SNPop, an application package for the collection, visualization and analysis of single nucleotide polymorphism data

Kyriakos Psallidas

Dissertation

Dissertation's Supervisor: Georgios Boulougouris, Assistant Professor

DEMOCRITUS UNIVERSITY OF THRACE HEALTH SCIENCES SCHOOL

DEPARTMENT OF MOLECULAR BIOLOGY & GENETICS

Department of Molecular Biology and Genetics
Democritus University of Thrace
Alexandroupolis 2022

SNPop, ένα πακέτο εφαρμογών για την συλλογή, οπτικοποίηση και ανάλυση δεδομένων μονονουκλεοτιδικών πολυμορφισμών

Κυριάκος Ψαλλίδας

Διπλωματική Εργασία

Επιβλέπων Καθηγητής: Γεώργιος Μπουλουγούρης, Επίκουρος Καθηγητής

ΔΗΜΟΚΡΙΤΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΡΑΚΗΣ **ΣΧΟΛΗ ΕΠΙΣΤΗΜΩΝ ΥΓΕΙΑΣ**

ΤΜΗΜΑ ΜΟΡΙΑΚΗΣ ΒΙΟΛΟΓΙΑΣ & ΓΕΝΕΤΙΚΗΣ

Τμήμα Μοριακής Βιολογίας και Γενετικής
Δημοκρίτειο Πανεπιστήμιο Θράκης
Αλεξανδρούπολη 2022

Acknowledgements

I would like to thank my supervisor, Professor Boulougouris G. for introducing me to the wonderful field of computational biology, as well as for his invaluable advice and continuous support during my undergraduate thesis.

Abstract

The constant improvement of high throughput sequencing technologies (HTS), as well as the steep decline of their cost, has resulted in a data “revolution” in the field of genomics. Genomic data are now catalogued in databases faster than they can be interpreted and the volume of data is increasing exponentially. Of this data, a big part is Single Nucleotide Polymorphisms (SNPs) data, which entail for most of human genetic variation. SNP research requires the retrieval of large volumes of data, available in different genomic databases. However, bulk retrieval of SNP data from various databases is inefficient when done non-programmatically. Software with user-friendly Graphical User Interfaces (GUIs), able to retrieve SNP data, are the exception to the rule. Additionally, while they provide great functionality on their own, the interaction of their file outputs to produce complete data sets appears inefficient. The application package “SNPop”, presented in the thesis, includes two programmes with a user-friendly GUI. "SNPfinder" written in python allows for the accurate retrieval of data for desired SNPs, by utilizing different search filters set by the user. “SNPanalysis” is written in R and handles reliably the analysis and visualization of the SNP data. Importance is given to SNPanalysis’s association of catalogued haplotypes, from the five main populations of the 1000 Genomes project, with the phenotype characteristics of the selected SNPs, a function unavailable in other software. The available information varies between different SNPs and databases, as a result, incomplete associations can be made. Consequently, SNPop’s role is to paint a picture regarding the association of SNPs, populations, phenotypes and haplotypes. The result shouldn’t be taken for granted but as a direction for the user’s further research.

Η συνεχή βελτίωση της απόδοσης, καθώς και η ραγδαία μείωση του κόστους των τεχνολογιών αλληλούχησης μεγάλης κλίμακας έχει οδηγήσει σε μια "επανάσταση" δεδομένων στο πεδίο της γονιδιωματικής. Γενωμικά δεδομένα πλέον καταχωρούνται ταχύτερα από όσο μπορούν να αναλυθούν σε βάσεις δεδομένων, με τον όγκο της πληροφορίας να αυξάνεται εκθετικά. Ένα μεγάλος μέρος αυτού του όγκου δεδομένων αφορά τους μονονουκλεοτιδικούς πολυμορφισμούς (SNPs), οι οποίοι καλύπτουν το μεγαλύτερο μέρος της γενετικής διαφορετικότητας του είδους μας. Η μελέτη των SNPs απαιτεί την συλλογή μεγάλου όγκου διαφορετικών δεδομένων από βάσεις γονιδιωματικής. Ωστόσο, μεγάλοι όγκοι καταχωρημένων δεδομένων, από διάφορες βάσεις, δεν είναι εύκολα προσβάσιμοι μη-προγραμματιστικά. Λογισμικά με γραφικά περιβάλλοντα φιλικά προς τον χρήστη που συλλέγουν SNP δεδομένα αποτελούν εξαίρεση στον κανόνα και ενώ διαθέτουν υψηλή λειτουργικότητα χωριστά, η αλληλεπίδραση των αρχείων που παράγουν για την παραγωγή πλήρη δεδομένων, εμφανίζει εμπόδια. Το πακέτο εφαρμογών "SNPop", που παρουσιάζεται στην διπλωματική, περιλαμβάνει δύο προγράμματα με πλήρη φιλικό προς τον χρήστη γραφικό περιβάλλον. Το πρόγραμμα "SNPfinder" γραμμένο σε python, επιτρέπει την εύστοχη συλλογή δεδομένων για επιθυμητά SNPs, με την χρήση διάφορων κριτηρίων αναζήτησης που μπορεί να θέσει ο χρήστης. Το "SNPanalysis", γραμμένο σε R πραγματοποιεί αξιόπιστα την οπτικοποίηση και ανάλυση των δεδομένων. Βάση δίνεται στην ικανότητα συσχέτισης καταχωρημένων απλότυπων , από τους πέντε κύριους πληθυσμούς του προγράμματος 1000 Genomes, με τα φαινοτυπικά χαρακτηριστικά των επιλεγμένων SNPs. Η διαθέσιμη αποθηκευμένη πληροφορία ποικίλει σημαντικά ανάμεσα στα διάφορα SNPs και βάσεις, έτσι είναι πιθανό μη ολοκληρωμένες συσχετίσεις να πραγματοποιηθούν. Συμπερασματικά ο ρόλος του SNPop είναι ή αποτύπωση μιας εικόνας συσχέτισης SNPs, πληθυσμών, φαινοτύπων και απλότυπων η οποία δεν πρέπει να θεωρείται δεδομένη, αλλά να κατευθύνει την περεταίρω μελέτη του χρήστη.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 A brief recent history of genomics in relation to the human genome

The completion of the Human Genome Project (HGP) in 2003, one of the largest in scale efforts in science ensued strong foundations in the study of large, biological data and the first complete reference of the human genome [6]. Meanwhile, since deoxyribonucleic acid (DNA) sequencing was first introduced in 1977 [26]. Its power has increased and its cost decreased exponentially with the emergence of high-throughput sequencing (HTS) technologies in 2007. Technologies capable of sequencing multiple DNA molecules in parallel, allowing hundreds of millions of DNA molecules to be read at once [9]. Having a clear point of reference as well as the means to achieve a larger scope the 1000 Genomes Project, launched in January 2008. A multinational scientific endeavour to create the most comprehensive record of human genetic variation. In 2015 phase 3 of the project concluded, discovering 88 million variants in 2,504 individuals from 26 populations [36]. Continuing the path of large scale studies in 2018 the 100,000 Genomes Project, after a runtime of five years completed having sequenced and analyzed genomes of United Kingdom National Health Service (NHS) patients with cancer and rare diseases [40].

Considering the brief history mentioned, it is clear that a revolution occurred in the field of genomics. The human genome has become as accessible as ever and data regarding variations in the genome appear faster than they can be interpreted. For example, DbSNP, a database for single nucleotide variants, micro-satellites, and small-scale insertions and deletions,doubled its data size from 557,939,960 submissions to 907,237,763 between 2016-2017 and then again from 907,237,763 to 1,828,331,768 from 2017 to 2018 as its shown in figure (1.1) [7]. In 2021 DbSNP has 3,341,554,567 submissions [7]. The evident acute growth in the amount of data available on human genetic variation as well as variation in other species classified genomics as a "big data" field and sparked the interest for this thesis [34]. How can this data be

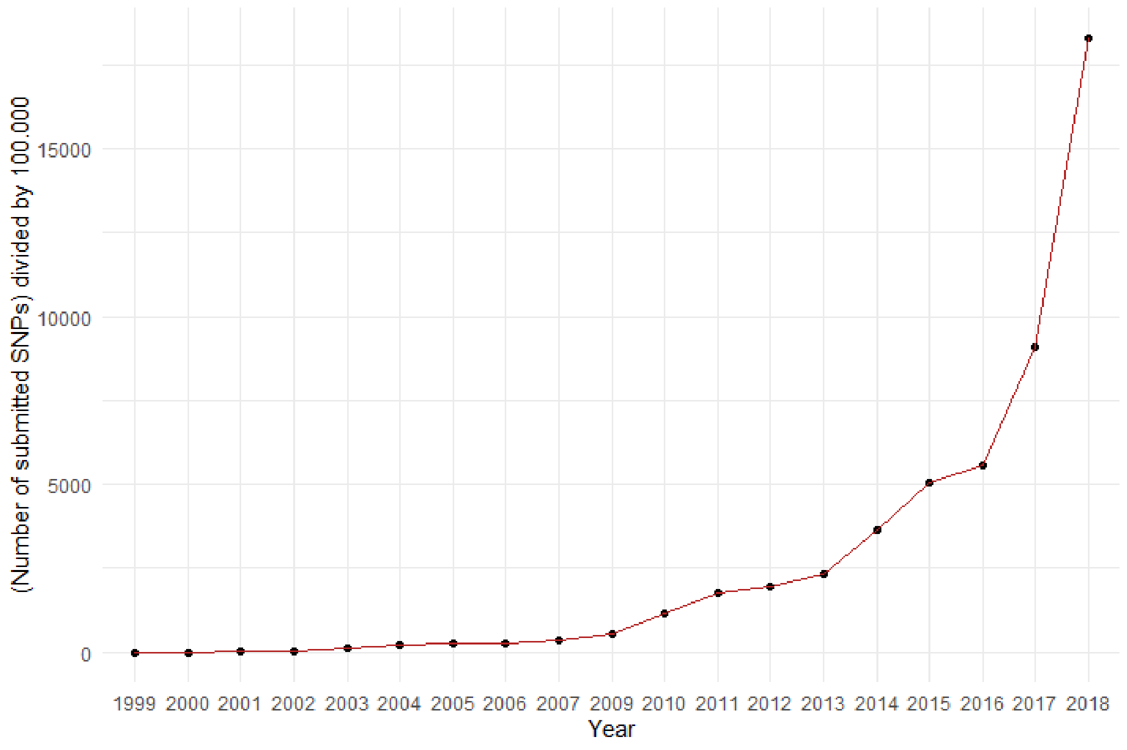


Figure 1.1: Submitted SNPs in DbSNP over the years

collected and understood in bulk for different populations? Is there a barrier for non power-users? And finally, if it exists, how can it be surpassed?

1.2 Theoretical Framework

1.2.1 Human genetic diversity

Although 99.9% of our genetic composition is identical as a species [29], the remaining 0.1% is critical since it correlates to our genetic variety. Of the 0.1% stated, >99.9% or 4-5 million variation sites consist of Single Nucleotide Polymorphisms (SNPs) and <0.01% or 2,100 to 2,500 variation sites are structural variants [1000 genomes]. Alternative alleles in these sites between individuals (within populations) account for 85%-90% of genetic diversity. The remaining 10%–15% is distributed throughout population groups [1]. Variation mechanisms are detailed thoroughly in the following chapters, in summary, the causes of genetic variation between individuals include mutations/polymorphisms and the independent assortment of variants except for variants in genetic linkage. Most SNP loci do not affect the phenotype, however, others affect a plethora of traits resulting in variation of important factors, namely disease susceptibility and treatment outcomes [29]. Consequently, discovering the possible alternative alleles for these sites and frequencies between individuals in populations and between whole populations provides numerous scientific applications. While structural variants are equally important, the scope of variation will narrow on SNPs from this point forward. This decision is based on the fact that SNPs assimilate 99.9% of the genetic variation found in humans. Subsequently, a much larger volume of data exists for SNPs which is crucial for the goal of this thesis.

