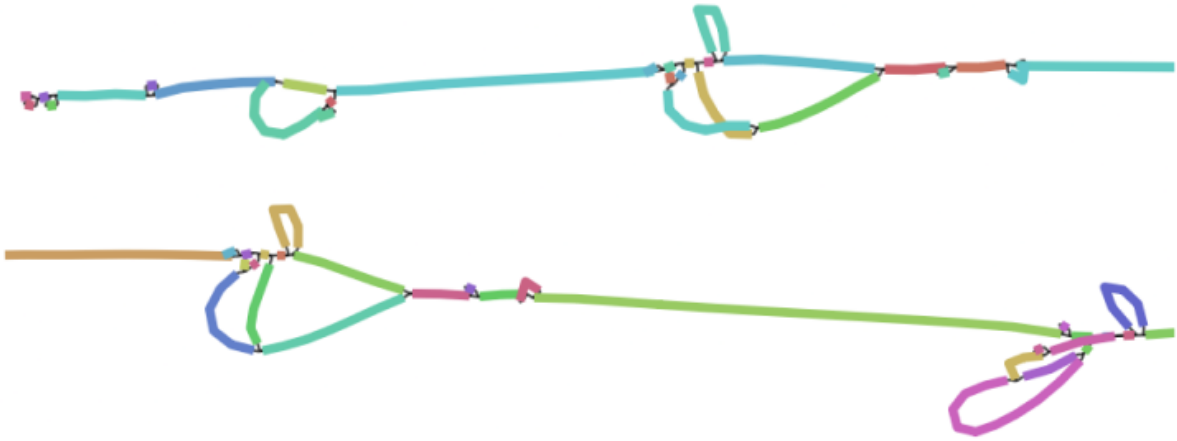


FIGURE 11 – Visualisation des 30 premiers segments tronqué de deux graphes de pangénome générés avec Minigraph à partir du même jeu de données `gd_chr10`, affichés via l'outil Bandage. Le graphe du haut a été construit en utilisant la séquence de l'individu `gd293` comme référence, tandis que celui du bas utilise celle de l'individu `gd442`.

3.1.3 Minigraph-Cactus

Cactus (https://github.com/ComparativeGenomicsToolkit/cactus) est une suite d'outils dédiée à l'alignement global de génomes ainsi qu'à la construction de graphes

pangénomiques. Parmi les différents outils proposés, Minigraph-Cactus [4] a été spécifiquement conçu pour analyser des génomes provenant d'individus appartenant à la même espèce, ce qui permet d'optimiser la représentation des variations intra-espèces. Étant donné que cette version de Cactus s'appuie sur Minigraph pour l'initialisation du graphe pangénomique, la fourniture d'une séquence de référence est indispensable, Minigraph nécessitant un point d'ancrage pour fonctionner. Comme pour l'utilisation directe de Minigraph, chaque fichier FASTA doit contenir une unique séquence, mais dans le cadre de Minigraph-Cactus, ces fichiers doivent être listés dans un fichier au format TSV afin d'organiser les séquences d'entrée.

```
cactus-pangenome    jobstore            # dossier pour logs et fichier temp.
                    seqfile             # fichier TSV
                    --outDir out        # dossier de sortie
                    --outName gd        # préfixe des fichiers
                    --reference gd293   # séquence de référence
                    --gfa full          # format de sortie souhaité
```

Lancement de cactus avec ses principales options.

À l'instar de Minigraph, les données de test bovines ne sont pas adaptées à l'utilisation de Minigraph-Cactus. En effet, Minigraph-Cactus s'appuie directement sur le graphe initial produit par Minigraph. Or, ce dernier repose sur la détection de variations structurales entre les séquences. Dans le cas de notre jeu de données, les séquences sont trop similaires et contiennent peu de variations de grande taille, ce qui aboutit à un graphe initial peu informatif. Cette limitation se répercute directement sur la construction du pangénome final, dont la qualité dépend fortement de la richesse du graphe de départ. C'est donc le jeu de données `gd_chr10` qui est utilisé pour la construction des pangénomes par Minigraph-Cactus.

Un fichier TSV (Tab-Separated Values) est un fichier texte dans lequel les valeurs sont séparées par des tabulations comme le montre la figure 12. Il est couramment utilisé pour représenter des tableaux ou des données structurées de manière simple et lisible par des scripts. Dans le cadre de l'utilisation de Cactus, ce format permet de fournir une liste où chaque ligne contient un identifiant de séquence suivi du chemin vers le fichier FASTA correspondant. Ce format se révèle particulièrement pratique, contrairement à Minigraph, qui requiert la mention explicite de chaque fichier dans sa commande Bash.

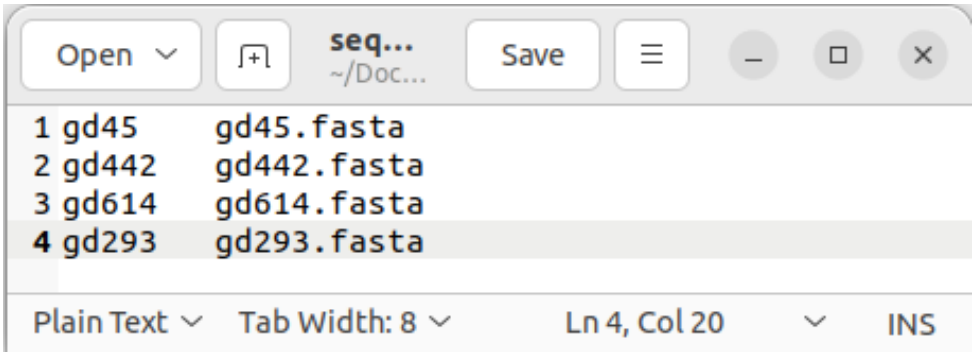


FIGURE 12 – Fichier TSV nommé `seqfile`. La première colonne contient les identifiants des séquences, tandis que la seconde spécifie le chemin d'accès vers chaque fichier FASTA correspondant. Dans la commande cactus-pangenome présentée précédemment, l'identifiant `gd293` a été choisi comme référence : c'est donc la séquence `gd293.fasta` qui sera utilisée par Minigraph pour initier la construction du graphe.

Lors de son exécution, Cactus commence par appeler l'outil Minigraph afin de générer un premier graphe basé sur la séquence de référence et les autres séquences listées dans le fichier TSV comme le montre la figure 13. Ce graphe initial ne comporte que les variations structurales d'au moins 50bp par défaut.

```
[MainThread] [I] [toil-rt] [minigraph]: [M::main] CMD: minigraph -c -xggs -t 6 gd293.fa gd442.fa gd614.fa gd45.fa
[MainThread] [I] [toil-rt] [minigraph]: [M::main] Real time: 276.457 sec; CPU: 276.511 sec; Peak RSS: 0.363 GB
```

FIGURE 13 – Logs du conteneur Cactus dans Docker Desktop. Minigraph-Cactus appelle l'outil Minigraph par la commande mise en évidence, on y voit la séquence de l'individu `gd293` bien utilisée comme référence.

Une fois le programme terminé, on retrouve dans le répertoire de sortie, renseigné dans la commande bash, les fichiers suivants :

- **Fichier .sv.gfa** : Contient le graphe de pangénome initial généré par Minigraph
- **Fichier .full.gfa** : Contient le graphe de pangénome final
- **Fichier .gaf** : Résultat d'alignement entre deux séquences.
- **Fichier .paf** : Permet de connaitre quel chemin prend une séquence d'entrée pour le graphe construit par Minigraph.

Cactus utilise le graphe généré par Minigraph comme point de départ pour intégrer les variations de toutes tailles, produisant ainsi un graphe pangénomique plus complet. Contrairement à Minigraph, qui est limité à la détection de variants structuraux et construit un graphe plus simple, Cactus construit un graphe bien plus dense et détaillé. Cette complexité accrue est visible sur la figure 14, qui illustre la différence de structure entre les deux graphes.

FIGURE 14 – Informations visualisées sur Bandage sur les deux graphes générés par Minigraph-Cactus : celles de gauche concernent le graphe initial produit par Minigraph, tandis que celles de droite se rapportent au graphe final obtenu après traitement par Cactus. Le jeu de données utilisé est `gd_chr10`. On peut remarquer que le graphe final contient près de 40 fois plus d'éléments que le graphe initial, avec 40 171 segments (nodes) et 53 717 liens (edges), contre seulement 1 139 segments et 1 617 liens dans le graphe contenant uniquement les variants structuraux.

La taille considérablement plus importante du graphe final s'explique non seulement par l'intégration d'un plus grand nombre de variations issues du traitement par Cactus, mais aussi par la fragmentation des séquences en segments de taille maximale 1024bp comme démontré dans la figure 15. Cette fragmentation entraîne une multiplication artificielle du nombre de segments, ce qui augmente fortement la complexité et la taille du graphe final par rapport au graphe initial généré par Minigraph.

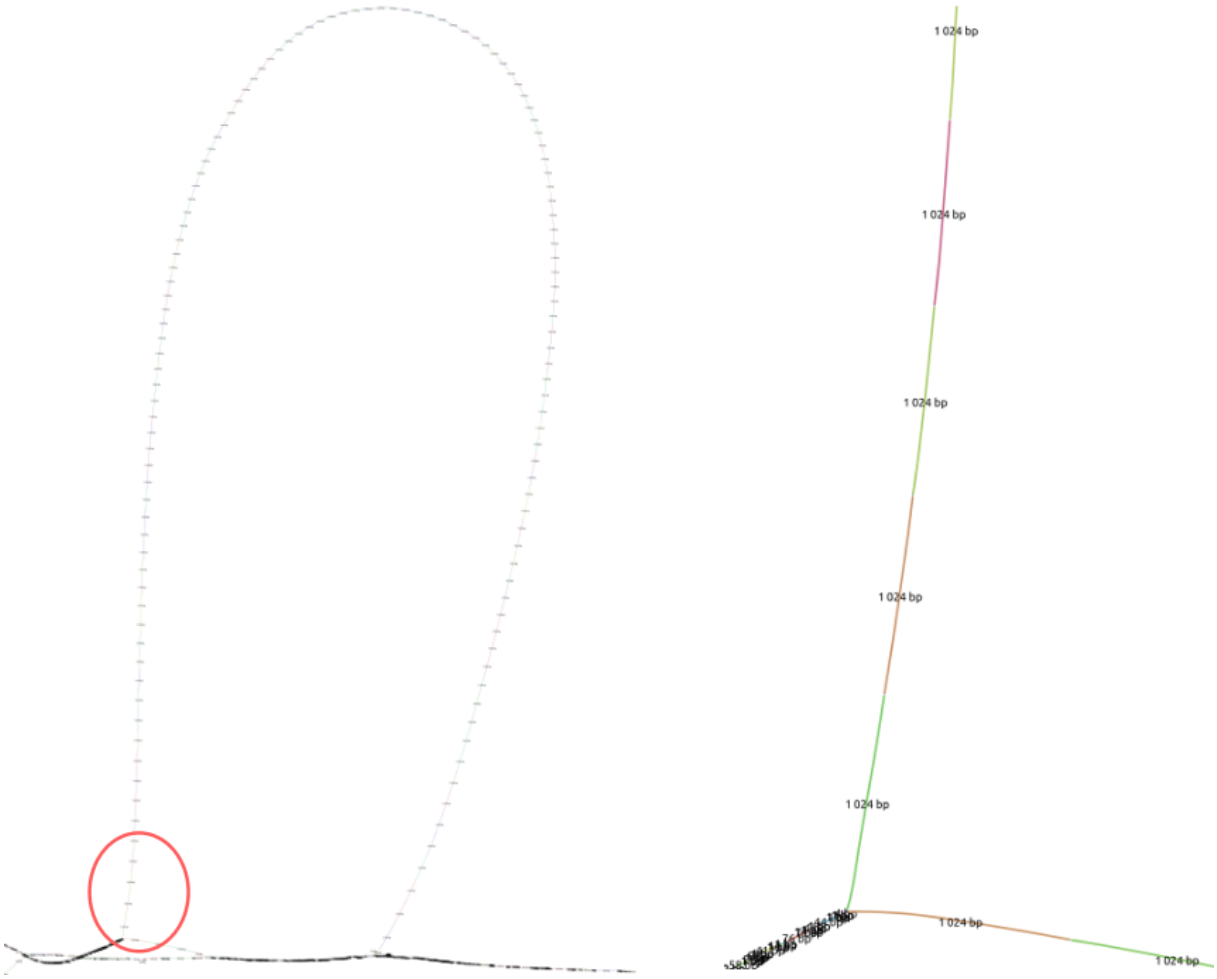


FIGURE 15 – Extrait de pangénome généré par Minigraph-Cactus, visualisé dans Bandage. L'enchaînement de segments mis en évidence dans la partie droite correspond à la région entourée en rouge dans le graphe plus large présenté à gauche. Les tailles des segments (en paires de bases) sont annotées, ce qui permet de constater que la boucle observée à gauche n'est en réalité qu'une succession de segments de 1024bp, issue de la fragmentation opérée par Cactus. Cette région pourrait donc correspondre à une seule séquence continue non fragmentée.

3.1.4 Comparaison des outils

Trois constructeurs de graphe ont été testés dans cette sous-section : PGGB, Minigraph et Minigraph-Cactus. Tous les outils ont été installés et utilisés avec un ordinateur équipé d'Ubuntu 22.04.2 LTS (64 bits), d'un processeur Intel Core i5-8400 (6 cœurs, 4.00 GHz) et de 8 Go de mémoire vive.

Concernant l'installation des différents outils testés, Minigraph a été installé en clonant son dépôt GitHub, puis compilé localement avec la commande `make`. PGGB a été installé via Conda à l'aide de la commande `conda create -n pggb-env -c bioconda pggb`, dans le but de créer un environnement dédié et ainsi éviter les conflits de dépendances. Enfin, Minigraph-Cactus a été installé à l'aide de Docker : le dépôt Github a été cloné et l'image Docker a été construite directement à partir du `Dockerfile` fourni.

