

# Bio-informatica groepswerk handleiding

## Table of contents

<b>1</b>	<b>Introductie</b>	<b>2</b>
<b>2</b>	<b>VS Code op de Vlaamse Supercomputer Centrum infrastructuur</b>	<b>3</b>
<b>3</b>	<b>Introductie tot Linux Shell/Bash</b>	<b>6</b>
3.1	Wat is een Shell? . . . . .	6
3.2	Basisbegrippen . . . . .	6
3.3	Basisopdrachten . . . . .	6
3.3.1	pwd (Print Working Directory) . . . . .	6
3.3.2	ls (List) . . . . .	6
3.3.3	cd (Change Directory) . . . . .	7
3.3.4	mkdir (Make Directory) . . . . .	7
3.3.5	cp (Copy) . . . . .	7
3.3.6	mv (Move) . . . . .	7
3.3.7	rm (Remove) . . . . .	7
3.3.8	cat (Concatenate) . . . . .	8
3.3.9	echo . . . . .	8
3.4	Opdrachtstructuur . . . . .	8
3.5	Tips . . . . .	8
3.6	Oefenopdrachten . . . . .	9
<b>4</b>	<b>Kwaliteitscontrole met FastQC</b>	<b>9</b>
4.1	Doel . . . . .	9
4.2	Het FASTQ-formaat . . . . .	9
4.2.1	Wat is FASTQ? . . . . .	9
4.2.2	Voorbeeld van een FASTQ-entry . . . . .	9
4.2.3	Structuur van een FASTQ-bestand . . . . .	10
4.3	Opdrachten . . . . .	10
4.4	Het interpreteren van het FastQC HTML-rapport . . . . .	11
4.5	Opdrachtvragen: . . . . .	11

<b>5</b>	<b>Read Mapping met BWA</b>	<b>12</b>
5.1	Doel . . . . .	12
5.2	SAM-formaat (Sequence Alignment/Map) . . . . .	12
5.2.1	Wat is SAM? . . . . .	12
5.2.2	Structuur van een SAM-bestand . . . . .	12
5.2.3	Voorbeeld van een SAM-entry . . . . .	12
5.2.4	Uitleg . . . . .	12
5.3	Opdrachten . . . . .	13
5.4	BAM-formaat (Binary Alignment/Map) . . . . .	13
5.4.1	Wat is BAM? . . . . .	13
5.4.2	Kenmerken van BAM . . . . .	13
5.4.3	Voorbeeld . . . . .	14
5.5	Opdracht 2 . . . . .	14
5.6	Opdrachtvragen: . . . . .	14
<b>6</b>	<b>Variant Calling met Mutect2</b>	<b>14</b>
6.1	VCF-formaat (Variant Call Format) . . . . .	15
6.1.1	Wat is VCF? . . . . .	15
6.1.2	Structuur van een VCF-bestand . . . . .	15
6.1.3	Voorbeeld van een VCF-entry . . . . .	15
6.1.4	Uitleg . . . . .	15
6.2	Opdrachten . . . . .	16
<b>7</b>	<b>Variant Annotatie met SnpEff</b>	<b>16</b>
7.1	Het SnpEff HTML-rapport begrijpen . . . . .	17
7.1.1	A. Algemene Statistieken Sectie . . . . .	17
7.2	Praktische Oefeningen . . . . .	17
7.2.1	Oefening 1: Basis Statistieken . . . . .	17
7.2.2	Oefening 2: Variant Effects Analyse . . . . .	18
7.2.3	Oefening 3: Genoomregio's . . . . .	18
7.2.4	Oefening 4: Codon Veranderingen . . . . .	18
<b>8</b>	<b>Variant filtering met SnpSift</b>	<b>18</b>
8.1	Opdrachtvragen . . . . .	19

## 1 Introductie

Somatische varianten zijn genetische veranderingen die niet overgeërfd zijn, maar tijdens iemands leven in specifieke cellen ontstaan. Deze varianten zijn vooral belangrijk in de context van kanker, waar ze een cruciale rol spelen bij het ontstaan en de progressie van de ziekte. Het doel van somatische variant calling is om deze niet-geërfde mutaties te identificeren in DNA sequencing data van tumoren en andere stalen.

