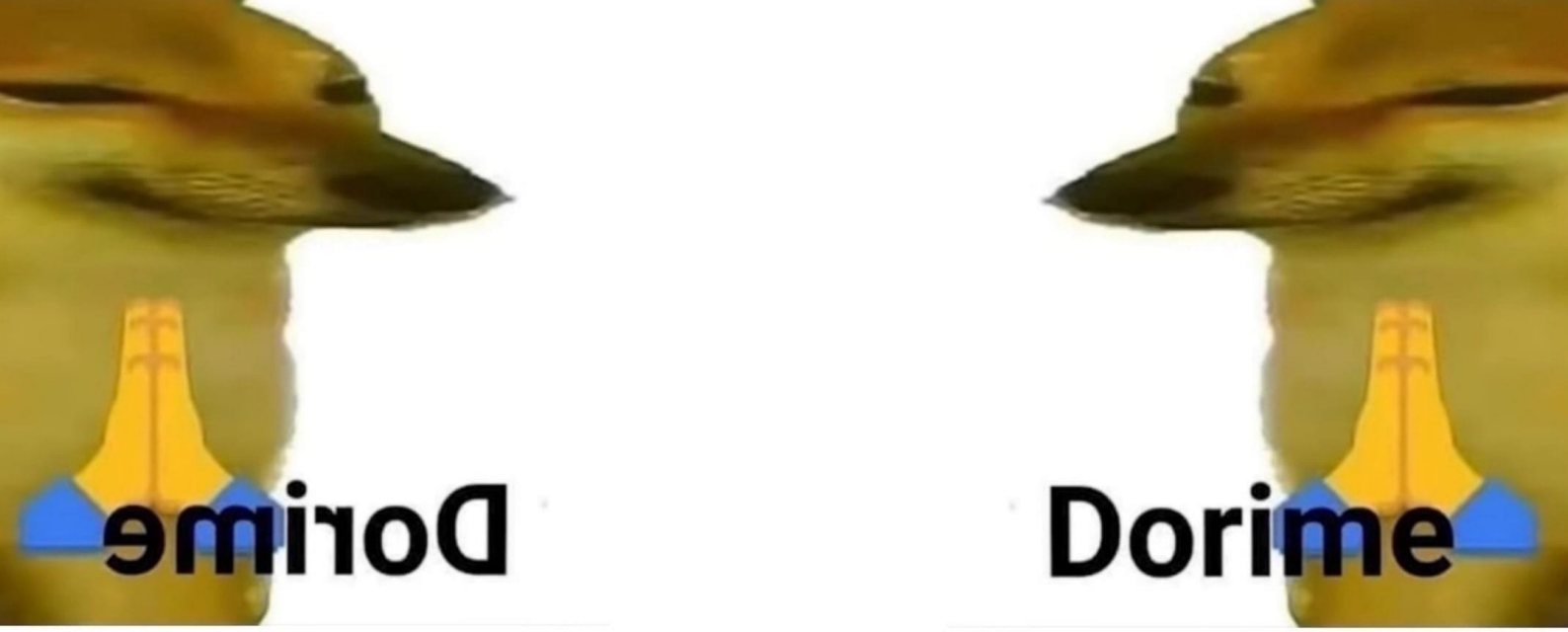


✨ ДОМАШНЕЕ ЗАДАНИЕ ✨
«АНАЛИЗ ГЕНОМОВ»

ОГЛАВЛЕНИЕ

ПРАКТИКУМ 11

Индексация референса с помощью hisat2

Для последующего картирования индексируем референсный геном, которым является 7 хромосома человека из референсной последовательности генома человека версии GRCh38.p14 (ensembl GCA_000001405.29).

```
hisat2-build chr7.fa chr7
```

Command: `hisat2-build` – строит индексы (в нашем случае используя 32-битные числа) референсного генома
Options: `chr7` – префикс имен выходных файлов
Input: `chr7.fa` – имя fasta-файла с референсным геномом
Output: `chr7.1.ht2, chr7.2.ht2, chr7.3.ht2, chr7.4.ht2, chr7.5.ht2, chr7.6.ht2, chr7.7.ht2, chr7.8.ht2`

Индексация референса с помощью samtools

Особая индексация референса необходима для некоторых программ.

```
samtools faidx chr7.fa
```

Command: `samtools`
Options: `faidx` – обеспечивает доступ к fasta (и fastq) файлам в input
Input: `chr7.fa` - имя fasta-файла с референсным геномом
Output: `chr7.fa.fai`

Рассмотрим поподробнее `chr7.fa.fai`:
В файле одна строка: 7 159345973 56 60 61
Попробую объяснить каждое число из файла!

Точное имя хромосомы: 7
Длина хромосомы в нуклеотидах: 159345973
56: индекс байта, с которого начинается сама нуклеотидная последовательность в fasta (данный в input) (то есть до самой последовательности находится 55 байтов, соответствующие, имени хромосомы и другим данным в первой строке после >)
60: длина строки fasta (данный в input) в нуклеотидах
61: длина строки fasta (данный в input) в байтах (загуглила, что перенос строки в Linux это 1 байт, поэтому разница 1 между длиной в нуклеотидах и байтах это символ переноса строки)

Описание образца чтений ДНК

Иду в NCBI в раздел SRA

SRR ID образца ДНК-чтений: SRR10720421
Ссылочка на информацию об образце в NCBI SRA (https://www.ncbi.nlm.nih.gov/sra/?term=SRR10720421)
Прибор для секвенирования: Illumina Genome Analyzer IIx
Организм: Homo sapiens
Стратегия секвенирования: whole-exome sequencing (экзомное)
Какие чтения: парноконцевые
Сколько чтений ожидается (spots): 31,417,056

Проверка качества исходных чтений

У меня есть 2 файла (так как парноконцевые чтения) - SRR10720421_1.fastq.gz (прямые) и SRR10720421_2.fastq.gz (обратные).

Запускаю fastqc, которые я положила в ~/public_html/term3, чтобы они висели на сайтике и я на зачете их могла просто открыть и показать.

```
fastqc SRR10720421_1.fastq.gz SRR10720421_2.fastq.gz
```

Получилось 2 html-файла, для прямых чтений (_1) и для обратных (_2).

Количество пар чтений: 31,417,056

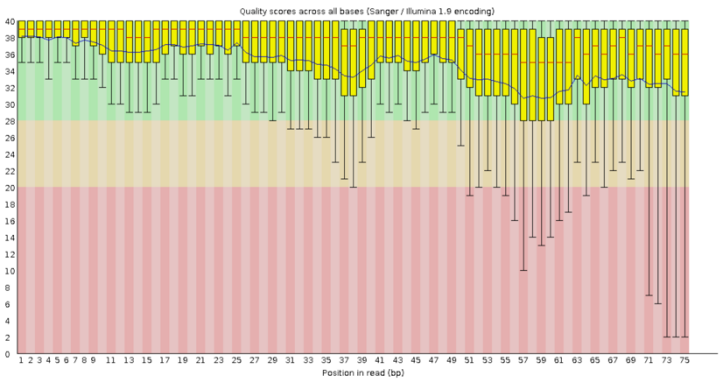
Совпадает ли количество чтений у прямых и обратных: да, совпадает также с ожидаемым количеством

Анализ качества пар чтений: если ориентироваться на медианы (красные линии) и среднее значение (синяя линия), то качество чтений хорошее (все выше 30), при этом нельзя сказать, что оно равномерно уменьшается к концу чтений.

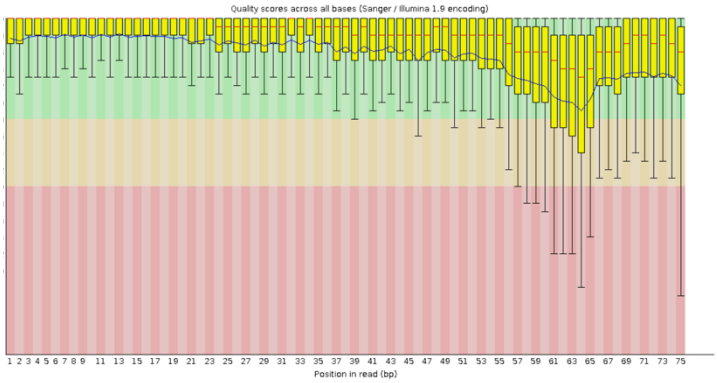
Координаты выделяющихся ухудшенным качеством участки, которые находятся НЕ в конце чтений: прямые чтения - 50-61, обратные чтения - 56-65. Несколько последних нуклеотидов обоих типов чтений с не самым волшебным качеством, но это резонно и ожидаемо.

При этом, если посмотреть на интервал квартилей уровня 25%-75%, то некоторые из них находятся на желтой границе (20-28), но так как их мало, думаю, это нормально.

Но вот если посмотреть на "усы" 10%-90%, то качество кажется сомнительным (достаточно много с качеством меньше 20). Но, наверное, стоит ориентироваться на медиану / среднее значение, тогда все нежно).



Прямые чтения. Участок 50-61 выбивается из общего уровня качества



Обратные чтения. Участок 56-65 выбивается из общего уровня качества

Рис. 1. Per base sequence quality

Анализ длины чтений: вставила только 1 картинку, потому что они одинаковые для обоих типов чтений. Длина всех чтений 75 нуклеотидов.