Pour comparer ces trois outils, plusieurs constructions de pangénomes ont été simulées à partir du jeu de données `gd_chr10` afin de réaliser la table 1. Minigraph et Minigraph-Cactus ont été utilisés avec la même séquence de référence lors de chaque simulation. La mémoire maximale utilisée ainsi que le temps d'exécution ont été évalués de manière uniforme pour les trois outils en utilisant la commande `/usr/bin/time -v`. Cette commande permet d'obtenir le pic de mémoire consommé et le temps total écoulé entre le lancement et la fin de chaque commande.
L'outil Bandage a été utilisé pour extraire des informations sur les graphes générés, notamment le nombre de segments, le nombre de liens et la taille totale. Ces données permettent de comparer la complexité des graphes produits par les trois outils de génération testés.

	PGGB	**Minigraph**	**Minigraph Cactus**
Référence	Non	Oui	Oui
Mémoire utilisée maximum	818 Mo	284 Mo	1525 Mo
Temps d'éxecution	5min	6min	22min
Nb de segments du graphe généré	37 790	1 180	40 162
Nb de liens du graphe généré	51 295	1 689	53 707
Taille du graphe généré	3 156 663 bp	3 483 611 bp	3 586 299 bp

TABLE 1 – Comparaison des outils de construction de pangénome en termes de performances et de caractéristiques des graphes générés. Les valeurs présentées correspondent à la moyenne de cinq simulations, dont les résultats détaillés sont disponibles en annexe A.7.

Pour la construction du pangénome de *Pseudogymnoascus destructans*, le choix s'oriente vers PGGB ou Cactus, car ces outils permettent de capturer non seulement les variants structuraux mais aussi les variations de plus petite taille. De plus, ils offrent la possibilité de reconstituer le chemin suivi par chaque séquence d'entrée dans le graphe. Pour chacun de ces outils, la documentation s'est révélée limitée, ce qui reflète le caractère encore émergent et en constante évolution du domaine de la construction de pangénomes.

3.2 Manipulation de graphe

L'analyse et la visualisation de graphes pangénomiques peuvent nécessiter de manipuler directement leurs structures. Plusieurs outils permettent d'interagir avec ces structures complexes. Parmi eux, Odgi propose un ensemble de commandes pour explorer, filtrer et visualiser, tandis que gfapy est une bibliothèque Python permettant de lire, modifier ou générer des fichiers au format GFA.

3.2.1 Odgi

Odgi (`https://github.com/pangenome/odgi`) est un programme en ligne de commande dédié à la manipulation de graphes utilisés dans l'étude des pangénomes. Il permet de visualiser, extraire et analyser les structures complexes d'un graphe. Pour l'installer, j'ai créé un environnement conda dédié en exécutant la commande suivante : `conda create -n odgi-env -c bioconda odgi`. Afin de manipuler un graphe avec cet outil, il est nécessaire de le convertir préalablement au format ODGI, un format binaire optimisé.

```
odgi build  -g pangenome.gfa      # graphe au format GFA
            -o pangenome.og       # fichier de sortie
```

Conversion d'un fichier GFA au format ODGI.

Une fonctionnalité particulièrement utile d'Odgi est la commande `odgi extract`, qui permet d'isoler un sous-graphe en précisant une plage de positions (en paires de bases) le long du chemin d'un individu donné. Il est important de noter que le sous-graphe extrait contiendra tous les segments situés entre les points d'ancrage déterminés par les positions spécifiées sur le chemin de l'individu sélectionné, même si certains segments intermédiaires ne sont pas directement parcourus par lui (voir figure 16). Par ailleurs, il est possible de convertir un fichier au format ODGI vers le format GFA à l'aide de la commande `odgi view`.

```
odgi extract    -i pangenome.og          # graphe au format ODGI
                -r red:12-43             # plage de position d'un individu
                -o pangenome_sub.og      # graphe de sortie au format ODGI
```

Extraction d'un sous-graphe grâce à une plage de position par rapport au chemin d'un individu.

```
odgi view   -i pangenome.og          # graphe au format ODGI
            -g                       # option pour obtenir le format GFA
            > pangenome.gfa          # fichier de sortie
```

Conversion d'un fichier au format ODGI au format GFA.

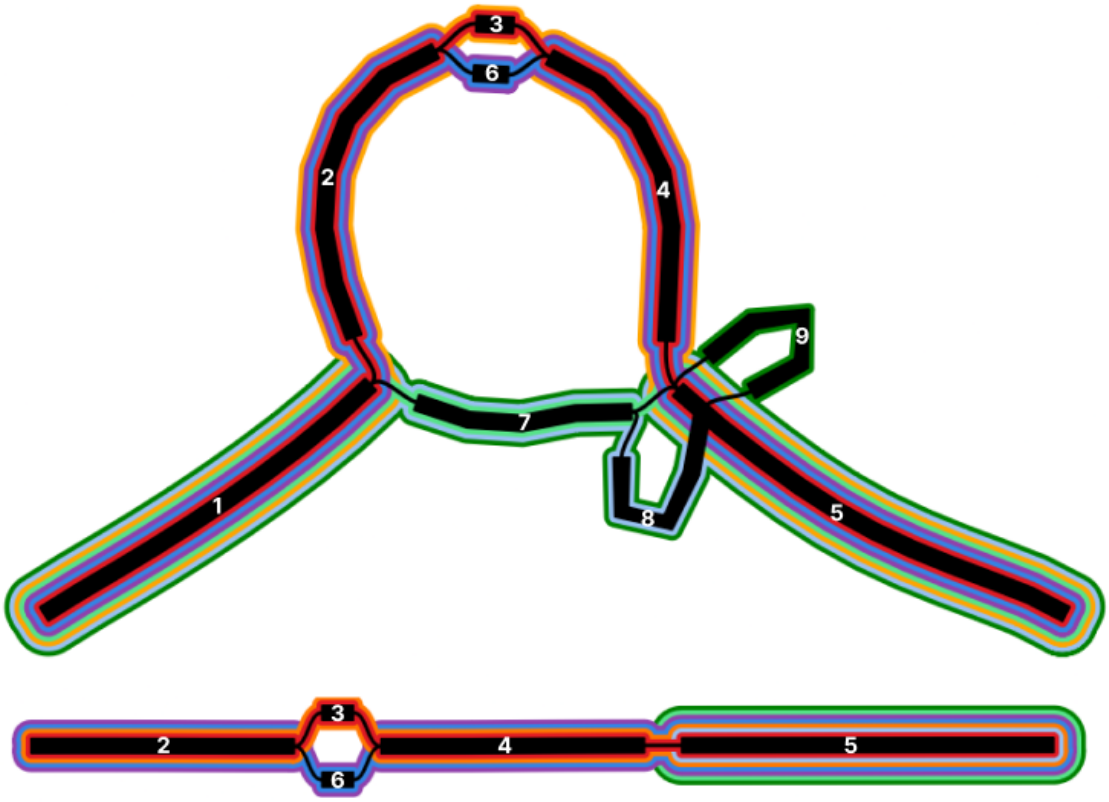


FIGURE 16 – Pangénomes construits à partir des séquences de test bovines et visualisés avec Gfaviz. Le graphe du haut correspond au graphe original obtenu directement depuis le fichier GFA créé manuellement, tandis que celui du bas est un sous-graphe extrait à l'aide de la commande `odgi extract`. Les chemins des individus ont été conservés et affichés en correspondant aux noms de couleur des individus. On observe que le sous-graphe contient tous les segments situés entre les segments ancres sélectionnées par la position sur le chemin de l'individu `red`, incluant également les segments parcourus par d'autres individus dans cette même plage.

Une autre commande utile est `odgi position`, qui permet de retrouver l'identifiant du segment correspondant à une position donnée sur un chemin (voir figure). Cela facilite la localisation précise de régions génomiques dans un pangénome représenté sous forme de graphe.

```
odgi position   -i pangenome.og  # graphe au format ODGI
                -p "red,10"       # identifiant de l'individu, position
                -v                # option pour obtenir la position
```

```
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pangenome.og -p "red,10" -v
#source.path.pos        target.graph.pos
red,10,+        1,10,+
```

FIGURE 17 – Résultats de la commande `odgi psoition`. On observe que la position 10bp en fonction de l'individu `red` correspond au segment 1, au nucléotide 10.

Odgi propose plusieurs commandes dédiées à la visualisation des graphes pangénomiques : `odgi viz` et `odgi draw`. La première permet de générer une image

représentant les différents individus qui parcourent le pangénome s'ils sont indiqués par les lignes P du fichier GFA d'origine.

```
odgi viz    -i pangenome.og
            -o pangenome.png
```

Conversion d'un fichier au format ODGI au format GFA.

La seconde, `odgi draw`, est une commande permettant de générer une image de la structure globale d'un graphe pangénomique mais ne montre pas explicitement les segments individuels. L'image produite ressemble à un réseau de connexions, sans détails sur les positions précises des segments. Afin d'utiliser cette dernière, il peut être nécessaire d'optimiser fichier ODGI l'aide de la commande `odgi sort` avec l'option `-optimize`. Cette étape permet de trier les nœuds du graphe selon un ordre, ce qui est indispensable pour exécuter la commande `odgi layout`. Cette dernière génère les coordonnées nécessaires à `odgi draw` pour produire une représentation visuelle cohérente de la structure du graphe pangénomique.

```
odgi sort   -i pangenome.og             # graphe au format ODGI
            -o pangenome.sorted.og      # fichier de sortie
            -O                          # option --optimize
```

Optimisation du fichier ODGI.

```
odgi layout    -i pangenome.og       # graphe au format ODGI
               -o layout.coords      # fichier de sortie
```

Génération des coordonnées du graphe.

```
odgi draw   -i pangenome.og         # graphe au format ODGI
            -c layout.coords        # coordonnées du graphe
            -s draw.svg             # image de sortie au format SVG
```

Création de l'image au format SVG.

3.2.2 Gfapy

Gfapy (`https://gfapy.readthedocs.io/en/latest/readme.html`) est une librairie Python qui permet de lire, manipuler et écrire des fichiers GFA. Elle permet de travailler directement sur les éléments du graphe (segments, liens et chemins) à travers des objets Python, ce qui facilite l'extraction et le traitement des données.

Dans le cadre du stage, j'ai utilisé cette librairie pour écrire un script permettant de générer un sous-graphe à partir d'une liste d'identifiants de segments, disponible en annexe A.5. Le script extrait les segments et les liens correspondants, puis génère un fichier GFA ne contenant que ce sous-ensemble, ce qui facilite l'analyse ciblée.

4 Visualisation

À présent que les pangénomes ont été construits et enregistrés au format GFA, l'étape suivante consiste à explorer des outils de visualisation capables d'interpréter ce format, afin d'obtenir une représentation graphique des structures pangénomiques. Dans cette section, le GFA créé manuellement, présenté dans la figure 8, est utilisé par les différents outils pour permettre une visualisation simple et compréhensible.

4.1 Bandage

Bandage (`https://github.com/rrwick/Bandage`) est un programme permettant de visualiser et d'interagir avec un graphe au format GFA1 (voir figure 18). Il dispose d'une interface graphique, rendue possible grâce à la bibliothèque Qt5, un ensemble d'outils dédié à la création d'interfaces visuelles. Pour l'utiliser, il est nécessaire de cloner le dépôt GitHub du projet et d'installer Qt5 au préalable. Une fois le graphe chargé, l'utilisateur peut le manipuler en déplaçant les segments et le personnaliser en ajoutant des étiquettes ou en modifiant la couleur des segments. [9]

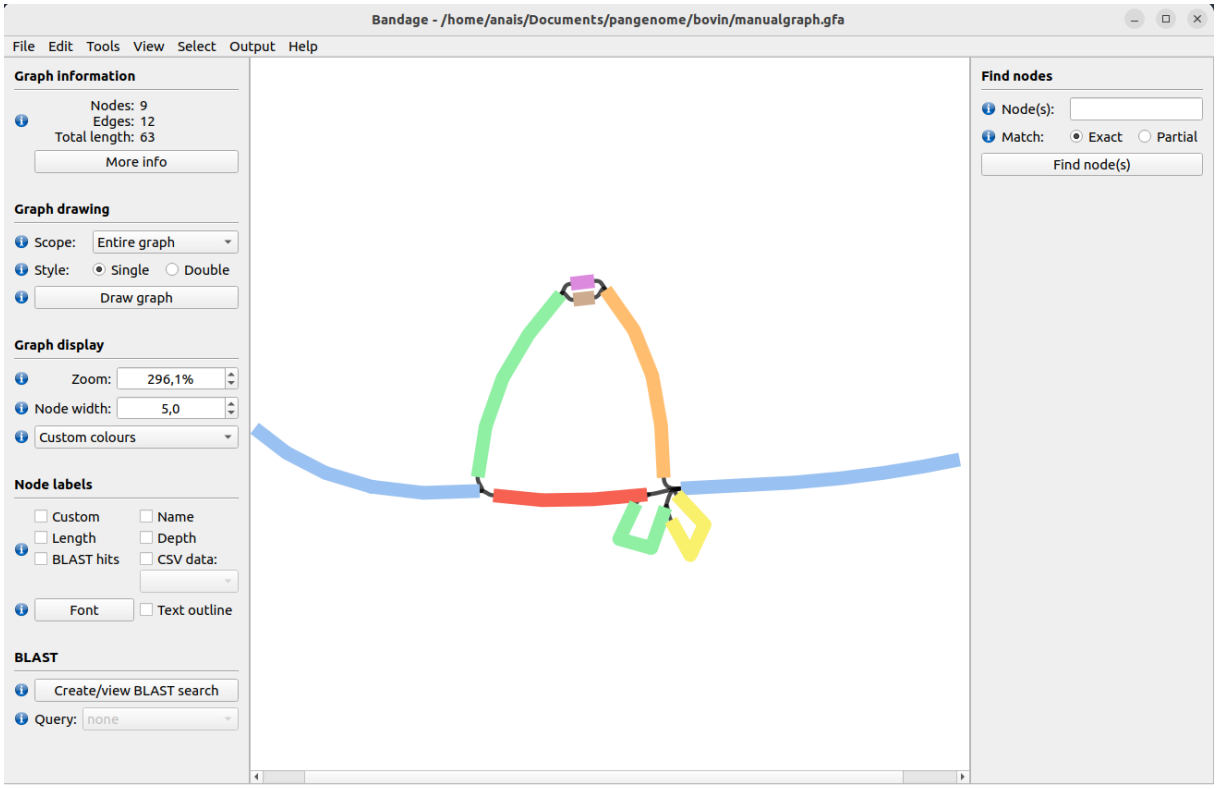


FIGURE 18 – Pangénome des espèces bovines affiché dans l'interface de Bandage. Le fichier GFA utilisé est celui construit manuellement, visible dans la figure 8.