Dit proces omvat:

1. Sequencing van tumorweefsel of bloedcellen van een patiënt.
2. Alignment van de sequentiedata aan een referentiegenoom.
3. Identificatie van posities waar de tumorsequentie verschilt van het referentiegenoom. Dit noemen we varianten.
4. Filteren van de gevonden varianten om prognostisch, diagnostisch of therapeutisch relevante varianten te vinden.

Het belang van somatische variant calling ligt in verschillende gebieden:

- **Kankerdiagnostiek:** Het helpt bij het identificeren van de specifieke mutaties die een rol spelen in een individuele tumor.
- **Gepersonaliseerde behandeling:** Kennis van de somatische varianten kan helpen bij het kiezen van de meest effectieve behandeling voor een patiënt.
- **Onderzoek:** Het draagt bij aan ons begrip van de genetische basis van kanker en andere ziekten.
- **Monitoring:** Het kan worden gebruikt om de evolutie van een tumor in de tijd te volgen en de respons op behandeling te evalueren.

In dit groepswerk doorlopen we de stappen die nodig zijn om somatische variant calling uit te voeren. We leren de basisprincipes van sequentieanalyse en specifieke uitdagingen van het identificeren van somatische mutaties. Hierbij reflecteren we bij elke stap over de verkregen resultaten.

## 2 VS Code op de Vlaamse Supercomputer Centrum infrastructuur

Om de bio-informatica stappen uit te voeren in dit groepswerk gaan we gebruik maken van de VSC (Vlaamse Supercomputer Centrum) infrastructuur. We kunnen de VSC infrastructuur op verschillende manieren benaderen. De meest gebruiksvriendelijke manier is het *KU Leuven OnDemand* platform. Om in te loggen op dit platform doorloop je volgende stappen:

1. Surf met je browser naar <https://ondemand.hpc.kuleuven.be>
2. Kies hier de optie om in te loggen met een VSC account: “Partner organizations: VSC account”
3. Log in met je UHasselt account
4. Bij de vraag “Authorize vsc-challenge?” antwoord je “Authorize”

Je bent nu ingelogd op het *KU Leuven OnDemand* platform. Hiermee kan je vanuit je web-browser een aantal populaire applicaties opstarten op de Vlaamse SuperComputer. Meer achtergrond kan je vinden in de [handleiding](#).

Voor het groepswerk gaan we gebruik maken van de Visual Studio Code (VS Code) applicatie. Dit is een populaire applicatie om code te schrijven die een ingebouwde bestands browser en

*terminal* vensters heeft. Om VS Code te starten via het OnDemand platform doorloop je de volgende stappen.

1. Klik op het “code-server” icoon.
2. Er verschijnt een formulier met een aantal opties. Controleer volgende opties en pas aan indien nodig.
  - **Account:** lp\_h\_edu\_bioinformatics\_2024
  - **Number of hours:** 1 (of meer als je langer wenst te werken aan de opdracht)
  - **Required memory per core in megabytes::** 6800
3. Klik op “Launch”
4. Een overzicht van jouw recente sessies verschijnt met bovenaan de VS Code sessie met status “Queued”

code-server (57115337)

Queued

Created at: 2024-09-20 14:32:29 CEST

Time Requested: 2 hours

Session ID: [477fe300-708e-448c-88a0-7f8292ccee87](#)

Cancel

Please be patient as your job currently sits in queue. The wait time depends on the number of cores as well as time requested.

Figure 1: Queued

Nu moet je even wachten tot de VS Code sessie gestart is. Normaal duurt dit maar enkele seconden. Het zou mogelijk zijn dat je ook wat langer moet wachten als de VSC infrastructuur druk bezet is. Wanneer de sessie gestart is verschijnt “Running” en wordt het kader groen:

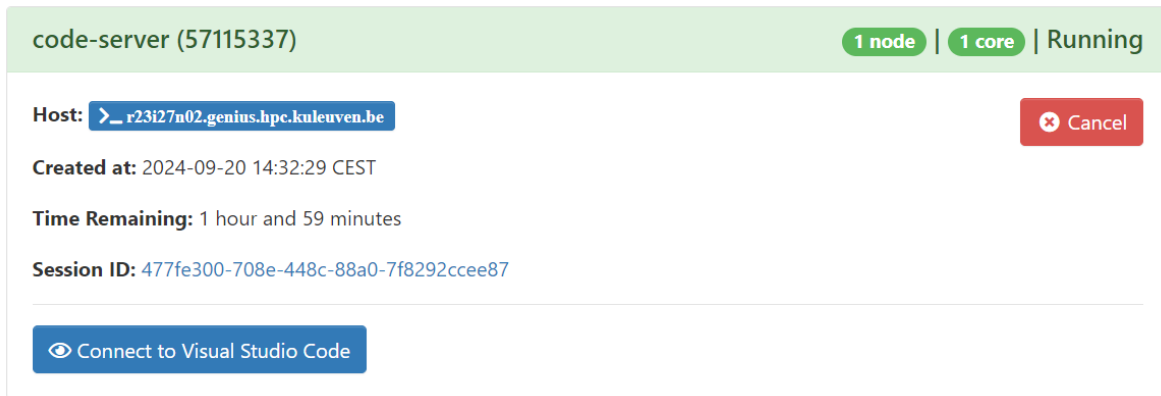


Figure 2: Ready

Klik nu op “Connect to Visual Studio Code”. De VS Code interface verschijnt.

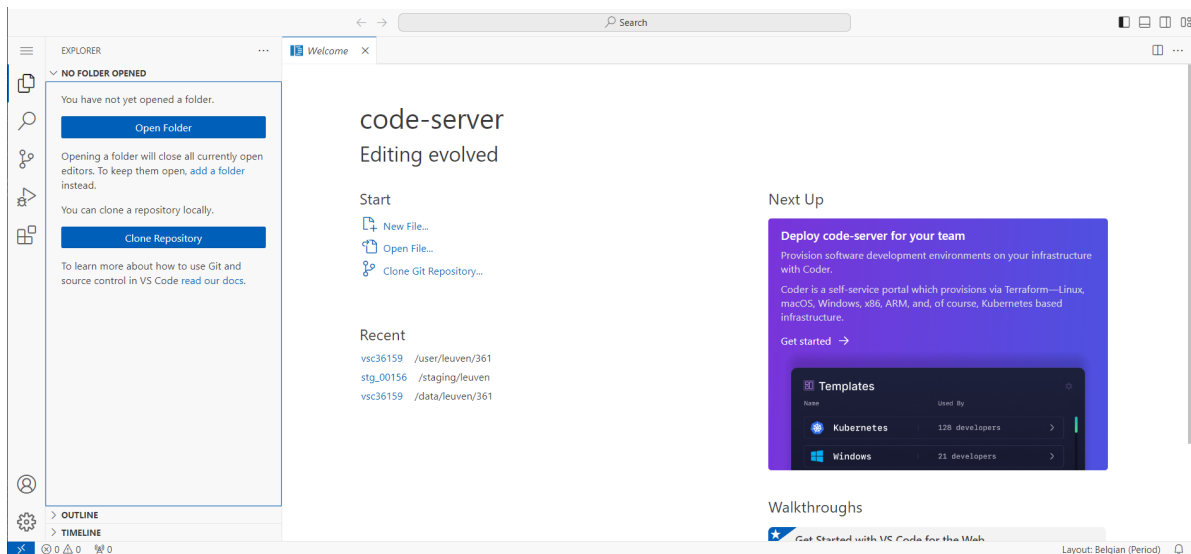


Figure 3: Alt text

De sessie op de VSC cluster zal blijven openstaan tot de gevraagde tijd voorbij is of de sessie manueel gestopt wordt (rode “cancel” knop). Als je klaar bent met werken en de sessie toch laat openstaan gaat de VSC cluster onnodig gereserveerd worden en zullen andere gebruikersodeloos moeten wachten. Sluit daarom de sessie af als je eerder klaar bent met werken dan de gevraagde tijd. Je kan steeds een nieuwe sessie starten!

## 3 Introductie tot Linux Shell/Bash

### 3.1 Wat is een Shell?

Een shell is een programma dat een interface biedt voor gebruikers om met het besturingssysteem te communiceren. De meest voorkomende shell in Linux-systemen heet Bash (Bourne Again SHell). Wanneer je de opdrachtregel gebruikt, typ je opdrachten in de shell.

### 3.2 Basisbegrippen

1. **Opdrachtprompt:** Hier typ je je opdrachten. Het eindigt meestal met een **\$**-teken.
2. **Opdrachten:** Dit zijn instructies die je aan de computer geeft.
3. **Argumenten:** Aanvullende informatie die je aan een opdracht geeft.
4. **Opties:** Wijzigen het gedrag van opdrachten, meestal beginnend met een streepje (-).