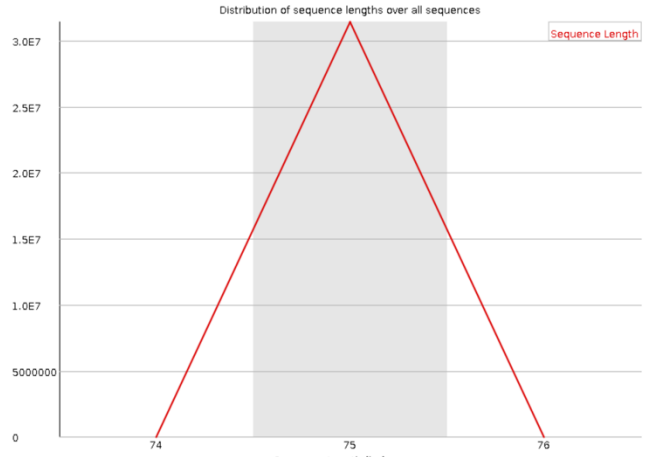
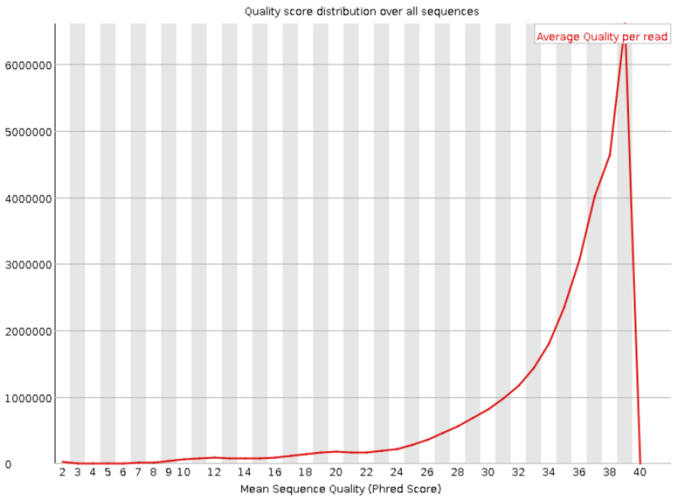


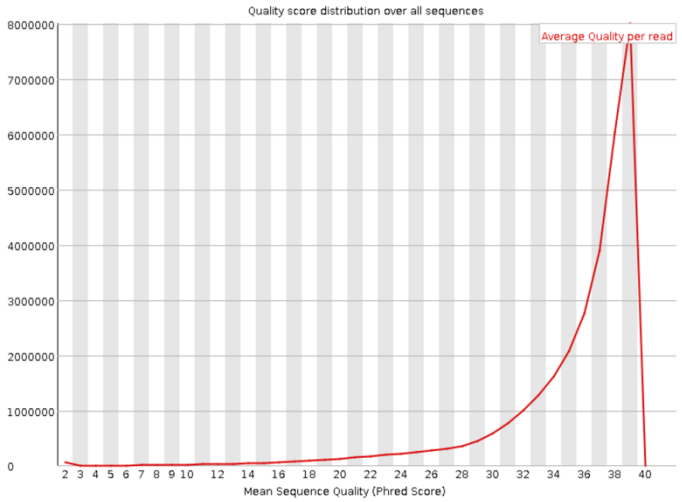
Рис. 2. Sequence length distribution

Пару слов про остальные графики:

Если продолжить тему качества чтений, то можно взглянуть на распределение чтений по среднему качеству. Конечно, большая часть с качеством больше 30, но доля не очень качественных все-таки достаточно весомая (ну, это было понятно и по графику Рис. 1 «Per base sequence quality»).



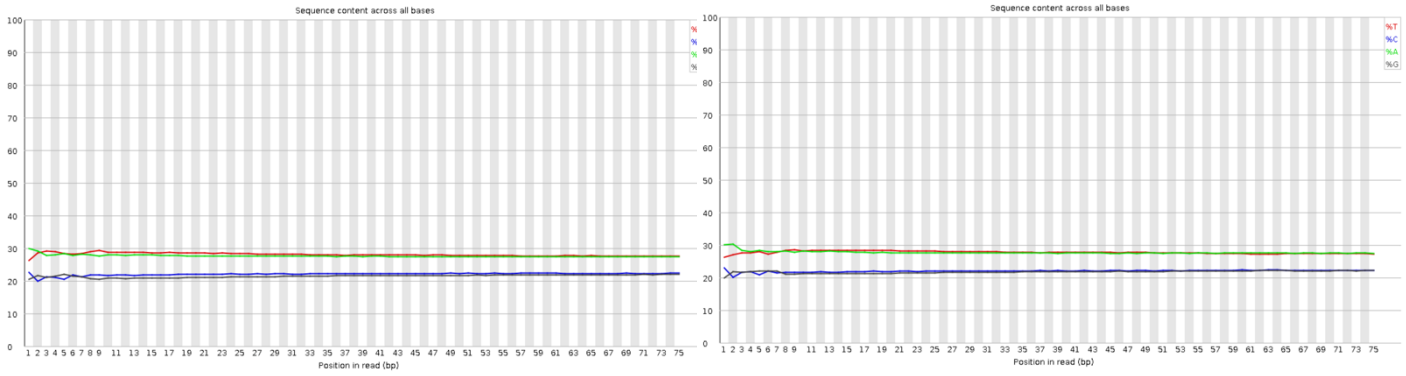
Прямые чтения



Обратные чтения

Рис. 3. Per sequence quality scores

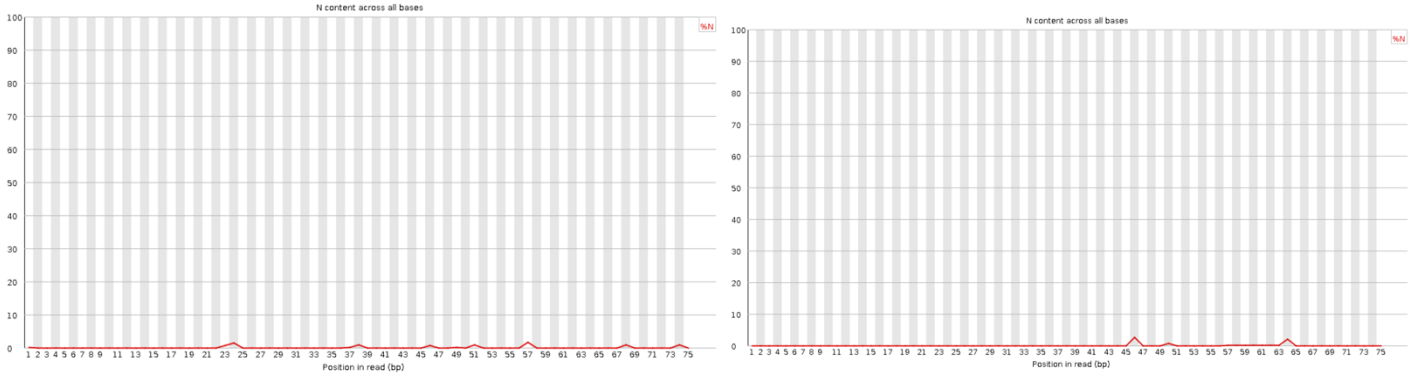
Процентное количество вида нуклеотидов в данной позиции. Считается, что, если в какой-то позиции разница |A-T| или |G-C| больше 10%, то это warning!! (думаем, что в процессе секвенирования что-то пошло не так, и получившиеся чтения лучше переделать). Но на моих графиках максимальная |delta| около 5%, так что все хорошо.



Прямые чтения Обратные чтения

Рис. 4. Per base sequence content

Процент знака N (неопределенный нуклеотид) в позиции. Насторожить должно значение выше 5%, на моих графиках максимум около 3%, то есть нормес.



Прямые чтения Обратные чтения

Рис. 5. Per base N content

Для оценки использовала мануал fastqc (https://www.bioinformatics.babraham.ac.uk/projects/fastqc/)

Общая оценка качества чтений: приемлемо (мем)

Фильтрация чтений с помощью trimmomatic

```
TrimmomaticPE -phred33 SRR10720421_1.fastq.gz SRR10720421_2.fastq.gz
trim_1_paired.fastq.gz trim_1_unpaired.fastq.gz trim_2_paired.fastq.gz
trim_2_unpaired.fastq.gz TRAILING:20 MINLEN:50
```

Command: `TrimmomaticPE` - запускаем trimmomatic для парноконцевых чтений (поэтому PE, SE - для одноконцевых)
Options: `-phred33` - данный Quality Score

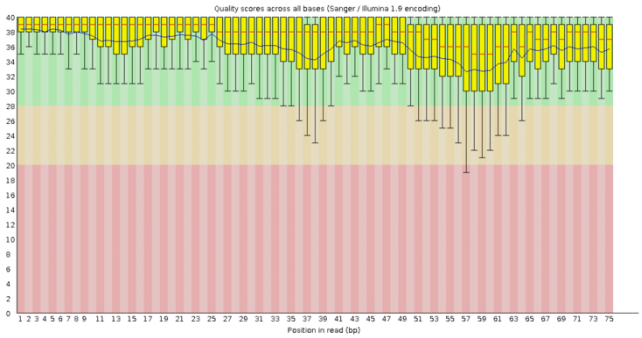
`TRAILING:20` - удаляем с конца чтений нуклеотиды с качеством ниже 20
`MINLEN:50` - удаляем чтения с длиной меньше 50
Input: `SRR10720421_1.fastq.gz SRR10720421_2.fastq.gz` - входные файлы с чтениями
Output: `trim_1_paired.fastq.gz trim_1_unpaired.fastq.gz trim_2_paired.fastq.gz trim_2_unpaired.fastq.gz` - выходные файлы, _paired - парные (оба чтения "выжили" после триммирования), _unpaired - непарные ("выжило" только одно чтение, а партнер не перенес триммирования...).