Bandage ne permet pas de visualiser directement les chemins empruntés par les séquences, bien qu'ils soient spécifiés par les lignes P ou W dans le fichier GFA. Il est néanmoins possible d'y accéder en passant par " Output " > " Specify exact path for copy/save " dans Bandage, puis en copiant-collant les chemins définis par les lignes P du GFA. Dans le cas des graphes générés par Minigraph et Minigraph-Cactus, les fichiers GFA ne contiennent pas de lignes P. Des scripts ont donc été développés pour permettre la visualisation des chemins dans les graphes produits par ces outils. Le script présenté en annexe A.2 permet

d'obtenir les chemins du graphe initial dans un format compatible à partir du fichier de sortie PAF généré par Minigraph-Cactus. Le script de l'annexe A.3 extrait quant à lui les chemins directement depuis les lignes W du fichier .full.gfa produit par Minigraph-Cactus, et les reformate pour qu'ils soient acceptés par Bandage. Pour que le chemin soit reconnu par Bandage, la succession de segments doit être séparée par des virgules, et l'orientation de chaque segment précisée à l'aide des signes + ou -, comme illustré dans la figure 19. Il est important que ce chemin corresponde exactement à un parcours valide dans le graphe, sans quoi Bandage n'affichera rien.

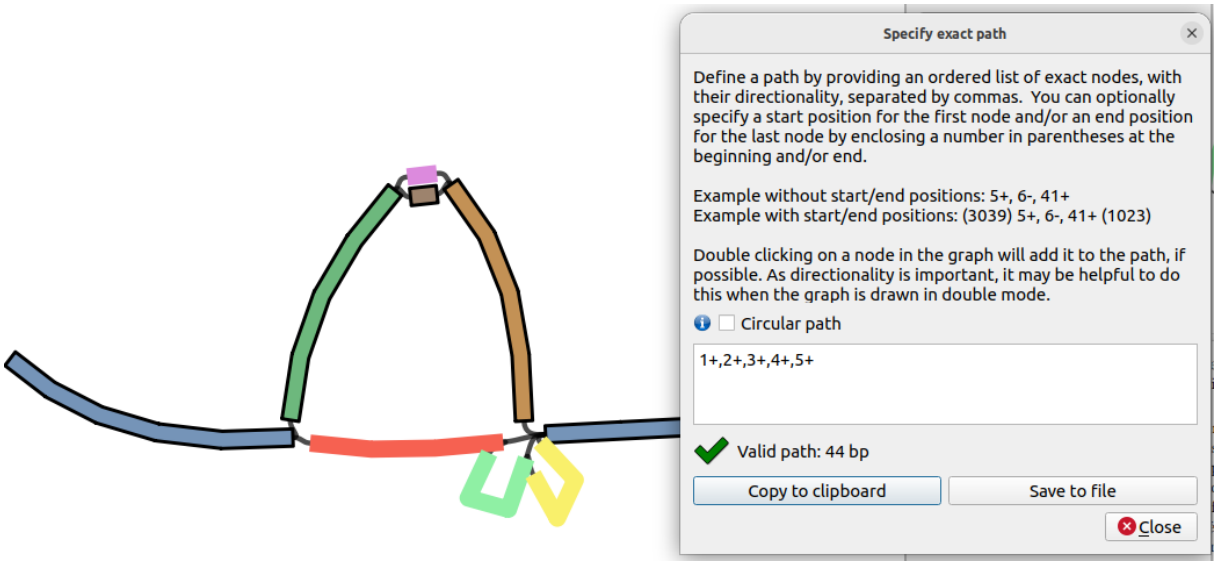


FIGURE 19 – Chemin de segments emprunté par l'individu rouge dans le pangénome des espèces bovines. Ce chemin a été extrait du fichier GFA, également utilisé pour construire le pangénome illustré dans la figure 8.

Bandage propose plusieurs fonctionnalités pour éditer un graphe :
— masquer les segments sélectionnés
— supprimer les segments sélectionnés
— fusionner les segments sélectionnés à condition qu'ils appartiennent à un chemin unique, sans embranchements ni arêtes supplémentaires
— fusionner automatiquement tous les segments compatibles (voir figure 20)

Il est possible de visualiser un sous-graphe en sélectionnant la partie souhaitée, puis en utilisant les options « Select » > « Invert selection » > « Edit » > « Remove selection from graph ». On peut aussi créer un nouveau graphe visuel en choisissant l'option « Scope : Around nodes », qui n'affiche que les segments renseignés. Il est ensuite possible d'enregistrer ces sous-graphes en allant dans « Output » > « Save visible graph to GFA ». Bandage offre également la possibilité d'enregistrer le graphe ou un sous-graphe au format FASTA. Dans ce fichier, chaque entrée identifiée par une ligne commençant par " > " correspond à un segment et contient sa séquence nucléotidique.

La sélection des segments s'effectue à la souris ou via la barre de recherche en utilisant leurs identifiants et en combinant différentes options de sélection. Bien qu'il soit possible d'obtenir la taille d'un ou plusieurs segments sélectionnés, leurs séquences nucléotidiques ne sont pas accessibles. Cet outil reste donc pertinent pour une visualisation rapide du

pangénome, mais ne permet pas une analyse approfondie. Par ailleurs, naviguer dans des graphes de grande taille (plus de 1000 bp) peut rapidement devenir complexe, en risquant de faire perdre le fil visuel. Enfin, les inversions présentes ne peuvent pas être visualisées, puisque l'outil ne fournit aucune indication sur le sens de lecture des segments.

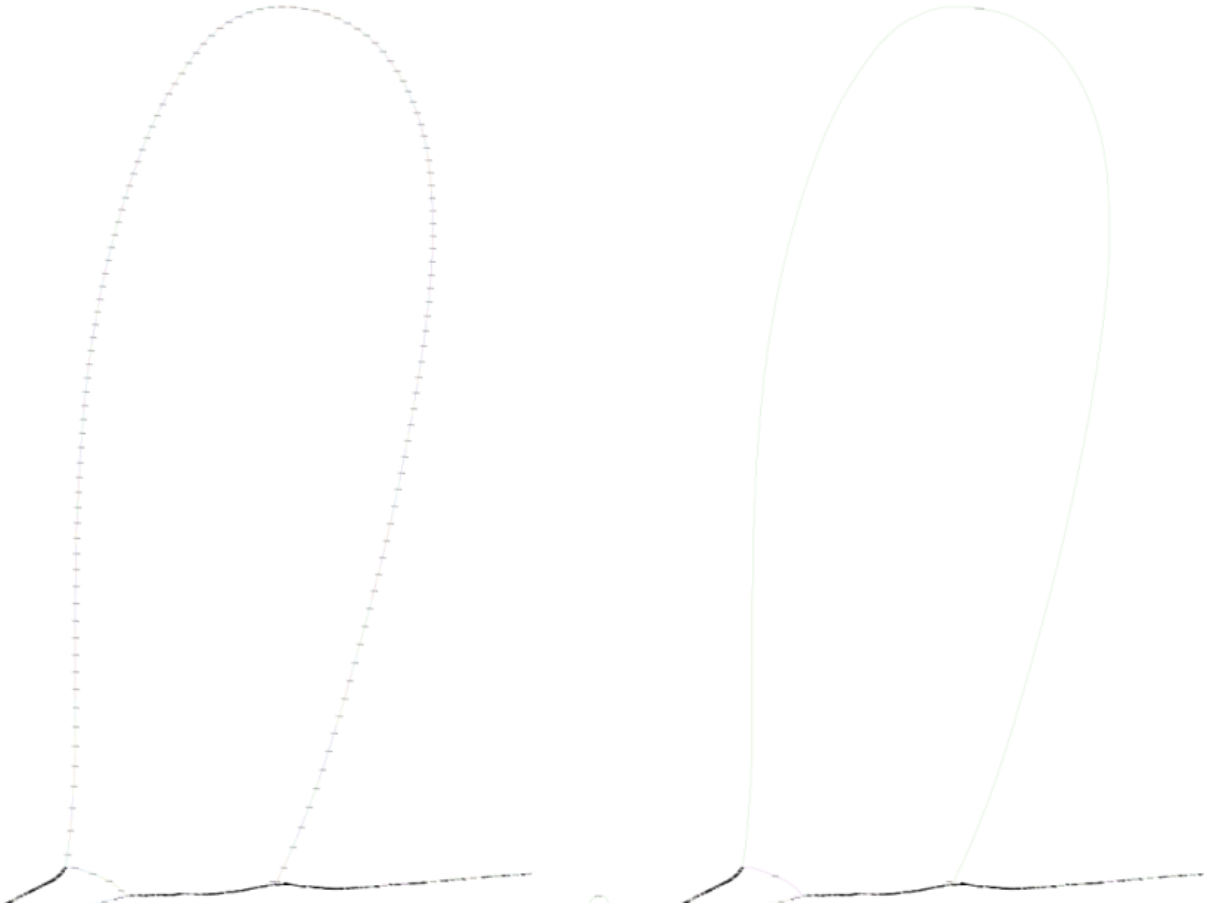


FIGURE 20 – Extrait du pangénome généré par Minigraph-Cactus à partir du jeu de données `gd_chr10`, visualisé dans Bandage et déjà présenté à la figure 15. À droite, le graphe d'origine ; à gauche, la même région après application de l'option « Merge all possible nodes », qui a permis de fusionner tous les segments fragmentés en un unique segment. Cette opération a réduit le nombre de nœuds de 40 154 à 38 275 et le nombre d'arêtes de 53 697 à 51 818 (soit une réduction d'environ 5%).

4.2 Gfaviz

GfaViz (`https://github.com/ggonnella/gfaviz`) est un outil de visualisation de graphes compatible avec les formats GFA1 et GFA2. Il s'utilise à partir d'un exécutable généré après clonage du dépôt GitHub du projet. Avant de pouvoir l'exécuter, il est nécessaire d'installer la bibliothèque Qt5, qui fournit les composants essentiels à l'affichage et à l'utilisation de l'interface graphique. L'outil permet de manipuler le graphe de manière intuitive, notamment en déplaçant les segments à la souris et en personnalisant leur apparence. En revanche, la navigation à l'intérieur du graphe se fait exclusivement à l'aide des barres de défilement, ce qui limite légèrement l'exploration interactive.

Si le fichier GFA contient des lignes de type P, les chemins sont automatiquement affichés, comme illustré dans la figure 21. Dans le cas de la construction du pangénome à partir des petites séquences bovines de test, cela permet une visualisation claire et directe des parcours. En revanche, pour des pangénomes réels plus complexes, il peut être préférable de spécifier manuellement le chemin à visualiser, récupéré à l'aide du script A.2. Cette approche permet de cibler plus précisément les segments d'intérêt et de les personnaliser, comme illustré dans la figure 22.

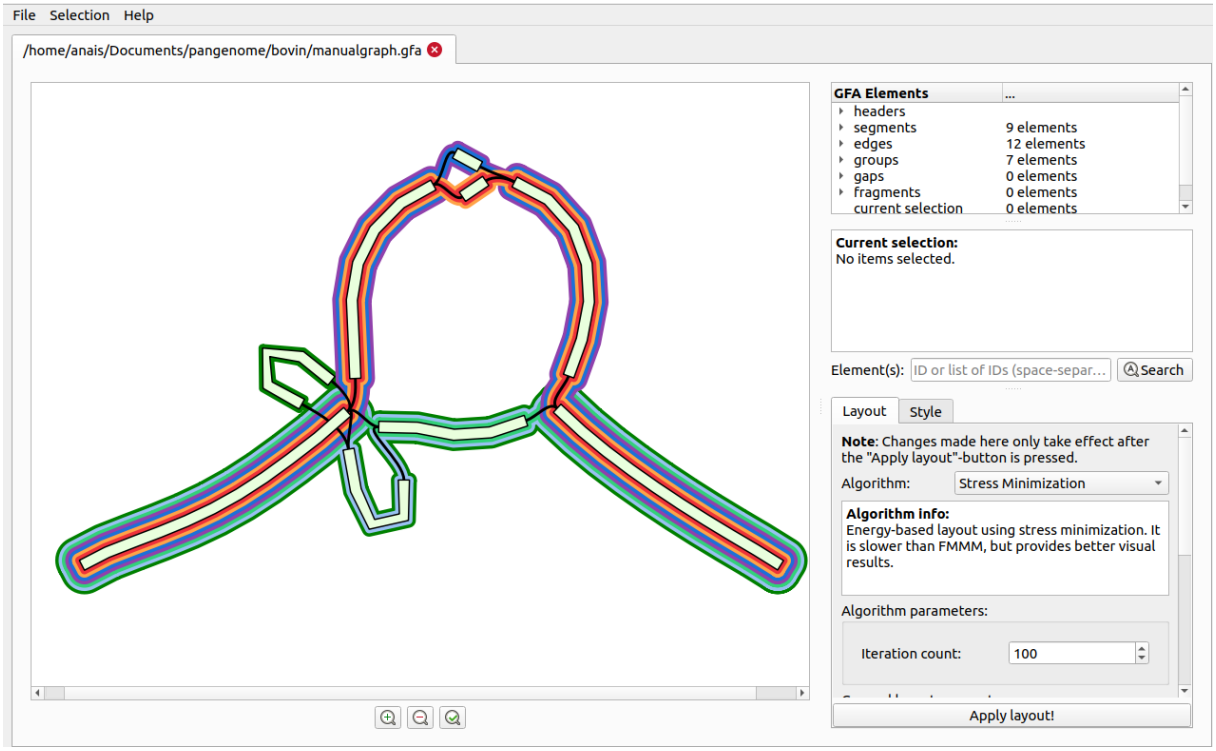


FIGURE 21 – Pangénome des espèces bovines affiché dans Gfaviz avec les chemins personnalisés afin de correspondre aux noms de couleurs de nos individus de test. Le fichier utilisé pour la création de ce graphe est le fichier GFA crée manuellement disponible dans la figure 8.

