UNIVERSITA' DEGLI STUDI DI VERONA

DEPARTMENT OF

Biotechnology

GRADUATE SCHOOL OF

Natural Sciences and Engineering

DOCTORAL PROGRAM IN

Biotechnology

XXXIV cycle

TITLE OF THE DOCTORAL THESIS

Precise variant calling in the clinical settings

S.S.D. BIO/18

Coordinator:	Prof. Matteo Ballottari
Tutor:	Prof. Massimo Delledonne

Doctoral Student: Denise Lavezzari

Precise variant calling in the clinical settings - Denise Lavezzari
Tesi di Dottorato
Verona, 10 Dicembre 2021

A mia nonna Giuseppina,

SOMMARIO

L'identificazione di varianti di alta qualità nell'analisi di sequenziamento dell'intero esoma può essere molto complessa a causa delle diverse modificazioni che possono essere fatte al protocollo di preparazione per il sequenziamento del campione, le quali possono influire in modo negativo sull'analisi bioinformatica, in particolare nell'identificazione delle varianti. La valutazione e la correlazione dei parametri di qualità di ogni fase dell'analisi potrebbero aiutare ad ottenere una migliore accuratezza e precisione nell'identificazione delle varianti. Inoltre, dopo aver identificato varianti ad alta qualità, l'uso di banche dati di riferimento in cui è possibile consultare il significato clinico e la frequenza delle varianti consente una diagnosi più accurata. Durante l'analisi di laboratorio e bioinformatica, è possibile calcolare molte metriche per valutare la qualità dei dati in elaborazione. Tutti questi dati vengono solitamente esaminati separatamente e la loro cronologia viene persa nel tempo. Inoltre, il processo di confronto di un nuovo protocollo di analisi del campione con quelli esistenti può essere molto dispendioso se eseguito manualmente. In aggiunta, per una diagnosi significativa di varianti rare, è importante considerare la variante nella popolazione di appartenenza del campione. Per questo motivo, è necessario un database che incorpori tutte le metriche di qualità dell'intera analisi whole exome sequencing (WES) nel tempo e raccolga varianti specifiche della popolazione per un'identificazione precisa del significato clinico delle varianti dell'individuo.

Questa tesi mira ad ottimizzare la valutazione delle metriche di qualità e la classificazione accurata delle varianti nella popolazione italiana attraverso la creazione di un database SQL direttamente collegato ad un sito web per un utilizzo più intuitivo. La tesi espone la struttura del database e la configurazione della pagina web creati. Inoltre, durante il lavoro di tesi, sono stati analizzati circa 2.500 esomi e sono stati raccolti tutti i parametri di controllo qualità derivati dalle analisi di laboratorio e bioinformatiche. Tutti i dati ottenuti sono stati caricati nel database per verificare l'utilità dell'applicazione nel controllare l'andamento della qualità dei dati nel tempo e nell'identificazione di possibili problemi. Vengono presentati due esempi di problemi identificati grazie all'applicazione implementata e successivamente risolti tramite delle modifiche al protocollo di laboratorio. Inoltre, viene mostrata la potenzialità del database nel semplificare i confronti tra protocolli di laboratorio in uso e nuovi tramite la memorizzazione dei parametri di controllo della qualità. In aggiunta, tutte le varianti

identificate nei campioni analizzati sono state caricate per creare un database di varianti della popolazione italiana. La corretta classificazione delle varianti italiane è mostrata in relazione ai database conosciuti che riportano solo una visione più ampia della popolazione europea. Questa nuova classificazione permette di classificare varianti mancanti nei database usati quotidianamente (gnomAD Exomes, ExAC, 1KgPhase3). Inoltre, permette di identificare varianti rare solitamente classificate come comuni che potrebbero presentarsi patogeniche nella popolazione italiana, ma anche varianti comuni nella popolazione italiana e quindi non dannose, che sono classificate come rare nella popolazione europea.

In conclusione, i risultati e gli esempi riportati hanno mostrato come l'applicazione creata (database esteso con il suo sito web) semplifichi e faciliti l'identificazione dei problemi nell'analisi WES clinica e il confronto tra i vari protocolli di laboratorio permettendo così una miglior precisione nell'analisi dell'esoma finalizzata all'identificazione delle varianti. Infine, l'indagine specifica delle varianti italiane potrebbe essere utile nelle applicazioni diagnostiche.

ABSTRACT

Identifying high quality variants in whole exome sequencing (WES) analysis can be very complex due to the different modifications that can be made in the sample sequencing preparation protocol. This can adversely affect bioinformatics analysis in the identification of variants. The evaluation and correlation of the quality parameters of each analysis stage could help to obtain a better accuracy and precision in the identification of the variants. Furthermore, after identifying high quality variants, the use of reference databases where the clinical significance and frequency of the variants can be consulted, allows for a more accurate diagnosis. During laboratory and bioinformatics analysis, it is possible to calculate many metrics to evaluate the quality of the data being processed. All this data is usually looked at separately and their history is lost over time. Besides, the process of comparing a new workflow to existing ones can be very time-consuming when done manually. In addition, for a significant diagnosis of rare variants, it is important to consider the variant frequency in the sample population. For this reason, a database that incorporates all quality metrics from the entire WES analysis over time and collects population-specific variants for accurate clinical variant identification, is needed.

This thesis aims to optimise the evaluation of quality metrics and the classification of variants in the Italian population through the creation of a Structured Query Language (SQL) database directly linked to a website for more intuitive use. The thesis sets out the structure of the database and the configuration of the web page created. Furthermore, during the writing of the thesis, approximately 2,500 exomes were analysed and all quality control parameters derived from both laboratory and bioinformatics analyses were collected. All the data obtained were uploaded to the database in order to verify the usefulness of the application in monitoring data quality trends over time and in identifying possible problems. Two examples of problems identified by the implemented application and subsequently solved by modifications to the laboratory protocol are presented. Moreover, the potential of the database to simplify comparisons between existing and new laboratory protocols storing quality control parameters, is shown. All variants identified in the analysed samples were uploaded to create an accessible reference of genetic variation in Italians. The correct classification of the Italian variants is shown in relation to renowned databases that only report a broader view of the European population.

This approach enables researchers to classify variants that are not observed in the most widely used databases (gnomAD Exomes, ExAC, 1KgPhase3). It also allows the identification of rare variants that are generally classified as common and might represent a disease predisposition in the Italian population. In addition, it is possible to recognize common and non-damaging variants in the Italian population that are classified as rare in the European population.

In conclusion, the reported results and examples have shown how the new application (extended database with its own website) simplifies and facilitates the identification of problems in clinical WES analysis. It also makes the comparison between the various laboratory protocols easier, allowing for more precision in exome analysis aimed at identifying variants. Finally, the specific investigation of the Italian variants could improve diagnostic accuracy in the specific population.

INTRODUCTION

NEUROLOGICAL DISEASE STUDY IN COLLABORATION WITH THE MEYER CHILDREN HOSPITAL

Next-generation sequencing (NGS) technologies are revealing new approaches for disease diagnosis. In particular, many different disease-causing genes were discovered for neurogenetic disease[1], [2].

Neurological disorders are one of the most widespread diseases affecting children. These diseases are cause of disability and in the worse cases of premature death[3]. The most frequent chronic neurological conditions are epilepsy, autism spectrum disorders (ASD), attention-deficit hyperactivity disorder (ADHD), intellectual development disorders (ID), learning disabilities and cognitive retardation.

In Italy, one of the highly specialised institutions for children's neurogenetic healthcare is the Meyer Children Hospital. Together with the University of Florence, professor Renzo Guerrini, dott.ssa Elena Parrini, dott. Davide Mei, dott.ssa Annalisa Vetro and dott.ssa Claudia Bianchini work on research aimed at diagnosing and treating patients. In this PhD work, a total of 2,535 samples deriving from a collaboration with the Meyer Children Hospital are analysed. Most of these samples derive from families in which the proband is affected by a neurological disorder.

As above mentioned, NGS technologies can be used for disease diagnosis. In particular, the most frequent applications are whole genome sequencing (WGS), whole exome sequencing (WES) and targeted gene sequencing[4]–[7]. WGS consists in the sequencing of an individual’s entire genome, whereas the WES is based on the sequencing of the coding portion of the genome only. Targeted gene sequencing involves the sequencing of specific target genes. WES is used in family trios because of the lower cost compared to WGS, for the wider coverage compared to the targeted gene sequencing[8], [9]. WES analyses are therefore widely used in the diagnosis of neurological diseases in paediatric patients, as it enables a diagnosis without invasive tests and a consequent cure of the children from 8% to 39% of the cases, depending on the disease (8.8% for autism spectrum disorder[10], 32.6% for epilepsy[11], 38.7% for neurodevelopmental delay[12]).

WHOLE EXOME SEQUENCING AND VARIANT IDENTIFICATION

WES is a genomic technique that allows the investigation of the exome of an individual, i.e., all the protein-coding regions of the genome. In order to examine an individual's WES, blood or saliva is taken from the patient and subjected to laboratory and bioinformatics analysis.

The laboratory analysis of the exome consists of three main steps: library preparation, exome capture and sequencing. First, during the library preparation, the genomic DNA deriving from blood or swab samples is fragmented into smaller pieces using a mechanic or enzymatic fragmentation. The mechanic fragmentation consists in the exposure of DNA to ultrasonic acoustic energy to shear it in smaller fragments. The enzymatic fragmentation is instead the break of the DNA based on a mixture of enzymes that cut the DNA in different sites. After the fragmentation, to perform sequencing on Illumina instruments, the blunt-ended fragments are repaired, and an Adenosine (A) is attached to the 3' end. Then, the universal sequencing adapters are ligated to the fragments. Afterwards, the Unique Dual Index (UDI) are added by PCR to discriminate the different samples sequenced together in a single pool (Figure 1).

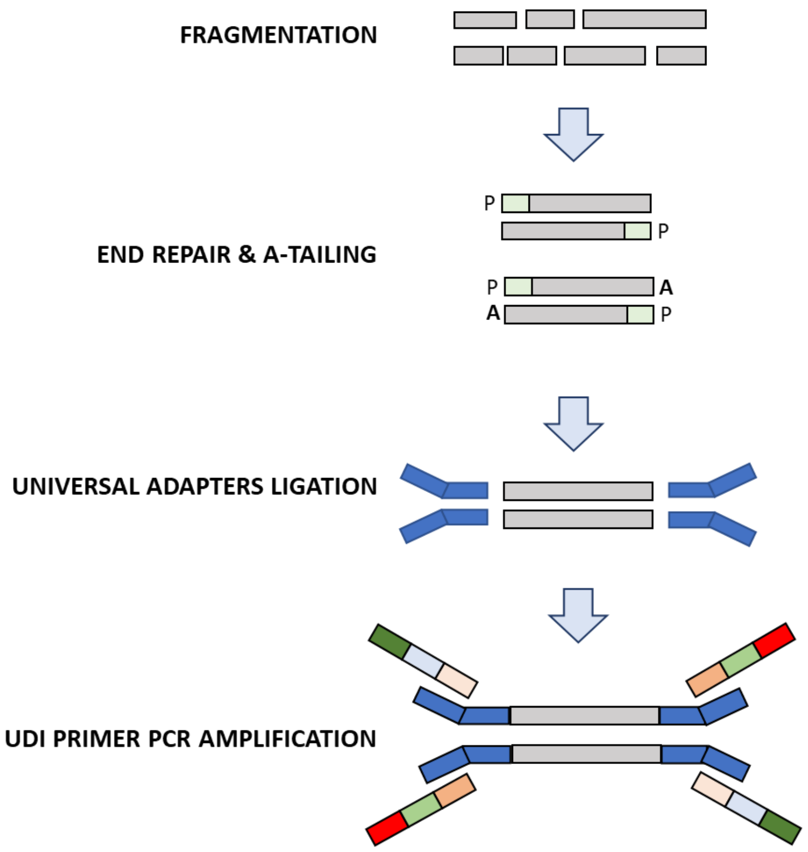


Figure 1. Library preparation workflow.

In the exome capture step, libraries derived from different samples (7/8 samples) are pooled and "captured" together in a single reaction. Then, specific biotinylate probes combined with streptavidin beads are used to extract (capture) only the exons from the DNA fragment library. At last, the captured fragments representing the exome are amplified and sequenced on a high-throughput DNA sequencer (Figure 2).

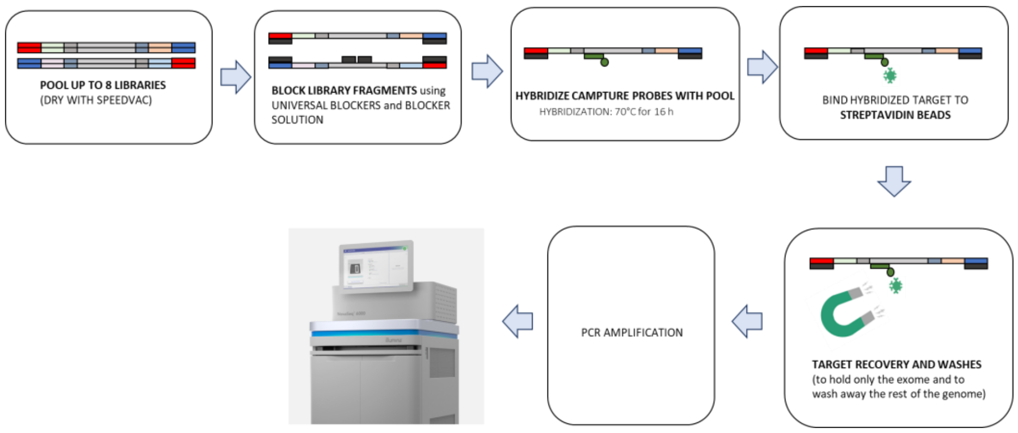

POOL UP TO 8 LIBRARIES (DRY WITH SPEEDVAC)
BLOCK LIBRARY FRAGMENTS using UNIVERSAL BLOCKERS and BLOCKER SOLUTION
HYBRIDIZE CAMPTURE PROBES WITH POOL
HYBRIDIZATION: 70°C for 16 h
BIND HYBRIDIZED TARGET TO STREPTAVIDIN BEADS
TARGET RECOVERY AND WASHES (to hold only the exome and to wash away the rest of the genome)
PCR AMPLIFICATION

Figure 2. Capture preparation workflow.

As above mentioned, WES analysis allows the identification of genetic variants causing disease in the coding regions of DNA.

To identify genetic variations of an individual, the sequencing reads are demultiplexed (sequences are divided according to the UDI index), collected in files (Fastq files), and analysed through a bioinformatics pipeline consisting of three main steps: 1) alignment, 2) variant calling, and 3) variant annotation and prioritization.

The process of alignment consists in the mapping of raw sequences to a well-characterized reference genome (Figure 3).

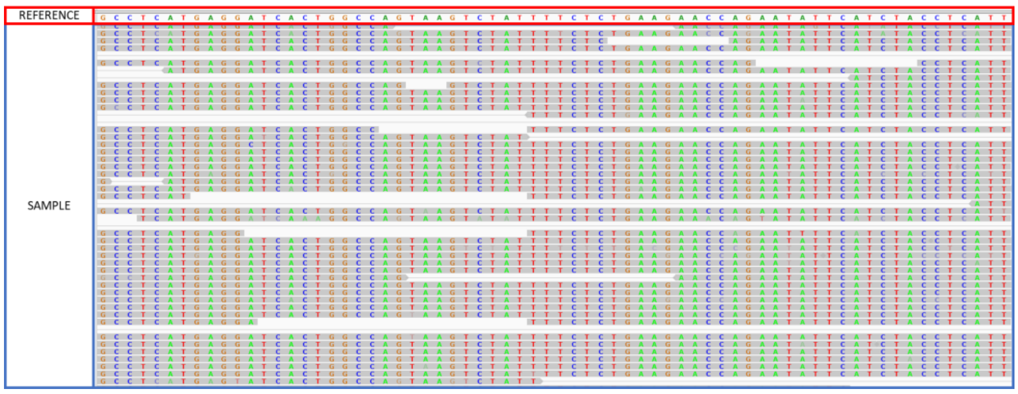


Figure 3. Example of alignment: above the reference genome sequence and below sequences deriving from a sequenced sample.

After the alignment, some quality controls are done on the alignment file: duplicated fragments are removed and overlapping reads are soft clipped (the overlapping portion of one of the two reads is hidden) to avoid multiple counts of the same base. Then, the resulting data are compared to a universally adopted reference to identify the sequence differences (i.e. the variants) in the analysed samples (Figure 4).

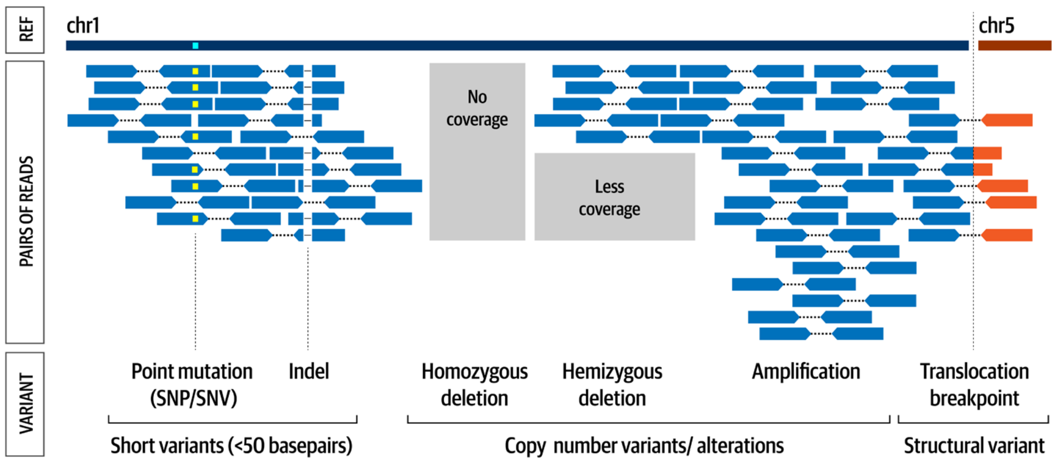


Figure 4. Graphical representation of different types of variants.

Variants can be divided into three main categories: short variants (SNV and small indels <50bp), copy number variants (CNV), and structural variants (SV). Whole exome sequencing is more employed for the detection of short variants, because of the lack of intronic regions which makes it difficult to correctly identify CNV and SV.

Subsequently, identified variants are prioritized and annotated based on clinical available information, on the frequency in the population, and on predictions.

EVALUATION METRICS

The evaluation of each step of the WES analysis, starting from the laboratory to the variant identification, is crucial to obtain high-quality samples to use for diagnosis.

Issues in laboratory procedures can indeed easily translate into poor-quality variant calling, namely missed identification of variants or even errors in variant calling. Different quality control parameters are assessed throughout the entire analysis, spanning from the steps performed in the laboratory to the computational identification of the genetic variants.

In laboratory analysis, the key parameters are the quality of the starting DNA sample, the concentration of the library and the capture.

Evaluation metrics used for the quality of the starting DNA sample are:

- DNA Integrity Number (DIN)
- 260/280 nm ratio
- 260/230 nm ratio
- Concentration ng/µl

DNA Integrity Number measures the degradation in terms of fragment size of the genomic DNA. DIN number range from 0 to 10, optimal DIN value should be ≥ 7.

The 260/280 nm ratio and the 260/230 nm ratio are metrics used to assess the nucleic acid purity. A 260/280 nm ratio < 1.8 can indicate protein contamination, whereas 260/230 nm ratio < 1.8 can indicate an organic compound contamination.

The concentration ng/µl measures DNA amount in a sample.

After the quality control, the sample undergoes library preparation. Each library kit can require a different quantity and purification, so it is important to have the right integrity, purity, and quantity.

For both library and capture, the most important metrics are:

- Average size
- Concentration ng/µl

The average size measures the length of the fragments and reflects the size of the sequenced insert, which is crucial to achieve higher mapping quality in repeat regions that is relevant in variant identification[13].

The right concentration of library and capture are important to proceed in the analysis.

In addition, it is important to keep track of lot numbers of library and capture kits, since slight differences in different batches of reagents can impact the library. For example, changes in the library preparation kit such as PCR buffer may affect the ability to amplify fragments of high GC content regions resulting in loss of sequencing of these target areas. Changes in the capture kit, on the other hand, affect the regions of interest that would be sequenced[14].

In the bioinformatics analysis, important data evaluation metrics are:

- Duplication rates
- Mean insert size
- Coverage
- Percentage of 5/10/20/30X
- PASS Percentage (genotypability)
- On/Near/Off target Percentage
- Fold80 penalty value
- Uniformity of Coverage (Pct > 0.2*mean)

The duplication rate is important to understand the complexity of the library, i.e. if the library is highly complex (low duplication rate), it means that the sequenced fragments represent my entire target; otherwise, if the library is not particularly complex (high duplication rate), it means that the sequenced fragments represent only some portions of my target.

Mean insert size is defined as the mean distribution of the distance between the two mate pairs, corresponding to the length of DNA fragments analysed (Figure 5).

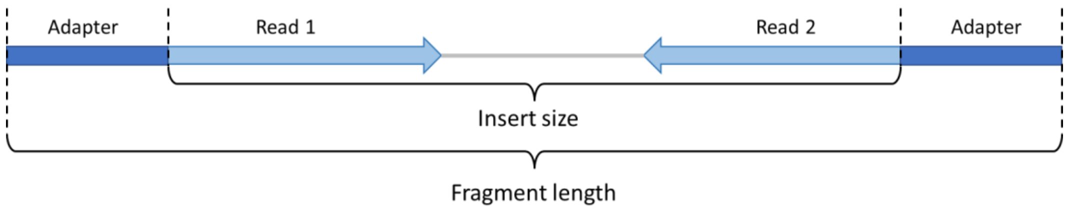


Figure 5. Graphical explanation of insert size.

Insert size distribution is important because fragments length influences the read mapping quality: if fragments are short, the read mapping quality will decrease since these fragments will map in repeated regions (no unique map). Increasing the fragment length allows to map sequences outside repeated regions and increases the mapping quality.

The quality of the alignment to the reference genome is extremely important for the right identification of the variants [15]. To assess this, different parameters could be considered (Figure 6):

- the mean coverage over the target, calculated as the number of aligned reads covering a genomic base
- the percentage of 5/10/20/30X -the fraction of target bases covered by at least 5/10/20/30 reads
- %PASS (genotypability), i.e., ability to call a base with good coverage (with at least 3 reads in support) and good mapping quality.

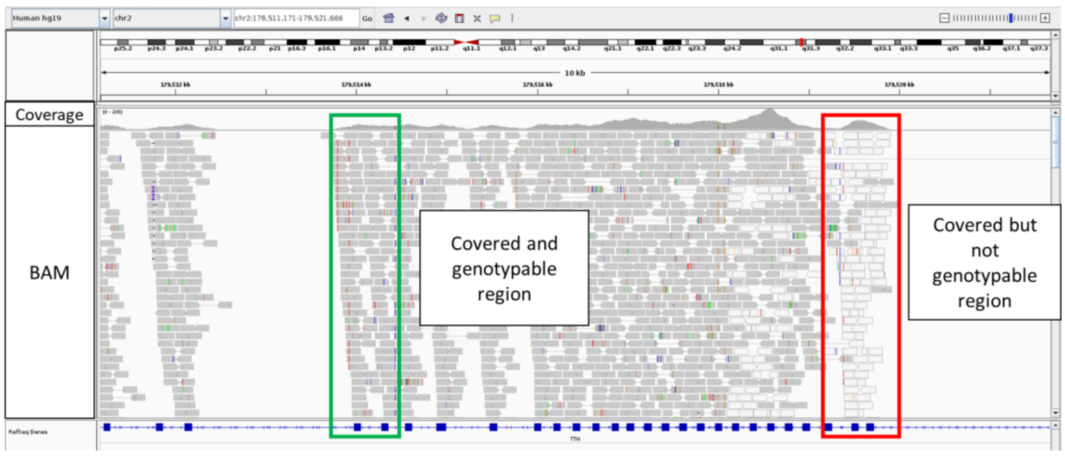

Human hg19
chr2
Coverage
BAM
Covered and genotypable region
Covered but not genotypable region
10 kb
RefSeq Genes
TTN

Figure 6. Graphical explanation of coverage and %PASS (genotypability).

Three additional metrics are required to understand if the sequenced reads are part of the desired target regions (Figure 7):

- On target: The number of aligned bases mapping to a targeted region
- Near target: The number of aligned bases mapping near a targeted region, where near is defined as the insert size length
- Off-target: The number of aligned bases mapping out of a targeted/near region

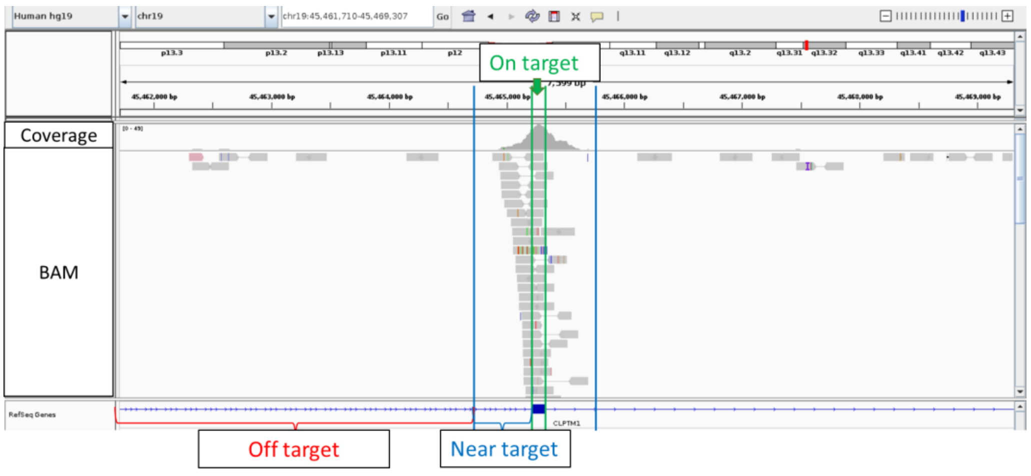


Figure 7. Graphical explanation of on/near/off-target.

Finally, the equal distribution of reads over the target is required to allow good coverage of all regions of interest and thus enable the identification of variants throughout the captured exome. To access the distribution of the reads over the entire target, the 2 following metrics must be evaluated:

- the fold80 penalty value, which is the fold over-coverage necessary to raise 80% of bases in covered targets to the mean coverage level in those targets,
- the uniformity of coverage (Pct > 0.2*mean), i.e., the calculated percentage of targeted base positions in which the read depth is greater than 0.2 times the mean region target coverage depth, can be calculated (Figure 8).