1.2.2 SNPs in depth

Mutation, the random change in the nucleotides of the DNA sequence is the driving force of genetic variation. These changes are inherited and/or stem from mistakes by repair enzymes during DNA replication, chemicals and ionizing radiation. Somatic mutations appear in single cells throughout life and cannot be inherited. Contradictory germline mutations occur in the gametes and can pass from parents to offspring, appearing in all of the cells. Furthermore, variations that occur due to the alteration of a single nucleotide are called point mutations, shown in figure (1.2). SNPs are

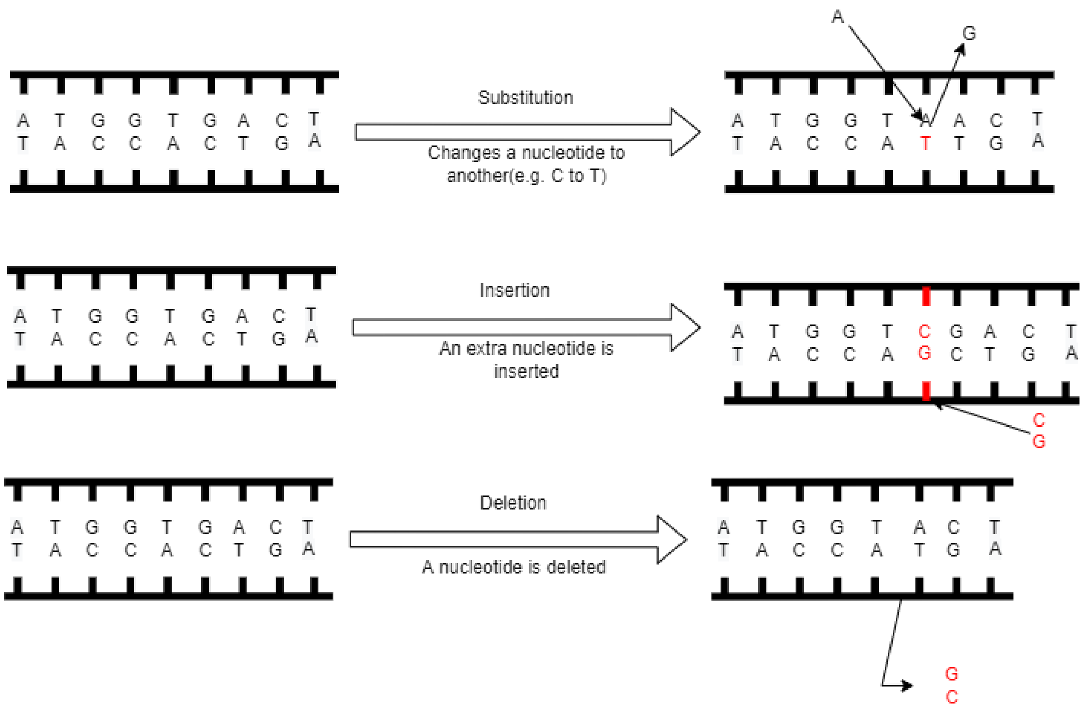


Figure 1.2: Types of point mutations

a loosely used term, often as an equivalent to point mutations. To preserve a clear scope, In this thesis SNPs are defined as single base loci (NOT mutagenic events) in the DNA sequence where:

- Different sequence variants (alleles) exist between individuals in some populations due to inherited germline substitution point mutations.

- Although an arbitrary requirement of >1% frequency for the least common allele in the population is often used to distinguish SNPs from rare point mutation loci, it won't be factored for this thesis, since it isn't universally applied [36][30].

Aiming to describe SNPs in detail a hypothetical nucleotide sequence is used,containing only one SNP, in its sixth base, sequence (1.1).

$$5' - AGCTA\mathbf{T}CA - 3' \tag{1.1}$$

Allelic count and frequency

If only one alternative allele, as shown in sequences (1.2), is observed in the population for the sixth base of sequence (1.1), this SNP would be a bi-allelic SNP with the alleles T, G.

$$1:\ 5' - AGCTA\mathbf{T}CA - 3'\ \ OR\ \ 2:\ 5' - AGCTA\mathbf{G}CA - 3' \tag{1.2}$$

While nearly all human SNPs are bi-allelic [2], multi-allelic variants exist. These carry three or more alleles. For example, if the SNP on the sequence 1.1 was tetra-allelic, four different alleles would be observed in the population, one for each possible nucleotide T, C, G, A.

$$\begin{gathered}1:\ 5' - AGCTA\mathbf{T}CA - 3'\\ OR\\ 2:\ 5' - AGCTA\mathbf{C}CA - 3'\\ OR\\ 2:\ 5' - AGCTA\mathbf{G}CA - 3'\\ OR\\ 3:\ 5' - AGCTA\mathbf{A}CA - 3'\end{gathered}$$

For both bi-allelic and multi-allelic SNPs, the most frequently observed allele is defined as the ancestral or major allele of the SNP, while the next most common in the population is characterized minor allele with its frequency defined as Minor Allele Frequency (MAF) [30].

Genotypes and inheritance

Humans are diploid organisms, thus they have two copies for each chromosome in each cell, one from each parent. The combination of the alleles a person has for a locus in each chromosome is defined as the person's genotype at that locus. As an illustration, if the parent's genotypes for the bi-allelic SNP ,shown in sequences 1.2, were **TG**. According to Gregor Mendel's theory of independent assortment of genes, homologous chromosomes are divided in half to generate haploid cells during meiosis as demonstrated in figure (1.3).

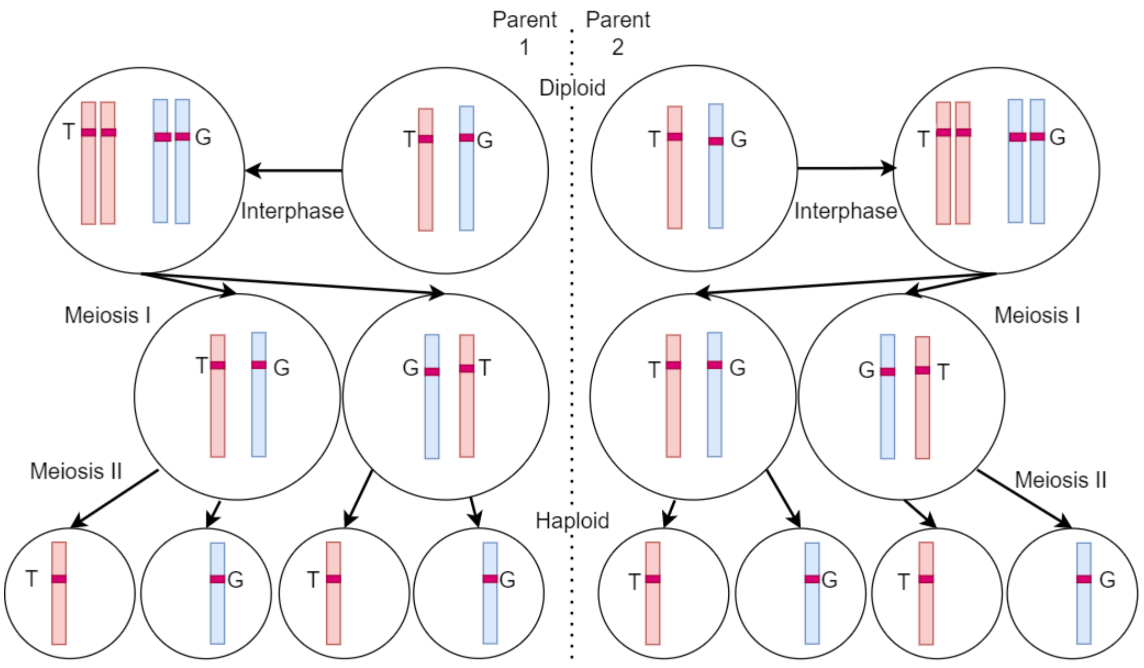


Figure 1.3: Example meiotic division

The parent's homologous chromosomes are copied during interphase forming sister chromatids. Meiosis I follows, the copied chromatids are separated and two diploid cells are created from each parent cell. In meiosis II the DNA isn't copied. The products are four haploid cells for each parent cell. These haploid cells represent the possible gametes[IGenetics : a Mendelian approach]. Since two of the gametes, one from each parent, form the zygote **G** and **T** are available from parent 1 as well as Parent 2, the offspring's genotype is either **GG** or **TT** or **TG**. For a bi-allelic SNP, only two alleles are available in a population, consequently, two distinct gametes exist for the loci. From the possible combinations of these gametes, three genotypes can exist in a population. Two having the same alleles (homozygous) and one having one of each (heterozygous).

Location and Function

SNPs are found naturally throughout a person's DNA averaging a frequency of one SNP per 1000 bases. However, their distribution isn't uniform. The vast majority of SNPs exist in non-coding areas of DNA, while only a small percentage is found in coding sequence. Most SNPs don't correspond to phenotype changes, however certain SNPs alter the genetic information in ways that directly affect the phenotype [37].

Non-coding region SNPs, as the name suggest, exist in genomic regions not translated into proteins. As depicted in figure (1.4). These include intergenic areas between genes and non-coding gene elements such as promoters, enhancers/silencers, 5'/3' untranslated regions and introns [37]. These elements cover about 97% of the

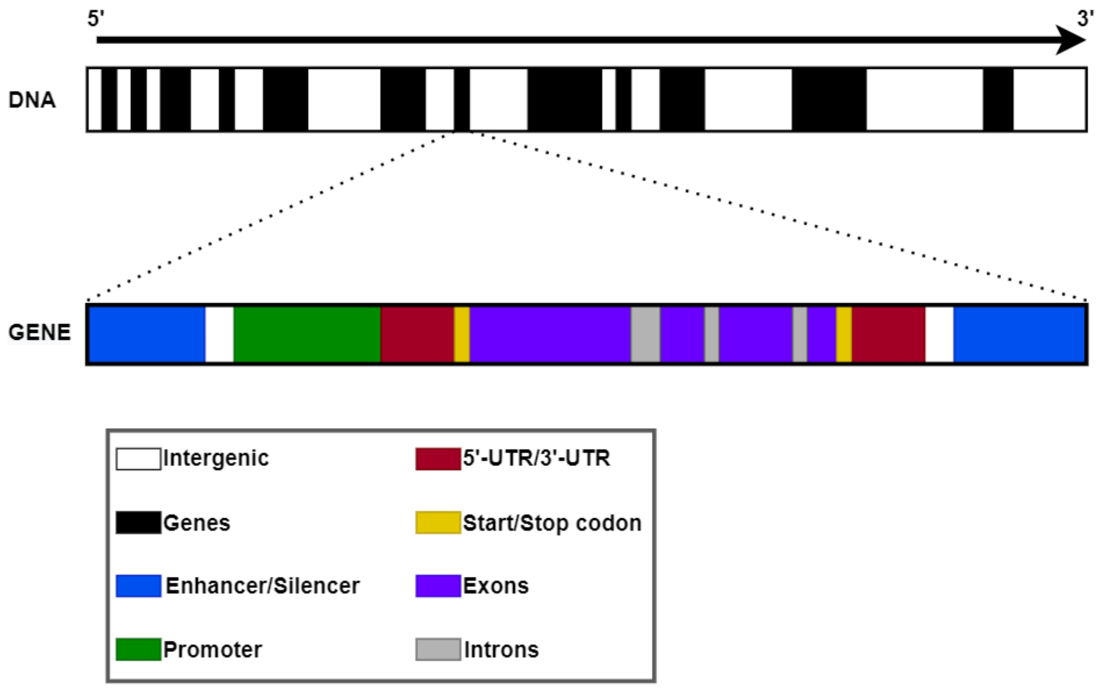


Figure 1.4: DNA and Gene regions

3 billion base pairs of our DNA. Although they were considered mostly functionless and 'junk' DNA, a term solidified in 1972, the Encyclopedia of DNA Elements (ENCODE) project in 2012 published data that assign biochemical functions for 80% of the genome [37].While ENCODE's definition of function was criticized and the number given might be an exaggeration [8], we know today that portions of non-coding DNA regulate gene expression. As a result, certain non-coding SNPs can result in unusual expression affecting the phenotype. These SNP are defined as expression Quantitative Trait Loci(eQTL)

Coding region SNPs, exist in exon sequences, depicted in purple in figure (1.4). These areas make up about 2-3% of our DNA sequence [37]. Assuming the gene in (1.4) had no introns, its theoretical exon DNA sequence is depicted in figure (1.5).Coding information stored as nucleotides in the double-stranded DNA act as a mould from which mRNA is created via the enzyme RNA polymerase. The nucleotides of mRNA are then read as triplets(codons) starting at the start codon(AUG) and ending at the stop codon(UAA/UAG/UGA), according to the genetic code. This results in the translation of the genetic information to amino acids, the building blocks of proteins. Coding region SNPs form three groups via their effect on the amino acid sequence.

- **Synonymous SNPs** are substitution variants that do not affect the amino

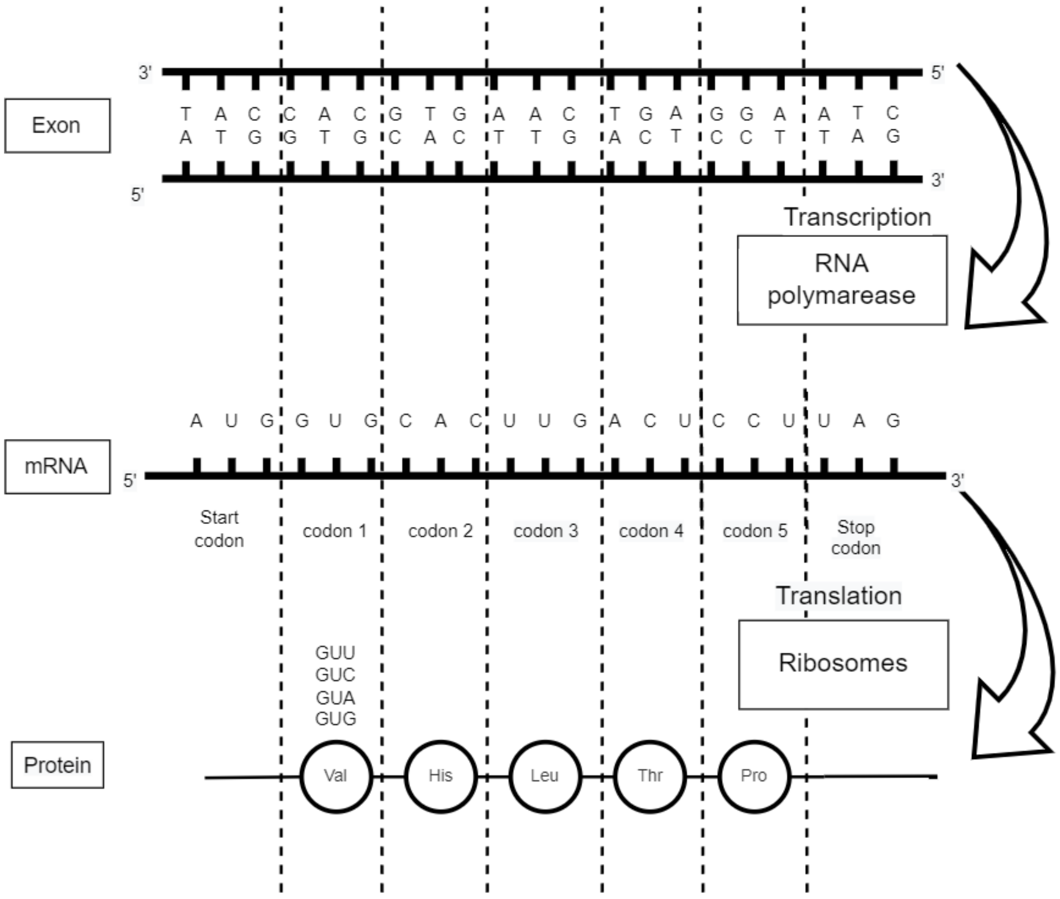


Figure 1.5: Exon transcription and translation

acid sequence of the gene. For example, if an SNP existed at the sixth base of the coding DNA strand in figure (1.5) with T as the minor allele then the first codons transcription goes as follows: CAT to GUA rather than CAC to GUG. However, the translation of the codon to amino acid remains unaffected since the codon GUA is also translated to the amino acid valine. While they are sometimes called "silent",some synonymous SNP can affect the splicing, stability and structure of mRNA, along with protein folding [15].