Pour des graphes simples, GfaViz permet une visualisation claire et lisible. En revanche, lorsque le graphe devient plus complexe, la représentation se dégrade rapidement et devient désordonnée, en raison de la longueur et du croisement des liens entre les segments. L'outil permet d'accéder à la séquence nucléotidique de chaque segment en plus de leurs tailles, ce qui est très utile pour analyser en détail la composition du graphe. Cependant, GfaViz plante systématiquement lorsqu'on tente d'ouvrir un graphe complexe contenant environ plus de 1500 segments et autant de liens. C'est notamment le cas avec les graphes de grande taille générés par PGGB ou Minigraph-Cactus contenant environ 40 000 segments, ce qui limite fortement l'usage de l'outil. Il reste toutefois pertinent pour la visualisation de graphes plus légers, comme ceux produits par Minigraph, ou du graphe initial généré par Minigraph-Cactus, représentant les variants structuraux (voir figure 22).

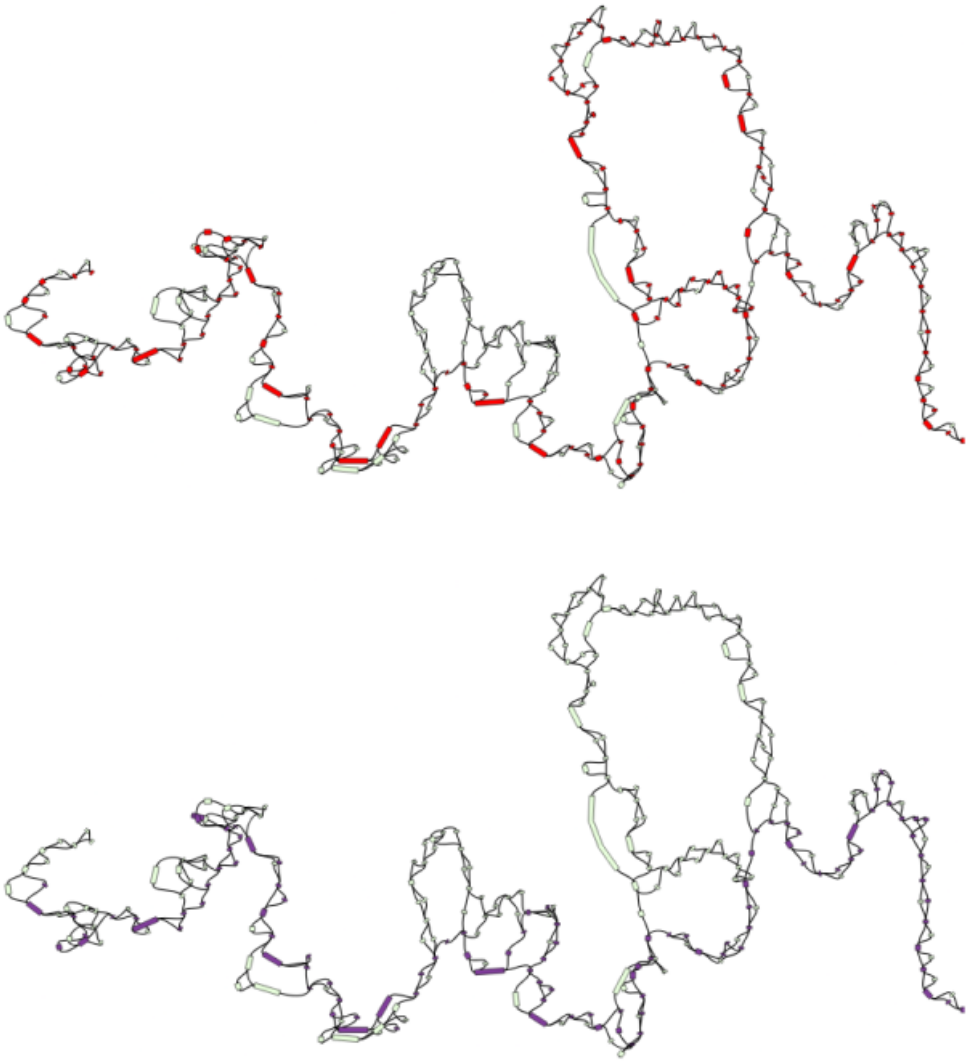


FIGURE 22 – Visualisation des 700 premiers segments tronqués du graphe initial généré par Minigraph-Cactus à partir du jeu de données `gd_chr10`. Il s'agit du même graphe représenté en haut et en bas, mais avec une coloration différente des segments en fonction du chemin suivi par deux individus. Dans la partie supérieure, les segments en rouge indiquent le chemin emprunté par l'individu `gd293`, utilisé comme référence pour la construction du graphe. Dans la partie inférieure, les segments en violet correspondent au chemin suivi par l'individu `gd442`.

Il est possible de sélectionner des segments à l'aide de la souris ou en entrant leurs identifiants, séparés par des espaces, dans la barre de recherche. Une fois sélectionnés ces segments peuvent être personnalisés ou masqués, ce qui permet de créer un sous-graphe. Le graphe visible ainsi obtenu peut être enregistré au format GFA1 ou GFA2, avec la possibilité de sauvegarder les personnalisations appliquées. Il est aussi possible d'exporter le graphe sous forme d'image aux formats PNG, JPG, SVG, etc.

4.3 GraphViz

GraphViz (`https://www.graphviz.org/`) est un programme à installer qui permet de visualiser un graphe mais qui ne possède pas d'interface graphique et fonctionne uniquement en ligne de commande. Il se distingue des autres outils présentés précédemment

car il affiche directement la séquence nucléotidique dans les segments (voir figure 23). Par conséquent, pour que ce programme génère un graphe visuel, celui-ci doit être de petite taille. On part donc du principe que l'on n'affichera qu'un sous-graphe. De plus, pour que GraphViz puisse traiter le graphe au format GFA, il est impératif que les identifiants des segments soient composés uniquement de nombres. Par exemple, pour les graphes générés par Minigraph, dont les identifiants suivent le format `s+nombre`, un renommage est nécessaire.

```
sed -E 's/\bs([0-9]+)\b/\1/g' subgraph.gfa > subgraph_rename.gfa
```

Commande bash pour renommer les segments nommés suivant le format `s+nombre`.

Pour pouvoir générer un graphe avec ce programme, il est nécessaire de convertir le fichier GFA en fichier DOT. Le format DOT est un format textuel utilisé par GraphViz pour représenter des graphes, dans lequel on retrouve une description des segments et des liens du graphe, ainsi que des options de mise en forme comme la couleur. Ce fichier sert ensuite de base pour produire une visualisation graphique du pangénome.

expliquer vg

```
vg view -F -p -d subgraph.gfa > subgraph.dot
```

Commande pour convertir un fichier GFA en fichier DOT.

Le programme génère une image du pangénome à partir du fichier DOT, avec la possibilité d'exporter le rendu aux formats PNG ou SVG. Le format SVG est particulièrement pratique pour explorer visuellement des graphes complexes, car il permet un zoom sans perte de qualité. Si le fichier GFA d'origine contient des lignes de type P, les chemins sont automatiquement intégrés à l'image, comme illustré dans la figure 23. Cet outil permet de visualiser des pangénomes, mais ne propose aucune fonctionnalité d'édition ni de création de sous-graphes.

```
dot -Tpng subgraph.dot -o subgraph.png      # format de sortie PNG
```

Commande pour obtenir la visualisation du pangénome au format PNG à partir du fichier DOT.

```
dot -Tsvg subgraph.dot -o subgraph.svg      # format de sortie SVG
```

Commande pour obtenir la visualisation du pangénome au format SVG à partir du fichier DOT.

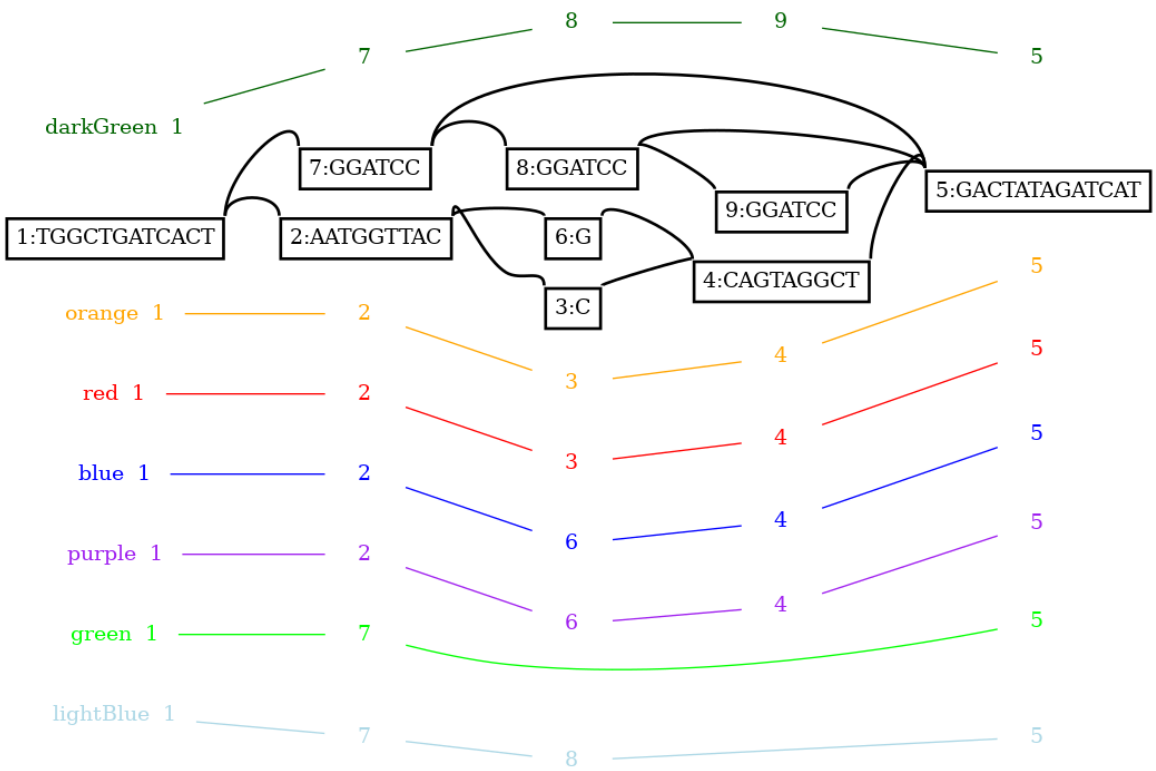


FIGURE 23 – Pangénome des espèces bovines généré par GraphViz à partir du fichier GFA écrit manuellement, disponible à la figure 8. Le fichier `.dot` a été modifié afin de personnaliser les chemins des individus en accord avec les couleurs associées à leurs noms.

5 Application à un cas réel

À présent que les différents outils de construction et d'analyse de pangénomes ont été explorés, il est temps de les mettre en application sur un cas concret. L'espèce choisie pour cette étude est le champignon *Pseudogymnoascus destructans*, connu pour son rôle dans le syndrome du nez blanc affectant les chauves-souris. Ce pathogène présente des variations structurales génomiques, notamment un grand nombre d'inversions chromosomiques, ce qui en fait un modèle pertinent pour évaluer la capacité des outils pangénomiques à détecter et représenter ces types de variations. La suite de cette section détaille le contexte biologique, les objectifs de l'étude, les moyens mis en œuvre pour analyser ces inversions ainsi qu'une discussion des résultats obtenus à partir des données génomiques disponibles.

5.1 Contexte

Pseudogymnoascus destructans, anciennement appelé *Geomyces destructans*, est une espèce fongique responsable du syndrome du nez blanc chez plusieurs espèces de chauves-souris (voir figure 24). Originaire de la région paléarctique, il s'est introduit en Amérique du Nord où il a entraîné une forte mortalité et un déclin des populations de chauves-souris. Dans sa région d'origine, le champignon entraîne rarement la mort de son hôte, ce qui suggère une coévolution avec les espèces locales de chauves-souris. Plus de quinze ans après son introduction en Amérique du Nord, certaines populations affectées commencent à montrer des signes de rétablissement, laissant envisager l'émergence progressive d'une coexistence entre l'hôte et le pathogène. [8]

FIGURE 24 – Chauves-souris atteintes du syndrome du nez blanc, présentant un anneau blanchâtre visible autour du museau. Crédit : CPEPESC FC.

En raison de sa préférence pour le froid, ce champignon ne se développe que chez des hôtes en hibernation. [2] Pendant l'hibernation, les chauves-souris passent par des phases de torpeur et des phases de réveil. Durant les phases de torpeur, elles sont inactives. Leur température corporelle chute et se rapproche de celle de leur environnement. Le métabolisme, la respiration, la circulation sanguine ainsi que le système immunitaire sont fortement ralentis. À intervalles réguliers, les chauves-souris sortent de cette torpeur.

Lors de ces réveils, elles peuvent se réhydrater, s'accoupler, chercher de la nourriture ou encore changer de site d'hibernation. Cependant, ces éveils sont très coûteux en énergie et consomment une grande partie des réserves de graisse accumulées avant l'hiver, nécessaires à leur survie jusqu'au printemps. *Pseudogymnoascus destructans* infecte les chauves-souris en colonisant les tissus cutanés, notamment au niveau des ailes et du museau. Cette infection perturbe profondément leur cycle d'hibernation : les individus infectés se réveillent plus fréquemment, consommant ainsi rapidement leurs réserves énergétiques accumulées pour l'hiver. Incapables de reconstituer ces réserves en raison de l'indisponibilité de nourriture pendant cette période, les chauves-souris finissent par mourir d'émaciation, affaiblies et épuisées par cette dépense énergétique anormale. [8]

Ce champignon appartient à l'embranchement des *Ascomycètes*, une vaste division regroupant une grande diversité d'espèces fongiques. Ces dernières présentent des différences marquées dans leurs modes de reproduction : certaines se reproduisent de manière sexuée entre deux types sexuels distincts (*mating-types*), tandis que d'autres se reproduisent par voie clonale ou par reproduction parasexuée. À ce jour, les données disponibles ne permettent pas de déterminer quel mode de reproduction est le plus couramment utilisé par *Pseudogymnoascus destructans*. On peut cependant noter la présence de deux types sexuels chez cette espèce : MAT1-1 et MAT1-2.[5]

5.2 Objectif

Chez *Pseudogymnoascus destructans*, un nombre particulièrement élevé d'inversions chromosomiques a été identifié en comparaison avec les autres espèces du genre *Pseudogymnoascus*, ainsi qu'avec la majorité des Ascomycètes. [5] Cette anomalie structurale de l'ADN motive l'exploration des représentations pangénomiques comme support d'étude afin d'y observer les inversions.