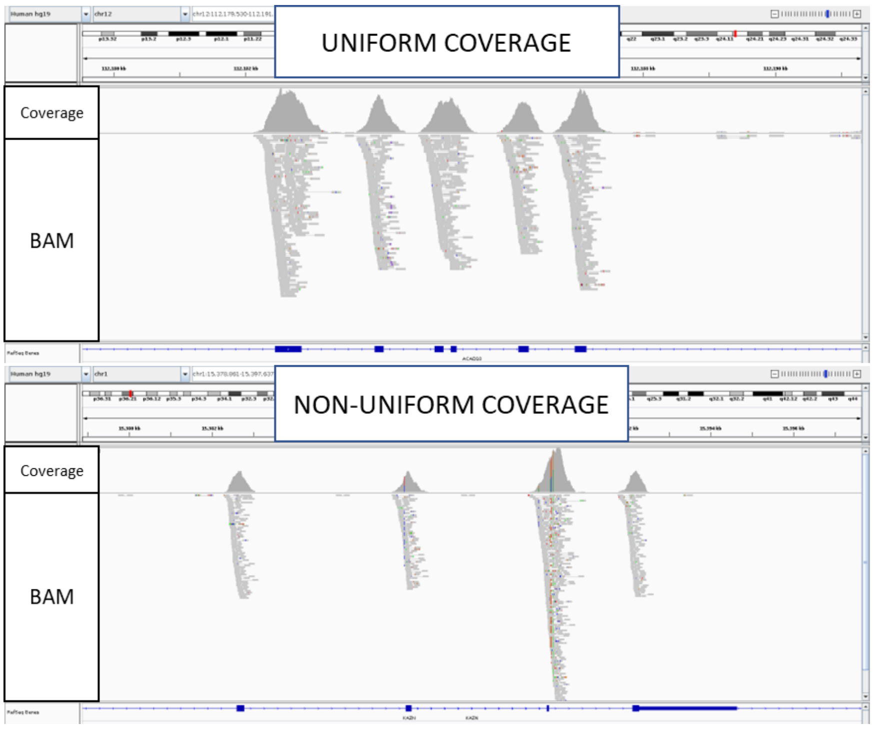


Figure 8. Graphical explanation of uniformity of coverage.

PRIORITIZATION METRICS

Other important metrics in the disease-causing variant identification are the ones used for the prioritization of the variants. Variants are annotated and prioritized based on i) clinical pathogenicity reported in clinical databases, ii) functional annotation and iii) frequency in the population.

First, it is crucial to understand if a variant has been already classified as disease-causing. The three most important databases used to understand the pathogenicity of a variant are ClinVar, the Human Gene Mutation Database (HGMD) and the Online Mendelian Inheritance in Man (OMIM). ClinVar is a public and free archive that reports the relationship between variants and phenotypes based on evidences[16]–[18]. Based on the 2015 ACMG guidelines, ClinVar classified variants in:

- pathogenic,
- likely pathogenic,
- uncertain significance (VUS),
- likely benign,
- benign.

The Human Gene Mutation Database is a collection of germline variants, published in the peer-reviewed literature, associated or closely associated to human inherited disease. HGMD is available in two different versions: the public and the professional one[19], [20]. HGMD classified variants in:

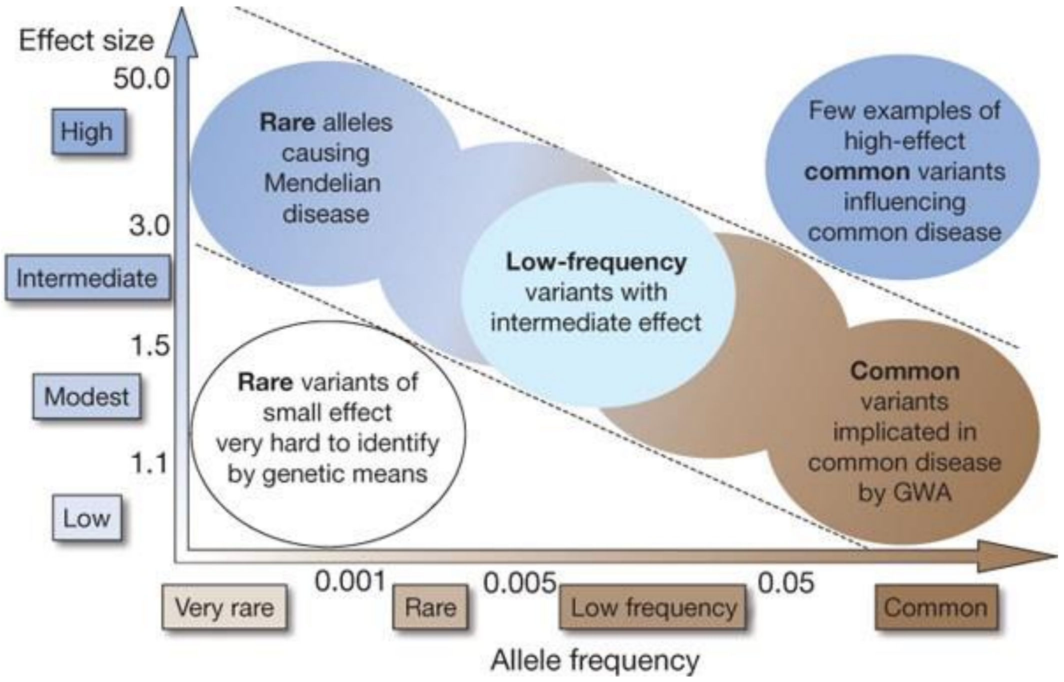
- disease-causing (pathological) mutation (DM),
- likely disease-causing (likely pathological) mutation (DM?),
- disease-associated polymorphism (DP),
- disease-associated polymorphism with additional supporting functional evidence (DFP),
- polymorphism affecting the structure, function, or expression of a gene but with no disease association reported yet (FP),

- frameshift or truncating variant with no disease association reported yet (FTV).

The Online Mendelian Inheritance in Man is a freely available catalogue based on peer-reviewed literature which contains a curated description of human genes and the genetic disorders and traits and the relationship between them [21], [22].

The second factor involved in the variants prioritization is the understanding of the functional effect of variants, i.e. whether a variant will cause a loss of gene function or encode a different amino acid. Several databases are currently available for the functional annotation of variants. In particular, one of the most widely used is the Reference Sequence Collection (RefSeq), which contains a non-redundant and well-annotated set of sequences derived from genomic DNA, transcripts and proteins [23], [24].

Finally, it is important to understand how much rare or common a variant is in a particular population. Figure 9[25] shows the relationship between the variant' allele frequency and its effect. In general, the rarer the variant, the higher the effect, i.e., the variant is very likely pathogenic. Although in some cases a rare variant could not be pathogenic and on the contrary a common variant could have high effect.


Effect size
50.0
High
3.0
Intermediate
1.5
Modest
1.1
Low
Rare alleles causing Mendelian disease
Few examples of high-effect common variants influencing common disease
Low-frequency variants with intermediate effect
Rare variants of small effect very hard to identify by genetic means
Common variants implicated in common disease by GWA
Very rare
0.001
Rare
0.005
Low frequency
0.05
Common
Allele frequency
Figure 9. Relationship between variant allele frequency and its effect.

Different frequencies databases are available to prioritize variants, such as gnomAD[26], ExAC[27], 1KgPhase3[27]. GnomAD contains variants deriving from 76,156 genome samples and is developed by an international alliance with the aim to combine data deriving from different sequencing projects available for the entire scientific community. GnomAD identifies individuals based on their origin as European, African, Latino, Ashkenazy Jewish, East Asian, South Asian, also divided for sub-population (e.g. Southern-European, Bulgarian, Korean etc). ExAC is now part of gnomAD project. ExAC contains variants deriving from 60,706 individuals. 1KgPhase3 is the first project that aims to discover >95% of the variants with Minor Allele Frequency (MAF) as low as 1% (across the genome) and 0.1-0.5% in gene regions. The third phase of the 1000 Genomes Project analysed a total of 2,548 samples, the dataset was produced using GRCh37 genome reference, and later re-analysed also with GRCh38. Both versions are available for download.

AIM OF THE THESIS

Whole exome analysis starts with a process of laboratory sample preparation for the sequencing, then the sample is sequenced and the resulting data are analysed through bioinformatic pipelines. The quality control parameters employed in such analyses are not stored in a database, making it impractical to have an overview of the overall long-term tendency. Moreover, the laboratory and the bioinformatics QC measurements are generally assessed individually and they are rarely interlinked.

Furthermore, for an accurate variant analysis, it is important to have an accurate variant interpretation. To this purpose, one of the most relevant criteria to consider is the population-specific frequency of variants. At the present time, the most commonly used databases take into account the European population, but no frequency database is currently available for the Italian population.

My thesis aims to develop a bioinformatics pipeline combined with an internal database to supervise the WES data and their quality, store and evaluate the entire sample information and genetic variants to ultimately achieve a more precise variant calling.

METHODS

SAMPLE COLLECTION AND LABORATORY PREPARATION

Blood samples deriving from children with neurogenic disorders and their parents are collected from Meyer Children's Hospital in Florence. Most of the affected children present autism, different types of epilepsy, epileptic encephalopathy, macrocephaly and microcephaly, and different types of retardation (mental, developmental, psychomotor). From 2018 till now, the whole exome of 2,535 samples underwent analysis.

First, biological samples deriving from probands, and parents undergo quality control by checking the purity, integrity and quantity of DNA. The purity check identifies the probable presence of contaminants, the integrity check shows if the sample has undergone a deterioration, and finally the quantity check indicates if available DNA is enough for the downstream analysis. After that, samples are subjected to the library preparation step, in which the Illumina adapters are added to permit the binding to the sequencing flow cells. Subsequently, target enrichment captures are performed on libraries to pick up only regions of interest.

Different enrichment captures protocols (Agilent XT2 - CRE v2, Twist Exome Core, Twist Exome Core + RefSeq v. 1.0/v.1.3, Twist Exome v2.0, Roche KAPA HyperExome) were used in this project. Finally, data were sequenced on Illumina platforms generating 2x150bp-reads.

PIPELINE FOR VARIANT CALLING

Initial FASTQ files were quality controlled using FastQC[28]. All sequenced fragments were trimmed for base quality and adapters using fastp v.0.21.0[29] and mapped using BWA-mem v.0.7.17[30] to the reference HG19 for clinical analysis and to HG38 for variants internal database. Overlapping reads were clipped using BamUtil v1.0.14[31] and the duplicated ones were marked and removed using Picard v2.21.1 MarkDuplicates[32]. GATK UnifiedGenotyper v.3.8.1.0[33], [34] was used for the variant calling for the clinical analysis, whereas HaplotypeCaller v.4.2.0.0[35], [36] was used for variant calling for the variants internal database. Concurrently, from alignment files the mean insert size, the coverage, the %PASS, and %PASS(DP>=10) were calculated using Picard CollectInsertSize v2.21.1 and GATK CallableLoci v.3.8.1.0, respectively. Furthermore, Picard CollectHsMetrics v2.21.1 was used to calculate on/near/off-target and fold80 penalty values, and an in-house script (Script 1 – calculate_unif_coverage_PCT.sh – Appendix A) was used to calculate the uniformity of coverage (Figure 10).

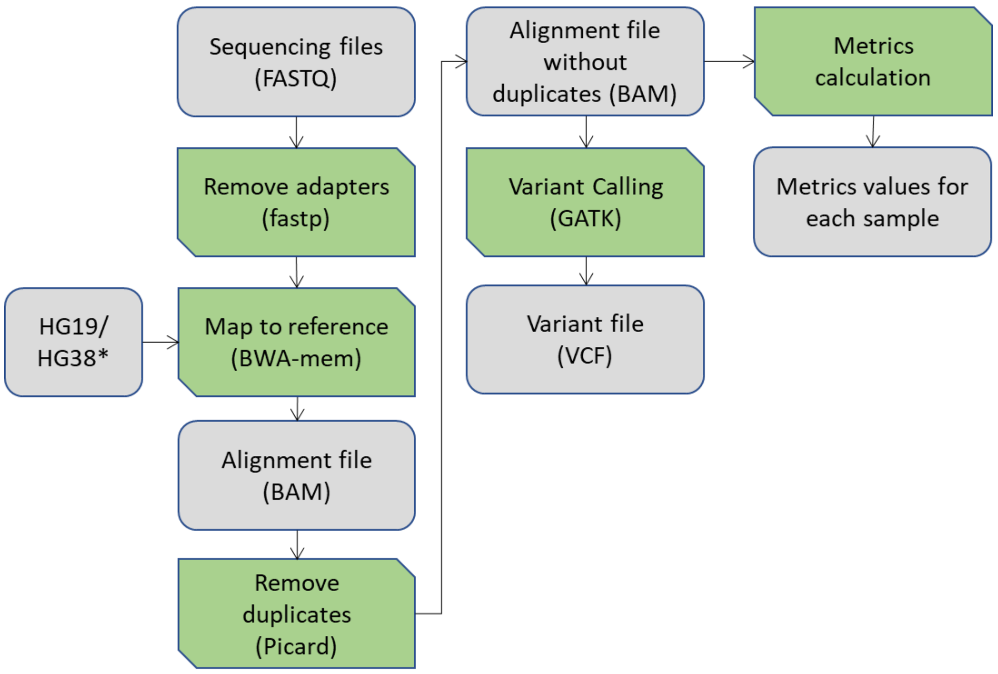


Figure 10. Schema of the used pipeline for the variant calling.

PIPELINE FOR DATABASE CREATION

A database to store all information deriving from the above mentioned procedures was created using Python v.3.6.8[37]. A Python script using Psycopg2 package was used to implement the database. Psycopg2 is a package implemented in C widely used as a PostgreSQL[38] database adapter. It contains cursors for both the client-side and server-side. Furthermore, to inspect the created database, PgAdmin4 v. 5.1[39] was employed. It is an Open-Source platform that allows the administration and development of PostgreSQL database.

PIPELINE FOR WEBSITE CREATION

To facilitate the use of the database, a website based on Python v.3.6.8 and HyperText Markup Language (HTML) scripts was developed. HTML is a language with specific rules used for the writing of documents destined for webpages. Thus, HTML was used to design the website pages layout. Flask v. 2.0.2 [40] framework, a structured architecture specific for the development of a dynamic website, was also used to build up the application that connects the database to the website, through the use of Python scripts. Different packages were integrated into the scripts:

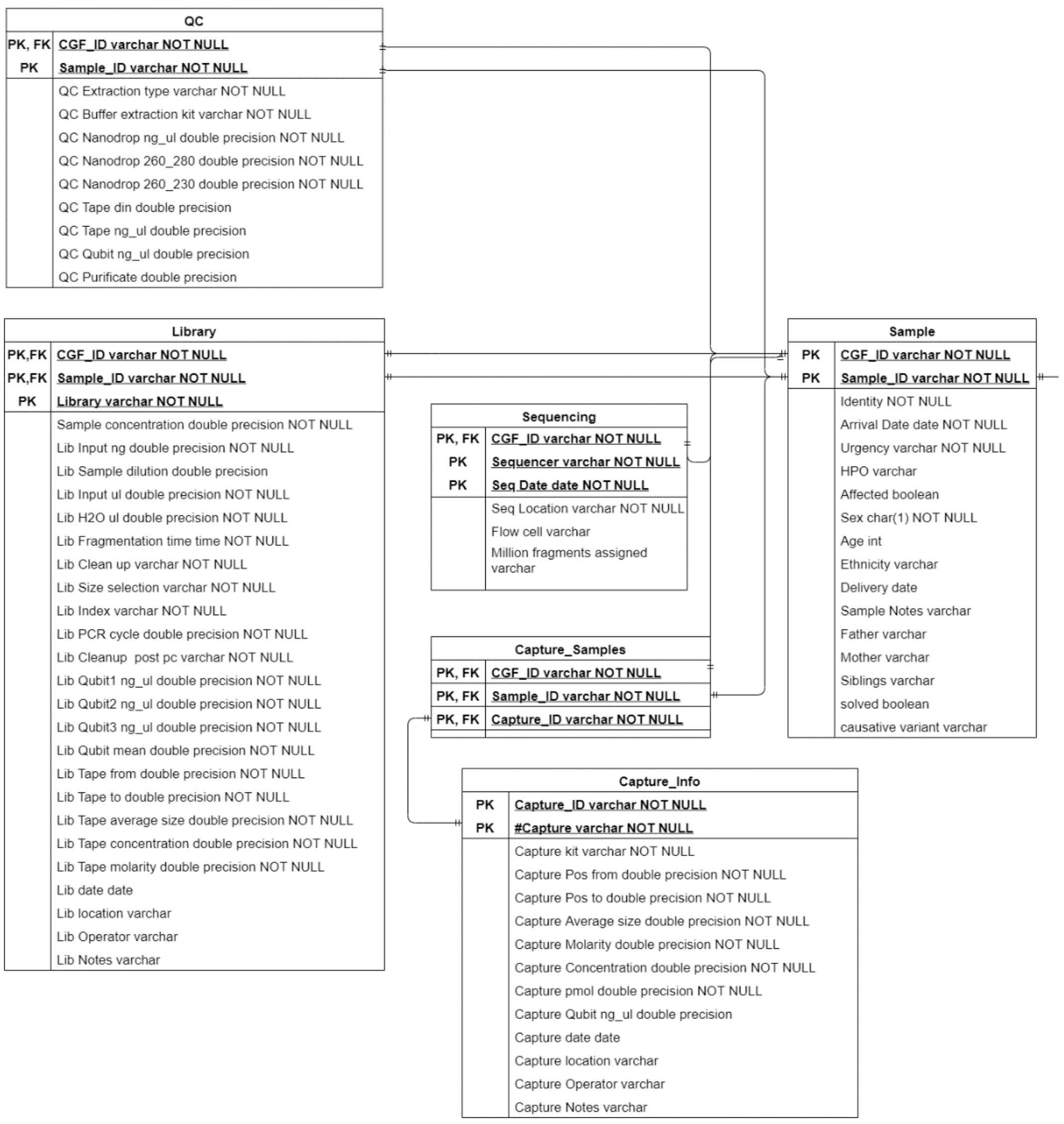
- *RetrieveExcel* to read the uploaded data as an excel file
- *Pandas*[41] to manipulate the data
- *psycopg2.extras, InterrogateDB, create_engine* to interrogate the PostgreSQL database
- *matplotlib.pyplot*[42] to create plots

RESULTS

DATABASE AND WEBSITE

A database containing (Figure 11) all sample's information was built. It comprises the patient phenotypes, the laboratory data, the data analysis statistics, and the sample variants. The database has been structured to connect the laboratory (Figure 11.A) and the bioinformatic information (Figure 11.B) so that it is easier to correlate the obtained results with changes in laboratory protocols. The DB was created using a Python script (Script 2 – db_creation.py – Appendix A) that communicates with PGAdmin4, where all data are stored. All tables and their attributes are described in Appendix B.

A



QC
PK, FK CGF_ID varchar NOT NULL
PK Sample_ID varchar NOT NULL
QC Extraction type varchar NOT NULL
QC Buffer extraction kit varchar NOT NULL
QC Nanodrop ng_ul double precision NOT NULL
QC Nanodrop 260_280 double precision NOT NULL
QC Nanodrop 260_230 double precision NOT NULL
QC Tape din double precision
QC Tape ng_ul double precision
QC Qubit ng_ul double precision
QC Purificate double precision
Library
PK,FK CGF_ID varchar NOT NULL
PK,FK Sample_ID varchar NOT NULL
PK Library varchar NOT NULL
Sample concentration double precision NOT NULL
Lib Input ng double precision NOT NULL
Lib Sample dilution double precision
Lib Input ul double precision NOT NULL
Lib H2O ul double precision NOT NULL
Lib Fragmentation time time NOT NULL
Lib Clean up varchar NOT NULL
Lib Size selection varchar NOT NULL
Lib Index varchar NOT NULL
Lib PCR cycle double precision NOT NULL
Lib Cleanup post pc varchar NOT NULL
Lib Qubit1 ng_ul double precision NOT NULL
Lib Qubit2 ng_ul double precision NOT NULL
Lib Qubit3 ng_ul double precision NOT NULL
Lib Qubit mean double precision NOT NULL
Lib Tape from double precision NOT NULL
Lib Tape to double precision NOT NULL
Lib Tape average size double precision NOT NULL
Lib Tape concentration double precision NOT NULL
Lib Tape molarity double precision NOT NULL
Lib date date
Lib location varchar
Lib Operator varchar
Lib Notes varchar
Sequencing
PK, FK CGF_ID varchar NOT NULL
PK Sequencer varchar NOT NULL
PK Seq Date date NOT NULL
Seq Location varchar NOT NULL
Flow cell varchar
Million fragments assigned varchar
Capture_Samples
PK, FK CGF_ID varchar NOT NULL
PK, FK Sample_ID varchar NOT NULL
PK, FK Capture_ID varchar NOT NULL
Capture_Info
PK Capture_ID varchar NOT NULL
PK #Capture varchar NOT NULL
Capture kit varchar NOT NULL
Capture Pos from double precision NOT NULL
Capture Pos to double precision NOT NULL
Capture Average size double precision NOT NULL
Capture Molarity double precision NOT NULL
Capture Concentration double precision NOT NULL
Capture pmol double precision NOT NULL
Capture Qubit ng_ul double precision
Capture date date
Capture location varchar
Capture Operator varchar
Capture Notes varchar
Sample
PK CGF_ID varchar NOT NULL
PK Sample_ID varchar NOT NULL
Identity NOT NULL
Arrival Date date NOT NULL
Urgency varchar NOT NULL
HPO varchar
Affected boolean
Sex char(1) NOT NULL
Age int
Ethnicity varchar
Delivery date
Sample Notes varchar
Father varchar
Mother varchar
Siblings varchar
solved boolean
causative variant varchar

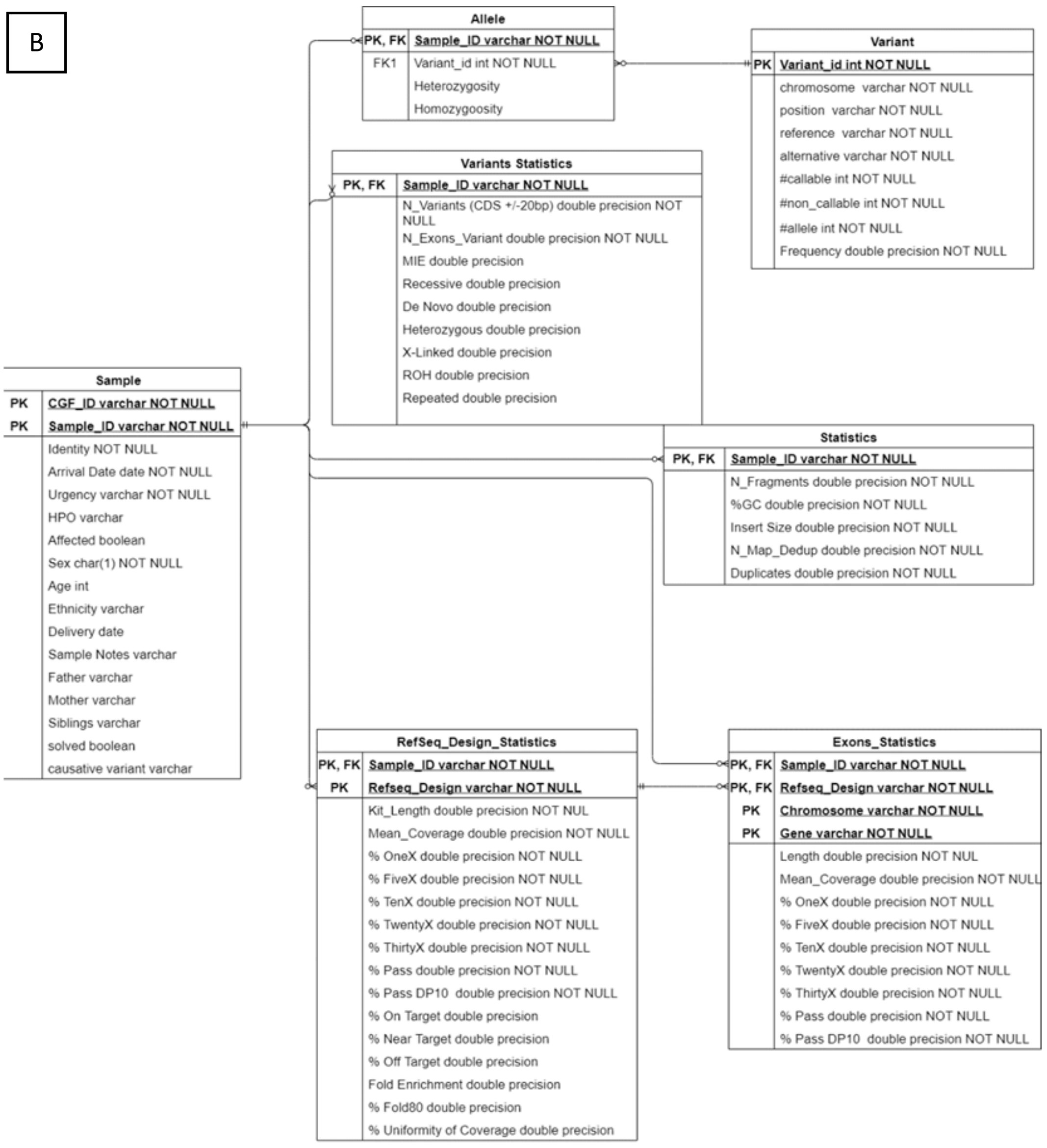

B
Allele
PK, FK Sample_ID varchar NOT NULL
FK1 Variant_id int NOT NULL
Heterozygosity
Homozygoosity
Variant
PK Variant_id int NOT NULL
chromosome varchar NOT NULL
position varchar NOT NULL
reference varchar NOT NULL
alternative varchar NOT NULL
#callable int NOT NULL
#non_callable int NOT NULL
#allele int NOT NULL
Frequency double precision NOT NULL
Variants Statistics
PK, FK Sample_ID varchar NOT NULL
N_Variants (CDS +/-20bp) double precision NOT NULL
N_Exons_Variant double precision NOT NULL
MIE double precision
Recessive double precision
De Novo double precision
Heterozygous double precision
X-Linked double precision
ROH double precision
Repeated double precision
Sample
PK CGF_ID varchar NOT NULL
PK Sample_ID varchar NOT NULL
Identity NOT NULL
Arrival Date date NOT NULL
Urgency varchar NOT NULL
HPO varchar
Affected boolean
Sex char(1) NOT NULL
Age int
Ethnicity varchar
Delivery date
Sample Notes varchar
Father varchar
Mother varchar
Siblings varchar
solved boolean
causative variant varchar
Statistics
PK, FK Sample_ID varchar NOT NULL
N_Fragments double precision NOT NULL
%GC double precision NOT NULL
Insert Size double precision NOT NULL
N_Map_Dedup double precision NOT NULL
Duplicates double precision NOT NULL
RefSeq_Design_Statistics
PK, FK Sample_ID varchar NOT NULL
PK Refseq_Design varchar NOT NULL
Kit_Length double precision NOT NUL
Mean_Coverage double precision NOT NULL
% OneX double precision NOT NULL
% FiveX double precision NOT NULL
% TenX double precision NOT NULL
% TwentyX double precision NOT NULL
% ThirtyX double precision NOT NULL
% Pass double precision NOT NULL
% Pass DP10 double precision NOT NULL
% On Target double precision
% Near Target double precision
% Off Target double precision
Fold Enrichment double precision
% Fold80 double precision
% Uniformity of Coverage double precision
Exons_Statistics
PK, FK Sample_ID varchar NOT NULL
PK, FK Refseq_Design varchar NOT NULL
PK Chromosome varchar NOT NULL
PK Gene varchar NOT NULL
Length double precision NOT NUL
Mean_Coverage double precision NOT NULL
% OneX double precision NOT NULL
% FiveX double precision NOT NULL
% TenX double precision NOT NULL
% TwentyX double precision NOT NULL
% ThirtyX double precision NOT NULL
% Pass double precision NOT NULL
% Pass DP10 double precision NOT NULL

Figure 11. Relational schema of the database.