- **Missense SNPs** result in changes in the amino acid sequence via nucleotide substitution. If the example SNP mentioned before existed in the fifth base of the exon rather than the sixth then the first mRNA codon would be GAG, which translates to the amino acid glutamine rather than valine. Amino acid changes possess strong impact on the function of the protein, often leading to illness, such as sickle cell disease [21].

- **Nonsense SNPs** produce a premature stop codon. Stop codons mark the end of translation. Normally they exist at the end of exons designating the end of the protein's amino acid sequence. However a substitution SNP, with T as the minor allele, at the eleventh base of the coding DNA strand in figure (1.5) produces a stop UAG codon instead of the codon for Leu and the amino acid sequence coded by the gene would be Val-His rather than Val-His-Leu-Thr-Pro. Nonsense SNPs have a stronger impact on function, since they lead to truncated or inactive proteins [42].

Discovering SNPs

SNP discovery revolves around the scanning of genomes for undiscovered variants by high-throughput sequencing (HTS) technologies. These can determine the nucleotide sequence in large sections of DNA or even the whole genome. To achieve that, distinct ways of differentiating the nitrogenous bases are used.sequencing by ligation consists of a fluorescent probe sequence that hybridizes to a DNA fragment and is then ligated to an adjoining oligonucleotide for imaging.The fluorophore's emission spectrum reveals the identity of the base or bases complementary to certain positions within the probe [9]. Alternatively, a polymerase is used in sequencing by synthesis, and a signal, such as a fluorophore or a change in ionic concentration, identifies the incorporation of a particular nucleotide into an elongating strand [9].After the sequencing of a large population of individuals, SNP loci can be discovered by identifying single nucleotide variants between the genomes and thus the loci's alleles and genotypes. However, sequencing provides no information about the position of SNP alleles between homologous chromosomes. For example in the following identified sequence with two SNPs,only the possibilities of allelic arrangement between homologous chromosomes are known.

$$ATC\mathbf{C/A}GT\mathbf{T/G}$$

$$\frac{ATC\mathbf{C}GT\mathbf{G}}{ATC\mathbf{A}GT\mathbf{T}} \; OR \; \frac{ATC\mathbf{C}GT\mathbf{T}}{ATC\mathbf{A}GT\mathbf{G}} \tag{1.3}$$

The group of alleles on each of the homologous chromosomes inherited from a parent is defined as a haplotype. In this case either **CG**, **AT**, **CT** or **AG**.

1.2.3 SNPs in population genetics

Population genetics is the study of the genetic makeup of populations, which includes genotype and phenotype distributions. In this scope, only determining the function and inheritance of isolated SNPs in individuals, detailed in section (1.2.2), is not enough. For instance, in complex traits, the function of various SNPs is linked to a specific phenotype. Additionally many SNPs are inherited together in haplotype groups of different frequencies between populations due to their distinct evolutionary history [2]. Considering these facts, understanding the mechanism of nonrandom association of SNP alleles at different loci is key for the analysis of SNPs in population genetics.

As stated, the principle of independent assortment of genes mentions that homologous chromosomes are divided in half to generate haploid cells during meiosis. However, when the inheritance patterns of more than one SNP on the homologous pair is of interest, the depiction of independent assortment in meiosis as demonstrated in 1.3 misses a key feature, the mechanism of crossing-over. By adding one more hypothetical SNP on the same chromosome, **A/C** the genotypes of Parent 1 and 2 become **TGAC** , as shown in figure (1.6).

The haplotypes of Parent's 1 are **TA**, **GC** and Parent's 2 are **GA**, **GC**. Based on the principle of independent assortment we expect to see the same allelic combinations in the gametes produced. However, when crossing-over occur at the beginning of meiosis I, genetic material is exchanged between the non-sister chromatids of Parent's 1 homologous pair. These chromosomes carrying exchanged DNA are called recombinants and in meiosis II produce gametes with new allele haplotypes that weren't present in the maternal and paternal chromosomesof Parent 1, namely, **GA** and **TC**. These can be inherited by the offspring increasing the genetic variation of a population. Crossing-over does not always occur and its frequency is neither constant nor random across different SNP combinations, it's linked to the distance of loci on the chromosome. The further apart two loci are on a chromosome the frequency of crossing-over between them increases. In contrast, the closer two loci are on a chromosome the crossing-over frequency decreases. Additionally, when two SNPs are in

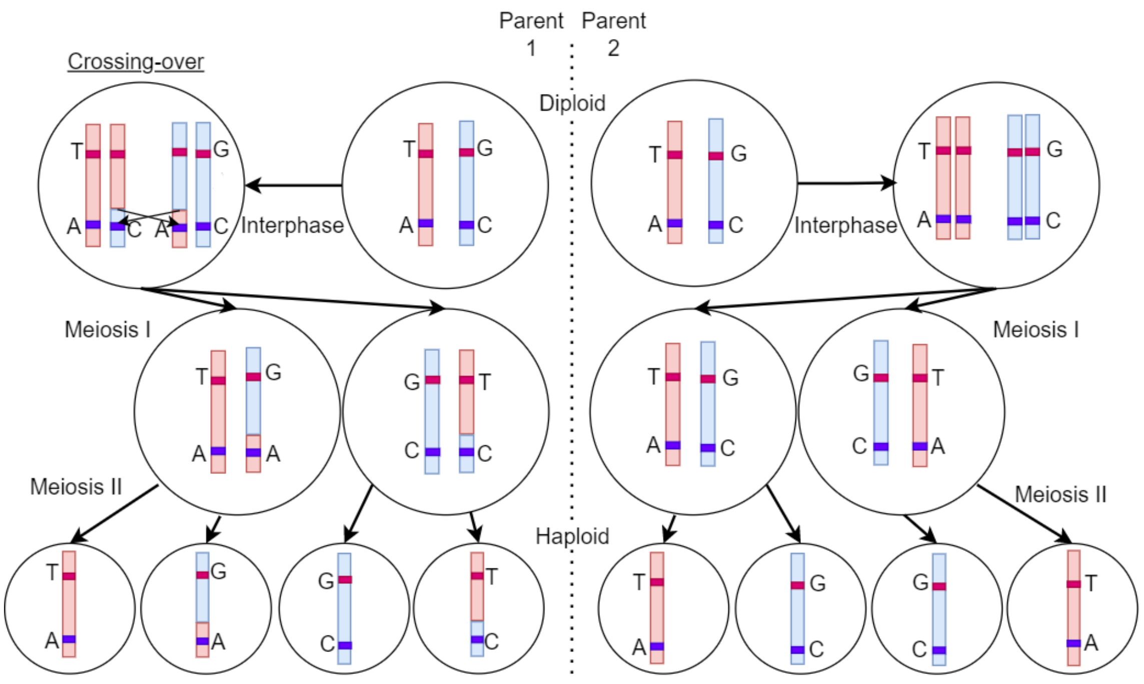


Figure 1.6: Crossing-over process

close proximity the frequency of crossing-over can be zero and the SNP alleles are in genetic linkage, they are co-inherited together in a haplotype.

Linkage disequilibrium (LD)

LD is a measurable population parameter concerning the non-random association of alleles of different loci [31]. LD is usually calculated pairwise between two distinct loci and its coefficient is D, which stands for the difference between the observed and expected frequency of their allelic haplotypes. In order to clarify LD calculations we consider the SNPs **T/G** and **A/C** from figure (1.6). Following genetic nomenclature the frequencies of their observed allelic haplotypes in a population are:

Haplotype	Frequency
TA	x_{11}
TC	x_{12}
GA	x_{21}
GC	x_{22}

Additionally their allelic frequencies are:

Allele	Frequency
T	$p_1 = x_{11} + x_{12}$
G	$p_2 = x_{21} + x_{22}$
A	$q_1 = x_{11} + x_{21}$
C	$q_2 = x_{12} + x_{22}$

Under crossing-over and according to the principle of independent assortment, the alleles are not associated and their distribution in haplotypes in the population is random. Consequently the expected haplotype frequencies equeal to the product of the allelic frequencies in each haplotype:

$$\mathbf{TA} = x_{11} = p_1 q_1 \;\; \mathbf{TC} = x_{12} = p_1 q_2 \;\; \mathbf{GA} = x_{21} = p_2 q_1 \;\; \mathbf{GC} = x_{22} = p_2 q_2$$

When crossing-over doesn't always occur. The loci's LD can be calculated for each haplotype as the deviation (**D**) between the observed haplotype frequency and the corresponding allelic frequencies expected under random association. The observed haplotypes are:

$$\mathbf{TA} = x_{11} = p_1 q_1 + D_{\mathbf{TA}} \;\; \mathbf{TC} = x_{12} = p_1 q_2 - D_{\mathbf{TC}}$$
$$\mathbf{GA} = x_{21} = p_2 q_1 - D_{\mathbf{GA}} \;\; \mathbf{GC} = x_{22} = p_2 q_2 + D_{\mathbf{GC}}$$

Solving for $D_{\mathbf{TA}}$:

$$D_{\mathbf{TA}} = x_{11} - p_1 q_1$$
$$= (Observed\ frequency\ of\ haplotype\ \mathbf{TA}) - (Expected\ frequency\ of\ haplotype\ \mathbf{TA})$$

Each allele pair has its own D. Furthermore, the allele frequencies at both loci and the haplotype frequencies must sum up to one. As a result In case both loci are bi-allelic, as almost all SNPs are, $D = D_{\mathbf{TA}} = \breve{}D_{\mathbf{TC}} = \breve{}D_{\mathbf{GA}} = D_{\mathbf{GC}}$ is true and D is utilized without a subscript. The sign of D is arbitrary and depends on whatever allele pair one begins with. The value of D can be $-0.25 \leq D \leq 0.25$. Since D can take negative values a normalized version of it is usually used to make D values range from 0 to 1, D' [31].

$$D' = \frac{D}{D_{max}} \;\; where \;\; D_{max} = \begin{cases} min(p_1 q_2, p_2 q_1) \;\; when \;\; D > 0 \\ min(p_1 q_1, p_2 q_2) \;\; when \;\; D < 0 \end{cases}$$

- If $D' = 0$, the two SNPs are in linkage equilibrium, no association exists between the alleles of the two loci. They are either on different chromosomes, or crossing-over always occurs.
- The value of D' gets further from 0 the closer two loci exist on a chromosome.
- D' can only be used to determine the random or non-random association between SNPs but not to determine complete association.

In order to determine the presence or absence of complete LD and hence association between two SNPs the correlation-coefficient r^2 is used with a value range of 0 to 1.

$$r^2 = \frac{D^2}{p_1 p_2 q_1 q_2}$$

- If $r^2 = 0$ the two SNPs are in complete linkage equilibrium and no association exist between their alleles.
- If $r^2 = 1$ the two SNPs are in complete linkage disequilibrium (genetic linkage) and their alleles are always co-inherited.

Genotype phasing

Another method of determining association between SNPs is by observing it via determining the SNP haplotype frequencies of populations after sequencing a substantial quantity of their individual's genomes. As stated in section (1.2.2), sequencing provides information about the genotypes of SNPs, but not about their allelic distribution in haplotypes. SNP genotypes identified after sequencing are defined as unphased and the process of of inferring haplotypes from genotype data is termed phasing. Phasing can be accomplished using family data, computational statistical estimation or often both. When the genotypes of the parents as well as the offspring are sequenced then using family trio data the haplotypes can be identified, when possible, via determining the possible inheritance patterns for the alleles. For example if the offspring's sequence is ATC**C/A**GT**T/G**, while the mother's is ATC**C/C**GT**T/G** and the father's is ATC**C/A**GT**T/T**. The offspring could have inherited the allele **A** of the first SNP only from the father since the mother is homozygous for the allele **C**. Considering the second SNP the allele **G** could have been inherited only from the mother since the father is homozygous for the allele **T**. As a result the offspring's haplotypes can be identified as:

$$\begin{cases} ATC\mathbf{A}GT\mathbf{T} \;\; From \;\; the \;\; father \\ ATC\mathbf{C}GT\mathbf{G} \;\; From \;\; the \;\; mother \end{cases}$$

Computational phasing utilizes statistical algorithms to determine haplotypes. Most phasing software implements algorithms to solve Hidden Markov Models(HMMs). Markov models define a chain of possible events in which the probability of each event depends only on the state acquired in the previous event. When it's not possible to observe the states of a Markov model, but only the result of some probability function of the model's states, an HMM is used. In an HMM model regarding genotype phasing the unknown haplotypes are treated as the hidden states of the model. The haplotypes accuracy is repeatedly evaluated backwards by algorithms via their dependence on observable genotype data until an accurate assessment of the individual's haplotypes is made [33]. Recognised phasing software include SHAPEIT, PHASE and BEAGLE.