### 3.3 Basisopdrachten

#### 3.3.1 pwd (Print Working Directory)

Toont je huidige locatie in het bestandssysteem.

```
$ pwd  
/home/gebruikersnaam
```

#### 3.3.2 ls (List)

Geeft een lijst van bestanden en mappen in de huidige directory.

```
$ ls  
Documenten  Downloads  Afbeeldingen  Muziek
```

Opties:

- **ls -l:** Lang formaat, toont meer details
- **ls -a:** Toont verborgen bestanden (die beginnen met een punt)
- **ls -lh:** Toont de grootte van de bestanden (in de kolom) in een leesbaar formaat (K: kilobyte, M: megabyte, G: gigabyte).

### 3.3.3 cd (Change Directory)

Verplaatst je naar een andere directory.

```
$ cd Documenten
```

Speciale directories:

- `.` : Huidige directory
- `..` : Bovenliggende directory
- `~` : Thuisdirectory van de gebruiker

### 3.3.4 mkdir (Make Directory)

Maakt een nieuwe directory aan.

```
$ mkdir NieuweMap
```

### 3.3.5 cp (Copy)

Kopieert bestanden of directories.

```
$ cp bestand.txt Documenten/
```

Om een directory en zijn inhoud te kopiëren, gebruik de `-r` (recursief) optie:

```
$ cp -r MapA MapB
```

### 3.3.6 mv (Move)

Verplaatst of hernoemt bestanden en directories.

```
$ mv bestand.txt Documenten/  
$ mv oudenaam.txt nieuwenam.txt
```

### 3.3.7 rm (Remove)

Verwijdert bestanden of directories. Wees voorzichtig met deze opdracht!

```
$ rm bestand.txt
```

Om een directory en zijn inhoud te verwijderen, gebruik de `-r` optie:

```
$ rm -r MapNaam
```

### 3.3.8 cat (Concatenate)

Toont de inhoud van een bestand.

```
$ cat bestand.txt
```

### 3.3.9 echo

Print tekst naar het scherm.

```
$ echo "Hallo, Wereld!"  
Hallo, Wereld!
```

## 3.4 Opdrachtstructuur

De meeste opdrachten volgen deze structuur:

opdracht [opties] [argumenten]

Bijvoorbeeld:

```
$ ls -l Documenten
```

Hier is `ls` de opdracht, `-l` een optie, en `Documenten` een argument.

## 3.5 Tips

1. Gebruik de pijltjestoetsen omhoog en omlaag om door je opdrachtgeschiedenis te navigeren.
2. Gebruik Tab voor automatische aanvulling van bestands- en mapnamen.
3. Gebruik `man` gevolgd door een opdrachtnaam om de handleiding te zien (bijv. `man ls`).



### 3.6 Oefenopdrachten

1. Maak een directory genaamd “BioinformaticaCursus” in je thuisdirectory.
2. Maak binnen “BioinformaticaCursus” drie subdirectories: “Data”, “Scripts” en “Resultaten”.
3. Maak een leeg bestand genaamd “notities.txt” in de “BioinformaticaCursus” directory.
4. Toon de inhoud van “BioinformaticaCursus” in lang formaat.
5. Verplaats “notities.txt” naar de “Resultaten” directory.
6. Kopieer “notities.txt” van “Resultaten” naar “Data”.
7. Verwijder het “notities.txt” bestand uit de “Data” directory.

## 4 Kwaliteitscontrole met FastQC

### 4.1 Doel

Het hoofddoel van FastQC is om een snelle kwaliteitscontrole uit te voeren op ruwe sequentiedata afkomstig van high-throughput sequencing pijplijnen (FASTQ formaat). Het helpt bij het identificeren van problemen die kunnen voortkomen uit de sequencer zelf of de bibliotheekvoorbereiding.

### 4.2 Het FASTQ-formaat

#### 4.2.1 Wat is FASTQ?

FASTQ is een tekstbestandsformaat voor het opslaan van zowel nucleotidesequenties (reads) als hun corresponderende kwaliteitsscores. Het wordt veel gebruikt voor het opslaan van gegevens die afkomstig zijn van sequencing-apparaten.