Flask web framework has been used to interrogate the database and to connect it to a web interface. For this purpose, as shown in Figure 12, four different Python scripts were developed.

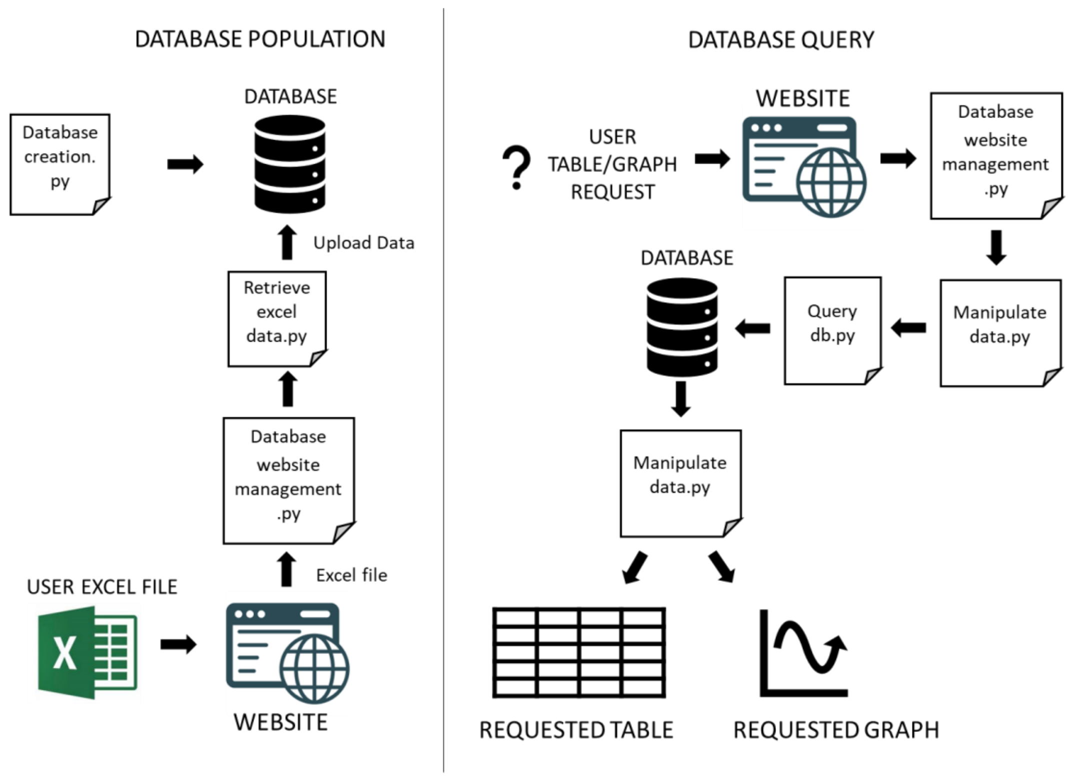
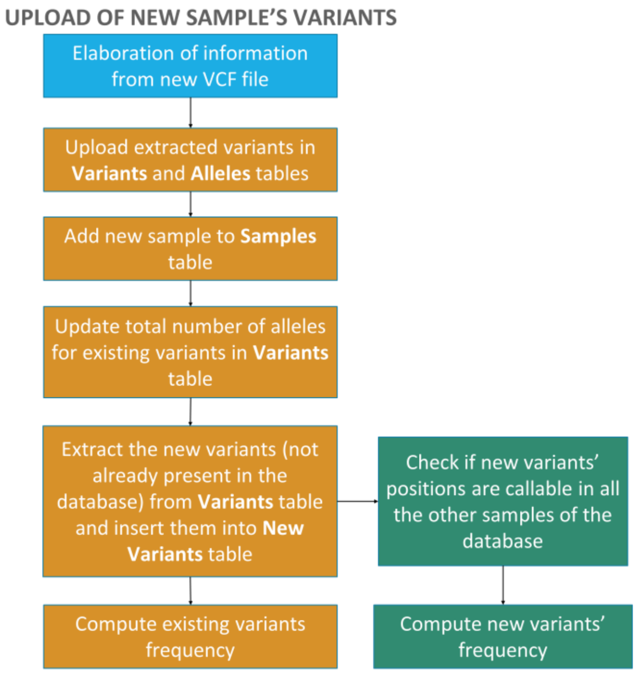


Figure 12. Schema of the relation between Python scripts.

The first one, "database website management" (Script 3 – db_website_management.py – Appendix A) is responsible for connecting the different Python scripts with HTML scripts. Depending on the user request, "database website management" can call the two scripts: "retrieve excel data" and "manipulate data".

The "retrieve excel data" (Script 4 – retrieve_excel_data.py – Appendix A) script takes as input the excel file uploaded from the user and the table to which the data must be uploaded. Then, it reads the file, saves the data in a dataframe and uploads the data in the database.

The "manipulate data" script (Script 5 – manipulate_data.py – Appendix A) takes as input the request from the user on tables, possible filters, and the type of data visualizations. The script "manipulate data" calls the script "query DB" (Script 6 – query_db.py – Appendix A) which converts the user request into a SQL query, interrogates the database and returns the results of the SQL query as a table or as a graph.


UPLOAD OF NEW SAMPLE'S VARIANTS
Elaboration of information from new VCF file
Upload extracted variants in Variants and Alleles tables
Add new sample to Samples table
Update total number of alleles for existing variants in Variants table
Extract the new variants (not already present in the database) from Variants table and insert them into New Variants table
Check if new variants' positions are callable in all the other samples of the database
Compute existing variants frequency
Compute new variants' frequency
Figure 13. Schema script for the upload of the variants on the database.

As far as the tables variants and alleles are concerned, a specific command line pipeline was developed to facilitate the upload of samples variants (Figure 13). First, the script "upload.py" (Script 7 – upload.py – Appendix A) extracts the sample name and all variants information, the chromosome, the position, the reference, the alternative bases and the genotype from the variant file (VCF) (Supplementary Table 11). Then, the extracted information are uploaded to the database while checking if the variant is already present. In this case, the total number of alleles and the number of samples are only updated with the callable or non-callable base, otherwise it creates a new entry in the database. When the variants are uploaded, the VCF file is compared with the bed files containing the regions that are genotyped, and only the callable bases are used to calculate the frequencies. After the upload of all the variants, the script "upload_frequency" (Script 8 – upload_frequency.py – Appendix A) recalculates all frequencies based on the new values of the total number of alleles (Supplementary Table 12) and the number of callable bases. Finally, the new values of the frequencies are updated.

The created database and the connected website result into a web page available at http://157.27.84.25/FGVDB/index/ (at the moment available only to the University of Verona intranet).

The home page is divided into three sections: the laboratory and the bioinformatics ones are dedicated to data upload while the third one allows users to retrieve all the data (Figure 14). Each section is accessible by clicking on the link.

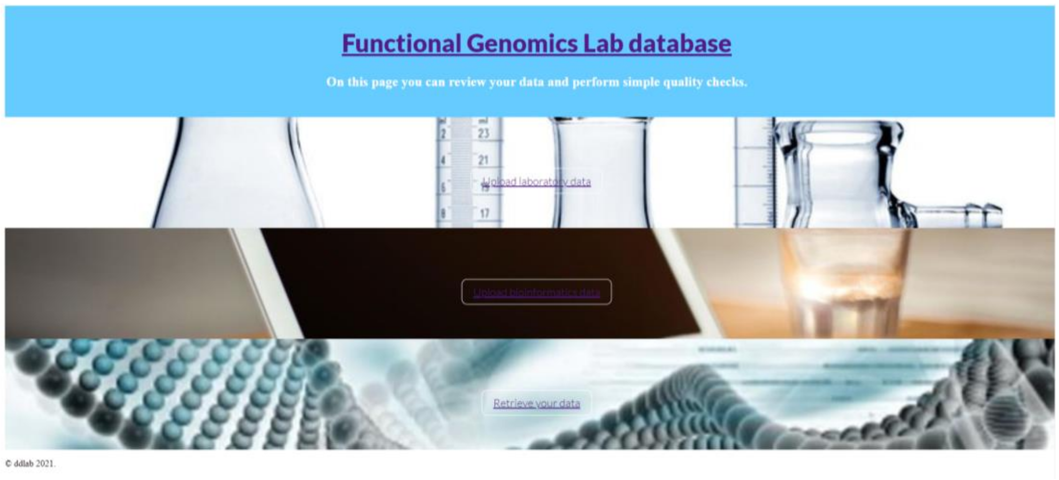
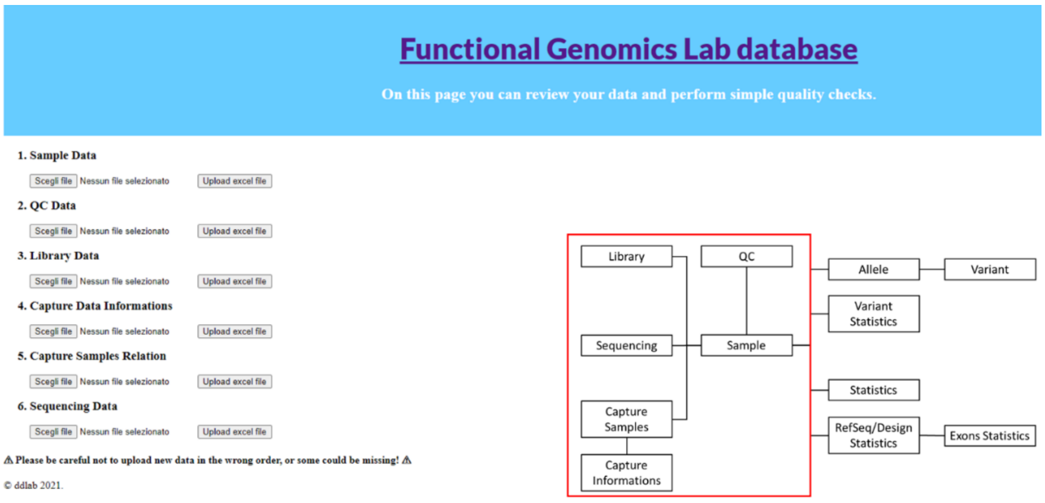

Functional Genomics Lab database
On this page you can review your data and perform simple quality checks.
Upload laboratory data
Upload bioinformatic data
Retrieve your data
© ddlab 2021

Figure 14. Starting page of the website.

In the wet-lab page (Figure 15), the different tables of the database schema can be uploaded as a .xls file. The sample data file must be the first to be uploaded in the "Sample Data" field to insert into the database the samples id and all its essential information (Supplementary Table 1). Then, all parameters about the DNA integrity, purity, and the quantity of the samples (Supplementary Table 2) deriving from the quality control process can be uploaded in the "QC Data" section. Afterwards, libraries are usually performed and parameters describing the quality, the concentration and the quantity of them can be uploaded in the "Library data" section (Supplementary Table 3).

Subsequently, in the case of whole-exome sequencing, the samples are subjected to the capture, whose quality specifications can be transferred to the database in the "Capture Data Information" field (Supplementary Table 5). The "Capture Samples Relation" area accepts a file that connects the samples id with the captures in which they are placed (Supplementary Table 4). Finally, the sequencing information can be uploaded in the "Sequencing Data" section (Supplementary Table 6).


Functional Genomics Lab database

On this page you can review your data and perform simple quality checks.

1. Sample Data

Scegli file Nessun file selezionato Upload excel file

2. QC Data

Scegli file Nessun file selezionato Upload excel file

3. Library Data

Scegli file Nessun file selezionato Upload excel file

4. Capture Data Informations

Scegli file Nessun file selezionato Upload excel file

5. Capture Samples Relation

Scegli file Nessun file selezionato Upload excel file

6. Sequencing Data

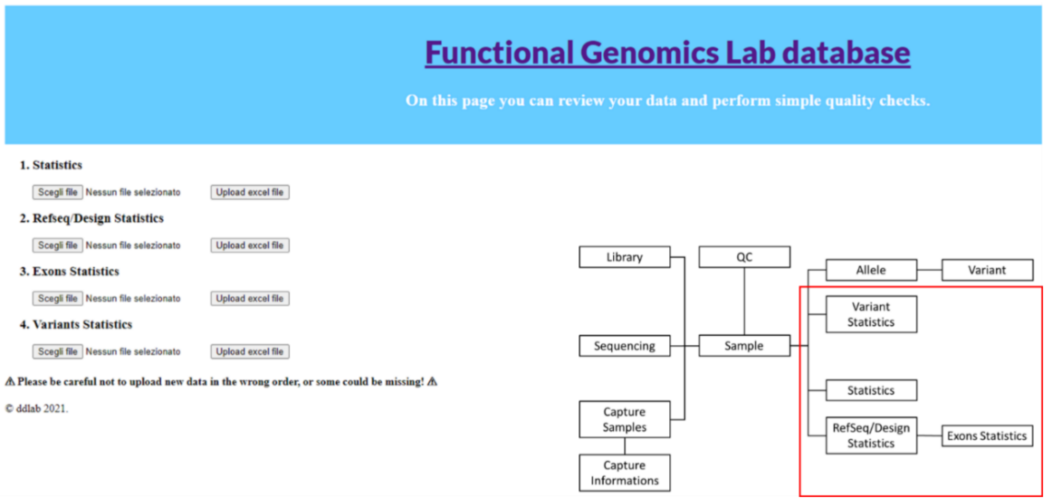
Scegli file Nessun file selezionato Upload excel file

⚠ Please be careful not to upload new data in the wrong order, or some could be missing! ⚠

© ddlab 2021.

Figure 15. Laboratory page.

Likewise, the bioinformatics page (Figure 16) contains sections to upload data deriving from the bioinformatic analysis. Firstly, the total number of sequenced fragments, the %GC, the average insert size, the number of fragments after deduplication, and the percentage of duplicates can be uploaded in the "Statistics" section (Supplementary Table 7). Secondly, all the statistics about the coverage, the uniformity of coverage, and the genotypability calculated on both RefSeq and on regions captured by a specific enrichment platform (i.e. Design) can be uploaded in the "RefSeq/Design Statistics" area (Supplementary Table 8). Then, for each sample, the excel file with the coverage statistics for each exon calculated on both RefSeq and Design can be transferred in the "Exons Statistics" field (Supplementary Table 9). Finally, in the "Variants Statistics" a file containing the total number of variants in CDS (+/- 20bp), the total number of exon variants and the number of variants identified in each category (MIE, Recessive, De Novo, Heterozygous, X-Linked, ROH, Repeated) can be reported (Supplementary Table 10).



Functional Genomics Lab database

On this page you can review your data and perform simple quality checks.

1. Statistics

Scegli file Nessun file selezionato Upload excel file

2. Refseq/Design Statistics

Scegli file Nessun file selezionato Upload excel file

3. Exons Statistics

Scegli file Nessun file selezionato Upload excel file

4. Variants Statistics

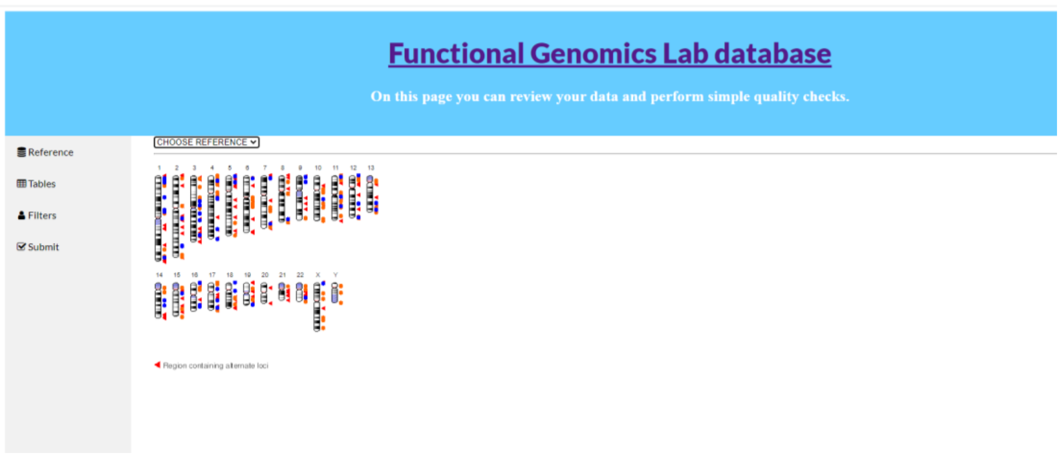
Scegli file Nessun file selezionato Upload excel file

⚠ Please be careful not to upload new data in the wrong order, or some could be missing! ⚠

© ddlab 2021.

Figure 16. Bioinformatic page.

The data mining page (Figure 17) allows users to retrieve information from the database. First of all, customers can choose which version of the human genome to consider (hg19 or hg38). In fact, data were analysed with both versions of the human genome. This happened because the Meyer hospital requests the hg19 version due to its reproducibility, while in our laboratory, the hg38 is employed since reads are better aligned to this newer version of the genome[43]–[45]. At the moment, 1,776/2,535 samples, analysed from 2019, were uploaded to the hg19 database, whereas 2,008/2,535 samples were re-analyzed and uploaded to the hg38 database (Table 1). Variants are uploaded and available only for the hg38 one.



Functional Genomics Lab database

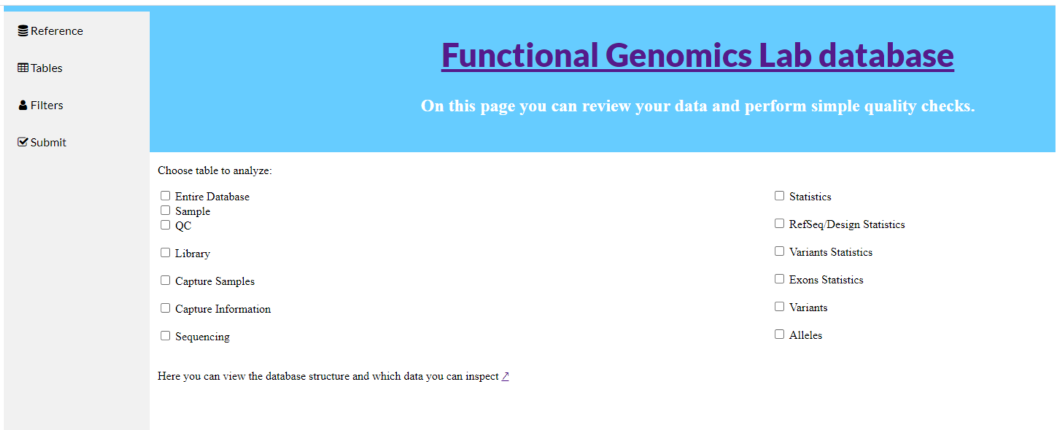
On this page you can review your data and perform simple quality checks.

Figure 17. Data mining page.

# Samples	Exome Capture Kit	Database
1,590	Twist Exome Core + RefSeq v.1.3	hg19
86	Twist Exome v2.0	
100	Roche KAPA HyperExome	
96	Agilent XT2 - CRE v2	hg38
99	Roche KAPA HyperExome	
389	Twist Exome Core + RefSeq v.1.0	
204	Nextera Rapid Capture Exome	
41	Roche SeqCap EZ MedExome	
1179	Twist Exome Core + RefSeq v.1.3	

Table 1. Table reporting the number of samples done with a specific kit and the database to which they are uploaded.

Additionally, users can retrieve data selecting one or more tables (Figure 18). The option "Entire Database" was created to collect all the main information about the sample by connecting tables: Sample, QC, Library, Capture Samples, Capture information, Sequencing, Statistics, RefSeq/Design Statistics, Variants Statistics.



Functional Genomics Lab database

On this page you can review your data and perform simple quality checks.

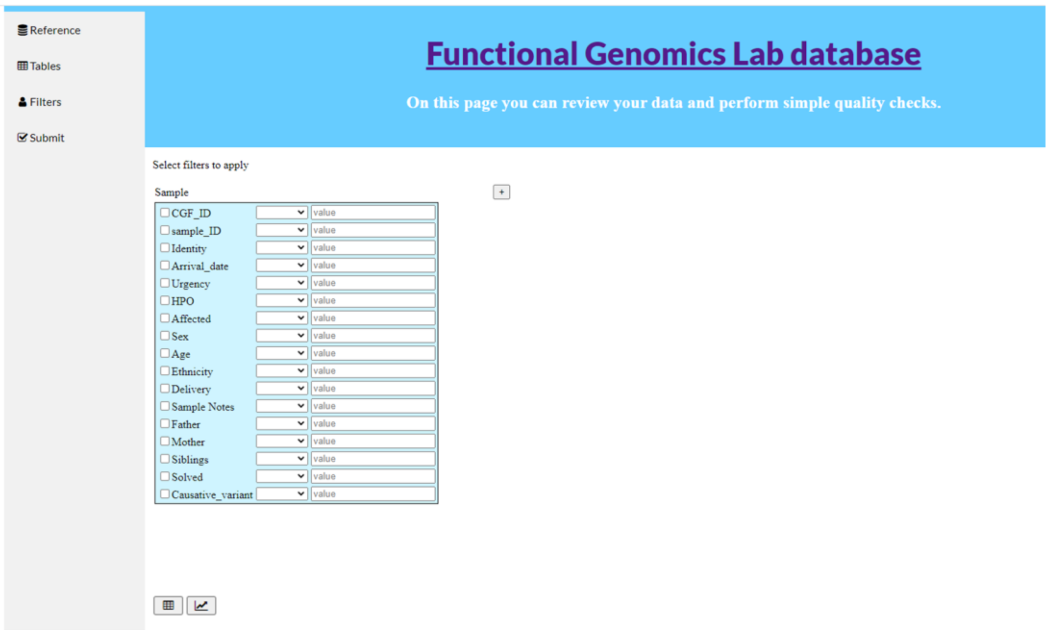
Choose table to analyze:

- ☐ Entire Database
- ☐ Sample
- ☐ QC
- ☐ Library
- ☐ Capture Samples
- ☐ Capture Information
- ☐ Sequencing
- ☐ Statistics
- ☐ RefSeq/Design Statistics
- ☐ Variants Statistics
- ☐ Exons Statistics
- ☐ Variants
- ☐ Alleles

Here you can view the database structure and which data you can inspect

Figure 18. Data mining page.

Finally, as shown in Figure 19, for each table in the database the user can select the column to examine and apply filters for specific parameters. Data can be reviewed as tables or as a plot.


Functional Genomics Lab database

On this page you can review your data and perform simple quality checks.

Select filters to apply

Sample

Figure 19. Data mining page.

EXAMPLES OF QUALITY CONTROL WITH THE CREATED DATABASE/WEBSITE

The created database and website provide a complete picture of the data and to help to solve frequent issues. During the years, due to high amount of analysed data, several problems occurred and somehow compromised the quality of the data and the results of the variant calls. Here are two examples of problems:

- Insert size decrease
- Coverage and genotypability decrease

The database enables not only the control of the issues but also the comparison of the different protocols results assisting the user in choosing the optimal workflow to carry out the most successful analysis. An example of this is reported in:

- Capture kit comparison

Insert size decrease

By plotting the WES experimental data using the website and the information stored in the database, we could notice that the samples analysed between September 2018 and July 2019 showed an insert size between 240bp and 290bp (Figure 20). As reported by Iadarola et al.[13] using a longer insert size improves the identification of variants in repeated regions. For optimal genotypability, the insert size of libraries should range between 290bp and 340bp.

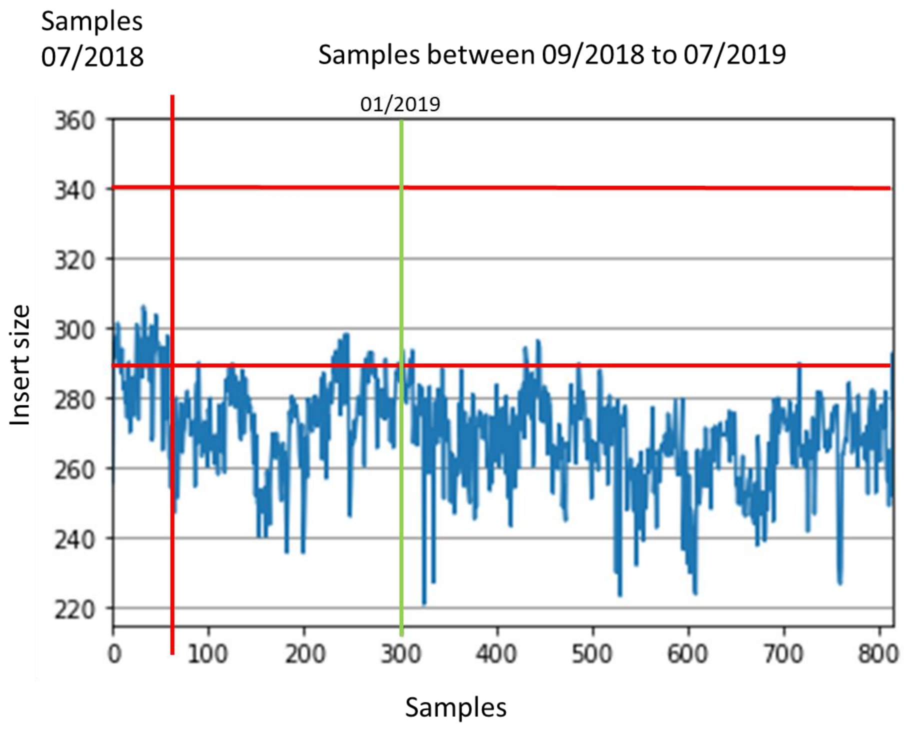


Figure 20. Insert size(y-axis) tendency of samples(x-axis) analysed between September 2018 and July 2019.