1.2.4 Scientific applications

Considering what has been discussed in the previous sections, it's apparent that identifying and analysing SNP data provides crucial applications in many fields, namely:

Pharmacogenetics and targeted therapy

By pinpointing the distribution of drug metabolism related SNP haplotypes in populations, targeted therapy could be suggested(e.g. different guidelines in dosage and/or alternative drug usage) as well as further genetic screening for individuals in risk populations. The gene arylamine N-acetyltransferase 2(NAT2) for example, which is active in the pharmacokinetics of various compounds is found mostly with its slow metabolism SNP alleles in European, west and north Asian populations and generally with its fast metabolism SNP alleles in East Asian populations, while in Africa and America high heterogeneity is observed between populations [25].

Identification and diagnosis of genetic factors contributing to disease and complex traits

Genome-wide association studies (GWAS) are case-control, phenotype first studies based on whole-genome screening. Comparison of SNPs between populations of healthy (control) individuals and populations affected by a disease (case) is utilized to determine phenotype-genotype relations and hence loci and alleles in high LD haplotypes, responsible for the disease. Over than 5,700 GWAS have been completed [11], with significant discoveries, such as underlying variants responsible for Crohn's disease, rheumatoid arthritis, hypertension, diabetes and bipolar disorder [41][39].

Forensic science

The polymorphic attributes of Short Tandem Repeats (STRs), DNA sequence repeats less than seven base pairs in length, have been used in DNA-profiling. The method is based on Polymerase Chain Reaction (PCR) for specific highly polymorphic STRs. The PCR products from the DNA extracted from the crime scene are then compared in length with the products from the suspect DNA using southern blotting. This assists in the identification of perpetrators of crime or absolves an innocent person from suspicion. Additionally SNPs cab used as a means of comparison rather than STRs since research supports that they offer better results for degraded DNA [13].

Evolutionary biology

Selection on SNPs in high LD is co-dependent. For example, when an SNP allele is in favourable selection, then its frequency in a population increases.Neutral SNP alleles in close proximity also rise in frequency due to the strong LD between them. As a result, these alleles tend to follow the fixation (100% frequency) of the advantageous allele and the genomic region around the latter losses variation. These areas can be detected and used for evidence of selection, mapping our evolutionary past [12].

1.2.5 Cataloguing SNPs

The abundant nature of SNPs, as well as the large scale reference panels needed to interpret them for scientific applications, defined the need for their cataloguing. As illustrated in figure (1.7), newly discovered SNPs are submitted to raw data primary

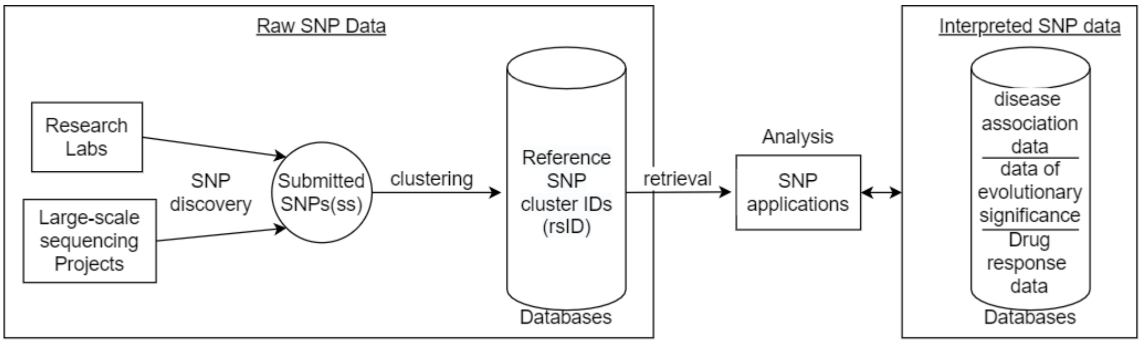


Figure 1.7: Flow of SNP data

databases. When overlapping reports are received, they are integrated into the same SNP, which is catalogued with a unique rsID accession number [31]. From this point, they are publicly available to be retrieved and analyzed by studies detailed in the prior section. These studies, in turn, produce interpreted SNP data to be archived in secondary databases and retrieved for field specific utilization. Examples of popular primary and secondary databases are presented below.

Primary SNP database:
Single Nucleotide Polymorphism Database (dbSNP): the largest database regarding human variation is dbSNP, created in 1998 by the American National Center for Biotechnology Information(NCBI) [30] . It is the single source of identified variants by defining the reference SNP cluster ID archival system (rsID). RsIDs in dbSNP can be categorized by their variation class, somatic or germline origin and minor allele frequency. Furthermore, for each rsID dbSNP provides info about its variant type, alleles, chromosome position, Gene and population sample. When available, dbSNP also provides the minor allele frequency for the sampling population or populations and the study that submitted it. At the time of this thesis, it contains 1,076,992,604 rsIDs, of which 930.786.700 are germline single nucleotide substitutions(dbsnp site), our definition of SNPs mentioned in section (1.2.2). For reference, part of the data available in dbSNP for rs1801279:

SNP rsID	rs1801279
Variant type	Single Nucleotide Variation(SNV)
Alleles	G>A
Chromosome	chr 8 : 18400194(base position)
Gene	NAT2
Functional Consequence	Missense Variant
Minor allele frequnecy(MAF)	A=0.0278 (1000Genomes)

Table 1.1: Example: rs1801279 data from dbSNP

In addition to the data mentioned above, when available , dbSNP presents some of the data available in other secondary databases regarding SNPs. Specifically IGSR (or other databases of population SNP variation) and ClinVar.

Secondary databases:
For population genetics
The International Genome Sample Resource (IGSR): was created in 2015 by the European Bioinformatics Institute after the completion of the 1000 Genomes Project and represents the world's largest open repository of human population variation. data to store the population data of its identified variations as well as to expand them by incorporating new populations. Data in IGSR can be filtered by population, data collection project and the technology used. Additionally,if available, the database provides not just the alleles but also the phased genotype data of individuals in the populations [5].

Population	Allele: frequency (count)	Genotype: frequency (count)
All individuals (ALL)	G: 0.972 (4869) A: 0.028 (139)	G|G: 0.947 (2372) A|A: 0.003 (7) A|G: 0.050 (125)
African (AFR)	G: 0.897 (1186) A: 0.103 (136)	G|G: 0.805 (532) A|A: 0.011 (7) A|G: 0.185 (122)
American (AMR)	G: 0.997 (692) A: 0.003 (2)	G|G: 0.994 (345) A|G: 0.006 (2)
East Asian (EAS)	G: 1.000 (1008)	G|G: 1.000 (504)
European (EUR)	G: 0.999 (1005) A: 0.001 (1)	G|G: 0.998 (502) A|G: 0.002 (1)
South Asian (SAS)	G: 1.000 (978)	G|G: 1.000 (489)

Table 1.2: Example: rs1801279 data from IGSR

For genotype-phenotype and disease association
ClinVar: is hosted by the NCBI and is a repository for interpreting the clinical relevance of SNPs and other variations for reported diseases [17]]. Data submitted to CLinvar stem from clinical testing facilities, research laboratories, and other

databases. Data for SNPs in ClinVar include the clinical interpretation, review status, condition, submitter and supporting information for the claims listed.

Interpretation	Review status	Condition	Submitter	Supporting information
drug response	no assertion criteria provided	slow acetylation	OMIM	Publications:PubMed(2)

Table 1.3: Example: rs1801279 data from ClinVar

GWAS Catalogue: was founded in 2008 by the American National Institute of Health (NIH) to archive the results of GWAS studies (1.2.4). In 2021 it contains 191,959 SNPs and 320,626 associations to traits. For each archived SNP, GWAS catalogue provides data about the risk allele, the trait associated with it, LD data with other variants and the GWAS studies from which these information arise [3].

1.3 Tools to retrieve large-scale catalogued SNP data

1.3.1 Tools provided by databases

Specific data of individual identified SNPs can easily be accessed via the web interface of databases, as detailed in section (1.2.5). However, for SNP data to make sense, large volumes of data need to be collected. Databases host File Transfer Protocol (FTP) sites, allowing the transfer of large computer files from a server to a client on a computer network. While it's possible to use the FTP site to manually download SNP data, the most efficient retrieval of SNP data is achieved programmatically through an application programming interface (API) in combination with the knowledge of the database's query method. NCBI for example provides the E-utilities (Entrez Programming Utilities) API[27], for the Entrez query system:

$$https://eutils.ncbi.nlm.nih.gov/entrez/eutils/andtheEnterez$$

The API converts a set of specific input query data in the form of a URL into values that various NCBI software components need to find and obtain the requested data. Services supported by E-utilities for SNP data access include:

- ESearch: $https://eutils.ncbi.nlm.nih.gov/entrez/eutils/einfo.fcgi?$
  $db = SNP\&term = Enterez\ query/$
  Provides the user with the corresponding rsIDs from dbSNP based on their search query.

- ESummary: *eutils.ncbi.nlm.nih.gov/entrez/eutils/esummary.fcgi* $db = SNP\&id = [rsID\ list]/$
  Uses a list of rsIDs to return their dbSNP data summary.

- EFetch : *eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi* $db = SNP\&id = [rsID\ list]/$
  Uses a list of rsIDs to return their complete data from the dbSNP database.

SNP data obtained using FTP can be in Extensible Markup Language (XML), JavaScript Object Notation (JSON) and Variant Call Format (VCF) file types. XML and JSON are both text-based hierarchical structures in a human-readable format. VCF is a variation data exclusive file type used to store sequence variations.

1.3.2 A time consuming barrier

FTP sites hosted by databases solve the issue of data volume. However another problem still exists, they provide specific data, while to proceed to SNP analysis, data types from different sources need to be collected in a single working environment. The solution of this issue usually requires work either pragmatically, to automate the gathering and formatting of data obtained from different API services. Or manually through exhaustively downloading, cutting and pasting different data with the added obstacle of file types, since not all databases provide data in the same format. Additionally the query systems of databases may differ, increasing the time, effort and knowledge needed to collect whole data-sets.

1.3.3 Tools to overcome the barrier

Ensembl genome database project

The most widespread tool that breaches this barrier is EBI's Ensembl genome database project [10]. A whole collection of genomic data for vertebrate species, including humans. For example, for rs1801279, discussed in database context in section (1.2.5), Ensembl provides data and their visualization about its genomic context, genes and regulation, flanking sequence, population genetics, phenotype data, phased genotypes, LD, phylogenetic context and relative citations. This data can be accessed in bulk programmatically via provided API services and manually via Ensembls API. Furthermore, Ensembl provides two tools of significant importance, since they provide ease of access to data through simple web interfaces.

BioMart, a data management system that is also available in a simple interactive web interface format, allowing researchers to seek and retrieve complex data without any programming requirement. Regarding SNP variants, BioMart provides the following options for researchers to filter their search and the information included in the output, detailed in tables 1.3.3 and 1.5 respectively.

Region filters:	Chromosome, Coordinates, Marker, Multiple regions
General filters:	Variant source, Variant name, Phenotype, Clinical significance, Minor Allele Frequency, Study name and type, Variant set name, Variant supporting evidence, Phenotype and it's source, Distance to transcript, scores from external SNP software
Gene associated variant filters:	Gene ID , Variant consequence
regulatory region associated filters:	Regulatory ID, Regulatory consequence, Binding matrix ID, Motif consequence

Table 1.4: Types of SNP search filters in BioMart [32]

Variant associated information:	Variant name, Variant source, Variant source description, Chromosome/scaffold name, Chromosome/scaffold position start (bp), Chromosome/scaffold position end (bp), Strand, Variant alleles, MapweightVariant name, Variant source, Variant source description, Chromosome/scaffold name, Chromosome/scaffold position start (bp), Chromosome/scaffold position end (bp), Strand, Variant alleles, Mapweight, Variant supporting evidence, Ancestral allele, Minor allele (ALL), Global minor allele frequency (all individuals), Global minor allele count (all individuals), Clinical significance, Synonym name, Synonym source, Synonym source description, Associated variant names, Study name, Study type, Study External Reference, Study Description, Source name, Associated gene with phenotype, Phenotype name, Phenotype description, Associated variant risk allele, P value, Variant Set Name, Variant Set Description, Title, Authors, Year, PubMed ID, PMC reference number (PMCID), UCSC ID, Digital Object Identifier, Variant supporting evidence, Ancestral allele, Minor allele (ALL), Global minor allele frequency (all individuals), Global minor allele count (all individuals), Clinical significance
Gene associated information:	Gene stable ID, Gene Name, Transcript stable ID, Transcript strand, Biotype, Variant consequence, Consequence specific allele, Protein allele, Variant start in cDNA (bp), Variant end in cDNA (bp), Variant start in translation (aa), Variant end in translation (aa), Variant start in CDS (bp), Variant end in CDS (bp),Distance to transcript, PolyPhen prediction, PolyPhen score, SIFT prediction, SIFT score
Regulatory region associated information:	Regulatory feature stable ID, Regulatory feature allele stringRRegulatory feature stable IDRegulatory feature allele string, Regulatory feature consequence type, Motif Feature stable ID, Motif feature allele string, Motif feature consequence type, High information position, Motif score change, Motif name, Motif position

Table 1.5: Types of SNP attribute filters in BioMart

For instance, if all the SNPs with minor allele frequency above 0.1, existing on the NAT2 gene, associated with drug response were needed for a study. A researcher can retrieve the SNPs by utilizing the filters: $Clinical\ significance \rightarrow drug\ response$, $Minor\ Allele\ Frequency \rightarrow \geq 0.1$, $Gene\ ID \rightarrow NAT2$. The search results can be further tailored to the needs of the user by selecting from a list of SNP data types for retrieval. These include but are not limited to: the variant name, variant source chromosome name, chromosome position start, chromosome position end, alleles, minor allele frequency, clinical significance, variant synonyms, phenotype annotations and gene annotations [32]. However, BioMart's web interface does not provide access to phased genotype, haplotype, or LD data about SNPs.