#### 4.2.2 Voorbeeld van een FASTQ-entry

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
+
! '*((((***+))%%%++) (%%%) .1***-+* ' ) **55CCF>>>>>CCCCCCC65
```

### 4.2.3 Structuur van een FASTQ-bestand

Een FASTQ-bestand bestaat uit blokken van vier regels per sequentie:

1. Een regelbeginlijn die start met '@', gevolgd door een sequentie-identificer
2. De ruwe sequentiegegevens
3. Een regel die begint met '+', optioneel gevolgd door dezelfde sequentie-identificer
4. De *Phred* kwaliteitsscores voor de sequentie in gecodeerde vorm, één karakter per nucleotide.

Een kwaliteitsscore kan gezien worden als de kans dat de nucleotide op die positie foutief gecalled werd. Zo komt ( overeen met een kans van 20% op een foute call. Hoe zuiverder het signaal is de sequencer, hoe zekerder de call. Het staal, de staal- of bibliotheekvoorbereiding en de sequencing(reagentia) kunnen aanleiding geven tot minder kwalitatieve data. Die slechte kwaliteit zal in de verdere bio-informatica analyse resulteren in een slecht of onbetrouwbaar resultaat. Daarom is de eerste stap in een bio-informatica analyse van sequencing data een kwaliteitscontrole op basis van de *Phred* kwaliteitsscores is de FASTQ bestanden. Het FastQC programma berekend samenvattende statistieken over alle entries in een FASTQ bestand en geeft deze weer in een rapport.

## 4.3 Opdrachten

Voor volgende commando's uit in de VS Code terminal.

Laad de FastQC module

```
module load FastQC/0.11.8-Java-1.8.0_162
```

Voer FastQC uit op één FASTQ read-bestand

```
fastqc naam_van_fastq_bestand.fastq.gz
```

Pas de namen van het FASTQ bestand aan naar de namen van jouw bestanden in het bovenstaande commando.

Het FastQC rapport wordt gegenereerd, dit duurt enkele seconden. Hierna zie je het rapport (bv `naam_van_fastq_bestand_fastqc.html`) verschijnen in de linkerbalk. Om het rapport te openen moeten we het eerst downloaden (rechstklikken + download).

## 4.4 Het interpreteren van het FastQC HTML-rapport

1. **Basic Statistics:** Geeft een overzicht van het bestand, waaronder totaal aantal sequenties, sequentielengte en GC-gehalte.
2. **Per base sequence quality:** Toont hoe de kwaliteitsscores (Phred scores) verlopen over de lengte van de reads. De blauwe lijn geeft de gemiddelde kwaliteitsscore mee over alle reads. Normaal neemt de kwaliteitsscore af met de lengte van de reads. Bij een goede gelukte sequencing blijft het gemiddelde ook naar het einde van de reads toe voldoende hoog.
  - Groen gebied: Goede kwaliteit
  - Oranje gebied: Redelijke kwaliteit
  - Rood gebied: Slechte kwaliteit
3. **Per sequence quality scores:** Geeft de verdeling van kwaliteitsscores over alle sequenties. We verwachten een normale verdeling met een gemiddelde hoger dan 30.
4. **Per base sequentie-inhoud:** Toont de verhoudingen van basen op elke positie.
5. **Per sequentie GC-inhoud:** Vergelijkt de waargenomen GC-inhoudverdeling met een theoretische normale verdeling.
6. **Per base N-inhoud:** Toont het percentage van basen op elke positie die niet konden worden bepaald (N).
7. **Sequentielengteverspreiding:** Voor de meeste platformen zou dit een scherpe piek moeten zijn.
8. **Sequentieduplicatieniveaus:** Hoge duplicatieniveaus kunnen duiden op PCR-bias.
9. **Overgerepresenteerde sequenties:** Lijst van sequenties die vaker voorkomen dan verwacht.
10. **Adapter-inhoud:** Toont de aanwezigheid van vaak gebruikte adapters in je bibliotheek.

## 4.5 Opdrachtvragen:

1. Hoeveel sequenties/reads zijn er in beide FASTQ files aanwezig?
2. Wat is de gemiddelde kwaliteitsscore over alle basen en reads?
3. Hoe verandert de kwaliteitsscore over de lengte van de reads?

## 5 Read Mapping met BWA

### 5.1 Doel

In deze stap worden de sequenties in de FASTQ-bestanden uitgelijnd (gealigneerd) met een referentiegenoom. Het resultaat zal de meest waarschijnlijke oorsprong van de read zijn alsook op welke manier deze verschilt van het referentie genoom. Dit proces gebruikt algoritmen om te bepalen waar elke read het best past op het referentiegenoom. Het resultaat van deze alignment wordt opgeslagen in SAM- of BAM-formaat. BWA (Burrows-Wheeler Aligner) wordt gebruikt om de reads te aligneren tegen een referentiegenoom.

### 5.2 SAM-formaat (Sequence Alignment/Map)

#### 5.2.1 Wat is SAM?

SAM is een tekstbestandsformaat voor het opslaan van sequentie-alignments. Het wordt gebruikt om te beschrijven hoe sequenties zijn uitgelijnd ten opzichte van een referentiegenoom.

#### 5.2.2 Structuur van een SAM-bestand

Een SAM-bestand bestaat uit: - Een optionele headergedeelte - Alignment-gedeelte met één regel per alignment

#### 5.2.3 Voorbeeld van een SAM-entry