Dans le cadre de ce stage, l'objectif sera donc d'explorer la capacité des pangénomes à représenter les inversions. Pour cela, il faudra déterminer si ce type de variation structurale est bien capturé dans les graphes générés et s'il est possible de les localiser afin de les analyser. Pour réaliser cet objectif, il sera question d'observer une inversion connue impactant le type sexuel (*mating-type*) du champignon étudié. Cette inversion mesure environ 40 000bp et est localisée sur le chromosome 1, le plus grand de l'espèce dont la taille dépasse 4 000 000 paires de bases. Elle traverse le gène responsable du *mating-type* qui s'étend sur 7704 bases pour le type MAT1-1 et sur 6210 bases pour le type MAT1-2.

En fonction des résultats, il sera possible d'évaluer la pertinence des pangénomes pour détecter et analyser les inversions chromosomiques.

5.3 Moyens

Pour visualiser l'inversion ayant un impact sur le type sexuel, nous disposons des données génomiques de neuf individus différents de l'espèce *Pseudogymnoascus destructans* de lignée 1, dont la plupart se trouvent en Europe (voir figure 25).

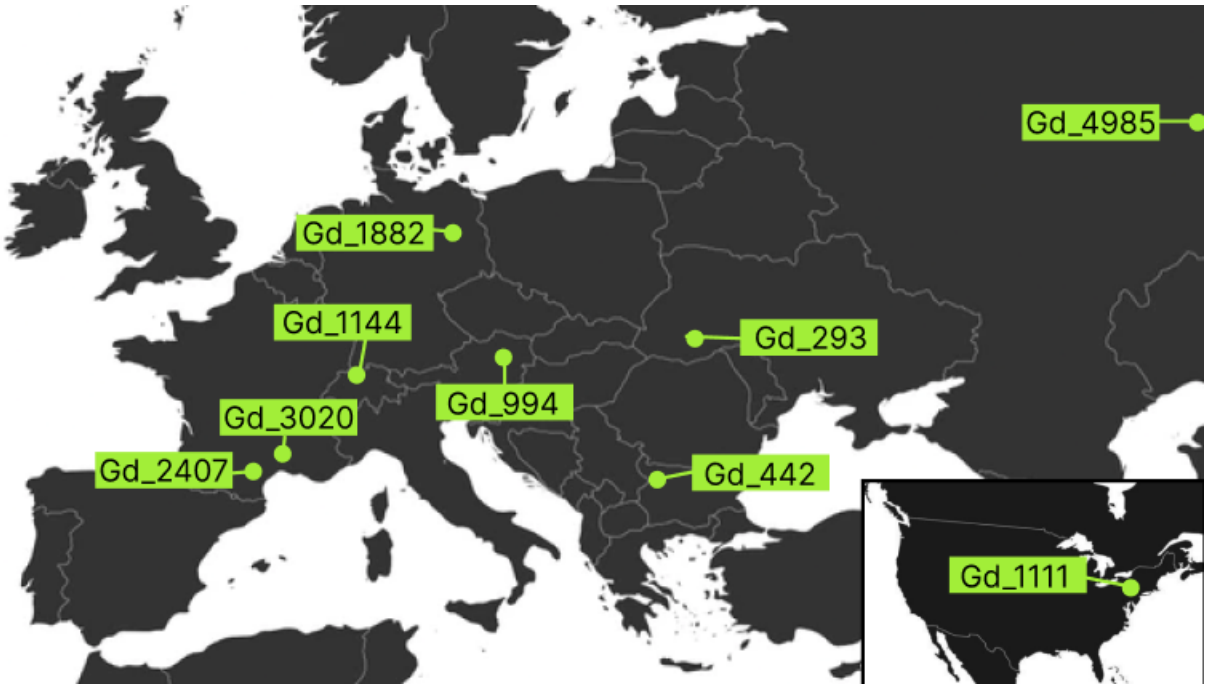


FIGURE 25 – Localisation approximative des individus de l'espèce *Pseudogymnoascus destructans* de lignée 1 dont les données génomiques sont possédées. Source : [5]

Tous les individus, excepté `Gd_1144`, possèdent cette inversion dans leur génome. Pour la visualiser, la séquence de l'individu `Gd_1144` sera donc notre référence. On ne connait pas exactement la position de cette inversion mais l'on connaît la position du gène responsable du type sexuel qu'elle traverse pour chacun de nos individus comme le montre la table 2.

Individu	**Position du gène (bp)**	**Taille du gène (bp)**	**Type sexuel**
Gd_293	1102953–1095255	7704	MAT1-1
Gd_442	1088641–1080943	7704	MAT1-1
Gd_994	1115705–1108007	7704	MAT1-1
Gd_1111	1097889–1090191	7704	MAT1-1
Gd_1144	1105731–1111937	6210	MAT1-2
Gd_1882	1103069–1095371	7704	MAT1-1
Gd_2407	1150996–1143298	7704	MAT1-1
Gd_3020	1792870–1785172	7704	MAT1-1
Gd_4985	1200877–1193179	7704	MAT1-1

TABLE 2 – Position du gène impliqué dans le type sexuel chez *Pseudogymnoascus destructans* sur le chromosome 1. La colonne colorée correspond à l'individu `Gd_1144`, le seul de notre échantillonnage à ne pas présenter l'inversion dans son génome, et par conséquent le seul individu de type *MAT1-2*. [5]

Pour entamer la recherche de l'inversion, j'ai choisi de construire deux pangénomes : l'un avec l'outil PGGB, l'autre avec Minigraph-Cactus, dans le but d'obtenir des graphes capables de représenter l'ensemble des variations structurelles. Cependant, Minigraph-Cactus n'a pas réussi à aboutir à une construction complète. À chaque tentative, une erreur

survient après environ 1h30 d'exécution. Le message affiché dans les logs du conteneur Docker est : `"error waiting for container: unexpected EOF"`. Cette erreur indique un arrêt brutal du processus, sans explication claire. En conséquence, la suite de l'analyse reposera uniquement sur le pangénome généré par PGGB.

Le graphe a ensuite été visualisé dans Bandage, mais j'ai rapidement constaté qu'effectuer une recherche dans ce pangénome serait impossible à cause de sa taille. En effet, sa longueur empêche une visualisation complète, rendant la navigation particulièrement difficile, comme l'illustre la figure 26. Par conséquent, il sera nécessaire de se concentrer sur un sous-graphe réduit, ciblant une région spécifique, afin de faciliter l'analyse et la recherche.

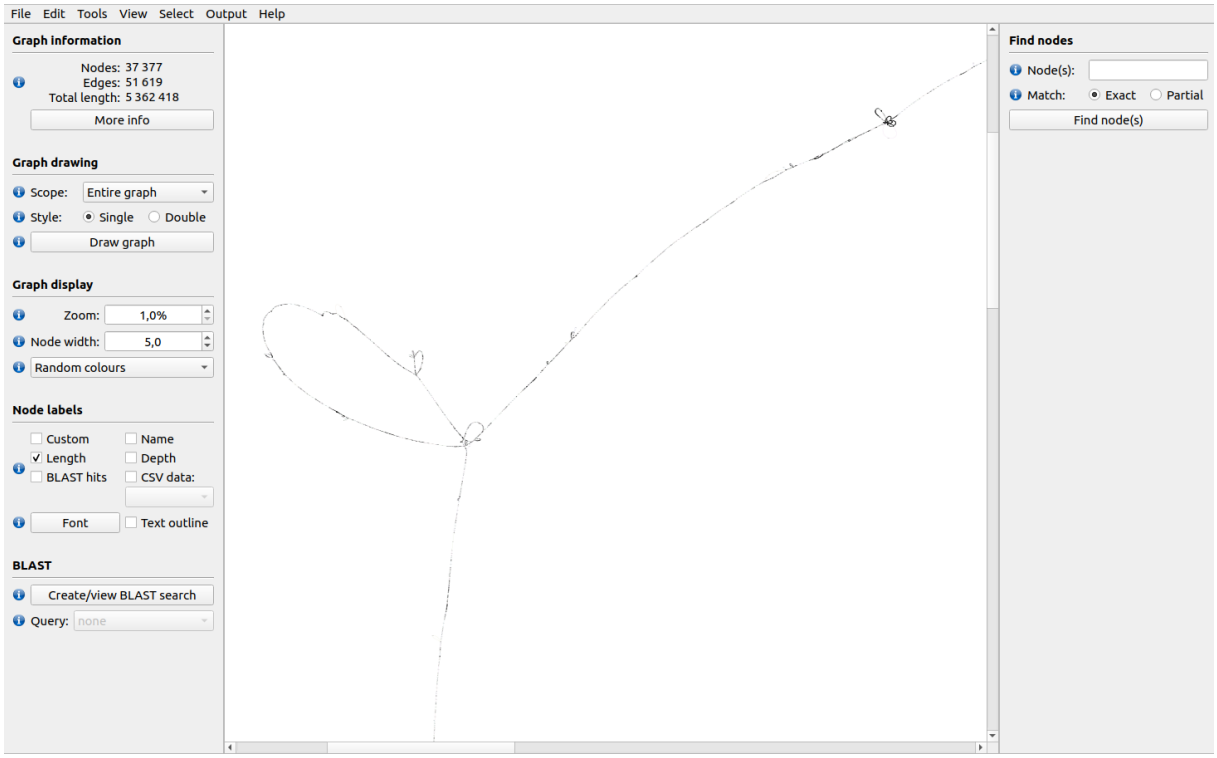


FIGURE 26 – Pangénome du chromosome 1 construit avec PGGB à partir des séquences de nos neuf individus de l'espèce *Pseudogymnoascus destructans* de lignée 1, affiché par Bandage. Même avec un zoom externe maximal, il reste impossible de visualiser le pangénome dans son intégralité.

Pour obtenir un sous-graphe réduit, j'ai choisi de partir du positionnement du gène chez l'individu `Gd_1144` (1105731–1111937 bp), en ajoutant 40kb de chaque côté, correspondant à la taille de l'inversion, afin de garantir que cette nouvelle région l'inclue entièrement. Le nouveau pangénome qui sera étudié sera donc dans l'interval [1066731 ; 1151937] de l'individu `Gd_1144`. Pour extraire ce sous graphe, j'ai utilisé la commande `odgi extract`, montrée dans la figure 27.

```
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi extract -i pggb_gd_chr1.og -r
"gd1144:1066731-1151937" -o pggb_gd_chr1_sub.og
```

FIGURE 27 – Commande `odgi extract` utilisée pour obtenir un sous graphe-réduit dans l'interval [1066731 ; 1151937] de l'individu `Gd_1144`.

Le sous-graphe obtenu, visible à la figure 28, s'avère bien plus adapté à la recherche de l'inversion. Il peut être visualisé dans son intégralité, et la navigation entre les segments y est nettement plus aisée.

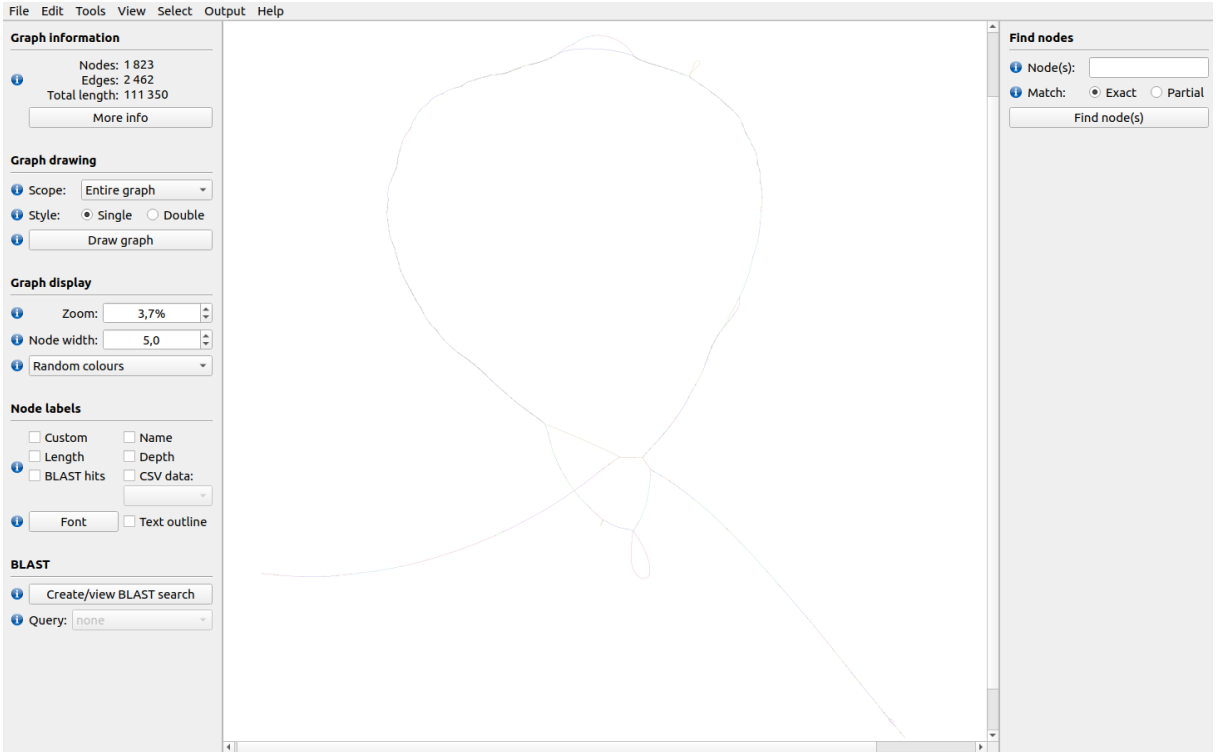


FIGURE 28 – Sous-graphe pangénomique de celui présenté dans la figure 26 obtenu grâce à `odgi extract`.