LDlink

Ldlink is a population LD analysis tool provided by NIH [18]. It's available both as a web-interface, as well as a package for the R programming language. LDlink references data from 5000 haplotypes of individuals spanning 26 populations provided by the 1000 Genomes phase 3, without the need to access the non user-friendly VCF files containing them. These data are utilized in ten tools. Tools able to analyze multiple SNPs across populations are showcased below:

- **LDexpress**, associates imported SNPs with gene expression in multiple tissue types.
- **LDhap**, calculates the haplotype frequencies in each population for a list of SNPs.
- **LDmatrix**, Creates pairwise linkage disequilibrium heatmap matrices.
- **LDtrait**, Finds out if a set of imported SNPs has been linked to a characteristic or illness in the past.
- **LDpop**, provides allele frequencies and LD across 1000G populations.

1.4 Goal and Outline

The goal of this thesis is to create an all in one user friendly application package with an efficient pipeline for the retrieval, visualization and small scale analysis of SNP data in the scope of population genetics which will bridge gaps existing in already established pipelines. To achieve this purpose, these core ideas made the framework of the software:

- Event-driven programming.

- GUI implementation
- The gathering, handling and saving of SNP data from various API services and databases in one file form.
- The use of methods and tools both in Python and R.

In chapter 2 the methods and logic used to build the apps, along with important code snippets, will be detailed. Following that, in chapter 3, the results of an example retrieval and analysis using the application package will be presented. Finally in chapter 4, the quality of the application's package results and pipeline efficiency will be assessed in comparison to existing ones.

Chapter 2

Methodology

2.1 Algorithm overview

Since the goal is a user-friendly tool, for the collection, visualization and small scale analysis of SNP data about population genetics, the software needed to provide the user tailored access to such data. As illustrated in figure (2.1), this is achieved by

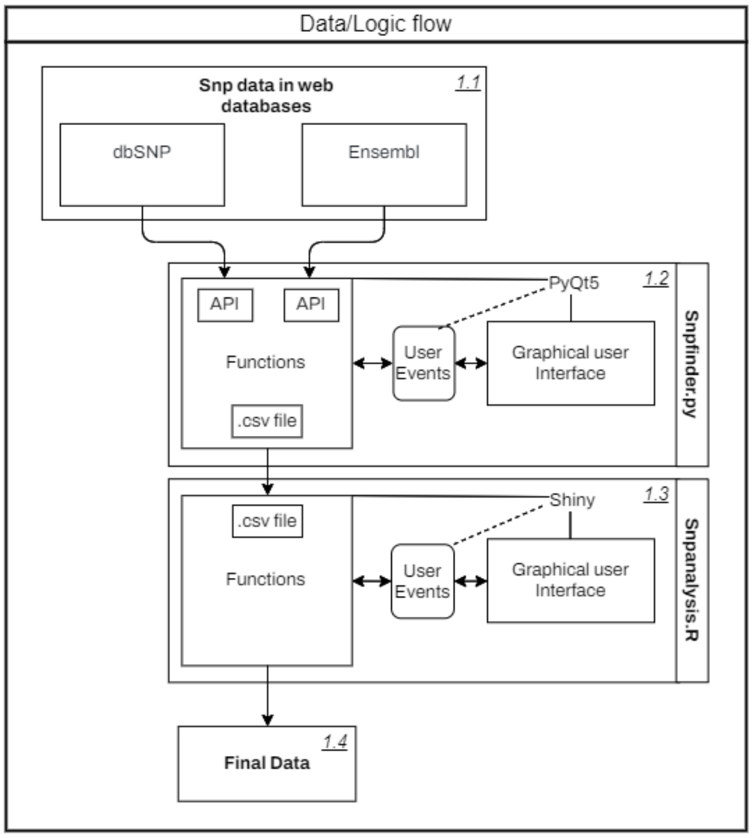


Figure 2.1: Algorithm overview

implementing various APIs services provided for dbSNP and Ensemble in an event-driven programming environment. Two distinct programs were coded to carry the task of retrieval and visualization, analysis respectively. *Snpfinder* is written in python and utilizes the PyQt5 framework for an intuitive graphical user interface (GUI) [22], as well as its threading capabilities to retrieve SNP data from multiple sources faster and process them to their final form in a single csv file. *Snpanalysis* is written in R and makes use of Shiny for a GUI [24][23]. Since R provides various packages for statistical computing and graphics, *Snpanalysis* accepts *Snpfinder's* CSV as input and covers the visualization and analysis workload.

```
Note
In the following sections, only some important code snippets for the
functionality and logic of the software are showcased.
The full code is available under GNU General Public License v3.0
at https://github.com/Cinnamonkk/SNPop
```

2.2 Data of choice

The data source selected for the algorithm is the secondary SNP data set of which the 1000 Genomes project provides a population frequency in dbSNP. This data set was chosen since the 1000 Genomes Project provides additional data for specific populations, as well as phased genotypes in other databases.

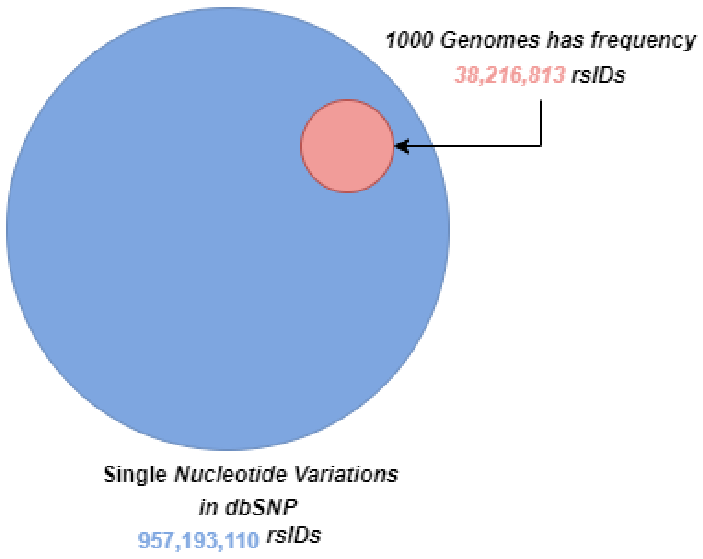


Figure 2.2: Data source

2.3 API calls

2.3.1 SNP retrieval base API

As a starting point the SNPs of these SNPs needed to be retrieved, a base API URL using the E-utilities ESearch API service plus certain Entrez filters were utilized for this goal [28]:

SNP Search API URL

```
#base API call
server = 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?'
#API extension
extension = 'db=snp&term=1000genomes+has+frequency+filter[Filter]+
             AND+snv[SNPClass]'
```

The "$db = SNP$" filter, informs the NCBI servers that the target database for data retrieval is dbSNP. Additionally the "$term = 1000genomes\ has\ frequency\ filter[Filter]\ AND\ snv[SNP\ Class]$" informs the Enterez system to retrieve the SNPs of SNPs that meet these criteria, this discards any SNPs with no data in the 1000 Genomes project and non-single and non-substitution variants. Using the requests module in python we retrieve the corresponding SNPs in JSON format [14]:

SNPs Search call

```
import requests
#base API call
server = 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?'
#API extension
extension = 'db=snp&term=1000genomes+has+frequency+filter[Filter]+
             AND+snv[SNPClass]&retmode=json'
#data request
r = requests.get(server+extension)
#passes data to a variable
data = json.loads(r.content)
```

2.3.2 Data retrieval base APIs

This API call returns the SNP rsID numbers but no data for any of them. To obtain the data additional API calls are required. Since Ensemble, as discussed, provides the largest repository of SNP data types, Ensemble's REST API Endpoints were chosen for this step [43]. Specifically the 'GET variation/species/id' call which returns variation characteristics of a species utilizing variation identifiers. Using the SNPs obtained from the Search API call,forms of the API call were implemented for different data types.

Genomic & population data

Genomic & population data call

```
import requests
#base API call
server = "https://rest.ensembl.org"
#API extension
extension = "/variation/homo_sapiens?pops=1"
data = '{"ids":"SNPs"}'
```

```
headers = {"Content-Type":"application/json","Accept":"application/json"}
#data request
r = requests.post(server+ext, headers=headers,data=data)
#passes data to a variable
data = json.loads(r.content)
```

Genotype Data

Genotype data call

```
import requests
#base API call
server = "https://rest.ensembl.org"
#API extension
ext = "/variation/homo_sapiens?population_genotypes=1"
data = '{"ids": "SNPs"}'
headers = {"Content-Type": "application/json","Accept": "application/json"}
#data request
r = requests.post(server+ext, headers=headers,data=data)
#passes data to a variable
data = json.loads(r.content)
```

Phenotype Data

Phenotype data call

```
import requests
#base API call
server = "https://rest.ensembl.org"
#API extension
ext = "/variation/homo_sapiens?phenotypes=1"
data = '{"ids": "SNPs"}'
headers = {"Content-Type": "application/json","Accept": "application/json"}
#data request
r = requests.post(server+ext, headers=headers,data=data)
#passes data to a variable
data = json.loads(r.content)
```

2.4 Snpfinder

Snpfinder, as seen in 2.1, handles the user's interaction with the APIs mentioned in sections (2.3.1 and 2.3.2), as well as the processing of the data obtained to a single CSV file. This was accomplished with the use of PyQt5's signals and slots system [22], where GUI elements send or respond to signals to or from internal functions according to user events. The API calls and results discussed could now be tailored around specific user inputs utilizing the four different GUI element groups shown in figure (2.3) and internal functions detailed in the following subsections.

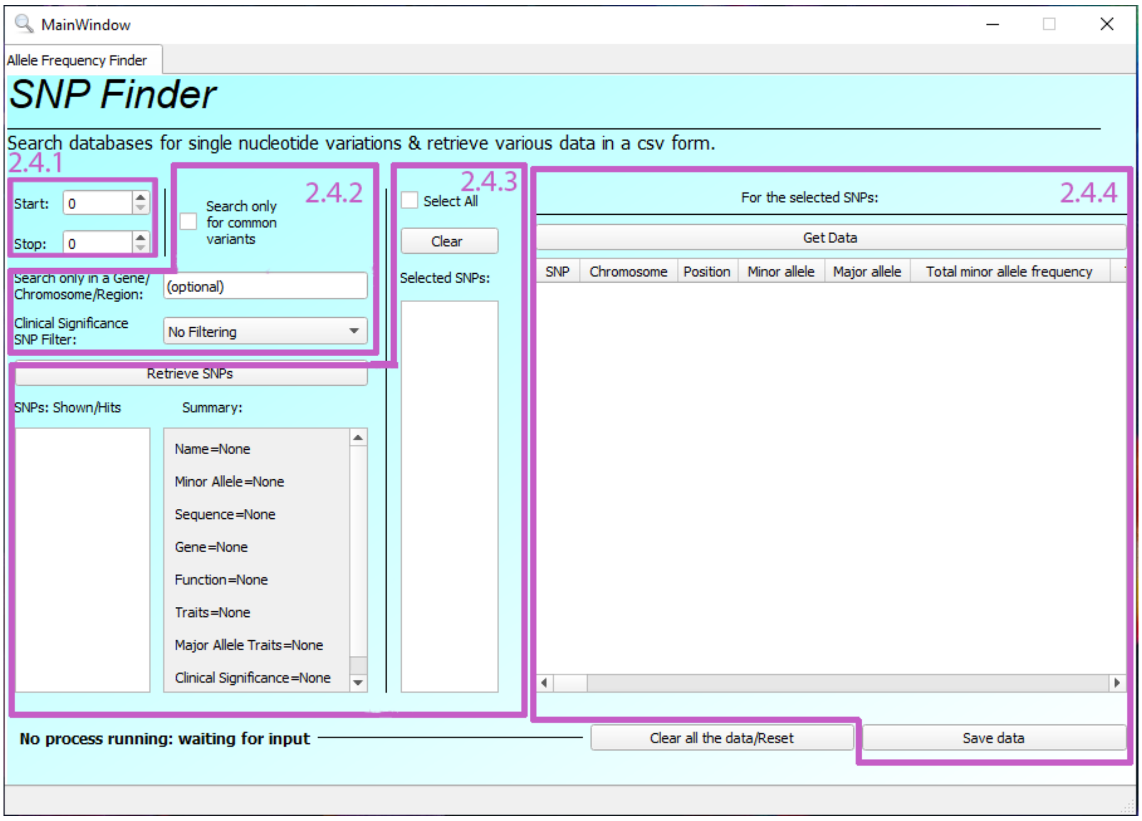


Figure 2.3: Snpfinder GUI sections

2.4.1 Implementing a numerical search window

Firstly a way to alter the number of SNPs obtained and set a numerical retrieval window was implemented. Two optional parameters provided by the E-utilities API, '*retmax*' and '*retstart*' were utilized. The value of '*retmax*' sets the number of SNPs to be retrieved by the API call, while the value of '*retstart*' corresponds to the sequential index of the first SNP retrieved from dbSNP.