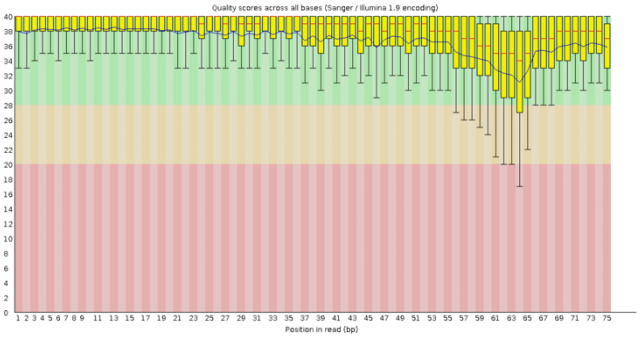
Анализ триммированных чтений fastqc

```
fastqc trim*
```

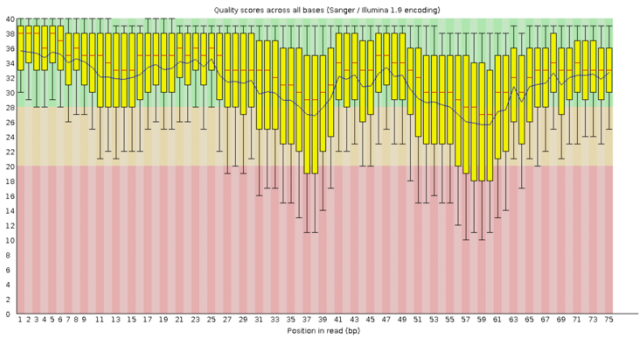
Количество пар чтений осталось: 29626256
Процент пар чтений осталось: 92.30%
Сравнение качества чтений после триммирования (paired vs unpaired): качество непарных чтений заметно хуже, чем качество парных

Парные чтения

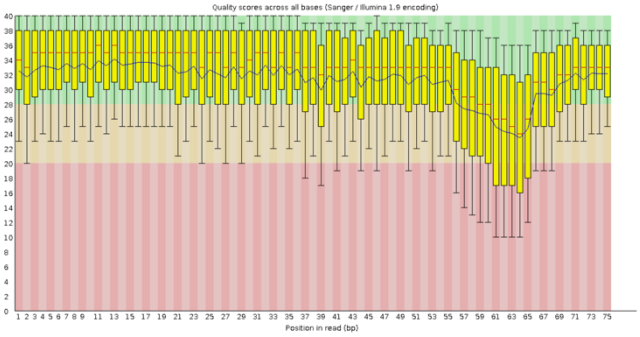




Прямые чтения



Обратные чтения

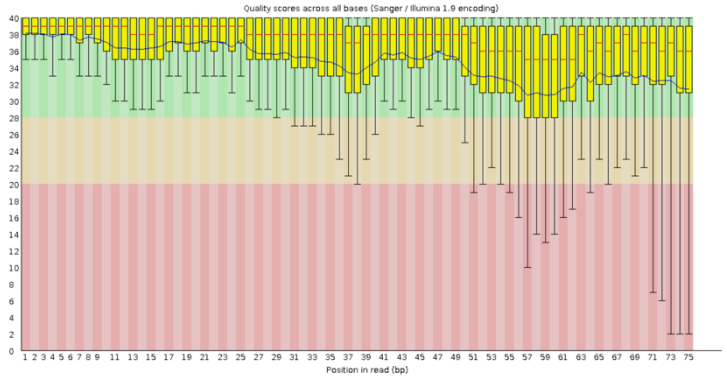


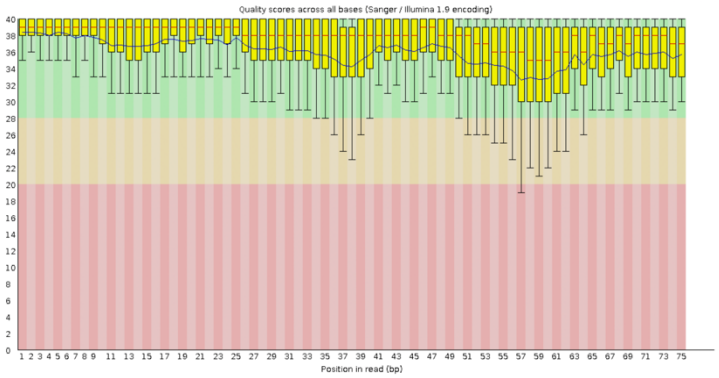
Непарные чтения

Рис. 6. Per base sequence quality после триммирования

Сравнение качества чтений до и после триммирования (только paired): после триммирования качество чтений улучшилось

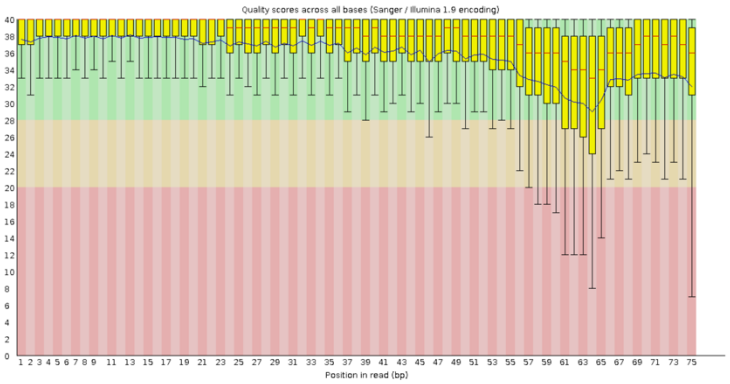
Прямые чтения

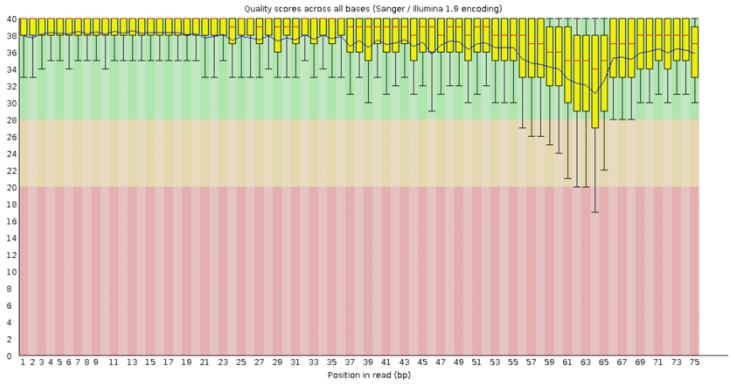




До триммирования

После триммирования



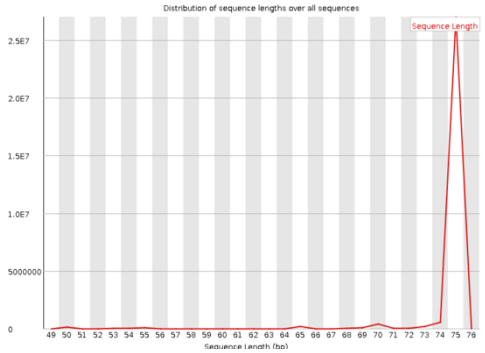


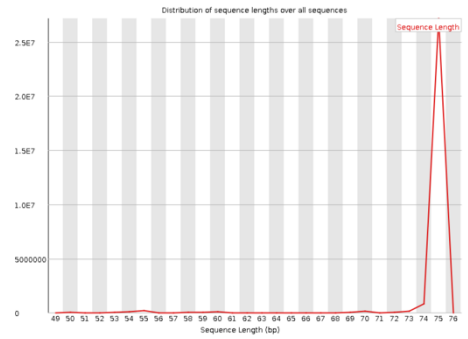
Обратные чтения

Рис. 7. Сравнение Per base sequence quality до и после триммирования

Изменение длины чтений после триммирования: Большая часть так и осталась 75 нуклеотидов, но: в парных чтениях появилось небольшое количество длиной меньше (74, еще меньше длиной 70); а вот у непарных длина совсем изменилась, так как появились значительной величины пики на графиках (подписала на графиках отдельно).

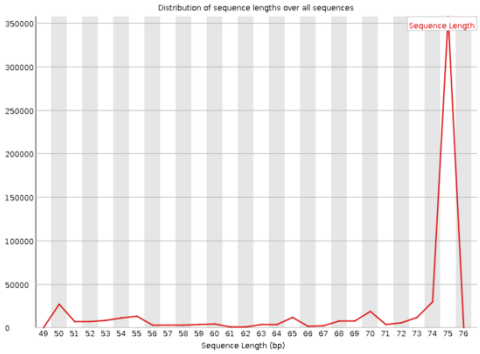
Парные чтения

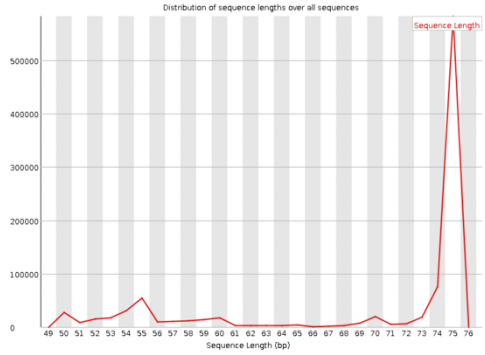




Прямые чтения

Обратные чтения





Непарные чтения

Рис. 8. Сравнение Sequence length distribution парных и непарных чтений

Сводный отчет о качестве чтений

Чтобы исключить ошибки при просмотре глазами, воспользуюсь multiqc на все файлы.

```
multiqc .
```

Command: `multiqc` – сводынй отчет о работе fastqc на нескольких файлах
Options/Input: `.` – все файлы из текущей папки с расширением fastqc.qz (прямые и обратные чтения, после триммирования: парные прямые и обратные, непарные прямые и обратные чтения)
Output: `multiqc_report.html` – сводный файл html (еще, кстати, делает отдельную папку с выходными файлами разных расширений (json, txt…))

Посмотрим на `multiqc_report.html`!!!

Начнем с картинки, которая заменяет 4 страницы текста… Зеленый цвет ячейки – результаты нормальные, желтый – на результаты стоит обратить внимание, красный – необычные результаты (warning!!)

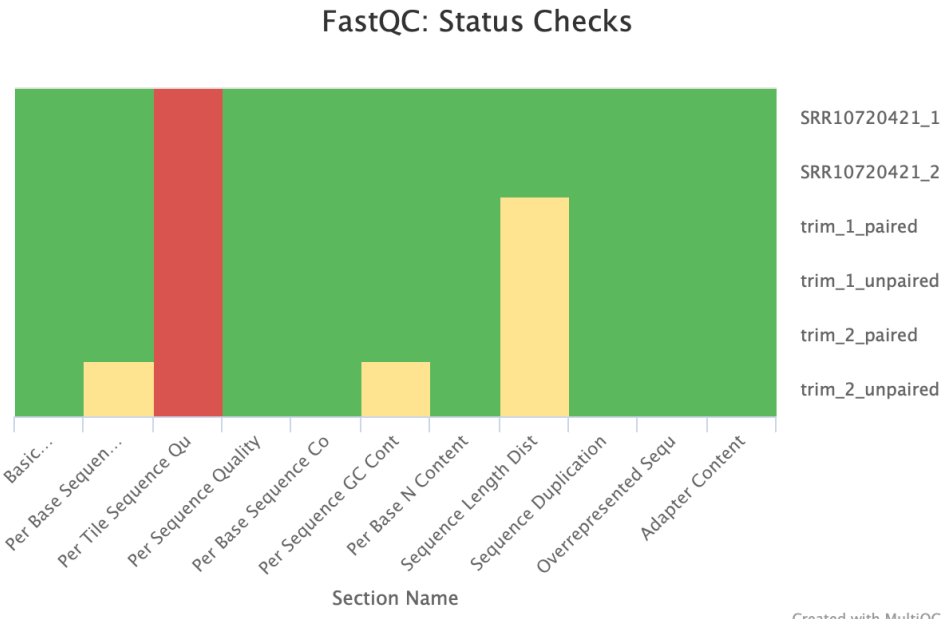


Рис. 9. Сводная статистка Status Checks по всем файлам и всем критериям

Красная колонка – per tile sequence quality – вообще не нашла такого раздела нигде… Не поняла, что это, если честно, но нигде это не затрагивала (а вообще, если смотреть в manual, то кажется, что она почти всегда красно-желтая).