```
@HD VN:1.6 S0:coordinate
@SQ SN:ref LN:45
r001 99 ref 7 30 8M2I4M1D3M = 37 39 TTAGATAAAGGATACTG *
```

#### 5.2.4 Uitleg

- Header: @HD en @SQ lijnen geven informatie over het bestand en de referentiesequentie.
- Alignment:
  - r001: Identifier van de sequentie uit het FASTQ bestand
  - 99: Bitwise flag
  - ref: Reference sequence name
  - 7: 1-based leftmost mapping position
  - 30: Mapping quality
  - 8M2I4M1D3M: CIGAR string (beschrijft hoe de sequentie is uitgelijnd)

- =: Reference name of the mate/next read
- 37: Position of the mate/next read
- 39: Template length
- TTAGATAAAGGATACTG: sequentie
- \*: Quality scores (hier niet weergegeven)

## 5.3 Opdrachten

Laad de BWA module

```
module load BWA/0.7.17-GCC-10.3.0
```

Aligneer de reads met volgende commando's.

```
# We slaan eerst de bestandslocatie van het referentiegenoom op in een variabele REF
REF=/staging/leuven/stg_00156/references/hg38.fa

# We voeren nu BWA
bwa mem -R '@RG\tID:samplename\tSM:samplename' -t 1 $REF naam_van_fastq_bestand_r1.fastq.gz
```

Pas de namen van de FASTQ bestanden aan naar de namen van jouw bestanden in het bovenstaande commando.

## 5.4 BAM-formaat (Binary Alignment/Map)

### 5.4.1 Wat is BAM?

BAM is de binaire versie van het SAM-formaat. Het bevat dezelfde informatie als SAM, maar in een gecomprimeerde, binaire vorm. Bestanden zijn dus een stuk kleiner en dat spaart ruimte bij het archiveren en tijd bij het kopiëren van bestanden. Om BAM files efficient te kunnen verwerken worden de reads vaak gesorteerd volgens positie in het referentiegenoom en wordt er een index aangemaakt.

### 5.4.2 Kenmerken van BAM

- Neemt minder opslagruimte in beslag dan SAM
- Sneller te verwerken door computers
- Kan worden geïndexeerd voor snelle toegang tot specifieke regio's

### 5.4.3 Voorbeeld

Omdat BAM een binair formaat is, kunnen we geen leesbaar voorbeeld geven zoals bij SAM. In de praktijk zou je speciale software gebruiken om BAM-bestanden te bekijken of te bewerken.

## 5.5 Opdracht 2

Zet het SAM bestand om naar een gesorteerd BAM bestand.

```
module load SAMtools/1.13-GCC-10.3.0

samtools sort aligned.sam -o aligned.sorted.bam
samtools index aligned.sorted.bam
```