Example numerical search window API call

```
import requests
#base API call
server = 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?'
#API extension
extension = 'db=snp&term=1000genomes+has+frequency+filter[Filter]+
              AND+snv[SNPClass]&retmax={user_input}
              &retstart={user_input}&retmode=json'
#data request
r = requests.get(server+ext)
#passes data to a variable
data = json.loads(r.content)
```

This example API call, with values retmax = 100 and retsart = 1000, would retrieve a hundred SNPs starting from the 1000th SNP catalogued in dbSNP that meets the thespeicfied criteria. Furthermore, two numerical input GUI elements were implemented in '*snpfinder*' to allow the user to enter values for retmax and restart.

GUI elements defining the numerical search window

```
# Retstart Spinbox GUI element
self.Retstart = QtWidgets.QSpinBox(self.gridLayoutWidget)
self.Retstart.setRange(0, 38216813)
self.UIDlayout.addWidget(self.Retstart, 1, 2, 1, 1)
# Retmax Spinbox GUI element
self.Retmax = QtWidgets.QSpinBox(self.gridLayoutWidget)
self.Retmax.setRange(0, 100)
self.Retmax.setObjectName("Retmax")
self.UIDlayout.addWidget(self.Retmax, 2, 2, 1, 1)
```

As it's shown in the code, the limit for SNP per request was set to 100 to achieve fast results and the moving search window can be modified from the zero index of dbSNP to the 38216813th SNP which is the final in the data-set.

2.4.2 Implementing search filters

To extend the search functionality of *snpfinder* beyond a numerical search window, five search filters were implented into the API, along with three GUI elements for the user to interact with them.

Common SNP filter

A checkbox GUI element was utilized to provide an extra parameter to the API call, when the user searches for SNPs with the element checked then, the API call is modified:

Common SNP filter

```
import requests
#base API call
server = 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?'
#API extension
extension = 'db=snp&term=1000genomes+has+frequency+filter[Filter]+
             AND+snv[SNPClass]
             +AND+00000.0100:+00001.0000[GLOBAL_MAF]
             &retmax={user_input}&retstart={user_input}&retmode=json'
#
r = requests.get(server+ext)
```

This filter informs the API call to obtain only SNPs with a minor allele frequency above 1% in the population.

Clinical significance filter

A drop-down menu GUI element was used to support user-choice between the different clinical significance values provided by ClinVar: benign, likely benign, uncertain significance, likely pathogenic, pathogenic, drug response, protective, risk facto r[17]. The users choice is established as a search filter in the API call, by the following parameter:

Clinical significance filter
```
import requests
server = 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?'
extension = 'db=snp&term=1000genomes+has+frequency+filter[Filter]+
             AND+snv[SNPClass]
             +AND+{user_input}[Clinical Significance]
             &retmax={user_input}&retstart={user_input}&retmode=json'
r = requests.get(server+ext)
```

DNA region filter

Finally a filter to restrict the SNP search in a user defined genomic area was defined. One text GUI input was utilized for three distinct search options. Either to bound the search to specific chromosome the user enters:

Chromosome filter
```
import requests
#base API call
server = 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?'
#API extension
extension = 'db=snp&term=1000genomes+has+frequency+filter[Filter]+
             AND+snv[SNPClass]
             +AND+{user_input}[Chromosome]
             &retmax={user_input}&retstart={user_input}&retmode=json'
#data request
r = requests.get(server+ext)
```

Or to a chromosomal region:

Chromosomal region filter
```
import requests
#base API call
server = 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?'
#API extension
extension = 'db=snp&term=1000genomes+has+frequency+filter[Filter]+
             AND+snv[SNPClass]
             +AND+{user_input}[Chromosome]
             +AND+{user_input}[CHRPOS]+:+{user_input}[CHRPOS]
             &retmax={user_input}&retstart={user_input}&retmode=json'
             &retmax={user_input}&retstart={user_input}&retmode=json'
#data request
r = requests.get(server+ext)
```

Or to a gene:

```
Gene filter
import requests
#base API call
server = 'https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?'
#API extension
extension = 'db=snp&term=1000genomes+has+frequency+filter[Filter]+
             AND+snv[SNPClass]
             +AND+{user_input}[Gene Name]
             &retmax={user_input}&retstart={user_input}&retmode=json'
#data request
r = requests.get(server+ext)
```

2.4.3 Initiating the search and displaying obtained SNPs

The search is Initiated when the user presses the 'Retrieve SNPs' GUI button. A function named *AvailableSNV* then takes as input and checks the states of all the GUI elements mentioned above. It then picks the API search URL matching the user's input. This was achieved with if, elif and else statements, in summary, the function checks for all the possible combinations of the filters and according to the user's input, allows or disregards certain API parameters. For example, If the user's input was a search window of 50 SNPs, starting from the 200th SNP available in dbSNP, which are pathogenic, exist in the gene TP53 and have a minor allele frequency above 0.01 then the API URL picked by the function would be:

```
https://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?
db=snp&term=1000genomes+has+frequency+filter[Filter]
+AND+snv[SNP Class]+AND00000.0100:+00001.0000[GLOBAL_MAF]
+AND+pathogenic[Clinical Significance]+AND+TP53[Gene Name]
&retstart=200&
retmax=50&retmode=json&sort=SNP_ID'
```

The SNPs obtained from the API call in JSON are extracted in a list and the function updates a text GUI element with the count of the available SNPs in dbSNP according to the filters. This displays the SNP amount that the user can retrieve chunks from, by modifying the search window.

```
Available SNV function
def available_SNV(self, retstart, retmax, gene, clinsignificance):
    '''Searches for the SNPs of the available SNPs in dbSNP that
    meet the API criteria'''
    #initiliaze the SNP list
    UIDlist = [ ]

        #Code omitted due to length
        {filter check with if, elif and else & API parameter choice}
        #Code omitted due to length

            #extracts JSON containing the SNP list
            strlist = (data['esearchresult']['idlist'])
            #Converts the SNP strings to integers
            numlist = [int(x) for x in strlist]
            #Sorts the list
            UIDlist = sorted(numlist)
            #Updates the count of the available SNPs
            self.label.setText(
            f"SNPs: len(UIDlist)/data['esearchresult']['count']")
    return UIDlist
```

To display the SNPs returned by the *Available_SNV* function, a list GUI element was utilized:

list element to display the obtained SNPs

```
self.UIDlist = QtWidgets.QListWidget(self.gridLayoutWidget)
       #Defines a double clicked signal connecting it to get_one function
       self.UIDlist.itemDoubleClicked.connect(self.get_one)
       #Defines a clicked signal connecting it to UIDclicked function
       self.UIDlist.itemClicked.connect(self.UIDclicked)
       self.UIDlist.setObjectName("UIDlist")
       self.UIDlayout.addWidget(self.UIDlist, 7, 1, 1, 2)
```

As shown in the code snippet above, click and double-click interactions with the SNPs of the list were also made available. Clicking a SNP provides a quick summary of the SNP's minor allele shown in line (10), sequence (9), gene (19-20), function (11), minor and major allele traits (21-24), as well as clinical significance(31-34) info, displayed in a text GUI element (36-51). This is achieved with the *infosum* function:

Infosum function

```
1  def infosum(self, uid):
2          '''creates the data summary for the SNP'''
3          rsSNV = 'rs' + str(uid.text())
4          server = "https://rest.ensembl.org"
5          ext = f"/variation/human/{rsSNV}?phenotypes=1"
6          r = requests.get(
7              server+ext, headers={"Content-Type": "application/json"}).json()
```

```
        #extracts summary SNP data from the JSON string.
        Seq = r['mappings'][0]['allele_string']
        Minor_Allele = r['minor_allele']
        func = r['most_severe_consequence']
        clinical = ''
        Traits = ''
        AltTraits = ''
        Gene = ''
        if r['phenotypes']:
            for items in r['phenotypes']:
                if 'risk_allele' in items:
                    if 'genes' in items:
                        Gene = items['genes']
                    if items['risk_allele'] == Minor_Allele:
                        Trait = items['trait']
                        if Trait not in Traits:
                            Traits += f'{Trait}\n'
                    elif items['risk_allele'] != Minor_Allele:
                        AltTrait = str(items['risk_allele']) + \
                            '->'+str(items['trait']).lower()
                        if AltTrait not in AltTraits:
                            AltTraits += f'{AltTrait}\n'
        try:
            for item in r['clinical_significance']:
                clinical += f'{item}\n'
        except KeyError:
            clinical += 'N\A'
        #Updates a text GUI element with the summary data extracted
        self.label_8.setText(f"Name={rsSNV}\n"
                             "\n"
                             f"Minor Allele={Minor_Allele}\n"
                             "\n"
                             f"Sequence={Seq}\n"
                             "\n"
                             f"Gene={Gene}\n"
                             "\n"
                             f"Function=\n{func}\n"
                             "\n"
                             f"Traits=\n{Traits.lower()}"
                             "\n"
                             f"Major Allele Traits=\n{AltTraits}"
                             "\n"
                             f"Clinical Significance=\n{clinical}"
                             )
```

The info provided by *infosum* can help the user choose the SNPs for which to obtain further data. Clicking a SNP moves it in a second list element containing the selected

SNPs. An option to select all the SNPs was also added in a form of a 'Select all' GUI checkbox. Up to a hundred SNPs can be selected and displayed. Additionally, an SNP can be removed from the list element, before committing to further data retrieval, by clicking it (7), or by pressing the 'clear' GUI button, if the user wants to empty the list.

list element to display the selected SNPs

```
1         # Selected UID list
2         self.SelectedUID = QtWidgets.QListWidget(self.gridLayoutWidget_4)
3         self.SelectedUID.setEnabled(True)
4         self.SelectedUID.setMouseTracking(False)
5         #Defines a click signal that connect the GUI element
6         #with the remove_item function
7         self.SelectedUID.itemClicked.connect(self.remove_item)
8         self.SelectedUID.setObjectName("SelectedUID")
9         self.gridLayout_2.addWidget(self.SelectedUID, 3, 1, 1, 1)
```

2.4.4 Retrieving SNP data

SNP data retrieval begins with the press of 'get Data' button. With the click, a signal is sent from the GUI element to execute the *Get_Data_Clicked* function:

Get_Data_Clicked function

```
    def Get_Data_Clicked(self):
        '''Retrieves data for the selected SNPs'''
        #Checks if the user has any SNPs selected,
        #if not it shows a warning message
        if self.SelectedUID.count() == 0:
            msg = QtWidgets.QMessageBox.warning
            (self, 'ERROR', 'Zero UIDs selected',
                QtWidgets.QMessageBox.Ok)
        else:
            #Cleans the table containing the data
            #and initialize the working threads
            self.FreqTable.model().removeRows(0, self.FreqTable.rowCount())
            worker1 = Worker(self.Popfinder, self.Selected_UID_list())
            worker2 = Worker(self.Phenfinder, self.Selected_UID_list())
            worker3 = Worker(self.Genotypefinder, self.Selected_UID_list())

            #Disable certain GUI elements while the functions run
            worker2.signals.started.connect(self.Ui_Get_Freqs_off)
            worker3.signals.started.connect(self.Ui_Get_Freqs_off)
            worker1.signals.started.connect(self.Ui_Get_Freqs_off)


            #Pass the functions results to functions filling the data-table
```

```
23              worker1.signals.result.connect(self.Tablemaker)
24              worker2.signals.result.connect(self.Tablemaker2)
25              worker3.signals.result.connect(self.Tablemaker3)
26
27              #When the functions finish, enable certain GUI elements
28              worker1.signals.finished.connect(self.Ui_Get_Freqs_on)
29              #Start the threads
30              self.threadpool.start(worker1)
31              self.threadpool.start(worker2)
32              self.threadpool.start(worker3)
```

Get_Data_Clicked creates three threads, workers 1, 2 and 3. As shown in the code snippet(9-11), each executes functions, specifically *popfinder*, *phenfinder* and *Genotypefinder* respectively. These functions run simultaneously and utilize the API calls discussed in section (2.3.2) to obtain distinct SNP data. If the user has not selected any SNPs it creates a pop up warning message (5-8).