В целом все файлы с чтениями хорошего качества, наиболее выделяется trim_2_unpaired, а из критериев самым подозрительным кажется sequence length distribution (в триммировании убирали чтения длиной меньше 50 и вырезали с конца нуклеотиды с качеством меньше 20, поэтому получился разброс, который и кажется multiqc странным).
Ну и немного сводных графиков с коротким комментарием, потому что все уже было расписано.

Количество чтений: прямых и обратных соответствует ожидаемому количеству чтений (100%), парных прямых и обратных после триммирования стало меньше (92.30%), очень мало (в отношении) непарных после триммирования.

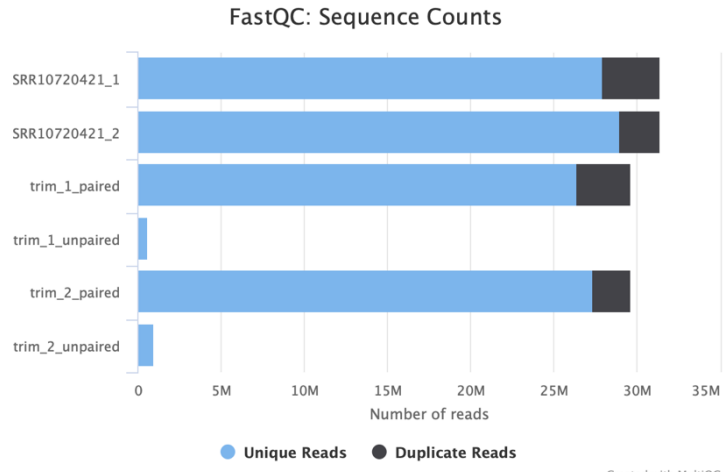


Рис. 10. Распределение количества чтений

Качество чтений по позициям нуклеотидов: все, кроме trim_2_unpaired, обратных триммирвоанных непарных чтений, хорошего качества («passed»), а вот сам trim_2_unpaired вызывает у multiqc warning!! – качество не очень хорошее (по графику распределения для него одного видно, что целый блок позиций в красной зоне со значением 17).

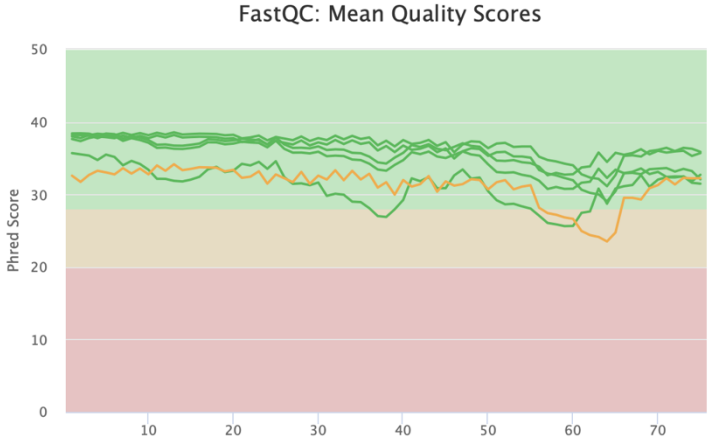


Рис. 11. Распределение качества чтений по позициям

Очень прикольным мне показалось распределение N по позициям, и вот почему: в позиции 46 у всех типов чтений находится пик. Не могу обосновать, с чем это связано, но занятно, что он очень выделяется у всех типов чтений.

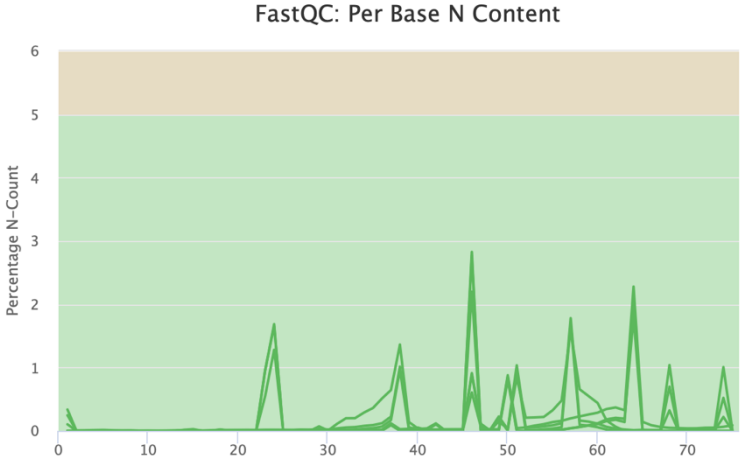


Рис. 12. Распределение N (неопределенного нуклеотида) по позициям

И последний график, на который я обращу внимание, это распределение последовательностей по (среднему) качеству. На графике видно только 3 пика на 6 файлов с чтениями: самый высокий пик – обратные чтения и тиммированные парные обратные чтения; средний пик (чуть ниже самого высокого) – прямые чтения и триммированные парные прямые чтения; а вот нижняя прямая – триммированные непарные прямые и обратные чтения.

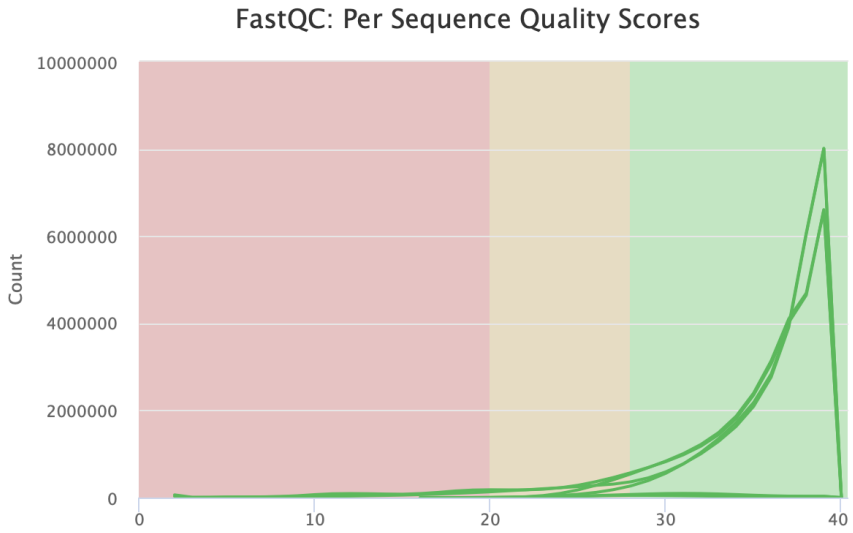


Рис. 13. Распределение качества последовательностей

ПРАКТИКУМ 12

Картирование чтений на референсный геном

Триммированные чтения лежат у меня просто в папке lizzzafomenko, а все про референс – в папке /lizzzafomenko/reference. Я сделала новую папку mapping, где буду делать само картирование чтений на референс (попыталась объяснить пути к файлам, которые написала в программе).

```
hisat2 -x ../reference/chr7 -1 ../trim_1_paired.fastq.gz -2
../trim_2_paired.fastq.gz -p 10 --no-spliced-alignment > map.sam 2>
map_log.txt
```

Command: hisat2 – картирование чтений
Options: -x ../reference/chr7 – префикс имен файлов с индексацией референса (которые были получены после индексации референса в hisat2-build, их было 8)
-1 trim_1_paired.fastq.gz - файл с прямыми парными триммированными чтениями
-2 trim_2_paired.fastq.gz - файла с обратными парными триммированными чтениями
-p 10 – использую 10 ядер процессора, чтобы быстро посчиталось
--no-spliced-alignment - параметр, запрещающий возможность сплайсинга (то есть запрещает картирование с разрывами)
Input: ../reference/chr7 – файлы с индексацией референса
../trim_1_paired.fastq.gz, ../trim_2_paired.fastq.gz - парные триммированные чтения
Output: > map.sam - записываю вывод программы в файл .sam
2> map_log.txt – сохраняю логи в txt-файл

Конвертация sam в bam

a) Вес sam файла – 12 Гб

Конвертируем этот тяжеленный файл в его бинарный аналог – map.bam

```
samtools sort -o map.bam map.sam
```

Command: samtools sort – сортирует файл .sam, данный в input
Options: -o map.bam – вывод программы в файл map.bam
Input: map.sam – sam файл, полученный при картировании чтений
Output: map.bam – bam файл

b) Вес bam файла – 3,5 Гб

Индексация bam файла с помощью samtools index:

```
samtools index map.bam
```

Command: samtools index – программа, которая индексирует файл
Input: map.bam – исходный файл
Output: map.bam.pai – выходной файл

Расскажу немного про файл sam, пока он не был навсегда стерт из моей папки…

Рис. 1. Файл .sam

(мне жаль, что картинка так плохо выглядит, но по-другому в одну строчку ничего не вмещалось)

Шапка:
@SQ SN – имя референса (7), LN – длина референса (159345973)
Еще в шапке можно найти информация о программе, которая сгенерировала этот sam файл (ID, имя и версия программы, копия полной команды для выравнивания)

Содержание по столбцам:
QNAME – имя чтения
FLAG – информация о чтении (говорит нам как картировано чтение, каждой характеристике есть свой набор цифр. Нужна, когда мы переходим к файлу bam и фильтруем чтения)
RNAME – имя референсного контига (в моем случае chr7, так как она везде одинаковая, то стоит *)
POS – Позиция первого нуклеотида, куда выровнялось чтение на референс
MAPQ – Качество картирования (Q = -10lgP)
CIGAR – сколько мэтчей, количество делеций и вставок в референсе (в формате CIGAR: xMyDzI)
RNEXT – Название последовательности для парного (в случае PE) или следующего (в случае SE) чтения
PNEXT – POS для первого выравнивания парного (в случае PE) или следующего (в случае SE) чтения
TLEN – длина участка референса, к которому прикрепилось чтение (может не совпадать с длиной чтения, так как есть делеции и вставки в референсе относительно чтения)
SEQ – последовательность чтения (как в fastq)
QUAL – качество чтения для каждой позиции нуклеотида (как в fastq)
Дополнительная информация (опционально; здесь находится tag: видов тэгов много, каждому соответствует какое-то значение, например MQ – качество картирования парного/следующего чтения, AS – вес выравнивания, подсчитанный программой выравнивания и тд)

файла map.sam больше нет с нами, но вся информация осталась в его бинарном аналоге map.bam, который сейчас рассмотрим!