Afin de mieux s'y repérer, j'ai souhaité localiser, au sein de ce sous-graphe, le gène responsable du type sexuel de *Pseudogymnoascus destructans*, en utilisant à nouveau ODGI et les positions pour chaque individu, disponible dans la table 2. Les résultats présentés en annexe A.8 sont très satisfaisants : PGGB a identifié l'inversion comme l'indiquent les orientations opposées (+ et −) selon les individus. Il est également parvenu à regrouper le début et la fin du gène dans un même segment, malgré le fait que cette région occupe des positions différentes dans le génome selon les individus. Seul l'individu `Gd_1144` présente un segment de départ du gène différent de celui des autres, ce qui est cohérent puisqu'il possède une version plus courte du gène (voir table 2) ; celui-ci débute donc dans un segment plus en aval comme le montre la figure 29.

FIGURE 29 – Zoom sur le sous-graphe pangénomique présenté dans la figure 26. Les segments correspondant aux limites du gène, identifiés à l'aide de la commande `odgi position`, sont annotés. Chez l'individu `Gd_1144`, la séquence du gène s'étend entre les segments 8232 à 8570, tandis que pour tous les autres individus, elle se situe entre les segments 8127 et 8570.

Bien que l'on sache qu'une inversion est présente dans cet extrait du pangénome, elle n'est pas directement visible avec Bandage. L'outil ne permet pas de visualiser ce type de variation structurelle de manière évidente, ce qui nous oblige à poursuivre l'analyse de façon plus ciblée et manuelle. Par ailleurs, la taille actuelle de l'extrait de pangénome reste encore trop importante pour pouvoir être utilisé par les autres visualisateurs présentés dans ce rapport (Gfaviz et GraphViz). Pour mettre en évidence l'inversion, il sera donc nécessaire d'extraire les chemins des différents individus ainsi que les identifiants des segments concernés par l'inversion. Cela n'est possible uniquement parce que PGGB a correctement identifié l'inversion dans le génome.

Grâce au script présenté en annexe A.4, j'ai pu récupérer l'ensemble des segments traversés par un individu, en particulier ceux affectés par une inversion. En copiant cette suite d'identifiants dans l'outil de visualisation Bandage, il devient alors possible de repérer clairement l'inversion, comme illustré à la figure 30.

FIGURE 30 – Pangénome généré par PGGB, avec le chemin de l'individu `Gd_293` mis en couleur : la séquence normale apparaît en vert, tandis que la région inversée est représentée en rouge. La largeur des segments a été augmentée pour mieux visualiser la différence de couleur.

Après analyse du sens des chemins empruntés par les individus, qui n'est pas évident à première vue, il apparaît dans la figure 31 que tous les segments concernés par l'inversion sont parcourus dans un sens opposé à celui de l'individu de référence `Gd_1144`, qui ne présente pas cette inversion dans la région observée.

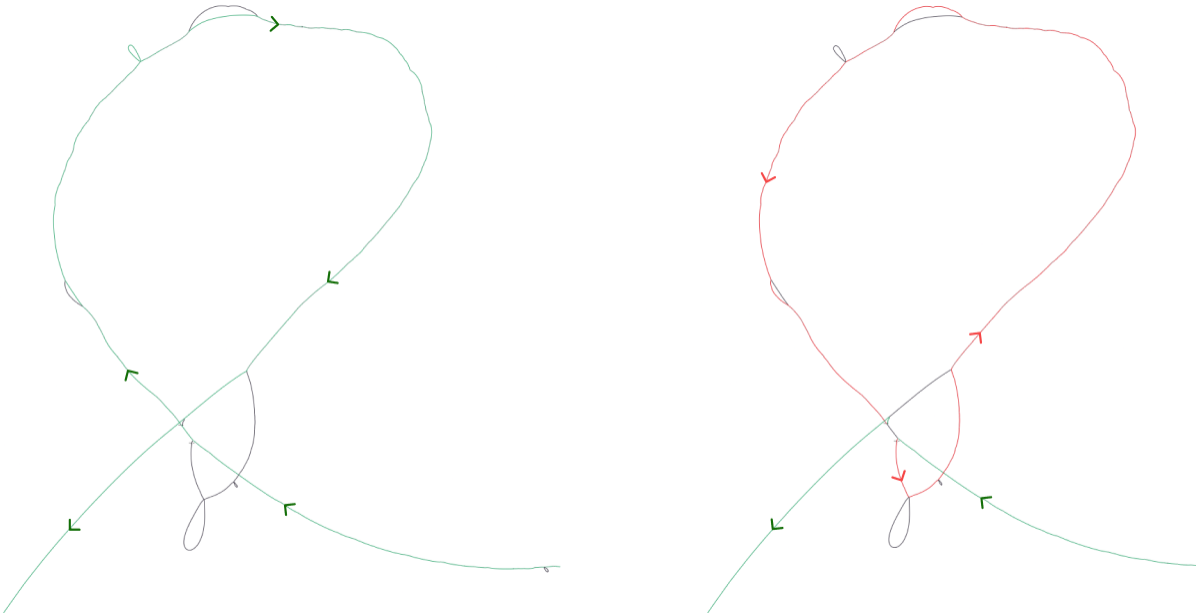


FIGURE 31 – Sous-graphe pangénomique généré avec PGGB et affiché dans Bandage, montrant à droite le chemin suivi par l'individu de référence `Gd_1144` et à gauche celui de l'individu `Gd_1882`. Les segments en vert correspondent à la séquence non inversée, tandis que ceux en rouge indiquent les régions affectées par une inversion. Des flèches ont été ajoutées pour visualiser le sens de parcours des chemins.

En superposant les chemins de chaque individu sur le pangénome, on s'aperçoit que, pour tous les individus concernés, l'inversion débute au segment n°7806 et se termine au segment n°7808. C'est dans cette zone plus précise qu'un nouveau sous-graphe est généré, en utilisant la fonctionnalité de création de sous-graphe de Bandage, afin d'obtenir un graphe plus petit et ainsi pouvoir l'observer avec le logiciel Gfaviz (voir figure 32).

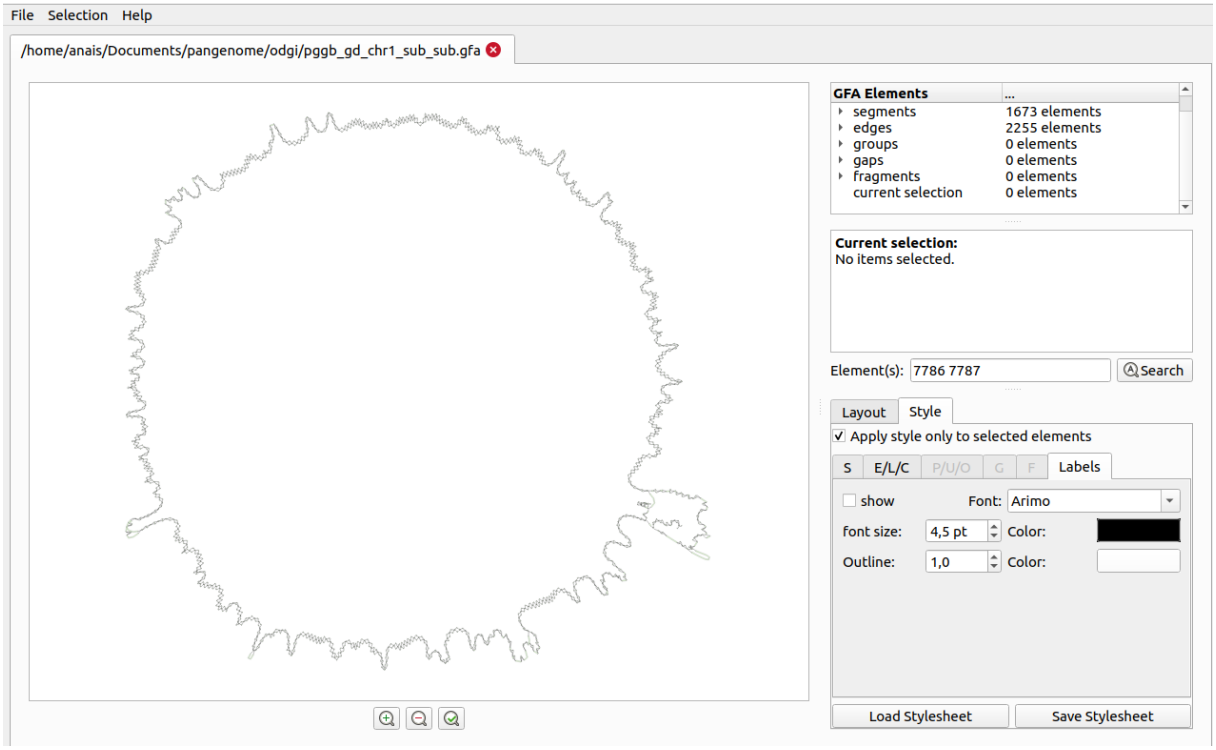


FIGURE 32 – Sous-graphe pangénomique généré par PGGB et visualisé avec GfaViz. La région sélectionnée, formant une boucle, apparaît sous la forme d'un cercle non connecté.

Encore un fois l'inversion n'est pas directement visible avec ce logiciel. Il est possible de visualiser les chemins des individus en extrayant les lignes P du fichier GFA d'origine, qui n'ont pas été incluses lors de la création du sous-sous-graphe avec Bandage, puis en les corrigeant à l'aide du script présenté en annexe A.6. Mais comme le montre la figure 33, cela n'avance en rien pour observer l'inversion.

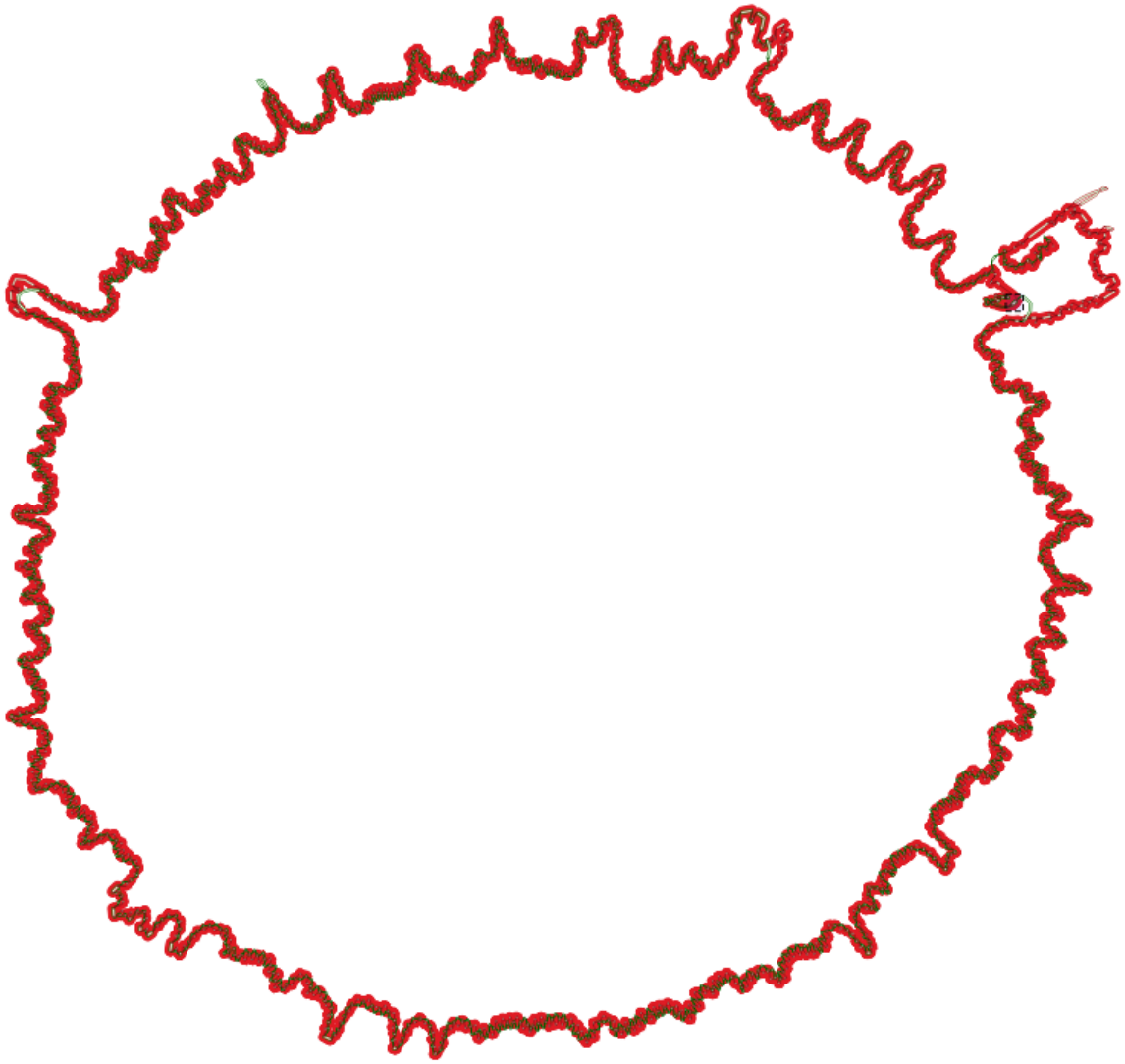


FIGURE 33 – Sous-graphe pangénomique généré par PGGB et visualisé avec GfaViz. Les lignes P ont été ajoutées dans le fichier GFA, ce qui permet de représenter les chemins de chaque individu. Tous les individus concernés par l'inversion ont leur chemin coloré en rouge, tandis que `Gd_1144` a son chemin colorié en vert. Malgré ces annotations, il reste impossible d'identifier l'inversion.

Il a donc fallu procédér de la même façon qu'avec le visuel de Bandage, c'est à dire extraire le chemin d'un individu en particulier ainsi que la zone touchée par l'inversion grâce au script disponible en annexe A.4 pour pouvoir la repérer sur le pangénome (voir figure 34).