Thanks to the tool created, we have been able to report the problem to the laboratory, which adapted the protocol to increase the fragment length by modifying either the size selection step or the fragmentation times. After the adjustment, the data was generally in line with the expected data (Figure 21), despite some samples remaining outside the desired length.

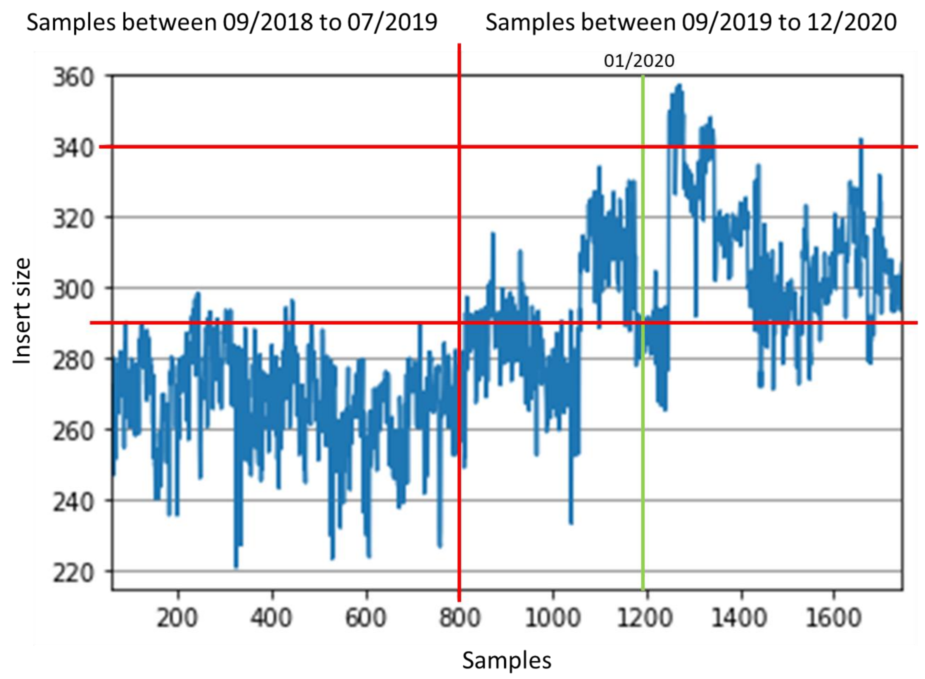


Figure 21. Insert size(y-axis) tendency of samples(x-axis) analysed between September 2018 and December 2020.

Data from the 2021 samples also confirmed the results obtained with the adjusted protocol, with data ranging in the desired length between 290bp and 340bp (Figure 22).

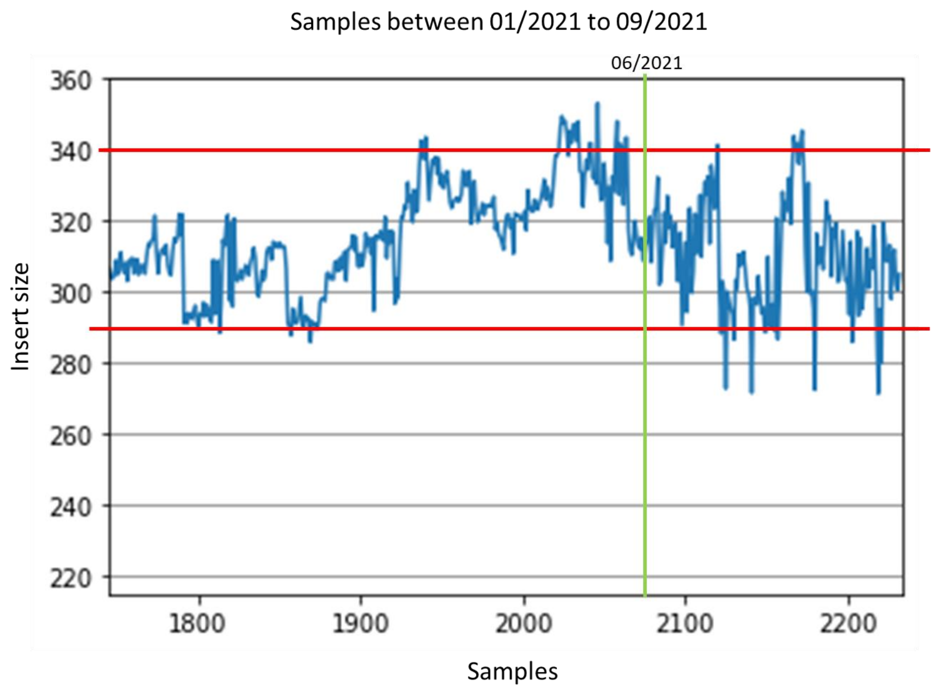


Figure 22. Insert size(y-axis) tendency of samples(x-axis) analysed between January 2021 and September 2021.

Coverage and genotypability

Other parameters that can be checked by exploiting the generated database are the coverage (e.g., %5X fraction of target covered by at least 5 reads) and genotypability (Percent PASS). During the analysis of more than 2000 samples over the years, we were able to establish that the optimal values for %5X coverage and genotyping are 99% and 96%, respectively.

In the last batch of 2021 generated with the kit Twist v.1.3, a decrease can be seen in in both %5X coverage (<99%) and in the genotypability (<96%) compared to the usual data (Figure 23).

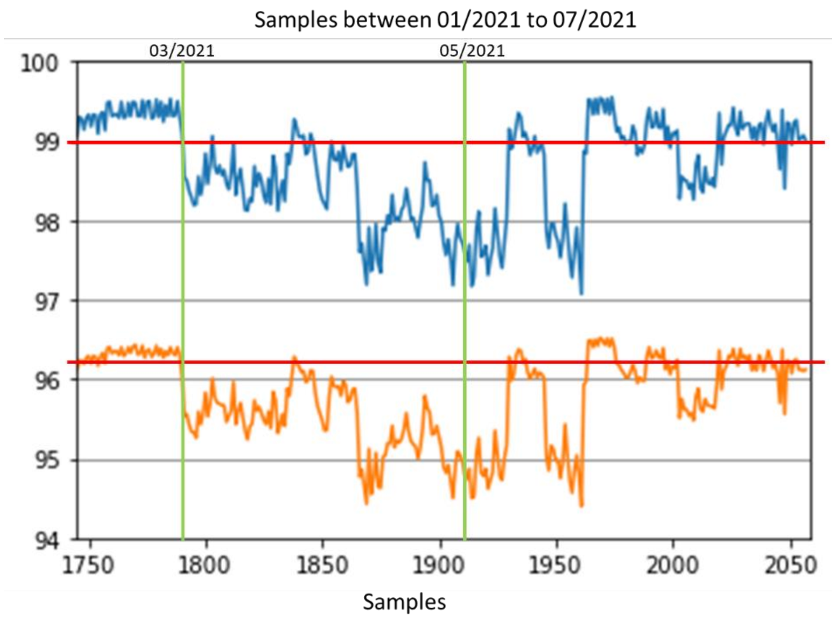


Figure 23. The tendency of percent 5X coverage(blue) and percentage of genotypability(orange) (y-axis) in samples analysed between January and July 2021(x-axis) calculated on design.

Furthermore, as shown in the following Figure 24, concerning the uniformity of coverage the findings showed also for this parameter a decrease to 94%, while the optimal values are supposed to be around 96-97%.

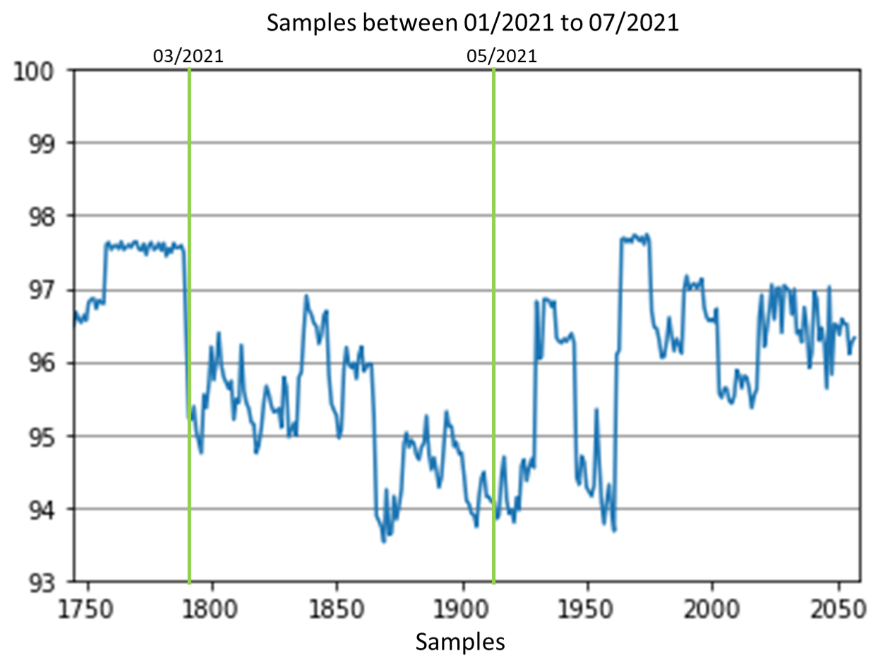


Figure 24. The tendency of percent uniformity of coverage(y-axis) in samples analysed in 2021(x-axis) calculated on design.

Additionally, as shown in Figure 25, some samples presented an increase in the fold80 value, suggesting that the uniformity of coverage decreased.

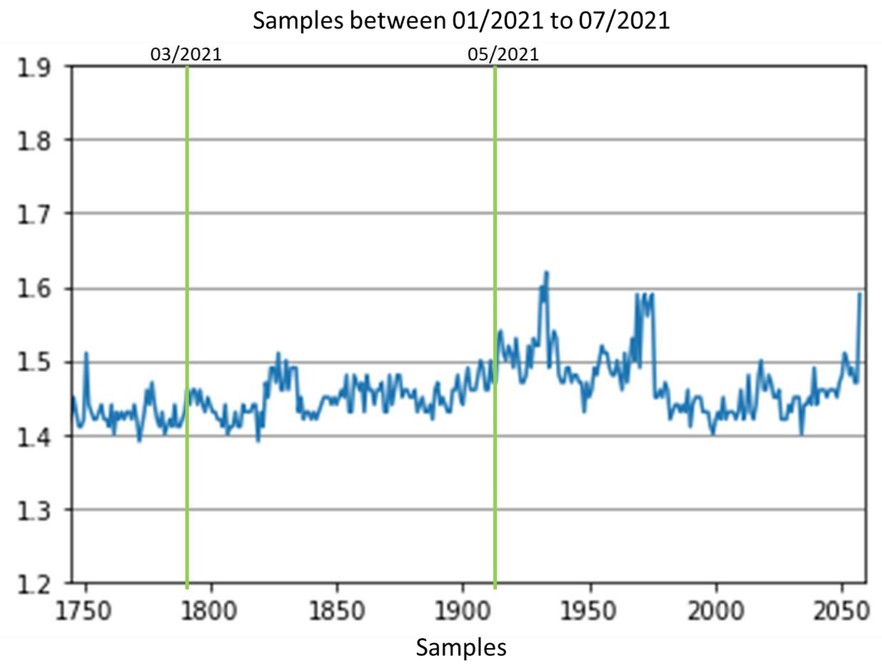


Figure 25. The tendency of fold80 penalty value(y-axis) in samples analysed in 2021(x-axis) calculated on design.

To investigate the problem in detail, a python script was created (Script 9 - graph.py - Appendix A). It takes as input all the sample coverage files keeping only the rows related to exons (Design_stats_all_design_SampleName.EXONS.tsv) and generates a heatmap based on %5X coverage, removing those regions that are 0 (not analysed at all) or 100 (optimal) in all samples (Figure 26).

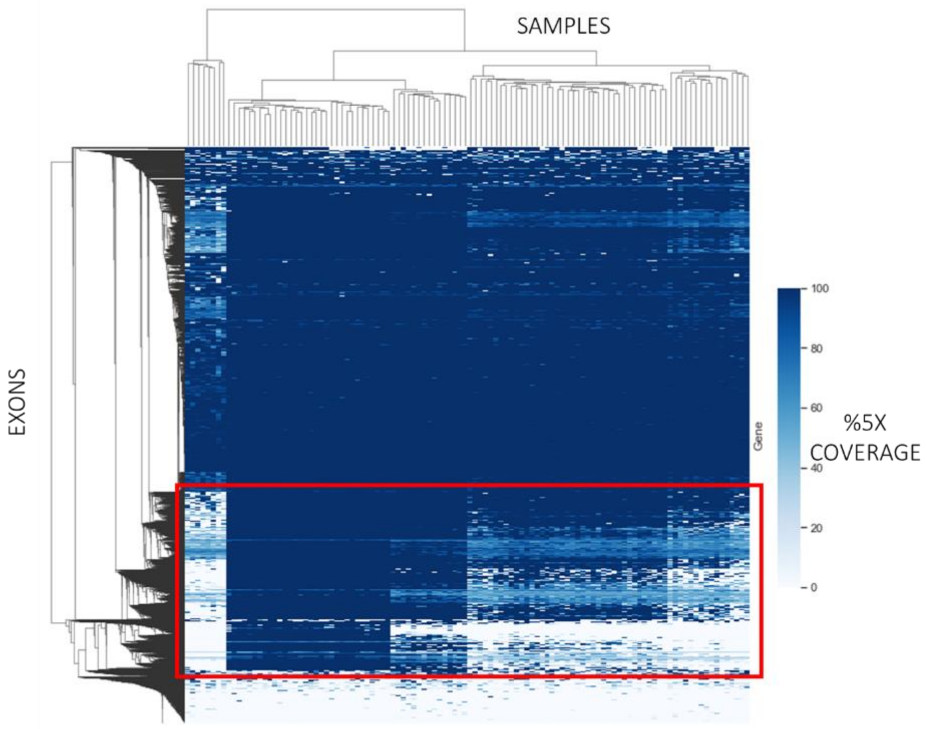


Figure 26. Heatmap with the %5X coverage for each exon(y-axis) and sample(x-axis).

Looking at Figure 26, it can be seen that certain exons showed a high percentage of 5X coverage in some samples, whereas they had no coverage at all in other ones. A deeper investigation of these variable exons exhibited that troublesome samples (low %5X coverage and genotyping) have all of these regions with poor or no coverage (Figure 27).

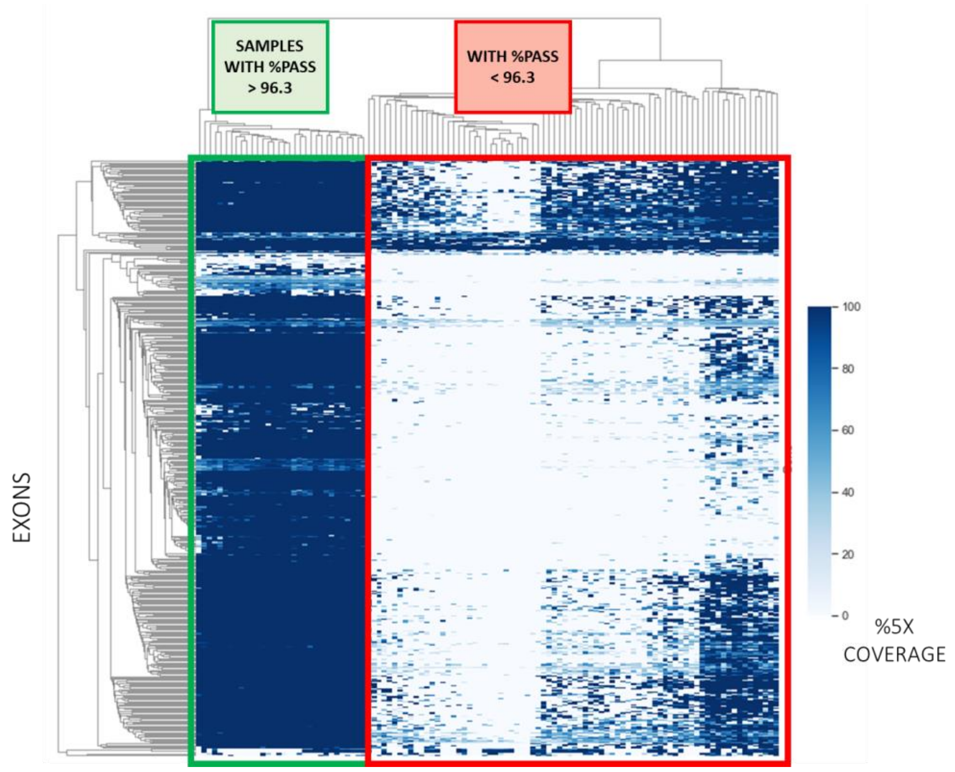


Figure 27. Heatmap with the %5X coverage for selected region(y-axis) and sample(x-axis).

The difficulties in performing the analysis of such regions could be caused by either issues during the library preparation or the capture. To better understand the work-flow issue in the laboratory and to determine if the problem happened during capture, an alternative capture kit (Roche) was tested on the 100 samples. Figure 28 illustrates the results of %5X and genotypability of samples that have been processed with the Roche capture kit.

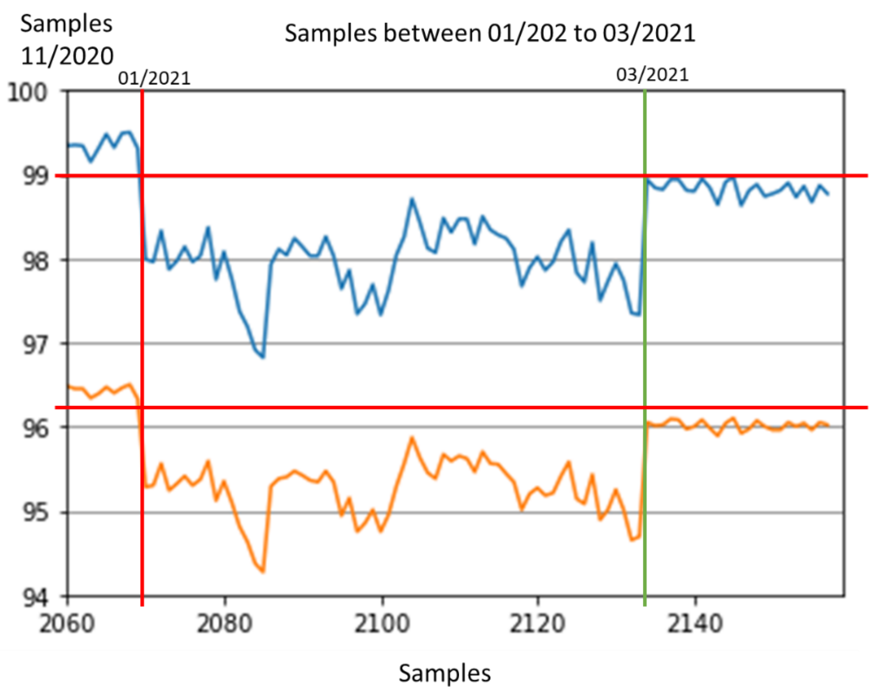


Figure 28. The tendency of percent 5X coverage(blue) and percentage of genotypability(orange)(y-axis) in samples analysed in 2021 with Roche(x-axis) calculated on design.

Figure 28 shows that the drop in quality metrics also persisted in the data that had been processed with the Roche kit, with some fluctuation. The same regions that revealed some troubles in Twist have been investigated in the Roche samples. Figure 29 shows that even in the Roche data most of the troublesome regions were lost in the samples with performance drop, which suggests that the issue might not be in the sample capture kit, but it might just be a shared step between both kits.

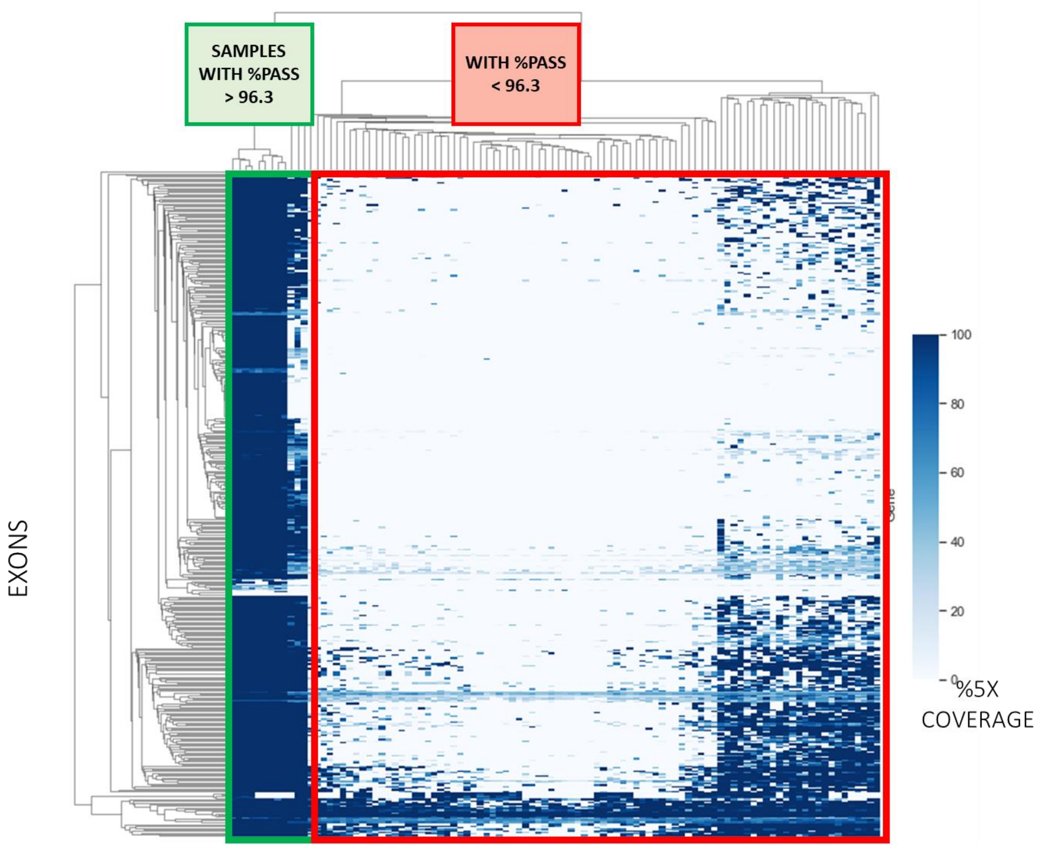


Figure 29. Heatmap with the %5X coverage for selected region(y-axis) and samples done with Roche(x-axis).

To make sure that the problem was not in the capture procedure, two captures made on the same day with Roche kit containing libraries from different days, have been compared (Figure 30).

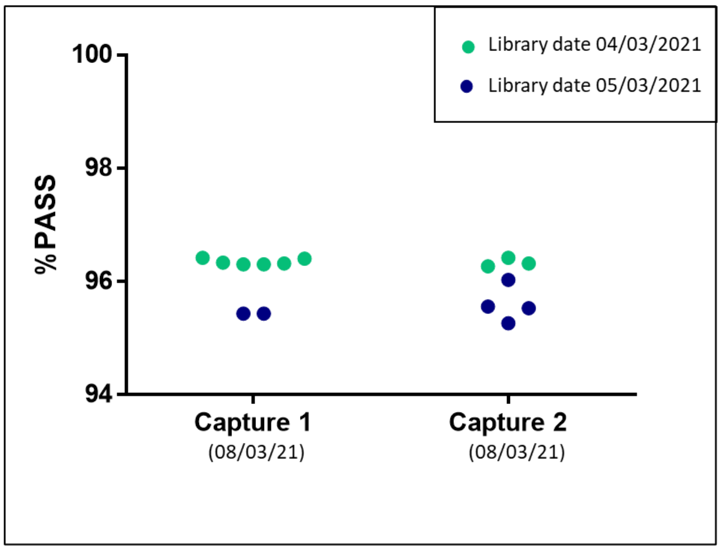


Figure 30. The plot shows the genotypability (%PASS) of libraries done in different days and in two different captures done with Roche kit.

The results from the Roche samples (Figures 29 and 30) suggested that the problem was not during the capture phase, hence the laboratory chose to adjust the library procedure. The regions were further studied to better understand their characteristics. They were found to be GC-rich regions. After discussing the outcome with Roche technical support, a loss of PCR polymerase efficiency was hypothesized and therefore it was agreed to modify the PCR polymerase mixture. Three replicates have been processed using different polymerase mixtures. Table 2 illustrates the results achieved with the different polymerase mixtures at the same mapped coverage (60X).

Polymerase buffer mix	**SAMPLE ID**	**%1X**	**%5X**	**%10X**	**%30X**	**%PASS**	**%PASS (RD>=10)**	**Fold80 Penalty Value**	**Uniformity of coverage (Pct > 0.2*mean)**
KAPA Hifi HotStart Ready Mix	1	99.39	98.53	97.63	89.3	95.76	94.76	1.56	95.56
	2	99.16	98.18	97.23	88.57	95.53	94.46	1.6	95.2
	3	99.1	98.06	97.03	88.14	95.43	94.26	1.61	94.98
KAPA HotStart Ready Mix +DMSO (5%)	4	99.55	99.04	98.36	89.81	96.19	95.49	1.56	96.33
	5	99.35	98.82	98.15	89.36	96.08	95.37	1.56	96.18
	6	99.34	98.82	98.14	89.32	96.09	95.37	1.56	96.20
KAPA HiFi HotStart Polymerase+ Hifi GC Buffer	7	99.54	99.06	98.27	86.35	96.23	95.40	1.70	95.98
	8	99.34	98.78	97.51	81.64	96.08	94.71	1.93	94.65
	9	99.35	98.85	97.96	85.06	96.11	95.17	1.75	95.58

Table 2. Table reporting the percentage of bases covered by at least 1, 5, 10, 20, 30 reads, the percentage of callable bases and the uniformity of coverage for data downsampled at 60X coverage.