Анализ bam файла

Так как файл бинарный, просто посмотреть его (с помощью less, например) не получится. Поэтому запускаю программу для анализа этого файла:

```
samtools flagstat map.bam > analysed_bam.txt
```

Command: `samtools index` – программа

Input: `map.bam` – исходный файл
Output: `analysed_bam.txt` – выходной файл

```
60271913 + 0 in total (QC-passed reads + QC-failed reads)
59252512 + 0 primary
1019401 + 0 secondary
0 + 0 supplementary
0 + 0 duplicates
0 + 0 primary duplicates
4845063 + 0 mapped (8.04% : N/A)
3825662 + 0 primary mapped (6.46% : N/A)
59252512 + 0 paired in sequencing
29626256 + 0 read1
29626256 + 0 read2
3352536 + 0 properly paired (5.66% : N/A)
3428486 + 0 with itself and mate mapped
397176 + 0 singletons (0.67% : N/A)
0 + 0 with mate mapped to a different chr
0 + 0 with mate mapped to a different chr (mapQ>=5)
```

Рис. 2. Анализ содержимого файла .bam

Все чтения прошли Quality Control!
a) **Сколько чтений картировано на референс**? 4845063 штуки
b) **Сколько чтений картировано на референс в % от количества триммированных?** 8.04%
c) **Сколько чтений картировано на референс в корректных парах?** 3352536
d) **Сколько чтений картировано на референс в корректных парах в % от количества триммированных?** 5.66%

Попробую объяснить такие на первый взгляд маленькие проценты. Так как у нас изначально чтения всего экзома, а референс у меня только 7ая хромосома, то очевидно, что не все чтения на нее картируются (увидела то же самое объяснение в следующем задании, эх(( ). А вот на вопрос «почему не все парные чтения корректно картировались» ответов может быть много, например: парные чтения могли картироваться не по направлению друг к другу; чтение могло картироваться на референс несколько раз (и получится, что расстояние между парными чтениями будет большим); а может быть одно из чтений вообще не картировалось.

Получение чтений, картированных на chr7

```
samtools view -h -bS map.bam 7 > chr7_map.bam
```

Command: `samtools view` – печатает все чтения из input картированные на референс
Options: `-h` – выводить в файл вместе с заголовком
`-b` – вывод в файл формата bam
`-S` – формат файла в input определить автоматически
Input: `map.bam` – исходный файл
7 – имя моей хромосомы (было получено в faidx)
Output: `chr7_map.bam` – файл bam с чтениями, картированными на 7 хромосому

Получение только правильно картированных пар чтений

Чтобы получить только правильно картированные чтения есть программа:

```
samtools view -f 0x2 -bS chr7_map.bam > true_pairs_chr7_map.bam
```

Command: `samtools view` – печатает все чтения из input картированные на референс
Options: `-f 0x2` – печатает в output только те чтения, которые прошли по критерию FLAG

со значением 0x2: это значение соответствует PROPER_PAIR, то есть выведутся только чтения, которые точно выравнены с референсом (нашла объяснение значение FLAG по ссылке http://www.htslib.org/doc/samtools-flags.html)
`-b` – вывод в файл формата bam
`-S` – формат файла в input определить автоматически
Input: `chr7_map.bam` – исходный файл
Output: `true_pairs_chr7_map.bam` – файл формата bam, в котором будут только правильно парно картированные чтения.

Посмотрим на этот файл!

```
samtools flagstat true_pairs_chr7_map.bam > true_pairs.txt
```

```
4126878 + 0 in total (QC-passed reads + QC-failed reads)
3352536 + 0 primary
774342 + 0 secondary
0 + 0 supplementary
0 + 0 duplicates
0 + 0 primary duplicates
4126878 + 0 mapped (100.00% : N/A)
3352536 + 0 primary mapped (100.00% : N/A)
3352536 + 0 paired in sequencing
1676268 + 0 read1
1676268 + 0 read2
3352536 + 0 properly paired (100.00% : N/A)
3352536 + 0 with itself and mate mapped
0 + 0 singletons (0.00% : N/A)
0 + 0 with mate mapped to a different chr
0 + 0 with mate mapped to a different chr (mapQ>=5)
```

Рис. 3. Анализ содержимого .bam файла с правильными парными чтениями

a) **Сколько чтений картировано на референс в корректных парах?** 3352536
b) **Сколько чтений картировано на референс в корректных парах в % от общего количества картированных чтений?** 100.00%

Теперь проиндексирую файл `true_pairs_chr7_map.bam`

```
samtools index true_pairs_chr7_map.bam
```

Output: `true_pairs_chr7_map.bam.bai`

ПРАКТИКУМ 13

Получение вариантов

Создаю новую папку variants (`/mnt/scratch/NGS/lizzzafomenko/variants`), в которой будут задания этого практикума.

```
bcftools mpileup -f ../reference/chr7.fa ../mapping/true_pairs_chr7_map.bam | 
bcftools call -mv -o variants.vcf
```

Command: `bcftools mpileup` – генерирует vcf файл, в котором находятся вероятности разных вариантов (на основании выравнивания)
Options: `-f ../reference/chr7.fa` – указывают референс
Input: `../reference/chr7.fa` – референс
`../mapping/true_pairs_chr7_map.bam` – файл bam с картированными ридами
Command: `bcftools call` – из output (stdout) программы bcftools mpileup берет только нужные строки (характеристики указаны в options)
Options: `-m` – модель, которая ищет мультиаллельные и редкие варианты
`-v` – на выдачу попадут только варианты
`-o variants.vcf` – выдача в файл variants.vcf
Input: output из `bcftools mpileup`
Output: `variants.vcf`

Посмотрим на variants.vcf

```
##fileformat=VCFv4.2
##FILTER=<ID=PASS,Description="All filters passed">
##bcftoolsVersion=1.11+htslib-1.11-4
##bcftoolsCommand=mpileup -f ../reference/chr7.fa ../mapping/true_pairs_chr7_map.bam
##reference=file://../reference/chr7.fa
##contig=<ID=7,length=159345973>
##ALT=<ID=*,Description="Represents allele(s) other than observed.">
##INFO=<ID=INDEL,Number=0,Type=Flag,Description="Indicates that the variant is an INDEL.">
##INFO=<ID=IDV,Number=1,Type=Integer,Description="Maximum number of raw reads supporting an indel">
##INFO=<ID=IMF,Number=1,Type=Float,Description="Maximum fraction of raw reads supporting an indel">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Raw read depth">
##INFO=<ID=VDB,Number=1,Type=Float,Description="Variant Distance Bias for filtering splice-site artefacts in RNA-seq data (bigger is better)",Version="3">
##INFO=<ID=RPB,Number=1,Type=Float,Description="Mann-Whitney U test of Read Position Bias (bigger is better)">
##INFO=<ID=MQB,Number=1,Type=Float,Description="Mann-Whitney U test of Mapping Quality Bias (bigger is better)">
##INFO=<ID=BQB,Number=1,Type=Float,Description="Mann-Whitney U test of Base Quality Bias (bigger is better)">
##INFO=<ID=MQSB,Number=1,Type=Float,Description="Mann-Whitney U test of Mapping Quality vs Strand Bias (bigger is better)">
##INFO=<ID=SGB,Number=1,Type=Float,Description="Segregation based metric.">
##INFO=<ID=MQ0F,Number=1,Type=Float,Description="Fraction of MQ0 reads (smaller is better)">
##FORMAT=<ID=PL,Number=G,Type=Integer,Description="List of Phred-scaled genotype likelihoods">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##INFO=<ID=ICB,Number=1,Type=Float,Description="Inbreeding Coefficient Binomial test (bigger is better)">
##INFO=<ID=HOB,Number=1,Type=Float,Description="Bias in the number of HOMs number (smaller is better)">
##INFO=<ID=AC,Number=A,Type=Integer,Description="Allele count in genotypes for each ALT allele, in the same order as listed">
##INFO=<ID=AN,Number=1,Type=Integer,Description="Total number of alleles in called genotypes">
##INFO=<ID=DP4,Number=4,Type=Integer,Description="Number of high-quality ref-forward , ref-reverse, alt-forward and alt-reverse bases">
##INFO=<ID=MQ,Number=1,Type=Integer,Description="Average mapping quality">
##bcftools_callVersion=1.11+htslib-1.11-4
##bcftools_callCommand=call -mv -o variants.vcf; Date=Fri Dec  8 19:45:55 2023
#CHROM  POS     ID      REF     ALT     QUAL    FILTER  INFO    FORMAT  ../mapping/true_pairs_chr7_map.bam
7       10523   .       G       A       3.77163 .       DP=2;SGB=-0.379885;RPB=1;MQB=1;MQSB=1;BQB=1;MQ0F=0;ICB=1;HOB=0.5;AC=1;AN=2;DP4=0,1,1,0;MQ=60    GT:PL   0/1:34,0,34
7       11001   .       G       A       3.73859 .       DP=2;SGB=-0.379885;RPB=1;MQB=1;BQB=1;MQ0F=0;ICB=1;HOB=0.5;AC=1;AN=2;DP4=0,1,0,1;MQ=60   GT:PL   0/1:34,0,23
7       11003   .       G       A       10.7923 .       DP=1;SGB=-0.379885;MQ0F=0;AC=2;AN=2;DP4=0,0,0,1;MQ=60   GT:PL   1/1:40,3,0
7       11153   .       A       G       37.4152 .       DP=2;VDB=0.02;SGB=-0.453602;MQ0F=0;AC=2;AN=2;DP4=0,0,2,0;MQ=60  GT:PL   1/1:67,6,0
7       11660   .       C       T       8.99921 .       DP=1;SGB=-0.379885;MQ0F=0;AC=2;AN=2;DP4=0,0,1,0;MQ=60   GT:PL   1/1:38,3,0
7       11671   .       G       A       7.30814 .       DP=1;SGB=-0.379885;MQ0F=0;AC=2;AN=2;DP4=0,0,1,0;MQ=60   GT:PL   1/1:36,3,0
7       11843   .       C       A       3.77034 .       DP=2;SGB=-0.379885;RPB=1;MQB=1;BQB=1;MQ0F=0;ICB=1;HOB=0.5;AC=1;AN=2;DP4=0,1,0,1;MQ=60   GT:PL   0/1:34,0,31
7       12302   .       A       G       36.4154 .       DP=2;VDB=0.52;SGB=-0.453602;MQ0F=0;AC=2;AN=2;DP4=0,0,0,2;MQ=60  GT:PL   1/1:66,6,0
7       12362   .       T       G       10.7923 .       DP=1;SGB=-0.379885;MQ0F=0;AC=2;AN=2;DP4=0,0,0,1;MQ=60   GT:PL   1/1:40,3,0
```

Рис. 4. Файл .vcf

Что можно сказать про файл .vcf?