```
samtools flagstat aligned.sorted.bam
```

## 5.6 Opdrachtvragen:

1. Hoeveel reads zijn er gealigneerd tegen het referentiegenoom?
2. Hoeveel van deze reads vormden een correct paar (beide reads mappen met correct orientatie en afstand op hetzelfde chromosoom)?
3. Wat is het percentage van de totaal aantal reads die gealigneerd zijn en een correct paar vormen.

## 6 Variant Calling met Mutect2

Variant calling is het proces waarbij we verschillen (varianten) identificeren tussen een sequentiedataset en een referentiegenoom. Het is een cruciale stap in veel genomische analyses, van het bestuderen van genetische ziekten tot het begrijpen van evolutie. De meeste software tools voor Variant Calling vertrekken van uitgelijnde reads in het BAM formaat. Door op elke positie in het referentiegenoom de uitgelijnde reads te vergelijken zal de software varianten kunnen opsporen. Mutect2 is een veelgebruikte bio-informatica tool om somatische varianten te identificeren uit de uitgelijnde reads.

Wanneer er een verschil met het referentiegenoom gevonden wordt, wil dit nog niet noodzakelijk zeggen dat het om een werkelijke variant in het staal gaat. Sequencing fouten of problemen met de alignment kunnen ook zulke verschillen veroorzaken. De kwaliteitsscores in de reads vormen hier een belangrijk hulpmiddel voor de software. Twee belangrijke metrieken die gebruikt worden om “echte” varianten van vals positieve te onderscheiden zijn de *depth*, het aantal reads dat overlapt met de positie van de variant, en de *allele frequency*, het percentage

van de reads waar de variant werd teruggevonden. Afhankelijk van de gebruikte technologie wordt een minimale *depth* en *allele frequency* voorop gesteld. Als de variant onder deze waarden valt wordt deze als vals positief beschouwd.

## 6.1 VCF-formaat (Variant Call Format)

### 6.1.1 Wat is VCF?

VCF is een tekstbestandsformaat voor het opslaan van genoomvariaties zoals SNPs, inserties, deleties en structurele varianten. Alle genoomvariates worden weergegeven ten opzichte van het gebruikte referentiegenoom.

### 6.1.2 Structuur van een VCF-bestand

Een VCF-bestand bestaat uit: - Meta-informatieregels (beginnen met ##) - Een headerregel (begint met #) - Dataregels met informatie over elke variant

### 6.1.3 Voorbeeld van een VCF-entry

```
##fileformat=VCFv4.3
#CHROM POS ID REF ALT QUAL FILTER INFO
20 14370 rs6054257 G A 29 PASS NS=3;DP=14;AF=0.5;DB;H2
```

### 6.1.4 Uitleg

- CHROM: Chromosoom
- POS: Positie van de variant
- ID: Optionele variant identifier (bijv. rs-nummer van gekende varianten in de dbSNP databank)
- REF: Referentie-allel zoals het in het referentiegenoom aanwezig is
- ALT: Alternatief allel dat gevonden werd in het staal
- QUAL: Kwaliteitsscore
- FILTER: Filter status (PASS betekent dat de variant alle filters heeft gepasseerd)
- INFO: Aanvullende informatie over de variant zoals de *depth* (DP) en de *allele frequency* (AF)

## 6.2 Opdrachten

Voer variant calling uit met Mutect2 op de gesorteerde BAM file.

```
module load GATK/4.1.9.0-foss-2018a-Java-11.0.4

REF=/staging/leuven/stg_00156/references/hg38.fa
gatk Mutect2 \
  -R $REF \
  -I simulated_data_1.sorted.bam \
  -O simulated_data_1.vcf
```

Open de aangemaakte VCF file in VS Code en beantwoord volgende vragen. Bovenaan in het bestand staan er ongeveer 500 meta-informatieregels (beginnen met ##). Deze mag je negeren en verder scrollen naar de regel die begint met #CHROM. Hieronder staan alle varianten, één per regel.

## 7 Variant Annotatie met SnpEff

In het VCF bestand dat je bekomen bent bij de vorige stap zitten bijzonder veel varianten. De overgrote meerderheid heeft geen belangrijke impact omdat ze bijvoorbeeld in intergenische regio's of intronen liggen of synonieme mutaties zijn (geen verandering in aminozuursequentie). Het identificeren van één of enkele varianten die aan de basis van een ziektebeeld liggen (zoals kanker) is dus een zoektocht naar een speld in een hooiberg! Een belangrijk hulpmiddel in deze analyse is *variant annotatie* waarbij we voor elke variant de impact op het eiwit gaan bepalen. Een veelgebruikte tool hiervoor is snpEff. SnpEff maakt gebruik van een databank die gen en eiwitposities ten opzicht van het referentiegenoom bevat. Op die manier kan de impact van een variant op de eiwitsequentie bepaald worden.