The data treated with “KAPA HotStart Ready Mix +DMSO (5%)” and "KAPA Hifi HotStart Polymerase + Hifi GC Buffer" showed a higher performance in the number of bases covered by at least 5 reads, as well as in genotypability. The “KAPA HotStart Ready Mix +DMSO (5%)” displayed an improvement also in the Fold80 Penalty value and in the uniformity of coverage. These findings confirmed that the issue occurred during the PCR amplification phase in the library preparation.

Capture kit comparison

Various capture kits can be used in WES analysis. Every year new exome capture kits are available for sale, but it might be challenging to choose the optimal solution for our analysis. For example, in 2021 Twist Bioscience released on the market their new kit "Twist Target Enrichment v.2.0" in which some new features were incorporated, such as extra non-coding regions containing pathogenic variants. To test the performances of the new kit, 152 genomic libraries have been analysed with Twist Library Preparation EF 2.0: half of them have been enriched using the Twist Enrichment Target v.1.3 protocol, and the remaining 76 have been enriched applying the new Twist Target Enrichment v.2.0 protocol.

As shown in Table 3, the median number of total fragments sequenced was similar in both kit designs, with a lower %GC in Twist Target Enrichment v.2.0 presumably attributable to the inclusion of non-coding regions, due to lower GC content in introns compared to exons[46]–[48]. The two kits presented a comparable average insert size and duplicate rate.

Capture kit	# Fragments	%GC	Mean insert size	% Duplicates
v1.3	30,757,058	49.00	314.99	9.92
v2.0	28,653,622.	47.44	308.55	10.61

Table 3. Table reporting the number of sequenced fragments, the GC content of the genome, the average insert size, the percentage of duplicates.

The Design and RefSeq were then compared. The Tables 4A and 4B show that on both Design (Table 4A) and RefSeq (Table 4B) the Target Enrichment v.2.0 kit exhibits a higher on/near target and a lower off-target, suggesting more target-specific reads have been sequenced. In addition, Twist Target Enrichment v.2.0 demonstrates a greater uniformity of coverage and a lower Fold80 Penalty value which indicates a more consistent homogeneity of coverage.

A.

Capture kit	MEAN COVERAGE (X)	ON TARGET BASES	NEAR TARGET BASES	OFF TARGET BASES	Fold80 Penalty value	Uniformity of coverage (Pct > 0.2*mean)
v1.3	81.53	40.83	40.79	18.11	1.46	96.58
v2.0	78.55	41.79	50.30	7.91	1.34	97.83

B.

Capture kit	MEAN COVERAGE (X)	ON TARGET BASES	NEAR TARGET BASES	OFF TARGET BASES	Fold80 Penalty value	Uniformity of coverage (Pct > 0.2*mean)
v1.3	81.76	38.24	38.73	22.83	1.46	96.92
v2.0	77.94	39.15	44.92	15.93	1.35	96.47

Table 4. Table reporting the average mapped coverage, the percentage of on/near/off target bases, the fold80 penalty value and the uniformity of coverage on design(A) and on RefSeq(B).

On the Design (Table 5.A) with similar mean mapped coverage, samples processed with Twist Target Enrichment v.2.0 exhibit higher genotyping (%PASS), which indicates an improved coverage and mapping quality over the design regions. In contrast, on RefSeq (Table 5.B), those samples processed with Twist Target Enrichment v.2.0 showed a slightly lower genotyping (%PASS), presumably because of the removal of some RefSeq regions which were not clinically relevant.

A.

Capture kit	MEAN COVERAGE (X)	%1X	%5X	%10X	%20X	%30X	%PASS	%PASS (RD>=10)
v1.3	81.53	99.55	99.18	98.85	98.02	96.82	96.24	95.82
v2.0	78.55	99.07	98.97	98.87	98.51	97.46	97.16	97.06

B.

Capture kit	MEAN COVERAGE (X)	%1X	%5X	%10X	%20X	%30X	%PASS	%PASS (RD>=10)
v1.3	81.76	99.64	99.39	99.10	98.33	97.20	96.72	96.36
v2.0	77.94	98.89	98.41	98.17	97.62	96.42	96.10	95.87

Table 5. Table reporting the average mapped coverage, the percentage of bases covered by at least 1/5/10/20/30 reads the genotypability and the genotypability (%PASS) with at least 10 reads (%PASS (RD>=10)) on design(A) and on RefSeq(B).

The same outcomes can be graphically reviewed for an easier and more immediate overview. Figure 31 shows the %5X and %PASS performance on the design of the two kits. As previously discussed, Twist Target Enrichment v.2.0 shows greater genotypability and similar %5X values, furthermore, the chart also reveals a greater reproducibility of the results.

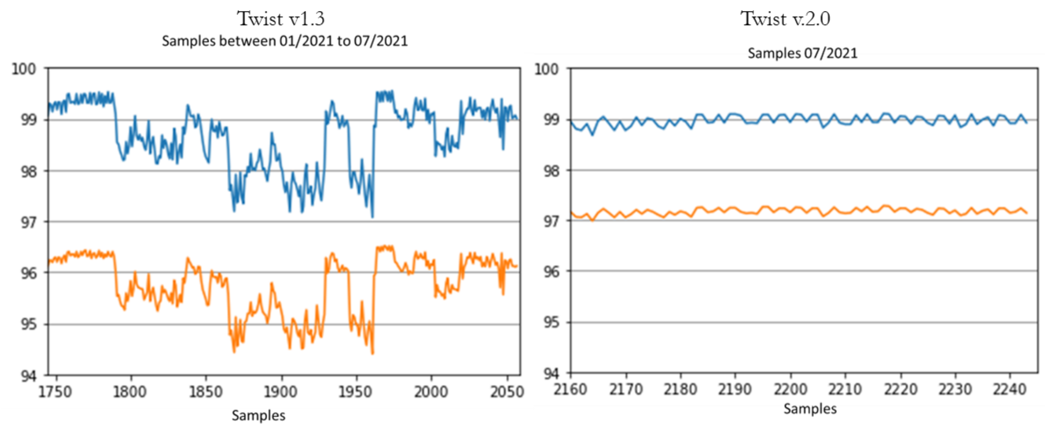


Figure 31. The tendency of percent 5X coverage(blue) and percentage of genotypability (orange)(y-axis) calculated on design in samples analysed with Twist Target Enrichment v.1.3 and with Twist Target Enrichment v.2.0.

Additionally, the percentage on target shows slightly higher values (41%); higher near target (50%), and lower off target (10%) in the newest version of the kit (Figure 32).

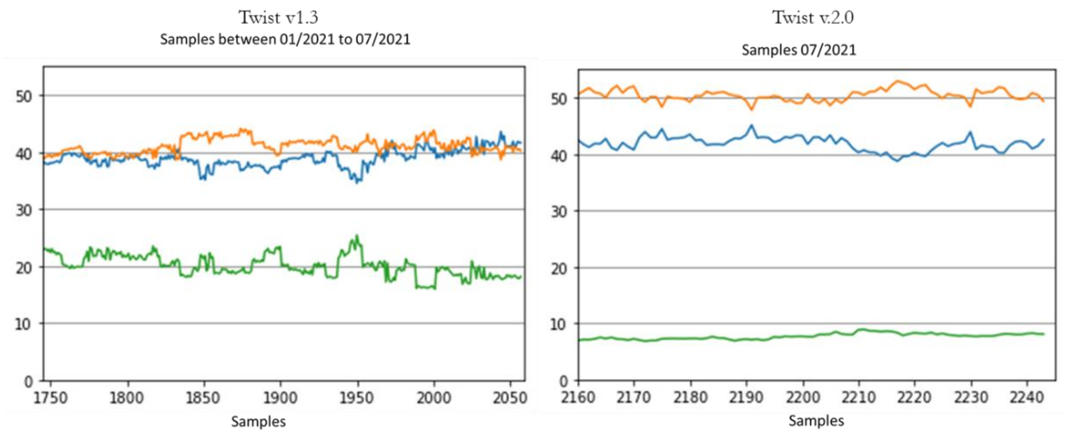


Figure 32. The tendency of on(blue)/near(orange)/off-target(green) (y-axis) calculated on design in samples analysed with Twist Target Enrichment v.1.3 and with Twist Target Enrichment v.2.0.

Finally, Figures 33 and 34 display the tendency of the Fold80 Penalty value and the uniformity of coverage over the design. As mentioned before, this confirms a lower Fold80 Penalty value and a higher and more reproducible uniformity of coverage.

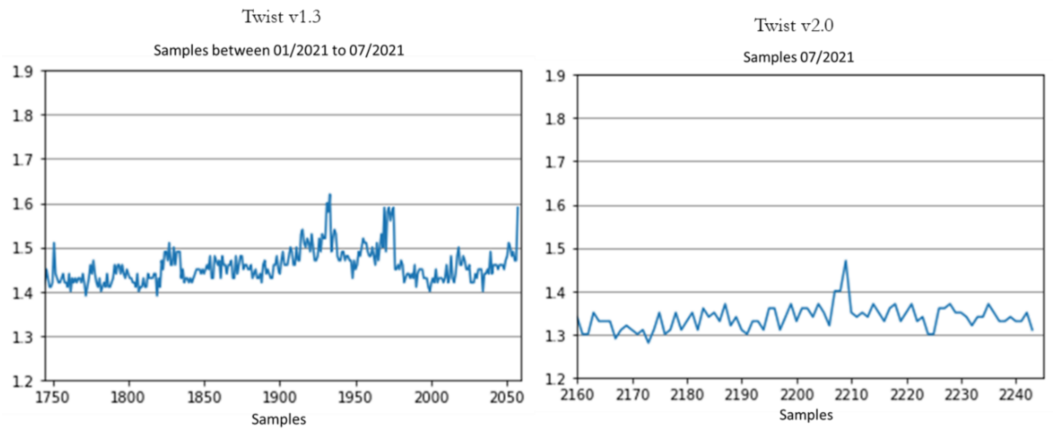


Figure 33. The tendency of Fold80 Penalty value calculated on design in samples analysed with Twist Target Enrichment v.1.3 and with Twist Target Enrichment v.2.0.

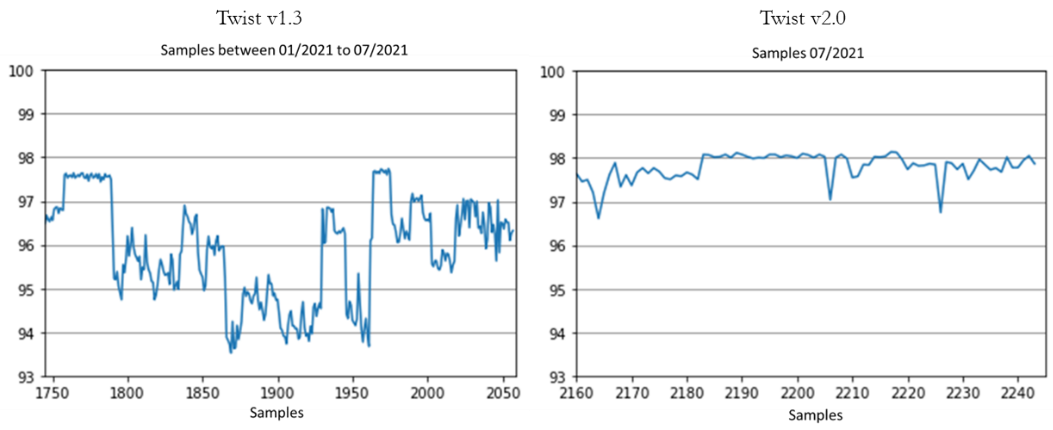


Figure 34. The tendency of uniformity of coverage (PCT > 0.2*mean) calculated on design in samples analysed with Twist Target Enrichment v.1.3 and with Twist Target Enrichment v.2.0.

THE ITALIAN VARIANT'S DATABASE

Variant frequency is an important parameter that assists physicians in the comprehension of the pathogenic nature of the variant under investigation. Currently, a population-based database focusing on the Italian population, is not available. For this purpose, the newly developed database includes a focused section to store the variants that were identified in the Italian population. The development of this database is important to determine the variants' frequency in Italian population in relation to other communities. In particular, it enables us to discriminate more accurately the variants reported as being rare in other databases, which are therefore frequent in the Italian population.

At the time of this writing, a total of 850,298 variants have been uploaded to the hg38-aligned data database. Out of these 279,155 were present in only one sample and thus were removed from the subsequent analysis because they could have resulted from errors in the sequencing or in the alignment, especially in repetitive regions. Of the remaining 571,143 total variants, 128,811 had frequency > 0.01, 62,905 > 0.05, and 442,332 had frequency ≤ 0.01. Of the 571,143 variants, 386,097 were in exons. Of these variants 11,562 were loss of function, whereas 226,454 were missense and 160 were splice region variants. All identified variants with frequency > 5%, > 1%, ≤ 1% have been compared to those in the gnomAD Exomes, ExAC, 1KgPhase3 databases.

The CSV file containing all the discovered variants uploaded in the Functional Genomics Variants Database (FGVDB) was downloaded from the database. The script "csvToVCF.sh" (Script 10 - csvToVCF.sh - Appendix A) was used to convert the CSV files to VCFs. Subsequently, the VCFs were loaded into Golden Helix VarSeq v.2.2.3[61] to be compared with gnomAD Exomes, ExAC, 1KgPhase3 databases.

Out of 571,143 variants found in the Italian Population database, the 13.37% (76,412) of them were missing in all three databases. 23.78%, 27.50%, 41.34% were not present respectively in gnomAD Exomes, ExAC, 1KgPhase3 in the European population (Table 6).

	Missing in gnomAD	Missing in ExAC	Missing in 1KgPhase3
All Italian' Variant	135,842	157,045	236,135
SNV	115,546	136,673	202,687
Indels	20,296	20,372	33,448

Table 6. The table reports the number of Italian' variants not present in gnomAD Exomes, ExAC, and 1KgPhase3.

Table 7 summarises the missing variants in the different databases considering the frequency. Of the total 128,811 variants with frequency > 1% in the developed database, 5,212 were missing in all the other databases. Of the total 62,905 variants with frequency > 5% in the developed database, 1,844 were missing in all the other databases. Of the 442,332 variants with frequency ≤ 1%, 71,200 were missing in all the other databases.

		Missing in gnomAD	Missing in ExAC	Missing in 1KgPhase3
All Italian' Variant	> 1%	30,042	29,942	12,887
	> 5%	9,654	9,592	4,669
	≤ 1%	105,800	127,103	223,248
SNV	> 1%	24,713	24,937	6,594
	> 5%	8,010	8,068	2,798
	≤ 1%	90,833	111,736	196,093
Indels	> 1%	5,329	5,005	6,293
	> 5%	1,644	1,524	1,871
	≤ 1%	14,967	15,367	27,155

Table 7. The table reports the number of Italian' variants with frequency > 1%, > 5%, ≤1% and not present in gnomAD Exomes, ExAC, and 1KgPhase3.

Afterwards, common variants in the Italian population were examined. Of the 123,599 variants with frequency > 1% and not missing in the other databases, 2,925 were rare (frequency ≤ 1%) in all the three databases. The 5.35%, 4.57% and 4.33% of the total variants with frequency > 1%, were respectively rare in gnomAD Exomes, ExAC, 1KgPhase3 in the European population. Of the 61,061 very common variants (frequency > 5%) and not missing in the other databases, 74 variants were rare (frequency ≤ 1%) in all the other databases (Table 8).

		0 ≤ gnomAD ≤ 1%	0 ≤ ExAC ≤ 1%	0 ≤ 1KgPhase3 ≤ 1%
All Italian' Variant	> 1%	6,608	5,651	5,479
	> 5%	746	656	197
SNV	> 1%	5,765	4,824	5,207
	> 5%	549	464	165
Indels	> 1%	843	827	272
	> 5%	197	192	32

Table 8. The table reports the number variants with frequency > 1%, > 5% in the Italian population and rare (frequency ≤ 1%) in gnomAD Exomes, ExAC, and 1KgPhase3.

Besides, rare variants in the Italian population were examined. Of the total variants 371,132 variants with frequency ≤ 1% and not missing in the other database, 25,164 were common (frequency >1%) and 11,336 were very common (frequency >5%) in all the three databases (Table 9).

	gnomAD		ExAC		1KgPhase3	
	> 1%	> 5%	> 1%	> 5%	> 1%	> 5%
All Italian' Variant (≤ 1%)	34,505	15,533	37,489	16,964	46,709	22,889
SNV (≤ 1%)	26,761	10,957	28,983	12,020	41,300	19,407
Indels (≤ 1%)	7,744	4,576	8,506	4,944	5,409	3,482

Table 9. The table reports the number of Italian' variants with frequency ≤ 1% and common (frequency > 1% and > 5%) in gnomAD Exomes, ExAC, and 1KgPhase3.

To better investigate variants in the created database, some examples of three classes of variants were reported:

- Common variant in the created database of the Italian population (frequency > 1% and > 5%) and rare in the other databases (frequency ≤ 1%)

- Variants classified in the created database of the Italian population and missing in the other databases
- Rare variant in the created database of the Italian population (frequency ≤ 1%) and common in the other databases (frequency > 1%)

Common variant in the created database of the Italian population (frequency > 1% and > 5%) and rare in the other databases (frequency ≤ 1%)

2,925 variants were found to be common in the Italian population (frequency > 1%) and rare in the Europe population in gnomAD Exomes, ExAC, and 1KgPhase3, 80 of them were very common (frequency > 5%). Of the total common variants, 121 variants were classified as pathogenic/likely pathogenic or damaging (DM/DM?) in ClinVar or HGMD. In particular, 3 clinically relevant SNPs were found to have frequency > 5% in the Italian population.

The first variant c.293-13C>G at position chr6:32039081 placed in the gene CYP21A2 (rs6467) was reported with a frequency of around 0.002 in all the three databases, whereas in the Italian database, it was reported with a frequency of 0.06. The variant was classified as pathogenic in ClinVar and damaging in HGMD for the "Adrenal hyperplasia" (Table 13).

POS	REF	ALT	AF gnomAD	AF ExAC	AF 1KG	AF FGVDB	ClinVar	HGMD
chr6: 32039081	C	G	0.002617	0.002788	0.001988	0.061439	Pathogenic	DM

Table 13. The table reporting the chromosome, the position, the reference and the alternative bases, the frequency of the variant in gnomAD Exomes, ExAC, and 1KgPhase3 and in the Italian database (FGVDB), the classification of the variant in ClinVar and in HGDM.

The second variant c.10552G>A at position chr11:92840745 placed in the gene FAT3 (rs10765565) was reported with a frequency of 0.002, 0.003, 0.001 in gnomAD Exomes, ExAC, and 1KgPhase3 respectively. In the Italian databases, instead, it was reported with a frequency of 0.06, as shown in Table 14. The variant was classified as damaging in HGMD for the "Primary hyperparathyroidism" and was not present in ClinVar.

POS	REF	ALT	AF gnomAD	AF ExAC	AF 1KG	AF FGVDB	ClinVar	HGMD
chr11: 92840745	G	A	0.002421	0.003558	0.000994	0.061968	-	DM?

Table 14. The table reporting the chromosome, the position, the reference and the alternative bases, the frequency of the variant in gnomAD Exomes, ExAC, and 1KgPhase3 and in the Italian database (FGVDB), the classification of the variant in ClinVar and in HGDM.

A third example variant c.1766G>C at position chrX:70925907 placed in the gene SLC7A3 (rs149447856) was reported with a frequency of 0.003, 0.003, 0.009 in gnomAD Exomes, ExAC, and 1KgPhase3 respectively. In the Italian databases, instead, it was reported with a frequency of 0.01, as shown in Table 14. The variant was classified as damaging in HGMD for the "Autism Spectrum Disorder" and was not present in ClinVar.

POS	**REF**	**ALT**	**AF gnomAD**	**AF ExAC**	**AF 1KG**	**AF FGVDB**	**ClinVar**	**HGMD**
chrX: 70925907	G	C	0.002810	0.003376	0.009138	0.010458	Likely Pathogenic	DM

Table 15. The table reporting the chromosome, the position, the reference and the alternative bases, the frequency of the variant in gnomAD Exomes, ExAC, and 1KgPhase3 and in the Italian database (FGVDB), the classification of the variant in ClinVar and in HGDM.

Variants classified in the created database of the Italian population and missing in the other databases

Of the 76,412 variants classified in the Italian population and missing in the cosmopolitan databases, 316 were classified as pathogenic/likely pathogenic or damaging (DM/DM?) in ClinVar or HGMD.

An example of these variants is the missense variant c.1270T>C at position chr21:34792308 placed in the gene RUNX1. It was reported with a frequency of 0.016759 but it was not present in the European population in gnomAD Exomes, ExAC, and 1KgPhase3.

The variant was classified with uncertain significance in ClinVar for the "Hereditary Thrombocytopenia" and damaging in HGMD for the "Platelet disorder" (Table 10).

POS	REF	ALT	AF gnomAD	AF ExAC	AF 1KG	AF FGVDB	ClinVar	HGMD
chr21: 34792308	A	G	-	-	-	0.016759	Uncertain Significance	DM

Table 10. The table reports the chromosome, the position, the reference and the alternative bases, the frequency of the variant in gnomAD Exomes, ExAC, and 1KgPhase3 and in the Italian database (FGVDB), the classification of the variant in ClinVar and in HGDM.

Another example is the loss-of-function variant c.4006C>T at position chr6:131587798 placed in the gene MED23. It was reported with a frequency of 0.000498, but it was not in the European population in gnomAD Exomes, ExAC, and 1KgPhase3. The variant was classified as pathogenic in ClinVar for the "Mental retardation" and damaging in HGMD for the "Intellectual disability" (Table 11).

POS	REF	ALT	AF gnomAD	AF ExAC	AF 1KG	AF FGVDB	ClinVar	HGMD
chr6: 131587798	G	A	-	-	-	0.000498	Pathogenic	DM

Table 11. The table reports the chromosome, the position, the reference and the alternative bases, the frequency of the variant in gnomAD Exomes, ExAC, and 1KgPhase3 and in the Italian database (FGVDB), the classification of the variant in ClinVar and in HGDM.

Rare variant in the created database of the Italian population (frequency ≤ 1%) and common in the other databases (frequency > 5%)

10,515 variants were found to be rare in the Italian population but very common (frequency >5%) in the Europe one in gnomAD Exomes, ExAC, and 1KgPhase3. Of these, 13 variants were classified as pathogenic or damaging in ClinVar or HGMD, below; an examples are reported below.

The variant c.6622+1G>A on the chr7:100779751 placed in the gene ZAN (rs1417635) was classified as probably damaging in HGMD for the "Autism spectrum disorder" and was not present in ClinVar. The analysed variant was reported with a frequency of 0.36, 0.41, 0.33 in gnomAD Exomes, ExAC, and 1KgPhase3 respectively, whereas it had a frequency of 0.007 in the Italian database (FGVDB). (Table 12).

POS	**REF**	**ALT**	**AF gnomAD**	**AF ExAC**	**AF 1KG**	**AF FGVDB**	**ClinVar**	**HGMD**
chr7: 100779751	G	A	0.356401	0.40918	0.332008	0.007007	-	DM?

Table 12. The table reports the chromosome, the position, the reference and the alternative bases, the frequency of the variant in gnomAD Exomes, ExAC, and 1KgPhase3 and in the Italian database (FGVDB); it also reports the classification of the variant in ClinVar and in HGDM.

DISCUSSION AND CONCLUSIONS

Variant analyses on WES data are nowadays used in the diagnosis of several diseases. These process produces a thousand of variants that must to be carefully filtered with specific techniques and prioritized with specific databases to easily identify the causative variant. In absence of accurate filtering, the identification of disease-causing variants could lead to focusing on wrong variants or could proceed with very long times that could be disadvantageous for the patient's status. The full exome processing workflow of the genomic DNA samples is subjected to a combination of laboratory and bioinformatics phases, each one having its specific quality control parameters. Several laboratory conditions are regulated in such a way that the right concentrations and purities of the sample are achieved. The desirable fragment length is also adjusted and selected during the sample preparation for sequencing. In addition, during WES analysis, several kits can be employed to capture samples together and select only regions of interests. Bioinformatics analyses are then conducted on the same samples and are used to verify different elements: the quality of the sequenced fragments, the mapping and coverage qualities, the overall number of variants and the number of variants for each of the categories in the trio family analysis (MIE, Recessive, Heterozygous, X -Linked, De Novo, Repeated). All these QC criteria are important for the precision of variant analysis identification. Relevant studies, such as gnomAD[26], ExAC[27], 1KgPhase3[27], report the creation of frequency databases based on different quality control parameters, e.g. duplication rate, coverage, read depth, variant QC and manual refinements. Nevertheless, all of them consider laboratory and bioinformatics parameters a separate way and only manual correlations is conducted. Such data separation in the commonly adopted process of variant calling hampers the capability to detect any potential issues. Furthermore, the absence of a data processing history does not provide an insight into the entire trend over time. Currently, there is no tool that takes into consideration all QC metrics for the automated comparisons of new and existing workflows.

For all these reasons, developing a database that aggregates all of the workflow information from the laboratory results to the bioinformatics might provide an invaluable tool. This would enable addressing potential problems and investigate new sample processing protocols. Hence, a new bioinformatics approach has been developed to easy access and compare data. A series of Python scripts have been implemented to relate a SQL database to a web page. Such scripts enable easy handling of the analysed data by uploading and re-examining the data on a simple webpage. Moreover, all relevant information is stored to prevent the loss of data processing history. To the best of my knowledge, the created database facilitates for the first time the automatic retrospective quality control of the data. The results shows the importance of computational inspection of essential metrics, e.g. the insert size and the genotypability, during the time. The two reported examples highlight the possibility of the implemented framework to recognize the lab-based root cause of sub-optimal results in the sample analysis. By pinpointing these problems, it was easier to investigate the data and make adjustments to the laboratory procedure. Due to the modifications, better results were evident in the generated chart analysis of the new data quality metrics. Furthermore, the literature does not report any tool to rapidly compare new and existing protocols. As reported in the case of the new Twist Capture Kit, the database enables an easy comparison of performance in terms of 5X coverage percentage and genotyping percentage, on/near/off target, fold 80 penalty value and coverage uniformity. The plot chart allows to efficiently and quickly assess the superior performance of the new Capture Kit, which suggests a possible advantage in the identification of variants due to an improved genotypability.