Сначала идет шапка файла, каждая строка начинается с ##.
Затем перед «телом файла» идет строка с заголовками столбцов, она начинается с #.

А теперь поподробнее о каждом столбце: что можно найти?
CHROM – имя хромосомы
POS – позиция варианта
ID – везде стоят «.» , но может быть любая информация о варианте
ALT – альтернативный аллель (так как у нас SNP, то здесь стоит одна бква)
QUAL – качество варианта
FILTER – везде «.» , так как ввели файл, который уже был маркирован по качеству

INFO – характеристики варианта
FORMAT – список параметров варианта (для конкретного образца)

А теперь проанализируем `variants.vcf`

```
bcftools stats variants.vcf > var_stats.txt
```

```
# SN    [2]id   [3]key  [4]value
SN      0       number of samples:      1
SN      0       number of records:      68580
SN      0       number of no-ALTs:      0
SN      0       number of SNPs: 66697
SN      0       number of MNPs: 0
SN      0       number of indels:       1883
SN      0       number of others:       0
SN      0       number of multiallelic sites:   31
SN      0       number of multiallelic SNP sites:       31
```

Рис. 5. Анализ вариантов из .vcf

a) **Сколько получилось вариантов?** 68580
b) **Сколько из них являются SNP?** 66697
c) **Сколько получилось коротких вставок и делеций?** 1883 (индели)

Фильтрация вариантов

```
bcftools filter -i'%QUAL>30 && DP>50' variants.vcf -o filt_variants.vcf
```

Command: `bcftools filter` – программа, которая отфильтрует варианты из input файла по заданным параметрам
Options: `-i'%QUAL>30 && DP>50'` – фильтруем по параметрам: качество больше 30 и длина больше 50
`-o filt_variants.vcf` – вывод программы в указанный файл
Input: `variants.vcf`
Output: `filt_variants.vcf`

И анализируем полученный файл!

```
bcftools stats filt_variants.vcf > filt_var_stats.txt
```

```
# SN    [2]id   [3]key  [4]value
SN      0       number of samples:      1
SN      0       number of records:      1776
SN      0       number of no-ALTs:      0
SN      0       number of SNPs: 1726
SN      0       number of MNPs: 0
SN      0       number of indels:       50
SN      0       number of others:       0
SN      0       number of multiallelic sites:   2
SN      0       number of multiallelic SNP sites:       2
```

Рис. 6. Анализ отфильтрованных вариантов из .vcf

a) **Сколько осталось вариантов после фильтрации?** 1776 штук, 2.59%
b) **Сколько осталось SNP?** 1276 штук, 1.91%
c) **Сколько осталось коротких вставок и делеций?** 50 штук, 2.66%

Аннотация вариантов

Category	Count
Variants processed	1776
Variants filtered out	0
Novel / existing variants	447 (25.2) / 1329 (74.8)
Overlapped genes	675
Overlapped transcripts	3164
Overlapped regulatory features	148

Всего в файле было 1776 вариантов
Из них отфильтровано 0, то есть все было проанntировано
Новые варианты – 447 (сведения о них нет ни в каком формате в датабазах, например, в clinvar), а существующих и уже аннотированных где-то вариантов – 1329
Перекрывающихся генов – 675
Перекрывающихся транскриптов – 3164
Перекрывающихся регуляторных областей- 148

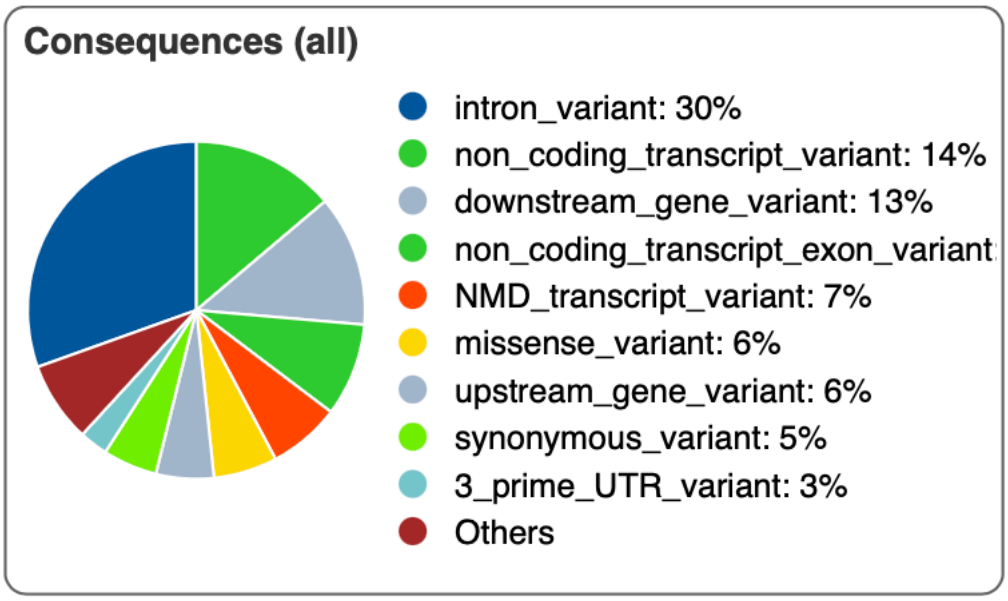


Рис. 7. Распределение эффектов (consequences) мутаций

Переписывать эффекты по-русски я смысла не вижу (только расшифрую NMD - Nonsense-mediated mRNA decay), поэтому дам обзорную характеристику:

82% мутаций (все, кроме тех, которые перечислены ниже) имеют влияние на транскрипт **«MODIFIER»,** то есть либо не влияют вообще, либо затрагивают некодирующие гены/области. Это и логично, ведь если бы было много мутаций, которые влияют на транскрипты, то количество болезней было бы огромным :(

5% мутаций относятся к synonymous_variant, то есть синонимичным заменам, и имеют влияние **«LOW»**, то есть в большинстве своем не меняют функцию/поведение белка (что и понятно по определению синонимичной мутации, хотя есть случаи, когда они влияют на структуру: например, если замена синонимичная, но при этом заменить на более редкий кодон, тогда скорость трансляции уменьшается, нарушается фолдинг (который идет во время трансляции), вследствие чего структура и функции белка могут нарушиться! Статья, красиво подкрепляющая растекание мысли по древу: doi: 10.3390/genes13081485)

6% мутаций относятся к missense_mutations и имеют влияние **«MODERATE»**. Белок транслируется, но неправильно, вследствие чего меняется его структура, а дальше либо теряется функция, либо меняется, либо… (можно очень много сказать!!!)

А теперь отдельно посмотрим на кодирующие области, ведь интересующую нас патогенность или изменения вызывают, в большинстве своем, именно они.

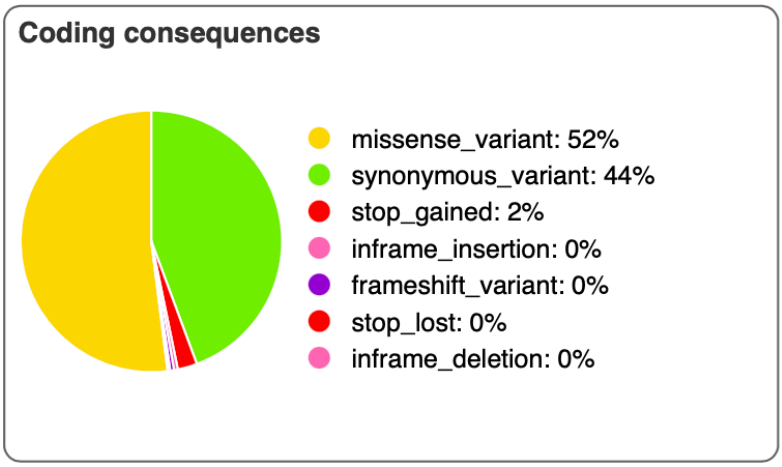


Рис. 8. Распределение эффектов (consequences) мутаций кодирующих областей

Очевидно, что большинство из них будут повторять мутации кодирующих областей из предыдущего графика. Описывать их по новой не хочу)

Опасно выглядит 2% **stop_gained**, то есть преждевременный стоп-кодон. Это приводит к укороченным транскриптам, которые могут терять/частично терять функцию. Такие мутации очень могут быть патогенными.
Остальные варианты встречаются 0% (но это, видимо, просто округление? Иначе зачем вообще вносить их в диаграмму), но в целом они с уровнями влияния на транскрипт «MODERATE» и «HIGH».

Отдельно поговорим про варианты с **HIGH IMPACT**, потому что именно они вызовут вопросы, если мы будем искать патогенные мутации.
Выпишу типы мутаций из выдачи, которые характеризуются HIGH влиянием.

Количество записей в выдаче на каждый тип мутации с HIGH IMPACT (x + y значит, что у мутации 2 эффекта)

Подавляющее большинство мутаций с HIGH IMPACT ожидается получить в экзонах (хотя, при альтернативном сплайсинге интрон одного транскрипта может быть экзоном другого… и много разного можно придумать, но логично ожидать в экзонах!). И точно все они должны попасть в ген.