- *Popfinder* 'reads' the JSON data obtained from the genomic & population data API call 2.3.2 to extract and return, if available, the chromosome number (44-45), position (46-47), major allele (48-76), minor allele (48-76), as well as the minor allele frequency in the Total, African, American, European, South Asian and East Asian populations (82-126) for the selected SNPs.

popfinder function

```
1   def Popfinder(self, UIDlist):
2       '''Finds the data to fill the table'''
3       newls = [("rs" + str(n)) for n in UIDlist]
4       UIDliststr = str(newls).replace("'", '"')
5       server = "https://rest.ensembl.org"
6       ext = "/variation/homo_sapiens?pops=1"
7       data = '{"ids":' + UIDliststr + '}'
8       headers = {"Content-Type": "application/json",
9                   "Accept": "application/json"}
10      r = requests.post(server+ext, headers=headers,
11                        data=data)
12
13      decoded = json.loads(r.content)
14      #Initialize list to fill with SNP data
15      chromosome = []
16      popdata = []
17      position = []
18      minor_allele = []
19      major_allele = []
20
21      decodeduidsorted = sorted([int(uid.strip('rs'))
22                                  for uid in decoded.keys()])
23      true_uidlist = ['rs' + str(uid) for uid in decodeduidsorted]
24      for uid in true_uidlist:
25          #Initialize a dictionary and lists for populations to fill the dictionary with.
26          popdict = {}
27          Allslist = []
```

```
            AFRlist = []
            AMRlist = []
            EASlist = []
            EURlist = []
            SASlist = []
            major = ''
            minor = ''
            #extract data from JSON
            try:
                minor += decoded[uid]["minor_allele"]
                minor_allele.append(minor)
            except TypeError:
                minor_allele.append('NA')

            if len(decoded[uid]["mappings"]) > 1:
                try:
                    chromosome.append(
                        (decoded[uid]["mappings"][0]["location"].split(':'))[0])
                    position.append(
                        (decoded[uid]["mappings"][0]["location"].split('-'))[1])
                    major = decoded[uid]["mappings"][0]["allele_string"].split(
                        '/')[0]
                    if major != minor:
                        major_allele.append(major)
                        major = ''
                    else:
                        major_allele.append(decoded[uid]["mappings"][0]["allele_string"].split(
                            '/')[1])
                        major = ''

                except KeyError:
                    chromosome.append('NA')
                    position.append('NA')
            else:
                for items in decoded[uid]["mappings"]:
                    try:
                        chromosome.append((items["location"].split(':'))[0])
                        position.append(
                            (items["location"].split('-'))[1])
                        if len(items["allele_string"]) > 3:
                            major = items["ancestral_allele"]
                            major_allele.append(major)
                        else:
                            major = items["allele_string"].split('/')[0]
                            if major != minor:
                                major_allele.append(major)
                            else:
                                major_allele.append(
                                    items["allele_string"].split('/')[1])
                    except KeyError:
                        chromosome.append('NA')
                        position.append('NA')
            #Check population data from JSON string and extract the data for
            #1000Genomes populations
            for items in decoded[uid]['populations']:
                if items['population'] == '1000GENOMES:phase_3:ALL':
                    Allslist.append(items["frequency"])
                    popdict['1000GENOMES:phase_3:ALL_major'] = max(Allslist)
                    if min(Allslist) == 1:
```

```
                    popdict['1000GENOMES:phase_3:ALL_minor'] = 0
                else:
                    popdict['1000GENOMES:phase_3:ALL_minor'] = min(
                        Allslist)
            if items['population'] == '1000GENOMES:phase_3:AFR':
                AFRlist.append(items["frequency"])
                popdict['1000GENOMES:phase_3:AFR_major'] = max(AFRlist)
                if min(AFRlist) == 1:
                    popdict['1000GENOMES:phase_3:AFR_minor'] = 0
                else:
                    popdict['1000GENOMES:phase_3:AFR_minor'] = min(AFRlist)
            if items['population'] == '1000GENOMES:phase_3:AMR':
                AMRlist.append(items["frequency"])
                popdict['1000GENOMES:phase_3:AMR_major'] = max(AMRlist)
                if min(AMRlist) == 1:
                    popdict['1000GENOMES:phase_3:AMR_minor'] = 0
                else:
                    popdict['1000GENOMES:phase_3:AMR_minor'] = min(AMRlist)
            if items['population'] == '1000GENOMES:phase_3:EAS':
                EASlist.append(items["frequency"])
                popdict['1000GENOMES:phase_3:EAS_major'] = max(EASlist)
                if min(EASlist) == 1:
                    popdict['1000GENOMES:phase_3:EAS_minor'] = 0
                else:
                    popdict['1000GENOMES:phase_3:EAS_minor'] = min(EASlist)
            if items['population'] == '1000GENOMES:phase_3:EUR':
                EURlist.append(items["frequency"])
                popdict['1000GENOMES:phase_3:EUR_major'] = max(EURlist)
                if min(EURlist) == 1:
                    popdict['1000GENOMES:phase_3:EUR_minor'] = 0
                else:
                    popdict['1000GENOMES:phase_3:EUR_minor'] = min(EURlist)
            if items['population'] == '1000GENOMES:phase_3:SAS':
                SASlist.append(items["frequency"])
                popdict['1000GENOMES:phase_3:SAS_major'] = max(SASlist)
                if min(SASlist) == 1:
                    popdict['1000GENOMES:phase_3:SAS_minor'] = 0
                else:
                    popdict['1000GENOMES:phase_3:SAS_minor'] = min(SASlist)
        popdata.append(popdict)

    return [newls, chromosome, position, minor_allele, major_allele, popdata]
```

- *Phenfinder* is utilized to process JSON data form the phenotype data API call, code snippet (2.3.2), in order to extract and return the functional impact (28-31), clinical significance (32-36), traits of the minor and traits of the major allele (42-59), as well as the associated gene (38-41) with the phenotype for the selected SNPs.

Phenfinder function

```
1  def Phenfinder(self, UIDlist):
2      '''Finds the data to fill the table'''
3      newls = [("rs" + str(n)) for n in UIDlist]
4      UIDliststr = str(newls).replace("'", '"')
5      server = "https://rest.ensembl.org"
6      ext = "/variation/homo_sapiens?phenotypes=1"
7      data = '{"ids":' + UIDliststr + '}'
```

```
        headers = {"Content-Type": "application/json",
                   "Accept": "application/json"}
        r = requests.post(server+ext, headers=headers,
                          data=data)
        decoded = json.loads(r.content)
        decodeduidsorted = sorted([int(uid.strip('rs'))
                                   for uid in decoded.keys()])
        #Initialize lists for the data extracted
        true_uidlist = ['rs' + str(uid) for uid in decodeduidsorted]
        clinicallist = []
        funclist = []
        AltTraitlist = []
        Traitlist = []
        Genelist = []
        for uid in true_uidlist:
            AltTraitsstring = ''
            Traitstring = ''
            AltTraitsstring = ''
            Gene = ''
            #Extract data from JSON
            try:
                funclist.append(decoded[uid]['most_severe_consequence'])
            except KeyError:
                funclist.append('Not specified')
            try:
                clinicallist.append(
                    str(decoded[uid]["clinical_significance"]).strip('[]'))
            except KeyError:
                clinicallist.append("Not specified")
            for items in decoded[uid]["phenotypes"]:
                try:
                    Gene = items['genes']
                except KeyError:
                    Gene = 'Not specified'
                try:
                    if items["risk_allele"] == decoded[uid]["minor_allele"]:
                        try:
                            Trait = items['trait']
                            if Trait.lower() not in Traitstring:
                                Traitstring += f'|{Trait.lower()}|'
                        except KeyError:
                            Traitstring = 'Not specified'
                    else:
                        try:
                            AltTrait = items['trait']
                            if AltTrait.lower() not in AltTraitsstring:
                                AltTraitsstring = f'|{AltTrait.lower()}|'
                        except KeyError:
                            AltTraitsstring = 'Not specified'
                except KeyError:
                    continue
            AltTraitlist.append(AltTraitsstring)
            Traitlist.append(Traitstring)
            Genelist.append(Gene)

        return [funclist, Genelist, Traitlist, AltTraitlist, clinicallist]
```

- *Genotypefinder* was implemented to manage the JSON data returned from the genotype API call,shown in code snippet (2.3.2). The API returns the genotypes in the form *alllele|allele* and their corresponding frequencies for various populations and different studies. Since Only the genotype data for 1000 Genomes study is needed, *Genotypefinder* 'reads' the JSON data, finds the genotypes for the main 1000 Genomes populations, inspects their alleles and catalogues them in major allele homozygous, minor allele homozygous and heterozygous along with their frequencies in the total, African, American, European, South Asian and East Asian populations (37-106).

Genotypefinder function

```
 1      def Genotypefinder(self, UIDlist):
 2          '''Finds the data to fill the table'''
 3          newls = [("rs" + str(n)) for n in UIDlist]
 4          UIDliststr = str(newls).replace("'", '"')
 5          server = "https://rest.ensembl.org"
 6          ext = "/variation/homo_sapiens?population_genotypes=1"
 7          data = '{"ids":' + UIDliststr + '}'
 8          headers = {"Content-Type": "application/json",
 9                     "Accept": "application/json"}
10          r = requests.post(server+ext, headers=headers,
11                            data=data)
12
13          decoded = json.loads(r.content)
14          decodeduidsorted = sorted([int(uid.strip('rs'))
15                                     for uid in decoded.keys()])
16          true_uidlist = ['rs' + str(uid) for uid in decodeduidsorted]
17
18          #Set the dictionary keys
19          keys = ['majorhomozygousALL', 'majorhomozygousAFR',
20                  'majorhomozygousAMR', 'majorhomozygousEAS',
21                  'majorhomozygousEUR', 'majorhomozygousSAS',
22                  'minorhomozygousALL', 'minorhomozygousAFR',
23                  'minorhomozygousAMR', 'minorhomozygousEAS',
24                  'minorhomozygousEUR', 'minorhomozygousSAS',
25                  'heterozygousALL', 'heterozygousAFR',
26                  'heterozygousAMR', 'heterozygousEAS',
27                  'heterozygousEUR', 'heterozygousSAS']
28          #Initialize the genotype list
29          Genotypelist = []
30
31          #For each SNP search the 1000 Genomes genotype data in the JSON string and
32          #catalogue them in major,minor heterozygous and homozygous.
33          for uid in true_uidlist:
34              gendict = {}
35              minorallele = decoded[uid]['minor_allele']
36
37              for items in decoded[uid]['population_genotypes']:
38                  if items['population'] == '1000GENOMES:phase_3:ALL':
39                      # filter heterozygotous genotypes
40                      if (items['genotype'].split('|')[0] == minorallele or
41                      items['genotype'].split('|')[1] == minorallele) and
42                      not items['genotype'] == f'{minorallele}|{minorallele}':
43                          gendict['heterozygousALL'] = items['frequency']
44                      # filter Minor allele Homozygous genotypes
45                      if items['genotype'] == f'{minorallele}|{minorallele}':
46                          gendict['minorhomozygousALL'] = items['frequency']
```

```
                    # filter Major allele Homozygous genotypes
                    if (items['genotype'].split('|')[0] != minorallele) and
                    (items['genotype'].split('|')[1] != minorallele):
                        gendict['majorhomozygousALL'] = items['frequency']
                if items['population'] == '1000GENOMES:phase_3:AFR':
                    if (items['genotype'].split('|')[0] == minorallele or
                    items['genotype'].split('|')[1] == minorallele) and
                    not items['genotype'] == f'{minorallele}|{minorallele}':
                        gendict['heterozygousAFR'] = items['frequency']
                    if items['genotype'] == f'{minorallele}|{minorallele}':
                        gendict['minorhomozygousAFR'] = items['frequency']
                    if (items['genotype'].split('|')[0] != minorallele) and
                    (items['genotype'].split('|')[1] != minorallele):
                        gendict['majorhomozygousAFR'] = items['frequency']
                if items['population'] == '1000GENOMES:phase_3:AMR':
                    if (items['genotype'].split('|')[0] == minorallele or
                    items['genotype'].split('|')[1] == minorallele) and
                    not items['genotype'] == f'{minorallele}|{minorallele}':
                        gendict['heterozygousAMR'] = items['frequency']
                    if items['genotype'] == f'{minorallele}|{minorallele}':
                        gendict['minorhomozygousAMR'] = items['frequency']
                    if (items['genotype'].split('|')[0] != minorallele) and
                    (items['genotype'].split('|')[1] != minorallele):
                        gendict['majorhomozygousAMR'] = items['frequency']
                if items['population'] == '1000GENOMES:phase_3:EAS':
                    if (items['genotype'].split('|')[0] == minorallele
                    or items['genotype'].split('|')[1] == minorallele) and
                    not items['genotype'] == f'{minorallele}|{minorallele}':
                        gendict['heterozygousEAS'] = items['frequency']
                    if items['genotype'] == f'{minorallele}|{minorallele}':
                        gendict['minorhomozygousEAS'] = items['frequency']
                    if (items['genotype'].split('|')[0] != minorallele) and
                    (items['genotype'].split('|')[1] != minorallele):
                        gendict['majorhomozygousEAS'] = items['frequency']
                if items['population'] == '1000GENOMES:phase_3:EUR':
                    if (items['genotype'].split('|')[0] == minorallele or
                    items['genotype'].split('|')[1] == minorallele) and
                    not items['genotype'] == f'{minorallele}|{minorallele}':
                        gendict['heterozygousEUR'] = items['frequency']
                    if items['genotype'] == f'{minorallele}|{minorallele}':
                        gendict['minorhomozygousEUR'] = items['frequency']
                    if (items['genotype'].split('|')[0] != minorallele) and
                    (items['genotype'].split('|')[1] != minorallele):
                        gendict['majorhomozygousEUR'] = items['frequency']
                if items['population'] == '1000GENOMES:phase_3:SAS':
                    if (items['genotype'].split('|')[0] == minorallele or
                    items['genotype'].split('|')[1] == minorallele) and
                    not items['genotype'] == f'{minorallele}|{minorallele}':
                        gendict['heterozygousSAS'] = items['frequency']
                    if items['genotype'] == f'{minorallele}|{minorallele}':
                        gendict['minorhomozygousSAS'] = items['frequency']
                    if (items['genotype'].split('|')[0] != minorallele)
                    and (items['genotype'].split('|')[1] != minorallele):
                        gendict['majorhomozygousSAS'] = items['frequency']
                for items in keys:
                    if items not in gendict.keys():
                        gendict[items] = 0
            Genotypelist.append(gendict)

```

```
106             return Genotypelist
```

2.4.5 Displaying and saving SNP data

As shown above, in lines (17-19), of *Get_Data_clicked* function's code snippet (2.4.4), the data returned from the three prior functions are loaded as input in three *Tablemaker* functions. These functions fill the rows and columns of a large table GUI element to display all of the SNP data. Each row corresponds to a SNP and each column to a SNP data type. The table contains forty columns:

' SNP ', 'Chromosome', 'Position', 'Minor allele', 'Major allele', 'Total minor allele frequency', 'Total major allele frequency', 'African minor allele frequency ', 'African major allele frequency', 'European minor allele frequency', 'European major allele frequency', 'American minor allele frequency', 'American major allele frequency', 'East Asian minor allele frequency', 'East Asian major allele frequency', 'South Asian minor allele frequency', 'South Asian major allele frequency', 'Function', 'Gene', 'Minor allele traits', 'Major allele traits', 'Clinical Significance', 'Total heterozygous', 'Total minor allele homozygous', 'Total major allele homozygous', 'African heterozygous', 'African minor allele homozygous', 'African major allele homozygous', 'European heterozygous', 'European minor allele homozygous', 'European major allele homozygous', 'American heterozygous', 'American minor allele homozygous', 'American major allele homozygous', 'East Asian heterozygous', 'East Asian minor allele homozygous', 'East Asian major allele homozygous', 'South Asian heterozygous', 'South Asian minor allele homozygous', 'South Asian major allele homozygous'.