Additionally, in the variant analysis it is highly important to estimate the frequency of variants in the sample population for a better comprehension of the variation pathogenicity. The most well-known and used databases mostly include the frequency of several populations such as the Europeans, the Africans and the Americans. Many studies have been published on the relevance of considering population-specific human differences [49]–[60]. For example, Gudbjartsson et al. 2015 found several rare mutations in the Icelandic population associated with early onset atrial fibrillation, liver and thyroid-stimulating hormone level diseases. In contrast, variants which are common in the surveyed population and rare in commonly used databases do not constitute disease-causing variants. Not only Iceland, but also United Kingdom, Sweden, Finland start to create their biobank in relation to the fact that different populations shows distinct alleles distributions which is reflected in both a different relationship between sequence and trait and a diverse clinical impact.

Besides these evidences, a database of the Italian variant frequencies is needed to have a clearer insight of the effect of a variant in the population in Italy. The developed database also provided an invaluable reference for the variant frequency in the Italian population, not present in literature at the moment. In the results, the variants stored in the created database have been compared with the conventional databases used for prioritisation of variants (gnomAD Exomes, ExAC, and 1KgPhase3). The majority of the non-missing variants present similar frequency both in the European databases and in the one that has been created for the Italian people. The framework pinpoint that the 4-5% of common variants in the Italian population that are rare in the other databases, which indicates that the variant does not gives a disorder predisposition in Italian people. Importantly, 121 of these variants were classified as pathogenic/likely pathogenic or damaging in ClinVar or HGMD and in particular two of them have frequency even higher than 5% in the FGVDB. As shown in the results, the two variants c.293-13C>G and c.10552G>A were implicated in adrenal hyperplasia and primary hyperparathyroidism, respectively. The Functional Genomics Variants Database (FGVDB) allows to identify the frequency of these variants as non-pathogenic for the Italian population as they present frequency around the 6%. Moreover, the tool demonstrate a lack of a high proportion of variants in the commonly databases, 13.37%, of the overall variants identified in the Italian database, the majority of which were rare in Italian populations (frequency < 1%). The absence of variants in the used databases did not enable the comprehension of the pathogenicity of the variants and hence did not simplify the diagnosis in the Italian population. As shown in the results, the created database allows us to improve the discriminatory capacity of 76,412 variants in the Italian population that were missing in conventional database. Two examples of variants were reported. Variant c.1270T>C implicated in the Hereditary thrombocytopenia had frequency >1% demonstrating that the FGVDB allows to classify it as non-pathogenic. Whereas, the variant c.4006C>T was classified as pathogenic for intellectual disability and was reported with frequency≤1%. Thus, it is potentially suitable candidates to be the cause of the disorder, even if further investigation must be done. Finally, the tools allows an identification of rare variants in the Italian population that are instead common in the conventional databases, which might be a first indicator of disease-causing variants in

Italian people, however in this case more specific and complex statistical, clinical and functional analysis must be necessary.

In conclusion, in this thesis the implementation of a database linked to a website facilitates the visualisation of the analysed WES data, improving the identification of potential errors. It also simplifies the comparison of the new protocol analyses, as well as enabling a chronological record of the elaborated data. The realized framework to store all the identified variants and their frequency in the Italian population provides an essential resource for the categorisation of the detected variants. Although some limitation due to the small size of the cohort, this database is extremely important to dismiss variants from their potential pathogenicity, when the variants are common in the Italian population and rare in the European one. Furthermore, although more complex analysis are needed, in some cases it might provides an help in a first identification of rare variants that could constitute a genetic predisposition for a disease in the Italian population. Finally, it also allows doctors to have a better insight into the diffusion of variants in Italy, especially the ones that are absent in other databases.

REFERENCES

[1] C. DI Resta, G. Pipitone, P. Carrera, and M. Ferrari, "Current scenario of the genetic testing for rare neurological disorders exploiting next generation sequencing," *Neural Regen. Res.*, vol. 16, no. 3, pp. 475–481, 2021, doi: 10.4103/1673-5374.293135.

[2] H. Sun *et al.*, "Next-Generation Sequencing Technologies and Neurogenetic Diseases," *Life 2021, Vol. 11, Page 361*, vol. 11, no. 4, p. 361, Apr. 2021, doi: 10.3390/LIFE11040361.

[3] C. R. Newton, "Global Burden of Pediatric Neurological Disorders," *Semin. Pediatr. Neurol.*, vol. 27, pp. 10–15, Oct. 2018, doi: 10.1016/J.SPEN.2018.03.002.

[4] P. Hartman *et al.*, "Next generation sequencing for clinical diagnostics: Five year experience of an academic laboratory," *Molecular Genetics and Metabolism Reports*, vol. 19. 2019, doi: 10.1016/j.ymgmr.2019.100464.

[5] S. E. Soden *et al.*, "Effectiveness of exome and genome sequencing guided by acuity of illness for diagnosis of neurodevelopmental disorders," *Sci. Transl. Med.*, vol. 6, no. 265, pp. 1–14, 2014, doi: 10.1126/scitranslmed.3010076.

[6] M. Ilyas, "Next-Generation Sequencing in Diagnostic Pathology," *Pathobiology*, vol. 84, no. 6, pp. 292–305, 2018, doi: 10.1159/000480089.

[7] C. Di Resta and M. Ferrari, "Genetic testing in neurology exploiting next generation sequencing: State of art," *Neural Regen. Res.*, vol. 15, no. 2, pp. 265–266, 2020, doi: 10.4103/1673-5374.265554.

[8] J. Aaltio *et al.*, "Cost-effectiveness of whole-exome sequencing in progressive neurological disorders of children," *Eur. J. Paediatr. Neurol.*, vol. 36, 2021, doi: 10.1016/j.ejpn.2021.11.006.

[9] K. Tsiplova *et al.*, "A microcosting and cost-consequence analysis of clinical genomic testing strategies in autism spectrum disorder," *Genet. Med.*, vol. 19, no. 11, pp. 1268–1275, 2017, doi: 10.1038/gim.2017.47.

[10] X. Du *et al.*, "Genetic Diagnostic Evaluation of Trio-Based Whole Exome Sequencing Among Children With Diagnosed or Suspected Autism Spectrum Disorder," *Front. Genet.*, vol. 9, p. 594, Nov. 2018, doi: 10.3389/FGENE.2018.00594/BIBTEX.

[11] L. Zhang *et al.*, "Pathogenic variants identified by whole-exome sequencing in 43

patients with epilepsy," *Hum. Genomics*, vol. 14, no. 1, pp. 1–8, Dec. 2020, doi: 10.1186/S40246-020-00294-0/TABLES/4.

[12] W. Tong *et al.*, "Whole-exome Sequencing Helps the Diagnosis and Treatment in Children with Neurodevelopmental Delay Accompanied Unexplained Dyspnea," *Sci. Reports 2018 81*, vol. 8, no. 1, pp. 1–9, Mar. 2018, doi: 10.1038/s41598-018-23503-2.

[13] B. Iadarola *et al.*, "Shedding light on dark genes: enhanced targeted resequencing by optimizing the combination of enrichment technology and DNA fragment length," *Sci. Reports 2020 101*, vol. 10, no. 1, pp. 1–11, Jun. 2020, doi: 10.1038/s41598-020-66331-z.

[14] S. Naz and A. Fatima, "Amplification of GC-rich DNA for high-throughput family-based genetic studies," *Mol. Biotechnol.*, vol. 53, no. 3, pp. 345–350, Mar. 2013, doi: 10.1007/S12033-012-9559-Y/FIGURES/3.

[15] H. Li, J. Ruan, and R. Durbin, "Mapping short DNA sequencing reads and calling variants using mapping quality scores," *Genome Res.*, vol. 18, no. 11, pp. 1851–1858, 2008, doi: 10.1101/gr.078212.108.

[16] M. J. Landrum *et al.*, "ClinVar: improving access to variant interpretations and supporting evidence," *Nucleic Acids Res.*, vol. 46, no. D1, pp. D1062–D1067, Jan. 2018, doi: 10.1093/NAR/GKX1153.

[17] M. J. Landrum *et al.*, "ClinVar: Public archive of interpretations of clinically relevant variants," *Nucleic Acids Res.*, vol. 44, no. D1, pp. D862–D868, 2016, doi: 10.1093/NAR/GKV1222.

[18] M. J. Landrum *et al.*, "ClinVar: Public archive of relationships among sequence variation and human phenotype," *Nucleic Acids Res.*, vol. 42, no. D1, Jan. 2014, doi: 10.1093/NAR/GKT1113.

[19] D. N. Cooper, E. V. Ball, and M. Krawczak, "The human gene mutation database," *Nucleic Acids Res.*, vol. 26, no. 1, pp. 285–287, Jan. 1998, doi: 10.1093/NAR/26.1.285.

[20] P. D. Stenson *et al.*, "The Human Gene Mutation Database: towards a comprehensive repository of inherited mutation data for medical research, genetic diagnosis and next-generation sequencing studies," *Hum. Genet.*, vol. 136, no. 6, pp. 665–677, Jun. 2017, doi: 10.1007/s00439-017-1779-6.

[21] J. S. Amberger, C. A. Bocchini, F. Schiettecatte, A. F. Scott, and A. Hamosh, "OMIM.org: Online Mendelian Inheritance in Man (OMIM®), an online catalog of

human genes and genetic disorders," *Nucleic Acids Res.*, vol. 43, no. D1, pp. D789–D798, Jan. 2015, doi: 10.1093/NAR/GKU1205.

[22] J. S. Amberger, C. A. Bocchini, A. F. Scott, and A. Hamosh, "OMIM.org: leveraging knowledge across phenotype–gene relationships," *Nucleic Acids Res.*, vol. 47, no. D1, pp. D1038–D1043, Jan. 2019, doi: 10.1093/NAR/GKY1151.

[23] K. D. Pruitt, T. Tatusova, and D. R. Maglott, "NCBI reference sequences (RefSeq): A curated non-redundant sequence database of genomes, transcripts and proteins," *Nucleic Acids Res.*, vol. 35, no. SUPPL. 1, Jan. 2007, doi: 10.1093/NAR/GKL842.

[24] N. A. O'Leary *et al.*, "Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation," *Nucleic Acids Res.*, vol. 44, no. D1, pp. D733–D745, Jan. 2016, doi: 10.1093/NAR/GKV1189.

[25] T. A. Manolio *et al.*, "Finding the missing heritability of complex diseases," *Nat. 2009 4617265*, vol. 461, no. 7265, pp. 747–753, Oct. 2009, doi: 10.1038/nature08494.

[26] K. J. Karczewski *et al.*, "The mutational constraint spectrum quantified from variation in 141,456 humans," *Nat. 2020 5817809*, vol. 581, no. 7809, pp. 434–443, May 2020, doi: 10.1038/s41586-020-2308-7.

[27] K. J. Karczewski *et al.*, "The ExAC browser: Displaying reference data information from over 60 000 exomes," *Nucleic Acids Res.*, vol. 45, no. D1, pp. D840–D845, 2017, doi: 10.1093/nar/gkw971.

[28] S. Andrews, "FastQC: A Quality Control Tool for High Throughput Sequence Data." 2010, [Online]. Available: http://www.bioinformatics.babraham.ac.uk/projects/fastqc/.

[29] S. Chen, Y. Zhou, Y. Chen, and J. Gu, "Fastp: An ultra-fast all-in-one FASTQ preprocessor," *Bioinformatics*, vol. 34, no. 17, pp. i884–i890, 2018, doi: 10.1093/bioinformatics/bty560.

[30] H. Li and R. Durbin, "Fast and accurate short read alignment with Burrows-Wheeler transform," *Bioinformatics*, vol. 25, no. 14, pp. 1754–1760, Jul. 2009, doi: 10.1093/bioinformatics/btp324.

[31] G. Jun, M. K. Wing, G. R. Abecasis, and H. M. Kang, "An efficient and scalable analysis framework for variant extraction and refinement from population-scale DNA sequence data," *Genome Res.*, vol. 25, no. 6, pp. 918–925, 2015, doi: 10.1101/gr.176552.114.

[32] Broad Institute, "Picard toolkit," 2018. http://broadinstitute.github.io/picard/.

[33] M. A. Depristo *et al.*, "A framework for variation discovery and genotyping using next-generation DNA sequencing data," *Nat. Genet.*, vol. 43, no. 5, pp. 491–501, 2011, doi: 10.1038/ng.806.

[34] G. A. Van der Auwera *et al.*, "From fastQ data to high-confidence variant calls: The genome analysis toolkit best practices pipeline," *Current Protocols in Bioinformatics*, no. SUPL.43. 2013, doi: 10.1002/0471250953.bi1110s43.

[35] R. Poplin *et al.*, "Scaling accurate genetic variant discovery to tens of thousands of samples," *bioRxiv*, pp. 1–22, 2017, doi: 10.1101/201178.

[36] Van der Auwera GA & O'Connor BD, *Genomics in the Cloud: Using Docker, GATK, and WDL in Terra*, 1st Editio. O'Reilly Media, 2020.

[37] F. L. Van Rossum, Guido and Drake, *Python 3 Reference Manual*. CreateSpace, 2009.

[38] PostgreSQL Global Development Group, "PostgreSQL." 1996, [Online]. Available: www.postgresql.org.

[39] The pgAdmin Development Team, "pgAdmin." https://www.pgadmin.org/.

[40] M. Grinberg, *Flask web development: developing web applications with python*. O'Reilly Media, Inc., 2018.

[41] W. McKinney, "Data Structures for Statistical Computing in Python," *Proc. 9th Python Sci. Conf.*, vol. 1, no. Scipy, pp. 56–61, 2010, doi: 10.25080/majora-92bf1922-00a.

[42] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 90–95, 2007.

[43] B. Pan *et al.*, "Similarities and differences between variants called with human reference genome HG19 or HG38," *BMC Bioinformatics*, vol. 20, no. Suppl 2, 2019, doi: 10.1186/s12859-019-2620-0.

[44] H. Li *et al.*, "Exome variant discrepancies due to reference-genome differences," *Am. J. Hum. Genet.*, vol. 108, no. 7, pp. 1239–1250, 2021, doi: 10.1016/j.ajhg.2021.05.011.

[45] Y. Guo, Y. Dai, H. Yu, S. Zhao, D. C. Samuels, and Y. Shyr, "Improvements and impacts of GRCh38 human reference on high throughput sequencing data analysis," *Genomics*, vol. 109, no. 2, pp. 83–90, 2017, doi: 10.1016/j.ygeno.2017.01.005.

[46] M. Amit *et al.*, "Differential GC Content between Exons and Introns Establishes

Distinct Strategies of Splice-Site Recognition," *Cell Rep.*, vol. 1, no. 5, pp. 543–556, 2012, doi: 10.1016/j.celrep.2012.03.013.

[47] A. Eyre-Walker and L. D. Hurst, "The evolution of isochores," *Nat. Rev. Genet.*, vol. 2, no. 7, pp. 549–555, 2001, doi: 10.1038/35080577.

[48] L. Zhu, Y. Zhang, W. Zhang, S. Yang, J. Q. Chen, and D. Tian, "Patterns of exon-intron architecture variation of genes in eukaryotic genomes," *BMC Genomics*, vol. 10, pp. 1–12, 2009, doi: 10.1186/1471-2164-10-47.

[49] S. Sanna *et al.*, "Variants within the immunoregulatory CBLB gene are associated with multiple sclerosis," *Nat. Genet.*, vol. 42, no. 6, pp. 495–497, 2010, doi: 10.1038/ng.584.

[50] E. T. Lim *et al.*, "Distribution and Medical Impact of Loss-of-Function Variants in the Finnish Founder Population," *PLoS Genet.*, vol. 10, no. 7, 2014, doi: 10.1371/journal.pgen.1004494.

[51] D. F. Gudbjartsson *et al.*, "Sequence variants from whole genome sequencing a large group of Icelanders," *Sci. Data*, vol. 2, pp. 1–11, 2015, doi: 10.1038/sdata.2015.11.

[52] A. B. V Halldorsson, H. P. Eggertsson, and K. H. S. Moore, "Title : The sequences of 150 , 119 genomes in the UK biobank," 2021.

[53] P. Francalacci *et al.*, "Peopling of three Mediterranean islands (Corsica, Sardinia, and Sicily) inferred by Y-chromosome biallelic variability," *Am. J. Phys. Anthropol.*, vol. 121, no. 3, pp. 270–279, 2003, doi: 10.1002/ajpa.10265.

[54] C. Sidore *et al.*, "Genome sequencing elucidates Sardinian genetic architecture and augments association analyses for lipid and blood inflammatory markers," *Nat. Genet.*, vol. 47, no. 11, pp. 1272–1281, 2015, doi: 10.1038/ng.3368.

[55] I. Moltke *et al.*, "A common Greenlandic TBC1D4 variant confers muscle insulin resistance and type 2 diabetes," *Nature*, vol. 512, no. 7513, pp. 190–193, 2014, doi: 10.1038/nature13425.

[56] M. Pala *et al.*, "Population- and individual-specific regulatory variation in Sardinia," *Nat. Genet.*, vol. 49, no. 5, pp. 700–707, 2017, doi: 10.1038/ng.3840.

[57] M. Zoledziewska *et al.*, "Height-reducing variants and selection for short stature in Sardinia," *Nat. Genet.*, vol. 47, no. 11, pp. 1352–1356, 2015, doi: 10.1038/ng.3403.

[58] J. D. Backman *et al.*, "Exome sequencing and analysis of 454,787 UK Biobank

participants," *Nature*, vol. 599, no. November, 2021, doi: 10.1038/s41586-021-04103-z.

[59] H. Jónsson *et al.*, "Data Descriptor: Whole genome characterization of sequence diversity of 15,220 Icelanders," *Sci. Data*, vol. 4, pp. 1–9, 2017, doi: 10.1038/sdata.2017.115.

[60] D. F. Gudbjartsson *et al.*, "Large-scale whole-genome sequencing of the Icelandic population," *Nat. Genet.*, vol. 47, no. 5, pp. 435–444, 2015, doi: 10.1038/ng.3247.

[61] M. Golden Helix, Inc., Bozeman, "VarSeq v.2.2.3." [Online]. Available: www.goldenhelix.com.

APPENDIX

APPENDIX A – SCRIPTS

Script 1 - calculate_unif_coverage_PCT.sh. The script calculates the uniformity of coverage.

```
#!/bin/bash


# @author: Denise Lavezzari

# The script calculates the uniformity of coverage defined as: "The percentage of
targeted base positions in which the read depth is greater than 0.2 times the mean
region target coverage depth."


tsv=$1; # coverage file from callable loci analysis

target=$2; # PER_BASE_COVERAGE.txt from Picard CollectHsMetrics

kit=$3; # used kit


if [[ -f $tsv && -f $target ]]; then

        # Retrieve the coverage

        t=$(head -1 $tsv | cut -f4);


        # Calculate the threshold

        t2=$(echo | awk '{print """$t"""*0.2}');

        threshold=$(printf "%.0f\n" $t2);

        echo "Threshold: $threshold" >> unif_of_coverage_$kit.txt;


        # Retrieve the total number of targeted base positions

        total=`wc -l $target | cut -d ' ' -f1`;

        echo "Total #of bases: $total" >> unif_of_coverage_$kit.txt ;

        bases=`awk -v t=$threshold '$4>t' $target | wc -l`;

        echo "Total bases: $bases" >> unif_of_coverage_$kit.txt;
        # Calculate the uniformity of coverage

        pct=`printf "%.2f\n" $(echo | awk "{print ($bases/$total)*100}")`;

        echo "Uniformity of coverage (Pct > 0.2*mean): $pct" >>
unif_of_coverage_$kit.txt;

else

        echo "

        SYNTAX:

        - $( basename $0 ) <tsv file with coverage> <PER_BASE_COVERAGE.txt
from collectHSMetrics> <kit>

        ";

fi
```