Stop_gained – 41 – в экзоне
Frameshift_variant – 7 – в экзоне
Stop_lost – 2 – в последнем экзоне
Frameshift_variant + **stop_lost** – 2 – в последнем экзоне
Stop_gained + **NMD_transcript_variant** – 5 – в экзоне
Splice_acceptor_variant – 8 – мутация 3’ конца экзона (акцепторного сайта)
Splice_donor_variant – 12 – мутация 5’ конца экзона (донорного сайта)
Splice_donor_variant + **non_coding_transcript_variant** – 3
Splice_donor_variant + **NMD_transcript_variant** - 2

#скрипт на практикумы 12-13 есть у меня на кодомо и текстом прописан здесь

ПРАКТИКУМ 14

Описание образца

a) **ID образца РНК-чтений:** ENCFF975AUW
b) **Ссылка на информацию об образце:**
https://www.encodeproject.org/search/?type=File&searchTerm=ENCFF975AUW
c) **Организм и ткань:** сердце из эмбриона человека (мужского пола, 120 дней)
d) **Стратегия секвенирования:** polyA plus RNA-seq (полиА РНК)
e) **Тип чтений:** SE (одноконцевые)
f) **Цепь-специфичность:** нет

Проверка качества исходных чтений

```
fastqc ENCFF975AUW.fastq.gz
```

На выходе получила файл `ENCFF975AUW.fastq.html` – путь к нему `/mnt/scratch/NGS/lizzzafomenko/ENCFF975AUW_fastqc.html`

a) **Количество чтений:** 87265266
b) **Качество чтений**
А что тут сказате... Среднее значение, медиана, интервалы между верхними и нижними квартилями убедительные, потому что большей частью находятся в зеленой зоне с качеством больше 30 (пониженное качество наблюдается на первых нуклеотидах и на последних, где среднее значение съезжает чуть ниже 30. Усы, конечно, растянуты прям до 2 (!!!), но чтобы оценить их реальную значимость лучше гляну на график Per sequence quality score (в пункте d)

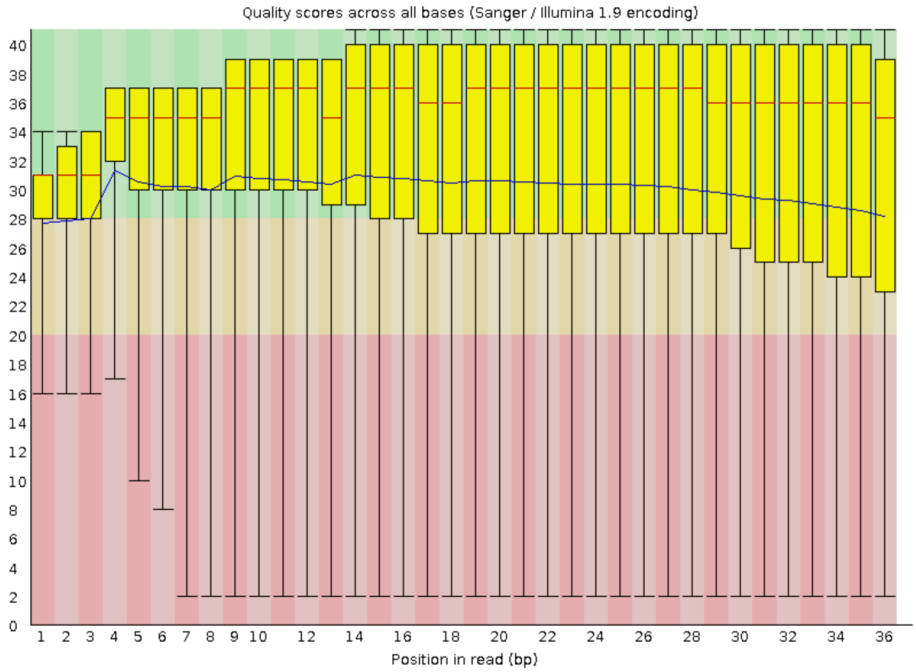


Рис. 1. Per base sequence quality

c) **Длина чтений**

Все чтения длины 36

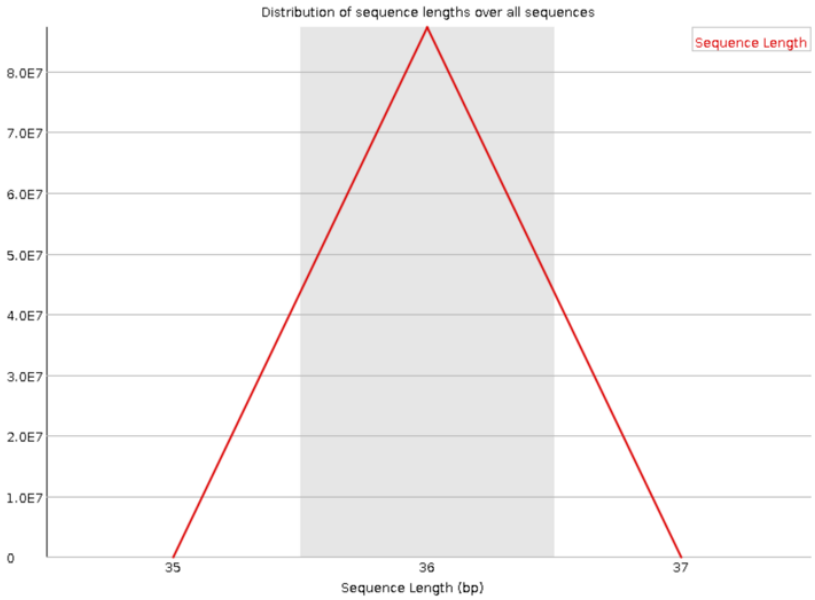


Рис. 2. Sequence Length Distribution

d) **Еще** посмотрим на Per sequence quality scores и Per base sequence content «ойойой с айайай». Мне не нравится, что много чтений с крайне низким качеством (Q=2). В штуках таких чтений около 7 миллионов, а это около 0.8% от всех чтений (что, кажется, не очень много, но все равно довольно существенно). Так как чтения не триммируем, то, возможно, такое качестве еще даст о себе знать при картировании.

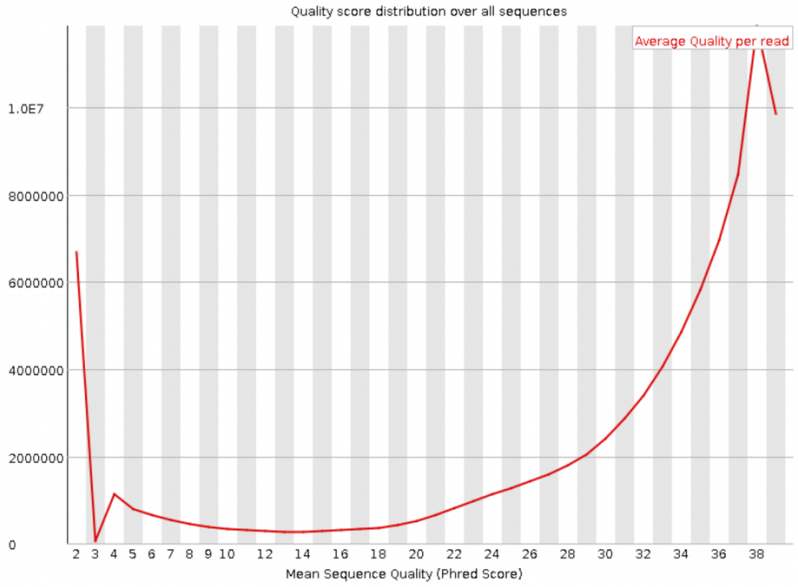


Рис. 3. Per sequence quality scores

Есть еще один график, который может мне подсказать, насколько можно доверять качеству чтений. Это Per base sequence content. Как я смотрела при проверке ДНК-чтений, на этом графике: «Считается, что, если в какой-то позиции разница |A-T| или |G-C| больше 10%, то это warning!! (думаем, что в процессе секвенирования что-то пошло не так, и получившиеся чтения лучше переделать)».
Первые 10 позиций происходит какая-то вакханалия)). В позициях 2, 4-6, 9, 10, а особенно в 7, |A-T|>10%. В позициях 1 и 5 |G-C|>10%. То есть, на самом деле, чтения на треть длины с омрачняющим качеством. Но «вы работаете с реальными данными, которые могут быть не идеальны, что-то может пойти не так, но в этом нет ничего ужасного», поэтому просто буду держать в голове этот факт и делать дальше.

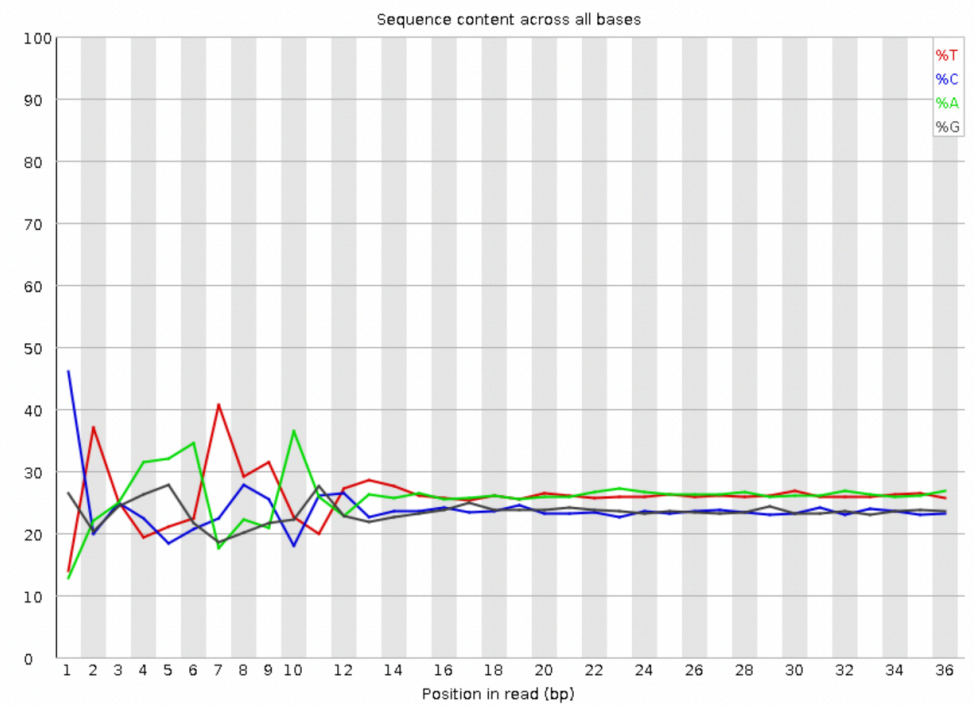


Рис. 4. Per base sequence content

Картирование чтений на референс

Сделаю папку rna_map, в которой будут все файлы про картирование РНК-чтений на референс

```
hisat2 -x ../reference/chr7 -k 3 -U ../ENCFF975AUW.fastq.gz >
rna_map.sam 2> rna_map_log.txt
```