```
module load snpEff/5.2c-GCCcore-10.3.0-Java-11

SNPEFF_REF=/staging/leuven/stg_00156/references/snpEff
snpEff -dataDir $SNPEFF_REF hg38 input_varianten.vcf > geannoteerde_varianten.vcf
```

Open de geannoteerde VCF in VS Code. Kijk naar het ANN= veld in de output VCF voor gedetailleerde annotaties. In dit veld staat de annotatie van deze variant als volgt:



## 7.1 Het SnpEff HTML-rapport begrijpen

SnpEff genereert naast geannoteerde varianten ook een uitgebreid HTML-rapport met statistieken en visualisaties (`snpEff_summary.html`). Download het bestand en open het in je browser. Laten we nu dit stap voor stap doornemen.

### 7.1.1 A. Algemene Statistieken Sectie

#### 1. Variants rate details:

- Aantal varianten per chromosoom
- Totaal aantal verwerkte varianten

#### 2. Number variants by type:

- SNP (Single Nucleotide Polymorphisms)
- MNP (Multiple Nucleotide Polymorphisms)
- INS (Inserties)
- DEL (Deleties)

#### 3. Number of effects by impact:

- HIGH: Grote impact op genfunctie (bijvoorbeeld stopcodons)
- MODERATE: Mogelijk effect op genfunctie (bijvoorbeeld missense varianten)
- LOW: Waarschijnlijk geen effect op genfunctie (bijvoorbeeld synonyme varianten)
- MODIFIER: Meestal in niet-coderende regio's

#### 4. Number of effects by functional class:

- Missense
- Nonsense
- Silent
- Splice site variants
- etc.

## 7.2 Praktische Oefeningen

### 7.2.1 Oefening 1: Basis Statistieken

Open het SnpEff HTML-rapport voor je geannoteerde varianten en beantwoord de volgende vragen:

1. Hoeveel varianten zijn er in totaal geanalyseerd?
2. Wat is de verdeling tussen SNPs, inserties en deleties?
3. Welk chromosoom heeft de meeste varianten?
4. Bereken het percentage varianten dat als 'HIGH impact' is geclassificeerd.

### 7.2.2 Oefening 2: Variant Effects Analyse

Bekijk de 'Number of effects by type and region' sectie:

1. Wat zijn de top 3 meest voorkomende effecten?
2. Hoeveel missense varianten zijn er gevonden?
3. Hoeveel splice site varianten zijn er?

### 7.2.3 Oefening 3: Genoomregio's

Bestudeer de 'Region counts' tabel:

1. Welk percentage van de varianten ligt in:
  - Exonen?
  - Intronen?
  - Intergene regio's?
2. Is deze verdeling wat je zou verwachten? Waarom wel/niet?

### 7.2.4 Oefening 4: Codon Veranderingen

Analyseer de 'Codon Changes' sectie:

1. Wat zijn de meest voorkomende codon veranderingen?
2. Hoeveel stop-gained mutaties zijn er?
3. Zijn er bepaalde aminozuur veranderingen die vaker voorkomen dan andere?

## 8 Variant filtering met SnpSift

Nadat we een geannoteerde VCF file hebben bekomen kunnen op basis van deze annotatie varianten filteren. Gezien de grote hoeveelheid varianten die typisch gevonden worden in een sequencing experiment is dit een cruciale stap.

Gebruik volgende commando om alle varianten met een *allele frequency* groter dan 50% met een hoge impact te filteren en in een nieuw bestand op te slaan.

```
SNPSIFT_JAR=/vsc-hard-mounts/leuven-apps/rocky8/skylake/2021a/software/snpEff/5.2c-GCCcore-10
java -jar $SNPSIFT_JAR filter "( GEN[*].AF > 0.50 ) & ( ANN[*].IMPACT = 'HIGH' )" $FILE_ID.a
```

Het ANN veld is moeilijk te interpreteren

```
java -jar $SNPSIFT_JAR extractFields -s "," -e "." $FILE_ID.annotated.filtered.vcf CHROM POS
```

## 8.1 Opdrachtvragen

1. Hoeveel varianten met HOGÉ impact heb je gevonden? Wat voor soort varianten zijn dit?
2. Vind een missense variant. Wat is de aminozuurverandering? In welk gen kwam het voor?
3. Zijn er varianten in bekende ziekte-geassocieerde genen?
4. Wat is het meest voorkomende type variant in je dataset?
5. Kun je varianten vinden die de eiwitfunctie kunnen beïnvloeden? Leg uit waarom je denkt dat ze impactvol kunnen zijn.