Script 2 - db_creation.py. Script to create the database

```
#!/usr/bin/env python3
# hg19 database creation script
# @author: Alessandro Facconi
# @modified by: Denise Lavezzari

import psycopg2

conn = psycopg2.connect(host="localhost", database="FGVD_hg19",
user="postgres")
with conn:
        with conn.cursor() as cur:

                # Sample table creation
                cur.execute("DROP TABLE IF EXISTS Sample CASCADE")
                cur.execute("""
                        CREATE TABLE Sample
                        (
                                cgf_ID VARCHAR NOT NULL UNIQUE,
                                sample_ID VARCHAR NOT NULL UNIQUE,
                                identity VARCHAR NOT NULL,
                                arrival_date DATE NOT NULL,
                                urgency VARCHAR,
                                hpo VARCHAR,
                                affected BOOLEAN,
                                sex CHAR(1) NOT NULL
                                        CHECK (sex = 'F' OR sex = 'M' OR sex =
'f' OR sex = 'm'),
                                age VARCHAR,
                                ethnicity VARCHAR,
                                delivery date,
                                sample_notes VARCHAR,
                                father VARCHAR,
                                mother VARCHAR,
                                siblings VARCHAR,
                                project VARCHAR,
                                solved BOOLEAN,
                                causative_variant VARCHAR,
                                PRIMARY KEY(cgf_ID, sample_ID)
                        )
                """)

                # QC table creation
                cur.execute("DROP TABLE IF EXISTS QC CASCADE")
                cur.execute("""
                        CREATE TABLE QC
                        (
                                cgf_ID VARCHAR
                                        REFERENCES Sample(cgf_ID)
                                        ON UPDATE CASCADE
                                        ON DELETE CASCADE,
                                sample_ID VARCHAR
                            REFERENCES Sample(sample_ID)
                            ON UPDATE CASCADE
                            ON DELETE CASCADE,
                                qc_extraction_type VARCHAR,
                                qc_buffer_extraction_kit VARCHAR NOT NULL,
                                qc_nanodrop_ng_ul DOUBLE PRECISION,
                                qc_nanodrop_260_280 DOUBLE PRECISION,
                                qc_nanodrop_260_230 DOUBLE PRECISION,
                                qc_tape_din DOUBLE PRECISION,
                                qc_tape_ng_ul DOUBLE PRECISION,
                                qc_purificate DOUBLE PRECISION,
                                qc_qubit_ng_ul DOUBLE PRECISION,
                                PRIMARY KEY(cgf_ID, sample_ID)
                        )
                """)


# Library table creation
                cur.execute("DROP TABLE IF EXISTS Library CASCADE")
                cur.execute("""
                CREATE TABLE Library
                (
                        library_kit VARCHAR NOT NULL,
                                cgf_ID VARCHAR NOT NULL
                                        REFERENCES Sample(cgf_ID)
                            ON UPDATE CASCADE
                            ON DELETE CASCADE,
                                sample_ID VARCHAR
                            REFERENCES Sample(sample_ID)
                            ON UPDATE CASCADE
                            ON DELETE CASCADE,
                        sample_concentration DOUBLE PRECISION NOT NULL,
                        lib_input_ng DOUBLE PRECISION NOT NULL,
                        lib_sample_dilution DOUBLE PRECISION,
                        lib_input_ul DOUBLE PRECISION NOT NULL,
                        lib_h2o_ul DOUBLE PRECISION NOT NULL,
                        lib_fragmentation_time TIME NOT NULL,
                        lib_cleanup DOUBLE PRECISION NOT NULL,
                        lib_size_selection DOUBLE PRECISION NOT NULL,
                        lib_used_index VARCHAR NOT NULL,
                        lib_pcr_cycle DOUBLE PRECISION NOT NULL,
                        lib_cleanup_post_pcr DOUBLE PRECISION NOT NULL,
                        lib_qubit1_ng_ul DOUBLE PRECISION NOT NULL,
                        lib_qubit2_ng_ul DOUBLE PRECISION NOT NULL,
                        lib_qubit3_ng_ul DOUBLE PRECISION,
                        lib_qubit_mean DOUBLE PRECISION NOT NULL,
                        lib_tape_from DOUBLE PRECISION,
                lib_tape_to DOUBLE PRECISION,
                lib_tape_average_size DOUBLE PRECISION,
                lib_tape_concentration DOUBLE PRECISION,
                lib_tape_molarity DOUBLE PRECISION,
                lib_date DATE,
                lib_location VARCHAR,
                lib_operator VARCHAR,
                lib_notes VARCHAR,
                        PRIMARY KEY(library_kit, cgf_ID, sample_ID)
            )
        """)

            # Capture table creation
            cur.execute("DROP TABLE IF EXISTS Capture_Info
CASCADE")
            cur.execute("""
                  CREATE TABLE Capture_Info
                  (
                        capture_kit VARCHAR NOT NULL,
                        capture_ID VARCHAR UNIQUE,
                        n_capture VARCHAR UNIQUE,
                        capt_pos_from DOUBLE PRECISION NOT
NULL,
                        capt_pos_to DOUBLE PRECISION NOT
NULL,
                        capt_average_size DOUBLE PRECISION NOT
NULL,
                        capt_concentration DOUBLE PRECISION NOT
NULL,
                        capt_molarity DOUBLE PRECISION,
                        capt_pmol DOUBLE PRECISION,
                        capt_qubit_ng_ul DOUBLE PRECISION,
                        capt_date DATE,
                        capt_location VARCHAR,
                        capt_operator VARCHAR,
                        capt_notes VARCHAR,
                        PRIMARY KEY (capture_ID, n_capture)
                  )
            """)

            # Capture Samples table creation
            cur.execute("DROP TABLE IF EXISTS Capture_Samples
CASCADE")
            cur.execute("""
                  CREATE TABLE Capture_Samples
                  (
                        cgf_ID VARCHAR NOT NULL
                    REFERENCES Sample(cgf_ID)
                    ON UPDATE CASCADE
                    ON DELETE CASCADE,
                        sample_ID VARCHAR NOT NULL
                    REFERENCES Sample(sample_ID)
                    ON UPDATE CASCADE
                    ON DELETE CASCADE,
                        capture_ID VARCHAR NOT NULL
                    REFERENCES Capture_Info(capture_ID)
                    ON UPDATE CASCADE
                    ON DELETE CASCADE,
                        PRIMARY KEY(cgf_ID, sample_ID, capture_ID)
                )
            """)

            # Sequencing table creation
            cur.execute("DROP TABLE IF EXISTS Sequencing CASCADE")
            cur.execute("""
                CREATE TABLE Sequencing
                (

                        cgf_ID VARCHAR NOT NULL
                    REFERENCES Sample(cgf_ID)
                    ON UPDATE CASCADE
                    ON DELETE CASCADE,
                        sample_ID VARCHAR NOT NULL
                            REFERENCES Sample(sample_ID)
                    ON UPDATE CASCADE
                    ON DELETE CASCADE,
                        sequencer VARCHAR NOT NULL,
                        seq_date DATE NOT NULL,
                        seq_location VARCHAR NOT NULL,
                        seq_flow_cell VARCHAR NOT NULL,
                        seq_millions_fragments DOUBLE PRECISION,
                        PRIMARY KEY (sample_ID, sequencer, seq_date)
                )
            """)

            # Statistics table creation
            cur.execute("DROP TABLE IF EXISTS Statistics CASCADE")
            cur.execute("""
                CREATE TABLE Statistics
                (
                        sample_ID VARCHAR PRIMARY KEY
                    REFERENCES Sample(sample_ID)
                    ON UPDATE CASCADE
                    ON DELETE CASCADE,
                        n_fragments DOUBLE PRECISION NOT
NULL,
                        gc DOUBLE PRECISION NOT NULL,
                        insert_size DOUBLE PRECISION NOT NULL,
                        n_map_dedup DOUBLE PRECISION NOT
NULL,
                        duplicates DOUBLE PRECISION NOT NULL
                )
            """)

            # RefSeq_Design_Statistics table creation
            cur.execute("DROP TABLE IF EXISTS RefSeq_Design_Statistics
CASCADE")
            cur.execute("""
                CREATE TABLE RefSeq_Design_Statistics
                (
                        sample_ID VARCHAR NOT NULL
                  REFERENCES Sample(sample_ID)
                  ON UPDATE CASCADE
                  ON DELETE CASCADE,
                        refseq_design VARCHAR NOT NULL,
                        kit_length DOUBLE PRECISION NOT NULL,
                        mean_coverage DOUBLE PRECISION NOT
NULL,
                        Percent_oneX DOUBLE PRECISION NOT
NULL,
                        Percent_fiveX DOUBLE PRECISION NOT
NULL,
                        Percent_tenX DOUBLE PRECISION NOT
NULL,
                        Percent_twentyX DOUBLE PRECISION NOT
NULL,
                        Percent_thirtyX DOUBLE PRECISION NOT
NULL,
                        Percent_pass DOUBLE PRECISION NOT
NULL,
                        Percent_pass_dp10 DOUBLE PRECISION NOT
NULL,
                        Percent_on_target DOUBLE PRECISION,
                        Percent_near_target DOUBLE PRECISION,
                        Percent_off_target DOUBLE PRECISION,
                        fold_enrich DOUBLE PRECISION,
                        fold80 DOUBLE PRECISION,
                        Percent_coverage_uniformity DOUBLE
PRECISION,
                        PRIMARY KEY (sample_ID, refseq_design)
                )
            """)

            # Exons_Statistics table creation
            cur.execute("DROP TABLE IF EXISTS Exons_Statistics
CASCADE")
            cur.execute("""
                CREATE TABLE Exons_Statistics
                (
                        sample_ID VARCHAR NOT NULL,
                        refseq_design VARCHAR NOT NULL,
                        FOREIGN KEY (sample_ID, refseq_design)
                                REFERENCES
RefSeq_Design_Statistics(sample_ID, refseq_design)
                                ON UPDATE CASCADE
                  ON DELETE CASCADE,
                        chr VARCHAR NOT NULL,
                        gene VARCHAR NOT NULL,
                        length DOUBLE PRECISION NOT NULL,
                        mean_coverage DOUBLE PRECISION NOT
NULL,
                        Percent_onex DOUBLE PRECISION NOT
NULL,
                        Percent_fivex DOUBLE PRECISION NOT
NULL,
                        Percent_tenx DOUBLE PRECISION NOT
NULL,
                        Percent_twentyx DOUBLE PRECISION NOT
NULL,
                        Percent_thirtyx DOUBLE PRECISION NOT
NULL,
                        Percent_pass DOUBLE PRECISION NOT
NULL,
                        Percent_pass_dp10 DOUBLE PRECISION NOT
NULL,
                        PRIMARY KEY (sample_ID, refseq_design, chr,
gene)
                )
            """)


            #Variants Statistics
            cur.execute("DROP TABLE IF EXISTS Variant_Statistics
CASCADE")
            cur.execute("""
                CREATE TABLE Variant_Statistics
            (
                 sample_ID VARCHAR PRIMARY KEY
                  REFERENCES Sample(sample_ID)
                  ON UPDATE CASCADE
                  ON DELETE CASCADE,
                 n_variants_cds20bp DOUBLE PRECISION,
                 n_exons_variants DOUBLE PRECISION NOT NULL,
                 mie DOUBLE PRECISION,
                 recessive DOUBLE PRECISION,
                 de_novo DOUBLE PRECISION,
                het DOUBLE PRECISION,
                x_link DOUBLE PRECISION,
                roh DOUBLE PRECISION,
                repeated DOUBLE PRECISION
            )
        """)

            # Variant table creation
            cur.execute("DROP TABLE IF EXISTS Variants CASCADE")
            cur.execute("""
                CREATE TABLE Variants
                (
                    variant_ID VARCHAR PRIMARY KEY,
                    chr VARCHAR(5) NOT NULL,
                    position DOUBLE PRECISION,
                    reference VARCHAR NOT NULL,
                    alternative VARCHAR NOT NULL,
                    n_callable DOUBLE PRECISION NOT NULL
DEFAULT 1,
                    n_non_callable DOUBLE PRECISION NOT
NULL DEFAULT 0,
                    n_allele DOUBLE PRECISION NOT NULL
DEFAULT 0,
                    frequency DOUBLE PRECISION NOT NULL
DEFAULT 0.00
                )
            """)

            # Allele table creation
            cur.execute("DROP TABLE IF EXISTS Alleles CASCADE")
            cur.execute("""
                CREATE TABLE Alleles
                (
                    sample_ID VARCHAR NOT NULL PRIMARY
KEY
                        REFERENCES Sample(sample_ID)
                        ON UPDATE CASCADE
                        ON DELETE CASCADE,
                    variant_ID VARCHAR NOT NULL
                        REFERENCES Variants(variant_ID)
                        ON UPDATE CASCADE
                        ON DELETE CASCADE,
                    heterozygosity BOOLEAN NOT NULL,
                    homozygosity BOOLEAN NOT NULL
                )
            """)

            # History table creation (for debugging purposes only)
            cur.execute("DROP TABLE IF EXISTS History CASCADE")
```

```
                    cur.execute("""
                              CREATE TABLE History
                              (
                                        id SERIAL PRIMARY KEY,
                                        idModeApp VARCHAR NOT NULL,
                                        instant TIMESTAMP NOT NULL,
                                        methodName VARCHAR NOT NULL
                              )
                    """)

          cur.close()
conn.commit()
conn.close()
```

Script 3 - db_website_management.py. Script to join the database to the web page.

```
#!/usr/bin/env python3
# dbhg19 database website creation/management script
# @author: Alessandro Facconi
# modified by Denise Lavezzari

#pip install sqlalchemy
#pip install pandas
#pip install flask
#pip install secure_filename

import logging, os, re
from flask import Flask, flash, redirect, request, render_template, url_for,
send_from_directory
from werkzeug.utils import secure_filename
from retrieve_excel_data import RetrieveExcel
from manipulate_data import ShowGraph
from manipulate_data import ShowTable

# Excel files uploading variables
UPLOAD_FOLDER = "/var/www/dbwebsite/dbwebsite/uploads/"
UPLOAD_FOLDER_TSV = "/var/www/dbwebsite/dbwebsite/uploads/tsv"
DOWNLOAD_FOLDER = "/var/www/dbwebsite/dbwebsite/downloads/"
ALLOWED_EXTENSIONS = {'xls','xlsx'}

logging.basicConfig(level=logging.DEBUG )
dbwebsite = Flask(__name__)
dbwebsite.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0
dbwebsite.secret_key = "ddlabno1PG"
dbwebsite.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
dbwebsite.config['UPLOAD_FOLDER_TSV'] = UPLOAD_FOLDER_TSV
dbwebsite.config['DOWNLOAD_FOLDER'] = DOWNLOAD_FOLDER
dbwebsite.excel = RetrieveExcel()
dbwebsite.graph = ShowGraph()
dbwebsite.table = ShowTable()

################ FUNCTION DEFINITION ###############

# This function checks if uploaded file extensions are valid
def allowed_file(filename):
        return "." in filename and filename.rsplit(".", 1)[1].lower() in
ALLOWED_EXTENSIONS

# Sample data uploading
def upload(name):
    input_name = name + '_excel'
    # Excel file uploading
```

```
        if request.form['identifier'] == input_name:
                # Check if the post request involves excel file uploading
                if input_name not in request.files:
                        flash('No excel')
                        return redirect(request.url)
                file = request.files[input_name]
                # If the user does not select a file, the browser submits an empty file
without filename
                if file.filename == '':
                        flash('No selected file')
                        return redirect(request.url)
                # Excel file uploading
                if file and allowed_file(file.filename):
                        filename = secure_filename(file.filename)
                        file.save(os.path.join(dbwebsite.config['UPLOAD_FOLDER'],
filename))
                        uploaded_file =
os.path.join(dbwebsite.config['UPLOAD_FOLDER'], filename)
                        if dbwebsite.excel.uploadFunction(uploaded_file, name) == -1:
                                return render_template("error_page.html")
                        else:
                                return render_template("upload_done.html")

############ WEBSITE ################

# Site layout
@dbwebsite.route("/layout")
def layout():
        return render_template("layout.html")

# Index page
@dbwebsite.route("/index/", methods=["GET", "POST"])
def index():
        return render_template("index.html")

############### WETLAB PAGE ####################

# Wetlab index page
@dbwebsite.route("/wetlab/", methods=["GET", "POST"])
def wetlab():
        return render_template("wetlab_page.html")

@dbwebsite.route('/sample_upload', methods=['GET', 'POST'])
def sample_upload():
        if request.method == 'POST':
                upload('sample')
        return render_template("wetlab_page.html")

@dbwebsite.route('/qc_upload', methods=['GET', 'POST'])
```

```
def qc_upload():
        if request.method == 'POST':
                upload('qc')
        return render_template("wetlab_page.html")

@dbwebsite.route('/library_upload', methods=['GET', 'POST'])
def library_upload():
        if request.method == 'POST':
                upload('library')
        return render_template("wetlab_page.html")

@dbwebsite.route('/capture_info_upload', methods=['GET', 'POST'])
def capture_info_upload():
        if request.method == 'POST':
                upload('capture_info')
        return render_template("wetlab_page.html")

@dbwebsite.route('/capture_samples_upload', methods=['GET', 'POST'])
def capture_samples_upload():
        if request.method == 'POST':
                upload('capture_samples')
        return render_template("wetlab_page.html")

@dbwebsite.route('/sequencing_upload', methods=['GET', 'POST'])
def sequencing_upload():
        if request.method == 'POST':
                upload('sequencing')
        return render_template("wetlab_page.html")

############## BIOINFO PAGE ################

@dbwebsite.route("/bioinfo/", methods=["GET", "POST"])
def bioinfo():
        return render_template("bioinfo_page.html")

@dbwebsite.route('/statistics_upload', methods=['GET', 'POST'])
def statistics_upload():
        if request.method == 'POST':
                upload('statistics')
        return render_template("bioinfo_page.html")

@dbwebsite.route('/refseq_design_statistics_upload', methods=['GET', 'POST'])
def refseq_design_statistics_upload():
        if request.method == 'POST':
                upload('refseq_design_statistics')
        return render_template("bioinfo_page.html")

@dbwebsite.route('/exons_statistics_upload', methods=['GET', 'POST'])
def exons_statistics_upload():
        if request.method == 'POST':
                upload('exons_statistics')
        return render_template("bioinfo_page.html")

@dbwebsite.route('/variant_statistics_upload', methods=['GET', 'POST'])
def variant_statistics_upload():
        if request.method == 'POST':
                upload('variant_statistics')
        return render_template("bioinfo_page.html")


############### DATABASE PAGE ################

@dbwebsite.route("/db/", methods=["GET", "POST"])
def db():
            if request.method == "POST":
                        print(request.form['button'])
                        # Database download
                        if request.form['button'] == 'table':
                                    db = request.form['reference']
                                    tabella = request.form.getlist('one_checkbox')
                                    print("SCELTA:", tabella)
                                    filtri = request.form.getlist('second_checkbox')
                                    print("SCELTA:", filtri)
                                    simbolo = request.form.getlist('option')
                                    simbolo = [i for i in simbolo if i]
                                    print("SCELTA:", simbolo)
                                    valori = request.form.getlist('valore')
                                    valori = [i for i in valori if i]
                                    print("SCELTA:", valori)
                                    dbwebsite.table.download_data(db, tabella, filtri, simbolo,
valori)
                                    return render_template("table_show.html")
                        elif request.form['button'] == 'graph':
                                    db = request.form['reference']
                                    tabella = request.form.getlist('one_checkbox')
                                    filtri = request.form.getlist('second_checkbox')
                                    simbolo = request.form.getlist('option')
                                    simbolo = [i for i in simbolo if i]
                                    valori = request.form.getlist('valore')
                                    valori = [i for i in valori if i]
                                    dbwebsite.graph.data_plotting(db, tabella, filtri, simbolo,
valori)
                                    return render_template("plot_show.html")

            return render_template("table_page.html")

@dbwebsite.route("/ER_Schema/", methods=["GET", "POST"])
def ER_Schema():
            return render_template("ER-Schema.html")
```

```
@dbwebsite.route("/tableView/", methods=["GET", "POST"])
def tableView():
        return render_template("table_page.html")

if __name__ == '__main__':
    app.run(debug=true)
```

Script 4 - retrieve_excel_data.py. Script to read the excel file and upload the data on the database.

```
#!/usr/bin/env python3

# The script retrieve data from the excel and upload it on the database

# @author: Denise Lavezzari

import glob, os, sys, stat
import pandas as pd
import psycopg2.extras

param_dic = {
        "host" : "localhost",
        "database" : "FGVD_hg19",
        "user" : "postgres",
        "password" : None
}

def connect(params_dic):
   """ Connect to the PostgreSQL database server """
   conn = None
   try:
      # connect to the PostgreSQL server
      print('Connecting to the PostgreSQL database...')
      conn = psycopg2.connect(**params_dic)
   except (Exception, psycopg2.DatabaseError) as error:
      print(error)
      sys.exit(1)
   print("Connection successful")
   return conn

class RetrieveExcel():
        # Sample excel data converter
        def uploadFunction(self, file, table):
                conn = connect(param_dic)
                cursor = conn.cursor()
                # Excel file opening
                df = pd.read_excel(file, header=0)
                tmp_df = "/var/www/dbwebsite/dbwebsite/uploads/tmp.csv"
                df.to_csv(tmp_df, header=False, index=False, sep=';')
                f =  open(tmp_df,'r')
                try:
                        cursor.copy_from(f,table, sep=';', null="")
                        conn.commit()
                except (Exception, psycopg2.DatabaseError) as error:
                        os.remove(tmp_df)
                        print("Error: %s" % error)
                        conn.rollback()
                        cursor.close()
```

```
                    return 1
                print('Data copied to DB!!!')
                cursor.close()
                os.remove(tmp_df)
```

Script 5 - manipulate_data.py. Script to create tables/graphs based on user request.

```
#!/usr/bin/env python3
# The script creates tables or graphs based on user request
# @author: Denise Lavezzari

import glob, os, sys, stat
import pandas as pd
import matplotlib.pyplot as plt
import psycopg2.extras
from sqlalchemy import create_engine
import logging
from query_db import InterrogateDB)
logging.getLogger('matplotlib.font_manager').disabled = True

POSTGRES_ADDRESS = 'localhost'
POSTGRES_USERNAME = 'postgres'
POSTGRES_PASSWORD = None
POSTGRES_DBNAME_hg19 = 'FGVD_hg19'
POSTGRES_DBNAME_hg38 = 'FGVD_E'

def connection(db):
    if db == "hg19":
        dbname = POSTGRES_DBNAME_hg19
        return dbname
    elif db == "hg38":
        dbname = POSTGRES_DBNAME_hg38
        return dbname


###############  TABLES ######################
class ShowTable():
        def download_data(self, db, tab, filters, symbol, values):

                ## connection to the choosen database
                database = connection(db)
                postgres_str =
('postgresql://{username}:{password}@{ipaddress}/{dbname}'.format(usernam
e=POSTGRES_USERNAME, password=POSTGRES_PASSWORD,
ipaddress=POSTGRES_ADDRESS, dbname=database))
                cnx = create_engine(postgres_str)
                con = cnx

                ### write the query
                query = InterrogateDB.write_query(con,tab, filters, symbol, values)
                print("QUERY:", query)

                ### save the table results from query in a dataframe
                df = pd.DataFrame()
                df = pd.read_sql(query, con=cnx)
                print("DATAFRAME:", df.head())

                ### Create a html page with the table of interest
                strTable = """{% extends \"layout.html\" %}
{% block content %}
        <table>
                <tr>"""
                for col in df.columns:
                          strCOL = "<th>" + col + "</th>"
                          strTable = strTable + strCOL
                strTable = strTable + "</tr>"
                for row in range(len(df)):
                          for col in range(len(df.columns)):
                                    strRW = "<td>" + str(df.iloc[row,col]) + "</td>"
                                    strTable = strTable + strRW
                          strTable = strTable + "</tr>"
                strTable = strTable+"</table>"
                strTable = strTable+"{% endblock %}"

                hs =
open("/var/www/dbwebsite/dbwebsite/templates/table_show.html", 'w')
                hs.write(strTable)
                return print("Data where retrieved.")


#################### GRAPHS ################

class ShowGraph():
        def data_plotting(self, db, tab, filters, symbol, values):
                ## connection to the choosen database
                database = connection(db)
                postgres_str =
('postgresql://{username}:{password}@{ipaddress}/{dbname}'.format(usernam
e=POSTGRES_USERNAME, password=POSTGRES_PASSWORD,
ipaddress=POSTGRES_ADDRESS, dbname=database))
                cnx = create_engine(postgres_str)
                con = cnx
                df = pd.DataFrame()
                ### write the query
                query_full = InterrogateDB.write_query(con,tab, filters, symbol,
values)
                print("QUERY", query_full)

                ### save the table results from query in a dataframe
                df = pd.DataFrame()
                df = pd.read_sql(query_full, con=cnx)
                cols = df.columns.to_list()

                ## remove columns not numerical
                if 'refseq_design' in cols: cols.remove('refseq_design')
                print("COLONNE:", cols )

                ### create the linear plot
                plt.figure()
                for i in range(len(cols)):
                          plt.plot(df[cols[i]], label=cols[i])

                plt.grid(color='grey', which='major', axis='y', linestyle='solid')
                lgd = plt.legend(loc='center left', bbox_to_anchor=(1, 0.5),
fontsize=7)

        plt.savefig('/var/www/dbwebsite/dbwebsite/static/image/line_plot.png',
dpi=175, format='png',bbox_inches='tight')
                plt.close()

                strImage = """{% extends \"layout.html\" %}{% block content
%}<img src="{{url_for('static', filename='image/line_plot.png')}}"/>{%
endblock %}"""
                hs =
open("/var/www/dbwebsite/dbwebsite/templates/plot_show.html", 'w')
                hs.write(strImage)
                return print("Plot done")
```

Script 6 - query_db.py. Script to transform the user request to a query.

```
#!/usr/bin/env python3
# Script to create the query to submit to the database
# @author: Denise Lavezzari

import pandas as pd

class InterrogateDB():
        def write_query(con, tab, filters, symbol, values):
                print("VALORII:", values)
                if 'fullDB' in tab:
                         ## create a view with all the data inside the database
                        con.execute("""DROP VIEW IF EXISTS fullDB;
                                CREATE VIEW fullDB AS
                                        (SELECT *
                                                FROM Sample s
                                                FULL JOIN qc USING (cgf_ID,
sample_ID)
                                                FULL JOIN library USING
(cgf_ID, sample_ID)
                                                FULL JOIN capture_samples
USING (cgf_ID, sample_ID)
                                                FULL JOIN capture_info USING
(capture_ID)
                                                FULL JOIN sequencing USING
(cgf_ID, sample_ID)
                                                FULL JOIN statistics USING
(sample_ID)
                                                FULL JOIN
refSeq_design_statistics USING (sample_ID)
                                                FULL JOIN variant_statistics
USING (sample_ID)
                                        )
                                        ORDER BY cast(cgf_ID as int) ASC;""")

                if len(filters)!=0 and len(values)!=0 and len(filters)!=len(values):
                        print("query 0")
                        for i in (range(len(filters))):
                                if i == 0:
                                        query = "SELECT " + filters[i] + " "
                                else:
                                        query = query + ","  + filters[i]

                        for i in (range(len(tab))):
                                if i == 0:
                                        query = query + " FROM " + tab[i]
                                elif tab[i] == "sample" or tab[i] == "qc" or tab[i]
== "library" or tab[i] == "capture_samples" or tab[i] == "sequencing":
                                        query = query + " FULL JOIN " + tab[i] +
" USING (cgf_ID, sample_ID)"
                elif tab[i] == "capture_info":
                    query = query + " FULL JOIN " + tab[i] +
" USING (capture_ID)"
                elif tab[i] == "alleles":
                    query = query + " FULL JOIN " + tab[i] +
" USING (variant_ID)"
                else:
                    query = query + " FULL JOIN " + tab[i] +
" USING (sample_ID)"

            for i in (range(len(values))):
                if values[i]=='':
                    continue
                else:
                    if i == 0:
                        if symbol[i] == "ilike":
                            query = query + " WHERE
" + filters[i] + " " + symbol[i] + " \'%" + values[i] + "%\'"
                        else:
                            query = query + " WHERE
" + filters[i] + " " + symbol[i] + "\'" + values[i] + "\'"
                    else:
                        if symbol[i] == "ilike":
                            query = query + " AND " +
filters[i] + " " + symbol[i] + " \'%" + values[i] + "%\'"
                        else:
                            query = query + " AND " +
filters[i] + " " + symbol[i] + "\'" + values[i] + "\'"

        elif len(filters)!=0 and len(values)!=0 and len(filters)==len(values):
            print("query 1")
            for i in (range(len(tab))):
                if i == 0:
                    query = "SELECT * FROM " + tab[i]
                elif tab[i] == "sample" or tab[i] == "qc" or tab[i]
== "library" or tab[i] == "capture_samples" or tab[i] == "sequencing":
                    query = query + " FULL JOIN " + tab[i] +
" USING (cgf_ID, sample_ID)"
                elif tab[i] == "capture_info":
                    query = query + " FULL JOIN " + tab[i] +
" USING (capture_ID)"
                elif tab[i] == "alleles":
                    query = query + " FULL JOIN " + tab[i] +
" USING (variant_ID)"
                else:
                    query = query + " FULL JOIN " + tab[i] +
" USING (sample_ID)"
```

```
                    for i in (range(len(values))):
                            if values[i]=='':
                                    continue
                            else:
                                    if i == 0:
                                            if symbol[i] == "ilike":
                                                    query = query + " WHERE
" + filters[i] + " " + symbol[i] + " \'%" + values[i] + "%\'"
                                            else:
                                                    query = query + " WHERE
" + filters[i] + " " + symbol[i] + "\'" + values[i] + "\'"
                                    else:
                                            if symbol[i] == "ilike":
                                                    query = query + " AND " +
filters[i] + " " + symbol[i] + " \'%" + values[i] + "%\'"
                                            else:
                                                    query = query + " AND " +
filters[i] + " " + symbol[i] + "\'" + values[i] + "\'"

            elif len(filters)!=0 and len(values)==0:
                    print("query 2")
                    for i in (range(len(filters))):
                            if i == 0:
                                    query = "SELECT " + filters[i] + " "
                            else:
                                    query = query + ","  + filters[i]

                    for i in (range(len(tab))):
                                    if i == 0:
                                            query = query + " FROM " + tab[i]
                                    elif tab[i] == "sample" or tab[i] == "qc" or
tab[i] == "library" or tab[i] == "capture_samples" or tab[i] == "sequencing":
                                            query = query + " FULL JOIN " +
tab[i] + " USING (cgf_ID, sample_ID)"
                                    elif tab[i] == "capture_info":
                                            query = query + " FULL JOIN " +
tab[i] + " USING (capture_ID)"
                                    elif tab[i] == "alleles":
                                            query = query + " FULL JOIN " +
tab[i] + " USING (variant_ID)"
                                    else:
                                            query = query + " FULL JOIN " +
tab[i] + " USING (sample_ID)"

            elif len(filters)==0 and len(values)==0:
                    print("query 3")
                    for i in (range(len(tab))):
                                    if i == 0:
                                            query = "SELECT * FROM " +
tab[i]
                                                  elif tab[i] == "sample" or tab[i] == "qc" or
tab[i] == "library" or tab[i] == "capture_samples" or tab[i] == "sequencing":
                                                            query = query + " FULL JOIN " +
tab[i] + " USING (cgf_ID, sample_ID)"
                                                  elif tab[i] == "capture_info":
                                                            query = query + " FULL JOIN " +
tab[i] + " USING (capture_ID)"
                                                  elif tab[i] == "alleles":
                                                            query = query + " FULL JOIN " +
tab[i] + " USING (variant_ID)"
                                                  else:
                                                            query = query + " FULL JOIN " +
tab[i] + " USING (sample_ID)"

                    print("LA MIA QUERY:", query)
                    return query
```