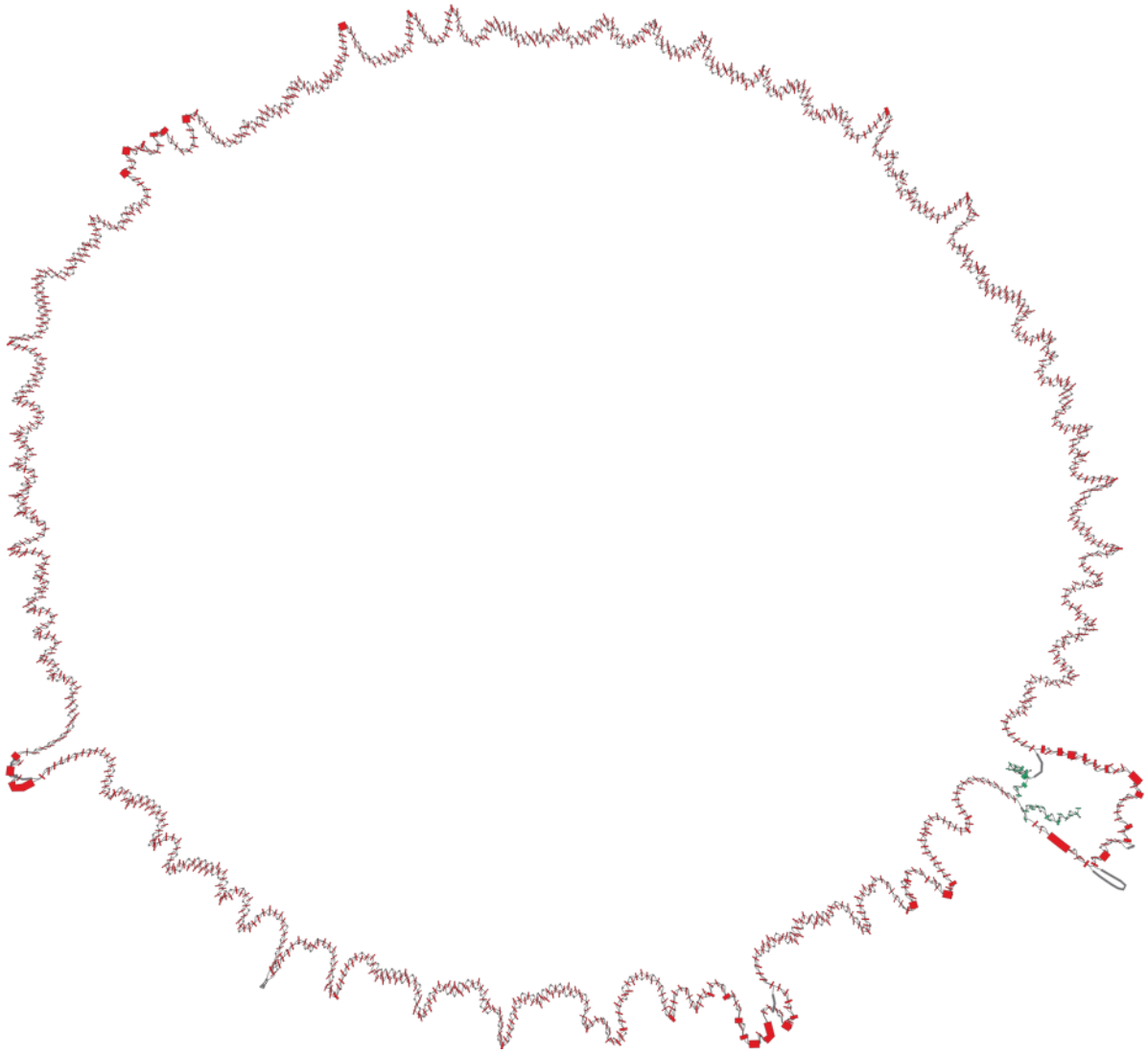


FIGURE 34 – Sous-graphe pangénomique généré avec PGGB et visualisé à l'aide de GfaViz. Le chemin correspondant à l'individu `Gd_293` est mis en évidence : la séquence normale est affichée en vert, tandis que les segments affectés par l'inversion apparaissent en rouge.

Pour visualiser l'inversion avec GraphViz, un sous-graphe encore plus restreint a été généré à partir du script présenté en annexe A.5. Les segments ont été choisis manuellement dans une région où une séquence « normale » est directement suivie d'une inversion, afin d'illustrer comment celle-ci apparaît dans le graphe. Cependant, il n'a pas été possible d'obtenir une représentation correcte de l'inversion. En effet, les individus porteurs parcourent la zone inversée dans le sens opposé à celui de l'individu de référence. Par conséquent, ce qui correspond au début de la région pour les individus inversés se trouve à la fin pour l'individu de référence. Si l'on extrait une portion trop courte du graphe, les segments conservés ne seront pas continus pour tous les individus, et la visualisation obtenue sera incorrecte.

Afin de valider les résultats obtenus par PGGB, la commande `odgi position` a été utilisée une seconde fois pour déterminer la position précise de l'inversion dans les génomes de chaque individu la possédant. Pour cela, les segments 7806 et 7808 du pangénome

construit, entre lesquels l'inversion se trouve, ont été utilisé. Cette commande, illustré dans la figure 35, a permis d'obtenir la table 3.

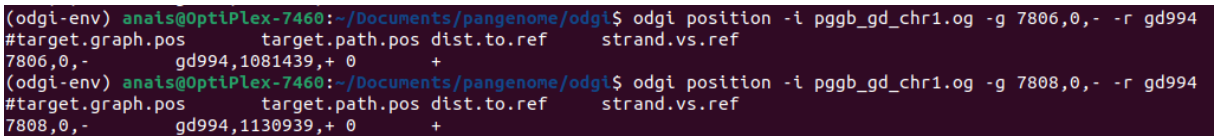


FIGURE 35 – Commande `odgi position` permettant de récupérer les positions génomiques de l'inversion dans le chromosome 1 de l'individu `Gd_994`.

Individu	Position (bp)	Longueur (bp)
gd293	1118188 - 1068699	49 489
gd442	1103875 - 1054375	49 500
gd994	1130939 - 1081439	49 500
gd1111	1113123 - 1063635	49 488
gd1882	1118303 - 1068803	49 500
gd2407	1166230 - 1115795	50 435
gd3020	1808108 - 1751600	56 508
gd4985	1216083 - 1166611	49 472

TABLE 3 – Positions et longueurs de l'inversion étudiée chez nos individus la possédant, selon le pangénome construit par PGGB. Les positions ont été obtenues grâce à la commande `odgi position`.

Dans le but de réaliser un alignement local, deux nouveaux fichiers FASTA ont été créés, contenant uniquement la région d'intérêt : l'un pour l'individu de référence, et l'autre pour un individu présentant l'inversion. Celui contenant l'inversion va être modifié grâce à la commande `seqkit seq` pour obtenir son complément inverse.

```
samtools faidx gd4985_chr1.fasta

samtools faidx gd4985_chr1.fasta gd4985_chr1001:1166611-1216083
> gd4985_inv.fasta

seqkit seq -r -p -t DNA gd4985_inv.fasta > gd4985_complementReverse.fasta
```

Commande permettant d'extraire la séquence de l'inversion au format FASTA et de générer son reverse complement.

Les deux fichiers FASTA générés ont été soumis au logiciel en ligne EMBOSS Matcher (`https://www.ebi.ac.uk/jdispatcher/psa/emboss_matcher`), qui permet d'effectuer un alignement local entre deux séquences. Les résultats de cet alignement sont présentés dans la figure 36.

```
########################################
# Program: matcher
# Rundate: Fri  4 Jul 2025 15:09:26
# Commandline: matcher
#    -auto
#    -stdout
#    -asequence emboss_matcher-I20250704-150835-0436-87147009-p1m.asequence
#    -bsequence emboss_matcher-I20250704-150835-0436-87147009-p1m.bsequence
#    -datafile EDNAFULL
#    -gapopen 16
#    -gapextend 4
#    -alternatives 1
#    -aformat3 pair
#    -snucleotide1
#    -snucleotide2
# Align_format: pair
# Report_file: stdout
########################################

#=======================================
#
# Aligned_sequences: 2
# 1: 1086262-1129168
# 2: 1166611-1216083
# Matrix: EDNAFULL
# Gap_penalty: 16
# Extend_penalty: 4
#
# Length: 42492
# Identity:   37212/42492 (87.6%)
# Similarity: 37212/42492 (87.6%)
# Gaps:        2530/42492 ( 6.0%)
# Score: 159108
#
#
#=======================================
```

FIGURE 36 – Résultat de l'aligmement local réalisé par EMBOSS MATCHER sur la région d'intéret entre l'individu de référence `Gd_1144` et le reverse complement de l'individu `Gd_4985`.

5.4 Résultats et discussion

L'objectif principal de ce travail était d'évaluer dans quelle mesure les outils de construction de pangénome ainsi que les visualisateurs de GFA permettent de représenter et de localiser les inversions chromosomiques. Pour cela, une inversion d'environ 40 000 paires de base traversant le type sexuel chez *Pseudogymnoascus destructans* a été étudiée.

La tentative de construction d'un pangénome à l'aide de Minigraph-Cactus s'est révélée

infructueuse. En revanche, l'outil PGGB a permis d'obtenir un graphe pangénomique complet pour le chromosome 1 de neuf individus. Bien que ce graphe ait initialement été trop volumineux pour une analyse directe, l'extraction d'un sous-graphe centré sur la région d'intérêt a rendu l'exploration possible.

PGGB s'est montré particulièrement performant pour capturer la variation structurale recherchée. En effet, l'inversion a été identifiée implicitement par des différences d'orientation dans les chemins suivis par les individus. Tous les individus MAT1-1 (porteurs de l'inversion) présentent une orientation opposée à celle de l'individu de référence MAT1-2 (`Gd_1144`) dans la région ciblée. Cette inversion affecte de manière cohérente les mêmes segments chez tous les individus concernés, ce qui témoigne de la robustesse de l'assemblage pangénomique, capable d'identifier cette variation structurelle même dans des régions génomiques pourtant divergentes entre individus.

La visualisation dans Bandage a permis de mettre en évidence cette inversion en colorant les chemins selon les individus et en annotant les orientations. En revanche, Bandage ne permet pas une détection automatique ou intuitive de l'inversion ; celle-ci n'est rendue visible qu'après un travail manuel de filtrage et d'annotation des segments concernés. GFaviz a été utilisé de manière similaire sur un sous-graphe encore plus réduit afin d'éviter les limitations techniques, mais cela n'a pas apporté d'informations supplémentaires par rapport à l'analyse réalisée avec Bandage. Pour que l'analyse de l'inversion à l'aide de GFaviz ou GraphViz soit pertinente, elle aurait dû être beaucoup plus petite en paire de base.

Ces résultats révèlent que les outils de visualisation ne sont pas encore pleinement adaptés pour une lecture directe des inversions, même si l'information est bien contenue dans le fichier GFA. Cependant, il est possible de contourner ces limitations en extrayant les données avec des scripts et des outils complémentaires comme ODGI, permettant ainsi de visualiser manuellement les inversions. En définitive, ce travail met en lumière le potentiel des pangénomes pour représenter toutes sortes de variations chromosomiques, tout en soulignant les limites actuelles en termes d'ergonomie et de visualisation. Dans une perspective plus large, ces résultats ouvrent la voie à une utilisation plus systématique des pangénomes pour étudier les variations structurales complexes, notamment dans des contextes évolutifs comme celui de *Pseudogymnoascus destructans* où la plasticité génomique semble jouer un rôle clé dans son type de reproduction.

6 Conclusion

Lors de ce stage, plusieurs outils ont été utilisés et explorés afin de mieux comprendre leur usage dans la construction et la visualisation de pangénomes. Une attention particulière a été portée à la recherche d'inversions dans les pangénomes générés, qui se sont avérées difficiles à identifier. Cela soulève des questions sur l'efficacité actuelle des pangénomes pour la visualisation des variations structurales de ce type.

Les pangénomes restent une approche récente, comme en témoigne le manque de documentation sur les différents outils utilisés. Cependant, on peut raisonnablement s'attendre à ce que ces méthodes et leurs logiciels associés s'améliorent rapidement dans le futur. En effet, les pangénomes offrent une perspective plus complète et pertinente que la simple utilisation d'une seule séquence de référence, car ils permettent d'intégrer et d'analyser la diversité génétique sous toutes ses formes, des petites variations aux plus grandes.

Malgré ces limites actuelles, les pangénomes restent très prometteurs, notamment pour explorer les variations structurales complexes dans les génomes. Leur évolution et amélioration permettront sans doute d'approfondir notre compréhension des mécanismes évolutifs et de la plasticité génomique qui façonnent la diversité au sein des populations.

Références

[1] Christine Tranchant Dubreuil. *Characterization of the pangenome variability and its role in domestication and adaptation in African rice*. 2023. URL : https://theses.hal.science/tel-04722481v1.

[2] Nicola M. Fischer, Serena E. Dool et Sebastien J. Puechmaille. "Seasonal patterns of *Pseudogymnoascus destructans* germination indicate host–pathogen coevolution". In : *Biology Letters* 16.6 (juin 2020), p. 20200177. ISSN : 1744-9561, 1744-957X. DOI : 10.1098/rsbl.2020.0177. URL : https://royalsocietypublishing.org/doi/10.1098/rsbl.2020.0177.

[3] *Graphical Fragment Assembly (GFA) Format Specification*. https://github.com/GFA-spec/GFA-spec. 2020.

[4] Glenn Hickey et al. "Pangenome graph construction from genome alignments with Minigraph-Cactus". In : *Nature Biotechnology* (2024). DOI : 10.1038/s41587-023-01793-w. URL : https://doi.org/10.1038/s41587-023-01793-w.

[5] Malo Lorazo. *Caractérisation et impact du système de reproduction sur l'évolution moléculaire de la fonge Pseudogymnoascus destructans (Ascomycète)*. Rapport de stage Diplôme de Master 2, Université de Montpellier. 2025.

[6] Camille Marchet. *Advances in colored k-mer sets : essentials for the curious*. arXiv :2409.05214. arXiv, 10 sept. 2024. DOI : 10.48550/arXiv.2409.05214. arXiv : 2409.05214[q-bio]. URL : http://arxiv.org/abs/2409.05214.