Command: `hisat2` – картирование чтений
Options: `-x ../reference/chr7` – префикс имен файлов с индексацией референса (которые были получены после индексации референса в hisat2-build, их было 8) `-k 3` - максимально возможное количество выравниваний (3), чей score больше или равен score любого другого выравнивания (но hisat2 не даёт гарантий, что это лучшие выравнивания)
`-U ../ENCFF975AUW.fastq.gz` – файл с чтениями
Input: `../reference/chr7` – файлы с индексацией референса
`../ENCFF975AUW.fastq.gz` – РНК-чтения
Output: `rna_map.sam` – записываю вывод программы в файл .sam
`rna_map_log.txt` – сохраняю логи в txt-файл

Заглянем в rna_map_log.txt

```
87265266 reads; of these:
  87265266 (100.00%) were unpaired; of these:
    81261318 (93.12%) aligned 0 times
    5157760 (5.91%) aligned exactly 1 time
    846188 (0.97%) aligned >1 times
6.88% overall alignment rate
```

Сколько чтений картировалось на хромосому? Всего картировалось 6003948 чтений (6.88%). Кстати, я думала, что число будет меньше (учитывая, что качество не лучшее, а еще у нас весь транскриптом и только 1 хромосома).

Перевод в bam файл: `samtools sort -o rna_map.bam rna_map.sam`
Индексация bam файла: `samtools index rna_map.bam`
Отбор только тех чтений, которые картировались на хромосому:
`samtools view -h -bS rna_map.bam 7 > chr7_rna_map.bam`

Поиск экспрессирующихся генов

Скачала файл с геномной разметкой Homo_sapiens.GRCh38.110.chr.gtf
Посмотрим, что у него внутри

Сначала идет шапка:
#!genome-build GRCh38.p14 - версия разметки
#!genome-version GRCh38 - версия генома, на которой строили
#!genome-date 2013-12 - дата публикации
#!genome-build-accession GCA_000001405.29- AC разметки
#!genebuild-last-updated 2023-03 - последняя дата обновления

После шапки идет тело файла. В каждой строке содержится информация о разметке, разделенная на 9 столбцов.

seqname - название последовательности, где аннотирован ген (в нашем случае имя хромосомы)
source - источник аннотации (просмотрела файл, чаще всего встречается ensembl и Havana)
feature - особенности гена
start - начало гена
end - конец гена
score – какое-то значение, вместо которого во всем файле стоит «.» (в мануале A floating point value, не особо понимаю, что это конкретно значит, но вероятно что-то о достоверности/качестве)
strand - цепь, на которой этот ген находится
frame – (сдвиг) рамка считывания
attribute - дополнительная информация

Сколько аннотировано генов на 7 хромосоме?

```
grep '^7' *gtf | cut -f3 | grep 'gene' | wc -l
```

Получаем ответ! 3147

(не буду врать, загуглила, сколько генов на 7 хромосоме. И почему-то по разным источникам их 1400-1800, а у меня получилось в 2 раза больше! Вроде по файлу нашла все логично: беру строки с 7 хромосомой, в третьем столбце (feature) считаю только те, которые описаны как ‘gene’…)

Теперь посчитаем для каждого гена из разметки число картированных на этот ген чтений.

```
htseq-count -f bam -s no -t exon -m union -o count_exons.sam
chr7_rna_map.bam ../Homo_sapiens.GRCh38.110.chr.gtf 1> count_log1.txt 2>
count_log2.txt
```

Command: `htseq-count` – картирование чтений
Options: `-f bam` – входной файл в формате bam
`-s no` – у чтений не указана цепь, поэтому они могут попадать и на прямую, и на обратную цепи
`-m union` – если чтения перекрываются, то их нужно объединить
`-t exon` – особенность гена (из 3 столбца), считаются только чтения, которые картировались на гены с этой особенностью. Так как по дефолту имеет значение exon, его и поставлю.
`-o count_exons.sam` – выходной файл
Input: `chr7_rna_map.bam` – чтения, картированные на 7 хромосому
`../Homo_sapiens.GRCh38.110.chr.gtf` – генетическая разметка
Output: `count_exons.sam` – sam файл с аннотированными выравниваниями
`count_log1.txt` (из stdout) – сводка о работе программы
`count_log2.txt` (из stderr) – файл с ошибками

Заглянем в получившиеся файлы

В файле count_log1.txt находится информация про гены.

```
__no_feature    1180579
__ambiguous     175058
__too_low_aQual 0
__not_aligned   0
__alignment_not_unique  846188
```

Сколько чтений попало в границы генов с (-t exon)? Посчитаем (не люблю баш, напишу на питоне)

```
counts = 0
with open('/Users/macbook/Desktop/count_log1.txt', mode='r') as file:
    for line in file:
        if line[0:2] == '__':
            print(counts)
            break
        counts += int(line.split('\t')[1])
```

Ответ! 3802123

Сколько чтений попало мимо границ экзонов (-t exon)? __no_feature 1180579

Попробую объяснить все строки с помощью
https://htseq.readthedocs.io/en/master/htseqcount.html

__no_feature – чтения, которые попали вне экзонов
__ambigous – 175058 чтений ассоциированы с более чем одной особенностью гена (-t, feature)
__too_low_aQual – чтения, которые были бы пропущены, если бы программе дали опцию -a (пропустить чтения с весом выравнивания ниже заданного)
__not_aligned – чтения в файле без выравнивания (таких нет, так на вход получен файл с чтениями, про которые точно известно картирование на 7 хромосому)
__alignment_not_unique – количество чтений, у которых больше одного выравнивания

СКРИПТ ДЛЯ ПРАКТИКУМОВ 12-13

В скрипте после каждой команды после # написан краткий комментарий о том, что эта команда делает. На кодомо скрипт без комментариев доступен:
/mnt/scratch/NGS/lizzzafomenko/script/

script.sh

```
#!/bin/bash


# Usage script.sh config_file ID N

##############################################
#  Soft
##############################################
#  FastQC v0.11.9
#  hisat2 version 2.2.1
#  TrimmomaticPE 0.39
#  multiqc version 1.15
#  samtools 1.17 (using htslib 1.17)
#  bcftools 1.11 (using htslib 1.11-4)
##############################################


config_file=$1
ID=$2
N=$3

. $config_file

hisat2-build $ref.fa $ref   #индексация референса hisat-2, ref - chrN

samtools faidx $ref.fa #индексация референса с помощью samtools

fastqc $forward_reads $reverse_reads #проверяем качества исходных чтений, для
прямых и обратных чтений, надо иметь в виду, что выходные файлы

TrimmomaticPE -$phred $forward_reads $reverse_reads $for_paired $for_unpaired
$rev_paired $rev_unpaired TRAILING:$trim_trailing MINLEN:$trim_minlen
#фльтруем=триммируем чтения, удаляя с конца чтений с качеством ниже
trim_trailing и удаляем чтения с длиной меньше trim_minlen

fastqc $for_paired $for_unpaired $rev_paired $rev_unpaired #анализ
триммированных чтений

multiqc . #общий сравнительный анализ всех чтений до и после триммирования

hisat2 -x $ref -1 $for_paired -2 $rev_paired  -p $nthreads --no-spliced-
alignment > $map_sam 2> $map_logs #картирование чтений на референс (N
хромосома), количество ядер процессора nthreads, без учета сплайсинга
(запрещаем большие разрывы внутри чтения)

samtools sort -o $map_bam $map_sam #конвертация sam в bam

#rm $sam_bam удаляем файл sam

samtools index $map_bam  #индексируем файл bam

samtools flagstat $map_bam > $an_bam  #смотрим на бинарный файл bam
```

```
samtools view -h -bS $map_bam $N > $true_map #получаем только те чтения, которые картировались на референс

samtools view -f 0x2 -bS $true_map > $true_paired #выдает только правильно картированные парные чтения

samtools flagstat $true_paired > $an_true #смотрим на этот файл

samtools index $true_paired #индексируем этот файл

bcftools mpileup -f $ref.fa $true_paired | bcftools call -mv -o $rough_vcf #ищем варианты

bcftools stats $rough_vcf > $rough_stat  #анализ

bcftools filter -i"%QUAL>${filt_qual} && DP>${filt_length}" $rough_vcf -o $filt_vcf
#фильтрация вариантов по критерию качество больше заданного в filt_qual и длинной больше filt_length. Двойные кавычки стоят потому, что в одинарных кавычках переменная читается как строка, а не как переменная

bcftools stats $filt_vcf > $filt_stat #анализ файла
```

config_file

```
ref=chr$N #префикс файла с хромосомой

forward_reads=${ID}_1.fastq.gz #прямые чтения

reverse_reads=${ID}_2.fastq.gz #обратные чтения

phred=phred33 #quality score для триммирования

trim_trailing=20 #обрезаем с конца качество меньше 20 при триммир

trim_minlen=50 #миним длина чтений после триммир (обрез конйа)

for_paired=trim_1_paired.fastq.gz #триммир прям пар

for_unpaired=trim_1_unpaired.fastq.gz #триммир прям непарные

rev_paired=trim_2_paired.fastq.gz #триммир обр парные

rev_unpaired=trim_2_unpaired.fastq.gz #триммир обрат непарные

nthreads=10 #количество ядер процессора

map_sam=map.sam #sam файл с выводом картирования

map_logs=map_logs.txt #файл с выводом stderr

map_bam=map.bam #файл конвертации sam в bam

an_bam=analysed_bam.txt #анализ bam файл

true_map=$ref_map.bam #картированные на рефер
```

```
true_paired=true_pairs_$ref_map.bam #правильно парно картир на рефер
an_true=true_pairs.txt #анализ предыд
rough_vcf=variants.vcf #все варинанты
rough_stat=var_stats.txt #анализ всех вариантов
filt_qual=30 #фильтр качества вариантов
filt_length=50 #фильтр длины вариантов
filt_vcf=filt_variants.vcf #фильтрованные варианты
filt_stat=filt_var_stats.txt #анализ фильтр вариантов
```