Script 7 - upload.py. Script to upload VCF information to the database.

```
import psycopg2
import subprocess
import sys
from datetime import datetime

# sys.argv[«] = sample ID   sys.argv[2] = design

print(sys.argv[1])
print('Start', datetime.now())

# connecting to the database
print('Connecting to db', datetime.now())
connessione = psycopg2.connect(host="localhost", database="FGVD_E",
user="postgres")
with connessione:
        with connessione.cursor() as cursore:


                #saving in data the last value of variant_ID
                cursore.execute("""SELECT MAX(variant_id) FROM variants""")
                dat = cursore.fetchone()
                data = int(dat[0]) + 1
                cursore.execute("""DELETE FROM table1""")
                print('Copying into table1', datetime.now())


                #uploading variants in table1
                f = sys.argv[1]
                f2 = f + "_3.vcf"
                fo = open(f2)
                cursore.copy_from(fo, '"table1"', columns=('sample_ID', 'chr',
'position', 'reference', 'alternative', 'genotype'), sep=" ")
                cursore.execute(
                        """UPDATE table1
                        SET heterozygosity = '1', homozygosity = '0'
                        WHERE split_part(genotype, '/', 1) <> split_part(genotype, '/',
2)"""
                )
                cursore.execute(
                        """UPDATE table1
                        SET homozygosity = '1', heterozygosity = '0'
                        WHERE split_part(genotype, '/', 1) = split_part(genotype, '/',
2)"""
                )


                # Adding a new line in the samples table
                d = sys.argv[2]
```

```
            cursore.execute(
                """SELECT DISTINCT sample_ID FROM table1"""
            )
            s = cursore.fetchone()
            cursore.execute(
                """INSERT INTO samples(sample_ID, design)
                VALUES (%s, %s)""", (s, d)
            )


        #creating a file that includes all the variant position we already have in the
database
        with open('/home/ghelix/database_scripts_esomi/posizioni.bed', 'w') as fw:
            with connessione.cursor() as cursore:
                cursore.copy_expert(
                    """COPY (SELECT DISTINCT chr, position, position
FROM variants) TO STDOUT DELIMITER '   '""", fw
                )
        print('Checking which old variants are callable', datetime.now())


        #intersecting the variant positions file with the bed of the new VCF to find
in how many files each position is callable
        subprocess.call("awk '{print $1, $2-1, $3}' OFS='\t' posizioni.bed >
posizioni2.bed", shell=True)
        subprocess.call("intersectBed -a posizioni2.bed -b Callable/" + f +
"_2.callable.bed.gz" + " -f 1.00 -c | cut -f1,2,4 | awk '{print $1, $2+1, $3}'
OFS='\t' > result.bed", shell=True)


        # Creating a file that includes all the variants we already have in the database
        with open('/home/ghelix/database_scripts_esomi/vecchievar.bed', 'w') as
fnw:
            with connessione.cursor() as cursore:
                cursore.copy_expert(
                    """COPY (SELECT chr, position, reference, alternative
FROM variants) TO STDOUT DELIMITER '     '""", fnw
            )
        print('Checking which variants are new', datetime.now())


        # creating a file with all the new variants we don't have in the database
        subprocess.call("sort -n vecchievar.bed > vecchievar_sorted.bed",
shell=True)
        subprocess.call("sed 's/ /      /g' " + f2 + " | cut -f2,3,4,5 > " + f +
"_cut.vcf", shell=True)
        subprocess.call("sort -n " + f + "_cut.vcf > " + f + "_sorted.vcf",
shell=True)
        subprocess.call("comm -23 " + f + "_sorted.vcf vecchievar_sorted.bed >
newvariants.vcf", shell=True)
```

```
    # for each variant position, saving the count of the first bedtools intersect in
a new table
    with connessione.cursor() as cursore:
        cursore.execute(
            """CREATE TABLE callable(
            chr VARCHAR,
            position INTEGER,
            n NUMERIC)"""
        )
    with open('/home/ghelix/database_scripts_esomi/result.bed', 'r') as fr:
        with connessione.cursor() as cursore:
            cursore.copy_from(
                fr, '"callable"', columns=('chr', 'position', 'n'), sep="	"
            )
    with connessione.cursor() as cursore:
        cursore.execute(
            """SELECT COUNT(*) FROM samples"""
        )
        rows = cursore.fetchone()
        cursore.execute(
            """SELECT DISTINCT chr, position, n FROM callable"""
        )


        # for each variant, updating the count of n_callable and n_noncallable
        for record in cursore:
            with connessione.cursor() as curs:
                curs.execute(
                    """UPDATE variants
                    SET n_callable = n_callable + %s
                    WHERE chr = %s AND position = %s""", (record[2],
record[0], record[1])
                )
                curs.execute(
                    """UPDATE variants
                    SET n_noncallable = %s - n_callable
                    WHERE chr = %s AND position = %s""", (rows,
record[0], record[1])
                )
    # copying new variants in the variants table, in the newvar table and updating
the alleles table with all the variants in the new VCF
    with connessione.cursor() as cursore:
        cursore.execute(
            """CREATE TABLE table2(
                chr VARCHAR(5),
                position INTEGER CHECK (position>0),
                variant_ID SERIAL,
                reference VARCHAR,
                alternative VARCHAR)"""
        )
```

```
            cursore.execute("""SELECT setval('table2_variant_ID_seq', %s)""",
dat)
            nv = open('/home/ghelix/database_scripts_esomi/newvariants.vcf')
            cursore.copy_from(nv, '"table2"', columns=('chr', 'position', 'reference',
'alternative'), sep="        ")
            cursore.execute(
                """INSERT INTO newvar(chr, position, variant_ID)
                    SELECT table2.chr, table2.position, table2.variant_ID
                    FROM table2"""
            )
            cursore.execute(
                """INSERT INTO variants(chr, position, variant_ID, reference,
alternative)
                    SELECT chr, position, variant_ID, reference, alternative
                    FROM table2"""
            )
            print('Updating alleles table', datetime.now())
            cursore.execute(
                """INSERT INTO alleles (variant_ID, sample_ID, heterozygosity,
homozygosity)
                    SELECT DISTINCT variants.variant_ID, table1.sample_ID,
table1.heterozygosity, table1.homozygosity
                    FROM table1 JOIN variants ON ROW(table1.chr,
table1.position, table1.reference, table1.alternative) = ROW(variants.chr,
variants.position, variants.reference, variants.alternative)"""
            )
        with connessione.cursor() as cursore:
            cursore.execute(
                """SELECT chr, position, reference, alternative FROM table1
WHERE heterozygosity = '1'"""
            )
            # counting the occurrences of each variant in the database
            for row in cursore:
                with connessione.cursor() as cur:
                    cur.execute(
                        """UPDATE variants
                        SET n_allele = n_allele + 1
                        WHERE chr LIKE %s AND position = %s AND
reference LIKE %s
                            AND alternative LIKE %s""", (row[0], row[1],
row[2], row[3])
                    )
            cursore.execute(
                """SELECT chr, position, reference, alternative FROM table1
WHERE homozygosity = '1'"""
            )
            # counting the occurrences of each variant in the database
            for row in cursore:
                with connessione.cursor() as cur:
                    cur.execute(
                        """UPDATE variants
                        SET n_allele = n_allele + 2
                        WHERE chr LIKE %s AND position = %s AND
reference LIKE %s
                            AND alternative LIKE %s""", (row[0], row[1],
row[2], row[3])
                    )
        with connessione.cursor() as cursore:
            cursore.execute(
                """DROP TABLE callable"""
            )
            cursore.execute(
                """DROP TABLE table2"""
            )
            cursore.execute(
                """ANALYZE"""
            )
connessione.commit()
connessione.close()

subprocess.call("rm posizioni.bed posizioni2.bed result.bed vecchievar.bed
vecchievar_sorted.bed " + f2 + " " + f + "_sorted.vcf " + f + "_cut.vcf
newvariants.vcf", shell=True)

print('End', datetime.now())
```

Script 8 – upload_frequency.py. Script to update frequency of the variants.

```
import psycopg2
import subprocess
import sys
from datetime import datetime

# sys.argv[«] = sample ID   sys.argv[2] = design

print('Start', datetime.now())

#connecting to the database
print('Connecting to db', datetime.now())
connessione = psycopg2.connect(host="localhost", database="FGVD_E",
user="postgres")
with connessione:
    print('Calculating frequency', datetime.now())


    #calculating the frequency of each variant
    with connessione.cursor() as cursore:
        cursore.execute(
            """SELECT DISTINCT n_allele, n_callable, variant_ID
            FROM variants"""
        )
        for tupla in cursore:
            freq = tupla[0]/(2*tupla[1])
            with connessione.cursor() as cur:
                cur.execute(
                    """UPDATE variants
                    SET frequency = %s
                    WHERE variants.variant_ID = %s""", (freq, tupla[2])
                )
    with connessione.cursor() as cursore:
        cursore.execute(
            """UPDATE variants
            SET frequency = 0.00
            WHERE variant_ID IN (SELECT variant_ID
                        FROM newvar)"""
        )

connessione.commit()
connessione.close()

print('End', datetime.now())
```

Script 9 – graph.py. Script to create heatmaps.

```
#pip install seaborn
#pip install matplotlib
#pip install pandas

import csv
import os
import sys
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns; sns.set_theme()

#folderpath = r"analisi_20-04-2020/Twist/all_regions/Twist/all/EXONS/no_ch
r_sessuali/" # make sure to put the 'r' in front
folderpath = r"analisi_20-04-2021/Roche/all_regions/EXONS/no_chr_sessuali/
"
filepaths  = [os.path.join(folderpath, name) for name in os.listdir(folderpath)]

# Extract for each sample and file the 5X column and add to a dataframe
list_samples = []
for tsv_file in filepaths:

    ## Extract file name
    sample = os.path.basename(tsv_file)
    #sample = sample.replace('Twist_Exome_v1_3_stats_all_design_', '')
    sample = sample.replace('Roche_stats_all_design_', '')
    sample = sample.replace('.EXONS.noSEX.tsv', '')

    ## Extract column of interest and rename it
    df1 = pd.read_csv(tsv_file, delimiter="\t", header=None, names=["Chr", "Gen
e", "Length", "Mean_Cov", "1X", "5X", "10X", "20X", "30X", "PASS","PASS_D
P10"])
    new_name = '5X_' + sample
    df1 = df1.rename(columns = {'5X': new_name }, inplace = False)

    ## Add column to the new list, if the sample is the first take also the gene colu
mn
    if filepaths.index(tsv_file) == 0:
        gene = df1[["Gene"]]
        numbers = df1[[new_name]]
        list_samples.append(gene)
        list_samples.append(numbers)
    else:
        numbers = df1[[new_name]]
        list_samples.append(numbers)

# Transform the list into a dataframe
```

```
large_df = pd.concat(list_samples, axis=1)
large_df

# set the Gene column as the index
large_df = large_df.set_index('Gene')
large_df

# Extract rows where columns are all equal zero or 100

# Remove lines all equal to zero
large_df = large_df[(large_df==0).all(axis=1) == False ]

# Remove lines all equal to 100
large_df = large_df[(large_df==100).all(axis=1) == False ]

large_df

# Create heatmap of the loci
sns.clustermap(large_df,cmap="Blues",cbar_pos=(1, .2, .03, .4), yticklabels=False)
plt.show()
```

Script 10 – csvToVCF.sh. Script to convert CSV to VCF file.

```
#!/bin/bash

# The script take as input the csv file downloaded from the database and convert i
t in vcf
# @author: Denise Lavezzari

csv=$1
# Extract csv name
name="$(basename $csv .csv)"

# Change to tab delimiters
sed -i 's/"//g; s/,/\t/g; s/chr//g' $csv

# Remove header line
tail -n +2 $csv > ${name}.no_header.csv

# Sort csv file
sort -k1,1V -k2,2V ${name}.no_header.csv | awk '{print $1"\t"$2"\t"$3"\t"$4"\t
"$5"\t""."\t""."\t""AF="$6"\t""GT""\t"1"/"1}' > ${name}.no_header.sorted.
csv

# Create header
grep '^##' $0 | sed 's/##/#/g' > header.vcf ;

# Create VCF
cat header.vcf ${name}.no_header.sorted.csv > ${name}.sorted_GENO.vcf
sed -i -e "s/\r//g"  ${name}.sorted_GENO.vcf
bgzip ${name}.sorted_GENO.vcf
tabix ${name}.sorted_GENO.vcf.gz

rm ${name}.no_header.csv ${name}.no_header.sorted.csv header.vcf

###fileformat=VCFv4.2
###FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
###INFO=<ID=AF,Number=.,Type=String,Description="AlleleFrequencies"
>
###INFO=<ID=AlleleFrequencies,Number=.,Type=Float,Description="The A
llele Counts divided by the total number of observed callable alleles (# Alleles).">
###contig=<ID=M,length=16571,assembly=hg38.fa>
###contig=<ID=1,length=249250621,assembly=hg38.fa>
###contig=<ID=2,length=243199373,assembly=hg38.fa>
###contig=<ID=3,length=198022430,assembly=hg38.fa>
###contig=<ID=4,length=191154276,assembly=hg38.fa>
###contig=<ID=5,length=180915260,assembly=hg38.fa>
###contig=<ID=6,length=171115067,assembly=hg38.fa>
###contig=<ID=7,length=159138663,assembly=hg38.fa>
###contig=<ID=8,length=146364022,assembly=hg38.fa>
```

```
###contig=<ID=9,length=141213431,assembly=hg38.fa>
###contig=<ID=10,length=135534747,assembly=hg38.fa>
###contig=<ID=11,length=135006516,assembly=hg38.fa>
###contig=<ID=12,length=133851895,assembly=hg38.fa>
###contig=<ID=13,length=115169878,assembly=hg38.fa>
###contig=<ID=14,length=107349540,assembly=hg38.fa>
###contig=<ID=15,length=102531392,assembly=hg38.fa>
###contig=<ID=16,length=90354753,assembly=hg38.fa>
###contig=<ID=17,length=81195210,assembly=hg38.fa>
###contig=<ID=18,length=78077248,assembly=hg38.fa>
###contig=<ID=19,length=59128983,assembly=hg38.fa>
###contig=<ID=20,length=63025520,assembly=hg38.fa>
###contig=<ID=21,length=48129895,assembly=hg38.fa>
###contig=<ID=22,length=51304566,assembly=hg38.fa>
###contig=<ID=X,length=155270560,assembly=hg38.fa>
###contig=<ID=Y,length=59373566,assembly=hg38.fa>
###reference=file:///home/db/hg38/hg38.fa
##CHROM POS     ID      REF     ALT     QUAL    FILTER  INFO FORMAT
SAMPLE
```

APPENDIX B – RELATIONAL DATABASE SCHEMA DEFINITIONS

For each table of the database, the attributes and their description are reported.

Supplementary Table 1. Sample

CGF ID	Internal Functional Genomics Center Identifier
Sample ID	Sample Identifier
Identity	Family relationship: Proband/Mother/Father/Siblings
Arrival Date	Date of sample arrival
Urgency	If TRUE sample must be processed in a short time
HPO	Specified Human Phenotype Ontology terms
Affected	If TRUE, sample is affected by the disease
Sex	Sex of the sample
Age	Age of the sample
Ethnicity	Ethnicity of the sample
Delivery	Date of sample delivery
Sample Notes	Notes given by doctors
Father	Father Identifier
Mother	Mother Identifier
Siblings	Siblings Identifier
Project	Project the sample belongs to.
Solved	If TRUE, the cause of the disease was identified
Causative Variant	Specification of the variant that has been found to cause the disease

Supplementary Table 2. QC

CGF ID	Internal Functional Genomics Center Identifier
Sample ID	Sample Identifier
QC extraction type	Extraction method: manual/automatic
QC buffer extraction kit	DNA elution buffer
QC NanoDrop ng/µl	Concentration of the sample calculate on NanoDrop

QC NanoDrop 260/280	260/280 absorbance ratio of the sample calculate on NanoDrop
QC NanoDrop 260/230	260/230 absorbance ratio of the sample calculate on NanoDrop
QC TapeStation DIN	DNA Integrity Number of the sample calculate on TapeStation
QC TapeStation ng/µl	Concentration of the sample calculate on TapeStation
QC purificate	Ampure XP Beads quantity and DNA quantity ratio for the purification
QC Qubit ng/ul	Concentration of the sample calculate on Qubit

Supplementary Table 3. Library

Library kit	Name of the library kit
CGF ID	Internal Functional Genomics Center Identifier
Sample ID	Sample Identifier
Sample concentration	Concentration of the sample
Lib input ng	Micrograms of the sample
Lib sample dilution	Dilution ratio
Lib input µl	Microliters of the sample
Lib h2o/µl	Microliters of water to dilute samples obtaining a desired concentration
Lib fragmentation time	Fragmentation time of the library
Lib cleanup	Cleanup to remove buffer from ETA and ligation steps
Lib size selection	Selection of the desired length fragments using AMPURE beads
Lib used index	Index used for the library
Lib pcr cycle	Number of PCR cycles for the library
Lib cleanup post pcr	Cleanup to remove the PCR component
Lib Qubit1 ng/µl	Concentration of the library calculate on Qubit (first check)

Lib Qubit2 ng/µl	Concentration of the library calculate on Qubit (second check)
Lib Qubit3 ng/µl	Concentration of the library calculate on Qubit (three check)
Lib Qubit mean	Mean concentration of the library calculate on Qubit
Lib TapeStation from	Minimum library size calculated on TapeStation
Lib TapeStation to	Maximum library size calculated on TapeStation
Lib TapeStation average size	Average library size calculated on TapeStation
Lib TapeStation concentration	Concentration of the library calculated on TapeStation
Lib TapeStation molarity	Molarity of the library calculated on TapeStation
Lib date	Date of library preparation
Lib location	Location of library preparation
Lib operator	Operator of library preparation
Lib notes	Notes about library

Supplementary Table 4. Capture Sample

CGF ID	Internal Functional Genomics Center Identifier
Sample ID	Sample Identifier
Capture ID	Capture Identifier

Supplementary Table 5. Capture Info

Capture Kit	Kit used for the capture
Capture ID	Capture Identifier
N capture	Capture Number
Capt pos from	Minimum capture size calculated TapeStation
Capt pos to	Maximum capture size calculated TapeStation
Capt average size	Average capture size calculated TapeStation
Capt concentration	Concentration of the capture calculated on TapeStation

Cap molarity	Molarity of the capture calculated on TapeStation
Capt pM	Picomol/l of the capture
Capt qubit ng/µl	Concentration of the capture calculate on Qubit
Capt date	Date of capture preparation
Capt location	Location of capture preparation
Capt operator	Operator of capture preparation
Capt notes	Notes about capture

Supplementary Table 6. Sequencing

CGF ID	Internal Functional Genomics Center Identifier
Sample ID	Sample Identifier
Sequencer	Sequencer used
Seq date	Date of capture preparation
Seq location	Location of capture preparation
Seq flow cell	Flow cell used for the sequencing
Seq millions of fragments	Millions of fragments assigned

Supplementary Table 7. Statistics

Sample ID	Sample Identifier
N fragments	Number of sequenced fragments
gc	Percentage of GC content
Insert size	Mean insert size on mapped data
N map dedup	Number of fragments after duplicates removal
Duplicates	Percentage of duplicates

Supplementary Table 8. RefSeq Design Statistics

Sample ID	Sample Identifier
RefSeq Design	RefSeq or Design version
Kit length	RefSeq or Design length
Mean coverage	Mean coverage over RefSeq / Design

Percent oneX	The fraction of target bases covered by at least 1 read
Percent fiveX	The fraction of target bases covered by at least 5 reads
Percent tenX	The fraction of target bases covered by at least 10 reads
Percent twentyX	The fraction of target bases covered by at least 20 reads
Percent thirtyX	The fraction of target bases covered by at least 30 reads
Percent pass	Percentage of base called a base with good coverage (with at least 3 reads) and good mapping quality
Percent pass dp10	Percentage of base called a base with good coverage (with at least 10 reads) and good mapping quality
Percent on-target	Percentage of bases on the on-target
Percent near-target	Percentage of bases on the near target
Percent off-target	Percentage of bases on the off target
Fold enrich	Fold Enrichment of RefSeq / Design regions
Fold80	Fold80 Penalty value
Percent coverage uniformity	Percentage of coverage uniformity (PCT > 0.2*mean)

Supplementary Table 9. Exons Statistics

Sample ID	Sample Identifier
RefSeq Design	RefSeq or Design version
Chr	Chromosome
Gene	Gene
Length	Gene Length
Mean coverage	Mean coverage over RefSeq / Design on the exon
Percent oneX	The fraction of target bases covered by at least 1 reads on the exon
Percent fiveX	The fraction of target bases covered by at least 5 reads on the exon
Percent tenX	The fraction of target bases covered by at least 10 reads on the exon
Percent twentyX	The fraction of target bases covered by at least 20 reads on the exon

Percent thirtyX	The fraction of target bases covered by at least 30 reads on the exon
Percent pass	Percentage of base called a base with good coverage (with at least 3 reads) and good mapping quality on the exon
Percent pass dp10	Percentage of base called a base with good coverage (with at least 10 reads) and good mapping quality on the exon

Supplementary Table 10. Variants Statistics

Sample ID	Sample Identifier
N variants cds20bp	Number of variants in CDS ± 20bp
N exons variants	Number of exons variants
Mie	Number of Mendelian Inheritance Error variants
Recessive	Number of recessive variants
De novo	Number of de-novo variants
Het	Number of heterozygous variants
X link	Number of X-linked variants
Roh	Number of variants in Region Of Homozygosity
Repeated	Number of variants in repeated regions

Supplementary Table 11. Variants

Variant ID	Variant Internal Identifier
Chr	Chromosome number
Position	Position in the chromosome
Reference	Reference allele
Alternative	Alternative alleles
N callable	Number of samples where the variant position is callable
N non-callable	Number of samples where the variant position is non-callable
N allele	Number of alleles
Frequency	Frequency of the variant calculated as #allele / #callable * 2

Supplementary Table 12. Alleles

Sample ID	Sample Identifier
Variant ID	Variant Internal Identifier
Heterozygosity	Boolean value that indicates if sample's variant is heterozygous
Homozygosity	Boolean value that indicates if sample's variant is homozygous

ACKNOWLEDGEMENTS

A conclusione di questa tesi vorrei ringraziare innanzitutto il professor e relatore Massimo Delledonne per avermi guidata, per il sostegno e i consigli ricevuti in questi anni. In particolare, per avermi insegnato a mettere in discussione i risultati, quella discussione che fa la differenza tra un tecnico ed un ricercatore.

Un doveroso ringraziamento va al professor Renzo Guerrini e alla dottoressa Elena Parrini, senza la cui collaborazione questa tesi non esisterebbe. Un ringraziamento, inoltre, a Annalisa Vetro, Davide Mei e Claudia Bianchini, sempre molto disponibili e con i quali ho condiviso questi tre anni di ricerca e di soddisfazioni quando i casi venivano risolti, nonostante non abbia ancora avuto il piacere di conoscervi personalmente.

Un ringraziamento va a tutti i membri presenti e passati del laboratorio di genomica funzionale. In primo luogo a Marzia che insieme al professor Delledonne ogni giorno ci guida e ci insegna instancabilmente il lavoro del ricercatore. Un grazie al dott. Salviati fonte infinita di conoscenza e che, oltre al sole, porta sempre un sorriso con i suoi aneddoti. Un grazie a tutti a gli instancabili colleghi: a Marta un'amica straordinaria per essermi stata accanto in questi anni non solo nel lavoro ma anche nella vita privata, nonostante la lontananza. A Barbara I. mentore, amica e collega paziente, per avermi aiutato nei momenti difficili, per aver assecondato le mie pazzie e per aver sopportato la mia memoria da pesce rosso. A Luciano, il mio guru della bioinformatica, sempre pronto ad aiutarmi, che mi ha insegnato tutto sulle varianti e dal quale ho ancora molto da imparare. A Chiara e Cristina, perfette bioinformatiche mancate, per aver condiviso in questi anni l'ansia e la gioia nell'ottenere nei campioni una buona genotipizzabilità. A Luca M. per essere sempre disponibile ad aiutare in qualsiasi momento e con il quale ho condiviso l'enorme gioia del demultiplexing. A Simone M. per aver sdrammatizzato ogni momento critico. A Emanuela che con le sue "supercazzole" rallegra le giornate. A Massimiliano, sempre pronto per un'arrampicata in compagnia, per aver condiviso quest'avventura assieme in Italia e dall'altra parte del mondo. A Giulia per aver allietato le giornate con canzoni e sorrisi, sempre pronta con un abbraccio nei momenti più ostici. A Stefania, indisposta agli abbracci, ma che ha sempre una parola di conforto tra un "maremma" e l'altro, che ci delizia con il thè delle 17 e vivacizza le giornate con il salvataggio di libri dalla spazzatura. A Valentina, ciclista prodigio, per avermi trasmesso un po' della sua grande passione per la ricerca. A Giovanni che nonostante le giornate buie, non perde mai l'allegria. A Martina, giovane ricercatrice, per aver condiviso con me l'avventura con il progetto Finotti e non essersi mai arresa. A Leonardo, radiofonico provetto, per aver seguito la mia follia nel progetto Illumina-MGI e per aver deciso d'intraprendere il piacere dei backup. A Luca D.A. per aver condiviso la sua passione per il vino. A Barbara G., Marzia V., Elena, per essere fonti di saggezza e ispirazione. A Emilia, Luca B., Simone M. e Matteo giovani ricercatori per la loro simpatia. A Valentina F., Antonio e Sandra per la loro allegria e simpatia. Un immenso grazie a tutti voi che mi avete accompagnato in questo percorso.

Un grande ringraziamento va ai miei genitori, il mio punto di riferimento, per aver sempre sostenuto le mie scelte, per avermi continuamente supportato e sopportato soprattutto nei momenti difficili. Grazie per essere sempre al mio fianco senza mai ostacolare il mio percorso, siete le mie colonne portanti. Un grazie a tutti i miei famigliari per l'affetto e l'appoggio che non mi hanno mai fatto mancare.

Un ringraziamento anche a tutti i miei cari amici. In particolare, Beatrice e Francesca per il supporto informatico e linguistico. Alice, Vanessa, Giulia e Giuliano per essermi sempre accanto e sostenermi da anni. Giulio, Silvia, Alessandra, Linda, Alice O., Aurora, Giovanna, Margherita, Luisa, Mara per l'affetto e il supporto morale.

Un grazie a Caterina, Michelangelo, Elena, Antonio e Michela per avermi accolta nella loro famiglia con affetto e dolcezza. Sono grata di poter condividere con voi questi momenti di gioia.

E infine, un grazie a Giovanni, che nonostante le avversità è sempre stato presente. Grazie per avermi incoraggiata, per avermi insegnato le nozioni più svariante (dai nomi degli attori, che non imparerò mai, alle nozioni di alpinismo), per portare felicità in ogni giorno della mia vita, soprattutto quelli più bui e soprattutto per essere al mio fianco.

Un sentito grazie a tutti coloro che ho incontrato sulla mia strada.