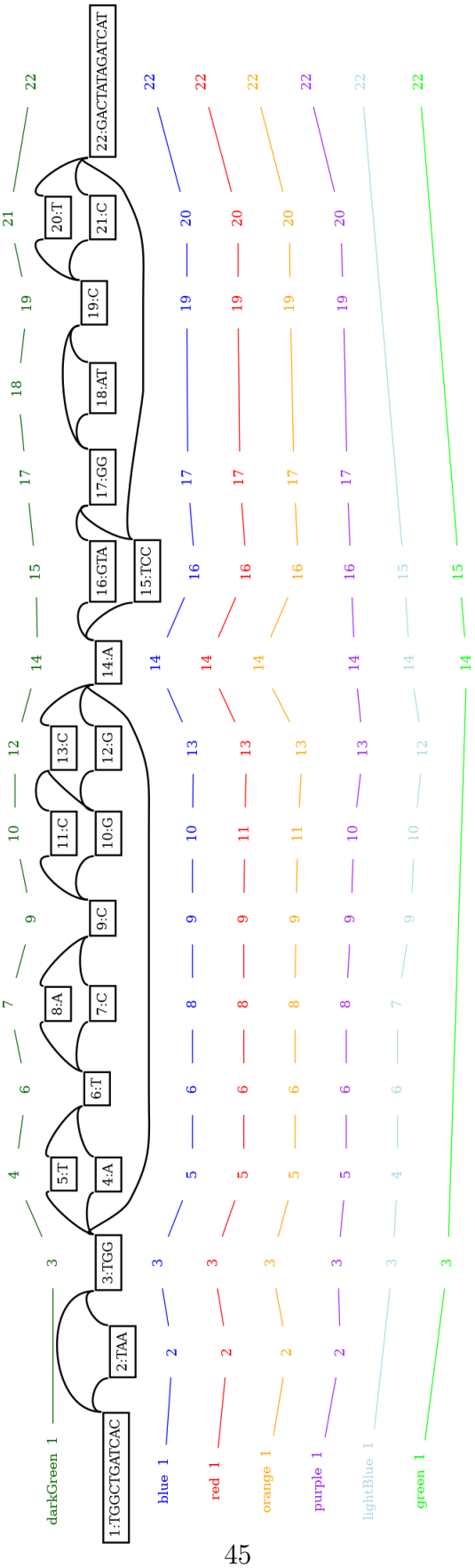
[7] Timothy P. L. Smith et al. "The Bovine Pangenome Consortium : democratizing production and accessibility of genome assemblies for global cattle breeds and other bovine species". In : *Genome Biology* 24.1 (19 juin 2023), p. 139. ISSN : 1474-760X. DOI : 10.1186/s13059-023-02975-0. URL : https://genomebiology.biomedcentral.com/articles/10.1186/s13059-023-02975-0.

[8] F. Whiting-Fawcett et al. "A Palearctic view of a bat fungal disease". In : *Conservation Biology* 39.1 (fév. 2025), e14265. ISSN : 0888-8892, 1523-1739. DOI : 10.1111/cobi.14265. URL : https://conbio.onlinelibrary.wiley.com/doi/10.1111/cobi.14265.

[9] Ryan R. Wick et al. "Bandage : interactive visualization of *de novo* genome assemblies". In : *Bioinformatics* 31.20 (15 oct. 2015), p. 3350-3352. ISSN : 1367-4811, 1367-4803. DOI : 10.1093/bioinformatics/btv383. URL : https://academic.oup.com/bioinformatics/article/31/20/3350/196114.

A Annexes

A.1 Pangénome construit à l'aide de PGGB avec les données bovines de tests, affiché grâce à GraphViz.



A.2 Script python pour récupérer les chemins du graphe initial depuis le fichier paf généré par Cactus.

```python
from collections import defaultdict

# Fonction pour recuperer le chemin traverse pour chaque individu d'entree
def getPathsMinigraph(paf_file):
    paths = defaultdict(list)

    # ouvre le fichier PAF genere par Minigraph-Cactus
    with open(paf_file) as file:
        # pour chaque ligne du fichier: on recupere l'id de la sequence concernee,
        # le segment qu'elle traverse et dans quelle direction
        for line in file:
            fields = line.strip().split('\t')
            seq_id = fields[0]
            direction = fields[4]
            segment_field = fields[5]  # ex: id=_MINIGRAPH_|s1
            segment_id = None
            if segment_field.startswith('id=_MINIGRAPH_|'):
                # on recupere uniquement l'id du segment
                segment_id = segment_field.split('|')[1]
            else:
                segment_id = segment_field

            paths[seq_id].append(( segment_id, direction))
    return paths

# on entre le lien du fichier paf
paf_path = "gd.paf"
paths = getPathsMinigraph(paf_path)

# On cree deux fichiers texte pour repertorier les chemins des individus :
# un pour Bandage et un pour GfaViz
with open("bandage_ver.txt", "w") as file:
    for seq_id, values in paths.items():
        file.write(f">{seq_id}\n")
        file.write(','.join(f"{seg_id}{dir_}" for seg_id, dir_ in values)+ "\n")


with open("gfaviz_ver.txt", "w") as file:
    for seq_id, values in paths.items():
        file.write(f">{seq_id}\n")
        file.write(' '.join(f"{seg_id}" for seg_id, dir_ in values)+ "\n")
```

A.3 Script python pour récupérer les chemins du graphe final depuis le fichier full.gfa généré par Cactus.

```python
import re
import os

input_gfa = "data/gd.full.gfa"
out_dir = "paths_cactus"

# permet de creer un dossier
os.makedirs(out_dir, exist_ok=True)

with open(input_gfa, "r") as file:
    for line in file:
        # on s'interesse uniquement aux lignes W
        # contenant le chemin de segment d'un individu
        if line.startswith("W"):
            # decoupe la ligne avec les tabulations comme separateur
            fields = line.strip().split("\t")

            # recuperation de l'id de la sequence et son chemin
            seq_id = fields[1]
            seq_path = fields[6]

            # parse le chemin avec son sens et le reformate
            # avec juste l'id du segment et un signe + ou -
            segments = re.findall(r'([><])(\d+)', seq_path)
            format_segment_bandage = [f"{num}{'+' if orientation == '>' else '-'}"
                for orientation, num in segments]

            # on cree un fichier au nom de l'id de la sequence
            filename = f"{seq_id.lower()}.txt"
            filepath = os.path.join(out_dir, filename)

            # ecriture du chemin dans le nouveau fichier
            with open(filepath, "w") as file:
                file.write(",".join(format_segment_bandage) + "\n")
```

A.4 Script python pour obtenir les segments traversés par un individu, dont ceux concernés par une inversion

```python
import os

# ce script fonctionne uniquement avec les fichiers GFA generes par PGGB, qui
# contiennent des lignes P (path)

# nom du fichier entrant
gfa_file = "data/pggb_gd_chr1_sub.gfa"
sub = True

# creation du dossier de sortie contenant les resultats
output_dir = "gd_chr1_inversion"
os.makedirs(output_dir, exist_ok=True)

if(sub):
    existing_segments = set()
    with open(gfa_file, 'r') as file:
        for line in file:
            if line.startswith("S"):
                parts = line.strip().split('\t')
                segment_id = parts[1]
                existing_segments.add(segment_id)

with open(gfa_file, 'r') as file:
    for line in file:

        # on ne s'interesse qu'aux lignes P, contenant les chemins des individus
        if line.startswith("P"):
            # decoupage la ligne en fonction des tabulations
            parts = line.strip().split('\t')
            # recuperation de l'identifiant de l'individu et des identifiants
            # des segments
            individual_id = parts[1]
            segment_part = parts[2].split(',')

            if(sub):
                segment_part  =
                    [seg for seg in segment_part if seg[:-1] in existing_segments]
            # creation d'une liste pour les segments concernes par une inversion (-)
            inversions = [seg[:-1] for seg in segment_part if seg.endswith('-')]
            # creation d'une liste pour tous les segments
            segments = [seg[:-1] for seg in segment_part ]

            # creation d'un fichier txt contenant les segments inverses
            # d'un individu
            inversion_path =
                os.path.join(output_dir, f"{individual_id}_inversions.txt")
            with open(inversion_path, 'w') as out_file:
                out_file.write(','.join(inversions))

            # creation d'un fichier txt contenant tous les segments
            # traverses par un individu
            all_path = os.path.join(output_dir, f"{individual_id}_path.txt")
            with open(all_path, 'w') as out_file:
                out_file.write(','.join(segments))
```

A.5 Script python pour créer un sous-graphe à partir d'un fichier GFA et une liste de segment voulus.

```
import gfapy

# renseignement des segments que l'on souhaite avoir dans le sous graphe
wanted = {"s1044", "s1042", "s1041", "s1043", "s5", "s4", "s3", "s2", "s1"}

# recuperation du gfa deja cree
gfa = gfapy.Gfa.from_file("data/pangenome.gfa")

# initialisation d'un nouveau graphe
subgraph = gfapy.Gfa()

#on copie chaque segment voulu dans le nouveau graphe
for segment in gfa.segments:
    if segment.name in wanted:
        subgraph.add_line(segment.clone())

# on recupere les liens entre les segments recuperes
for link in gfa.edges:
    from_seg = link.from_segment.name
    to_seg = link.to_segment.name

    if from_seg in wanted and to_seg in wanted:
        subgraph.add_line(link.clone())


# on enregistre ce nouveau graphe au format gfa
subgraph.to_file("subgraph.gfa")
```

A.6 Script python qui permet de corriger les chemins des individus d'un fichier GFA d'un sous-graphe pangénomique

```
input_file = "data/pggb_gd_chr1_sub_sub.gfa"

existing_segments= set()
fixed_lines = []

with open(input_file, 'r') as f:
    lines = f.readlines()

    # recuperation des segments existants
    for line in lines:
        if line.startswith('S'):
            seg_id = line.strip().split('\t')[1]
            existing_segments.add(seg_id)

    # creation de nouvelles lignes P
    for line in lines:
        if line.startswith('P'):
            fields = line.strip().split('\t')
            seq_id = fields[1]
            path_elems = fields[2].split(',')

            filtered_path = []

            for seg in path_elems:
                seg_id = seg[:-1]
                if seg_id in existing_segments:
                    filtered_path.append(seg)

            new_line = f"P\t{seq_id}\t{','.join(filtered_path)}\t*\n"
            fixed_lines.append(new_line)

    # reecriture du fichier d'origine hors pour les anciennes lignes P
    # + rajout des nouvelles
    with open(input_file, 'w') as f2:
        for line in lines:
            if not line.startswith('P'):
                f2.write(line)
        f2.writelines(fixed_lines)
```

A.7 Comparaison des ressources et résultats du graphe pour PGGB, Minigraph et Minigraph-Cactus

	PGGB	Minigraph	Minigraph-Cactus
1			
Mémoire max (Mo)	818	266	1376
Temps exécution (min)	4.9	5.9	22.3
Nb segments	37793	1180	40154
Nb liens	51317	1689	53697
Taille (bp)	3156550	3483611	3586304
2			
Mémoire max (Mo)	815	312	1565
Temps exécution (min)	5.0	6.1	22.3
Nb segments	37719	1180	40178
Nb liens	51178	1689	53727
Taille (bp)	3156568	3483611	3586298
3			
Mémoire max (Mo)	825	266	1517
Temps exécution (min)	5.0	6.0	22.1
Nb segments	37669	1180	40167
Nb liens	51120	1689	53715
Taille (bp)	3156705	3483611	3586293
4			
Mémoire max (Mo)	820	311	1600
Temps exécution (min)	4.8	5.9	22.4
Nb segments	37824	1180	40157
Nb liens	51330	1689	53701
Taille (bp)	3156959	3483611	3586295
5			
Mémoire max (Mo)	812	266	1566
Temps exécution (min)	4.8	5.9	22.3
Nb segments	37947	1180	40152
Nb liens	51528	1689	53697
Taille (bp)	3156534	3483611	3586304
Moyenne Totale			
Mémoire max (Mo)	818	284.2	1524.8
Temps exécution (min)	4.9	5.96	22.28
Nb segments	37790.4	1180	40161.6
Nb liens	51294.6	1689	53707.4
Taille (bp)	3156663.2	3483611	3586298.8

A.8 Position du gène en fonction du segment dans le graphe pangénomique pour chaque individu

```
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd1144,1105731" -v
#source.path.pos        target.graph.pos
gd1144,1105731,+        8232,2365,+
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd1144,1111937" -v
#source.path.pos        target.graph.pos
gd1144,1111937,+        8570,36,+
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd293,1102953" -v
#source.path.pos        target.graph.pos
gd293,1102953,+ 8127,14,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd293,1095255" -v
#source.path.pos        target.graph.pos
gd293,1095255,+ 8570,15,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd442,1088641" -v
#source.path.pos        target.graph.pos
gd442,1088641,+ 8127,14,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd442,1080943" -v
#source.path.pos        target.graph.pos
gd442,1080943,+ 8570,15,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd994,1115705" -v
#source.path.pos        target.graph.pos
gd994,1115705,+ 8127,14,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd994,1108007" -v
#source.path.pos        target.graph.pos
gd994,1108007,+ 8570,15,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd1111,1097889" -v
#source.path.pos        target.graph.pos
gd1111,1097889,+        8127,14,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd1111,1090191" -v
#source.path.pos        target.graph.pos
gd1111,1090191,+        8570,15,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd1882,1103069" -v
#source.path.pos        target.graph.pos
gd1882,1103069,+        8127,14,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd1882,1095371" -v
#source.path.pos        target.graph.pos
gd1882,1095371,+        8570,15,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd2407,1150996" -v
#source.path.pos        target.graph.pos
gd2407,1150996,+        8127,14,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd2407,1143298" -v
#source.path.pos        target.graph.pos
gd2407,1143298,+        8570,15,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd3020,1792870" -v
#source.path.pos        target.graph.pos
gd3020,1792870,+        8127,14,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd3020,1785172" -v
#source.path.pos        target.graph.pos
gd3020,1785172,+        8570,15,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd4985,1200877" -v
#source.path.pos        target.graph.pos
gd4985,1200877,+        8127,14,-
(odgi-env) anais@OptiPlex-7460:~/Documents/pangenome/odgi$ odgi position -i pggb_gd_chr1.og -p "gd4985,1193179" -v
#source.path.pos        target.graph.pos
gd4985,1193179,+        8570,15,-
```