The data displayed in the table can be saved in a CSV file by pressing the 'Save Data' GUI button. This event sends a signal which activates the *save_clicked* function. It utilizes the pandas module to create a data-frame from the table data(6-12) and then save it as a CSV file (14-26) [38]. If unsuitable file names are provided, *save_clicked* also uses PyQt5's GUI pop up messages to inform the user (27-46).

list element to display the selected SNPs

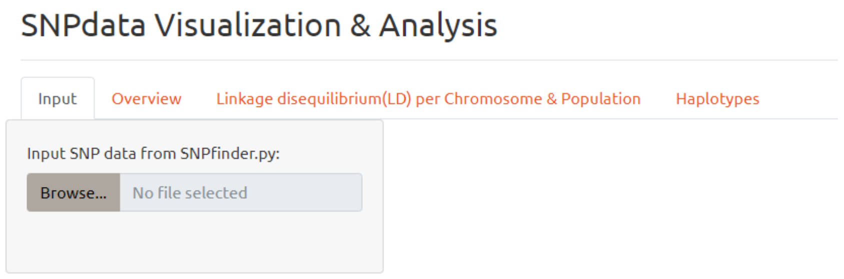
```
1       def save_clicked(self):
2           #
3           columnHeaders = []
4           for j in range(self.DataTable.model().columnCount()):
5               columnHeaders.append(self.DataTable.horizontalHeaderItem(j).text())
6           #Create the pandas dataframe
7           df = pd.DataFrame(columns=columnHeaders)
8           #Fill the data frame with table data
9           for row in range(self.DataTable.rowCount()):
10              for col in range(self.DataTable.columnCount()):
11                  df.at[row, columnHeaders[col]] = self.DataTable.item(
12                      row, col).text()
13
14          text, ok = QtWidgets.QInputDialog.getText(
            self, 'Saving csv', 'File Name:')

        if ok and text != '':
            textls = text.split('.')
            if len(textls) == 2:
                if textls[0].isalnum() == True and textls[1].isalnum() == True:
                    if textls[1] == 'csv':
                        #Convert the dataframe to a csv and save it
                        df.to_csv(f'{text[0]}.{textls[1]}', index=False)
                        msg = QtWidgets.QMessageBox.information(self, ' ',
                        'File Saved.',
                        QtWidgets.QMessageBox.Ok)
                    else:
                        df.to_csv(f'{textls[0]}.csv', index=False)
                        msg = QtWidgets.QMessageBox.information(self, ' ',
                        'File Saved.',
                        QtWidgets.QMessageBox.Ok)
                else:
                    msg = QtWidgets.QMessageBox.warning(self, 'ERROR!',
                    'Only use alphanumeric characters\nfor the file name.',QtWidgets.QMessageBox.Ok)

            elif len(textls) == 1 and text.isalnum() == True:
                df.to_csv(f'{text}.csv', index=False)
                msg = QtWidgets.QMessageBox.information(self, ' ', 'File
                Saved.',QtWidgets.QMessageBox.Ok)
            else:
                msg = QtWidgets.QMessageBox.warning(self, 'ERROR!',
                'Only use alphanumeric characters\nfor the file name.',
                QtWidgets.QMessageBox.Ok)
        else:
            msg = QtWidgets.QMessageBox.warning(self, 'ERROR', 'A file name cant be blank',
            QtWidgets.QMessageBox.Ok)
```

2.5 SNPanalysis

Snpanalysis, as illustrated in figure (2.1), accepts the CSV file produced by *Snpfinder* as input. The program uses the shiny package for R to support a GUI. The R code based on shiny is separated into two sections, UI and server. The UI section creates the GUI elements for the user to interact with, while the server section reacts to the user inputs according to coded events and updates the UI elements. Four GUI tab elements were implemented, with UI and server components, creating a SNP population data, visualization and analysis workflow. Since each tab covers a part of the workflow they are detailed in the following sections. However, given that each tab requires data to be loaded and returns results, these are presented in chapter 3.

2.5.1 Importing the CSV

To import the file a GUI tab named 'Input' was created. The tab contains a file input GUI element and a data table to display the imported CSV. When the user uploads the file, the server-side of the file input element stores it in a variable. This variable is then inserted and displayed using the data table.



SNPdata Visualization & Analysis

Input SNP data from SNPfinder.py:

No file selected

Figure 2.4: Snpanalysis tabs

2.5.2 Overview tab

An 'Overview' tab was created to handle the visualization of the CSV data. The tab contains five GUI elements. When the user selects it , a bar chart comparing the alleles frequencies in the total population of all the SNPs in the CSV is displayed using the *ggplot*2 package for R. Additionally the user can select any of the SNPs imported, by using a selection input element. Doing so provides a table element containing the function, gene and minor,major traits for the SNP, as well as twelve pie charts for the allele and genotype frequencies of the SNP in each population by using the *Plotmaker* functions. As an example the function for the allele frequencies is shown below.

Allele frequency Plotmaker function

```

Plotmaker <- function(SNP,df) {
#Function to plot the major,minor allele frequencies in different populations
  #Create a dataframe containing the data from the selected SNP row of
  the csv file
  x <- data.frame(which(df == SNP,arr.ind =TRUE))
  #pass the dataframe data into values
  rowcount <- x[['row']]
  valuesALL <-  c(df[rowcount,'Total.minor.allele.frequency'],
  df[rowcount,'Total.major.allele.frequency'])
  valuesAFR <-  c(df[rowcount,'African.minor.allele.frequency'],
  df[rowcount,'African.major.allele.frequency'])
  valuesAMR <-  c(df[rowcount,'American.minor.allele.frequency'],
  df[rowcount,'American.major.allele.frequency'])
  valuesEUR <-  c(df[rowcount,'European.minor.allele.frequency'],
  df[rowcount,'European.major.allele.frequency'])
  valuesSAS <-  c(df[rowcount,'South.Asian.minor.allele.frequency'],
  df[rowcount,'South.Asian.major.allele.frequency'])
  valuesEAS <-  c(df[rowcount,'East.Asian.minor.allele.frequency'],
  df[rowcount,'East.Asian.major.allele.frequency'])
  labels1 <- c(df[rowcount,'Minor.allele'],df[rowcount,'Major.allele'])

 #Plot the values
layout(matrix(c(1,2,3,4,5,6), 2, 3, byrow = TRUE))
   pie(valuesALL,paste(paste(round(valuesALL,2)*100,'%'),labels1),
   main = "Total Population Allele Frequencies",radius = 1.09)
   pie(valuesAFR,paste(paste(round(valuesAFR,2)*100,'%'),labels1),
   main = "African Population Allele Frequencies",radius = 1.09)
```

```
    pie(valuesAMR,paste(paste(round(valuesAMR,2)*100,'%'),labels1),
    main = "American Population Allele Frequencies",radius = 1.09)
    pie(valuesEUR,paste(paste(round(valuesEUR,2)*100,'%'),labels1),
    main = "European Population Allele Frequencies",radius = 1.09)
    pie(valuesSAS,paste(paste(round(valuesSAS,2)*100,'%'),labels1),
    main = "South Asian Population Allele Frequencies",radius = 1.09
    pie(valuesEAS,paste(paste(round(valuesEAS,2)*100,'%'),labels1),
    main = "East Asian Population Allele Frequencies",radius = 1.09)
```

2.5.3 Linkage disequilibrium tab

The third tab of *Snpanalysis* provides an interface to obtain and visualize pairwise Linkage Disequilibrium (LD) data for the imported SNPs. Two selection GUI elements allow the user to choose the chromosome and the population to perform the LD analysis on. Utilizing two functions The chromosome's SNPs from the CSV are then loaded, along with the chosen population, onto two functions containing LDlinkR: LDmatrix API calls (1-50) [20]. These return two pairwise LD matrices, for D' and r^2. Finally using the package heatmaply, a function was written to visualize the matrix data to heatmaps (52-59).

LD tab functions

```
1  Pr2finderfunc <- function(snp,population){
2  #Function to create an r2 matrix
3    snp <- sort(snp)
4
5    if (population == 'Total'){
6      r2matrix <- (LDmatrix(snp,pop ='ALL',r2d = 'r2',token ='user token'))
7    }
8    if (population == 'African'){
9      r2matrix <- (LDmatrix(snp,pop ='AFR',r2d = 'r2',token ='user token'))
10   }
11   if (population == 'American'){
12     r2matrix <- (LDmatrix(snp,pop ='AMR',r2d = 'r2',token ='user token'))
13   }
14   if (population == 'East Asian'){
15     r2matrix <- (LDmatrix(snp,pop ='EAS',r2d = 'r2',token ='user token'))
16   }
17   if (population == 'South Asian'){
18     r2matrix <- (LDmatrix(snp,pop ='SAS',r2d = 'r2',token ='user token'))
19   }
20   if (population == 'European'){
21     r2matrix <- (LDmatrix(snp,pop ='EUR',r2d = 'r2',token ='user token'))
22   }
23
24   return(r2matrix)
25 }
26 #Function to create a D' matrix
27 dfinderfunc <- function(snp,population){
28   snp <- sort(snp)
29
30   if (population == 'Total'){
31     dmatrix <- (LDmatrix(snp,pop ='ALL',r2d = 'd',token ='user token'))
32   }
33   if (population == 'African'){
34     dmatrix <- (LDmatrix(snp,pop ='AFR',r2d = 'd',token ='user token'))
35   }
```

```
  if (population == 'American'){
    dmatrix <- (LDmatrix(snp,pop ='AMR',r2d = 'd',token ='user token'))
  }
  if (population == 'East Asian'){
    dmatrix <- (LDmatrix(snp,pop ='EAS',r2d = 'd',token ='user token'))
  }
  if (population == 'South Asian'){
    dmatrix <- (LDmatrix(snp,pop ='SAS',r2d = 'd',token ='user token'))
  }
  if (population == 'European'){
    dmatrix <- (LDmatrix(snp,pop ='EUR',r2d = 'd',token ='user token'))
  }

  return(dmatrix)
}

#Heatmap function
normheatmapfunc <- function(r2matrix){
  row.names(r2matrix) <- r2matrix$RS_number
  r2matrix1 <- r2matrix[,2:(length(r2matrix$RS_number)+1)]
  heatmatrix <- data.matrix(r2matrix1)
  heatmaply(heatmatrix, grid_gap = 1,
  distfun = dist_no_na ,na.color="black")
}
```

2.5.4 Haplotypes tab

A final fourth tab, along with two subtabs, was added to *Snpanalysis* to support haplotype analysis for the loaded SNPs. Three selection inputs on the GUI side allow the user to select a chromosome and up to thirty SNPs from the CSV on the chromosome. Finally, the user chooses the population to obtain its haplotype frequencies for the picked SNPs at the 'Haplotype frequencies' subtab. To accomplish this, a function, lines (1-25), containing an LDhap API call was utilized. Once the user obtains the haplotypes, the subtab 'Haplotype to traits' provides a selection input for them, selecting one executes a function(28-56) that maps the SNP alleles on the haplotype to their traits from the CSV file and showcases them.

Haplotype tab functions

```
1   #Function to retrieve the haplotype frequencies
2   for the selected SNPs and population
3   hapfinderfunc <- function(snplist,population){
4
5     snp <- sort(snplist)
6
7     if (population == 'Total'){
8       hapmap <- (LDhap(snp,pop ='ALL',token ='user token'))
9     }
10    if (population == 'African'){
11      hapmap <- (LDhap(snp,pop ='AFR',token ='user token'))
12    }
13    if (population == 'American'){
14      hapmap <- (LDhap(snp,pop ='AMR',token ='user token'))
15    }
16    if (population == 'East Asian'){
17      hapmap <- (LDhap(snp,pop ='EAS',token ='user token'))
```

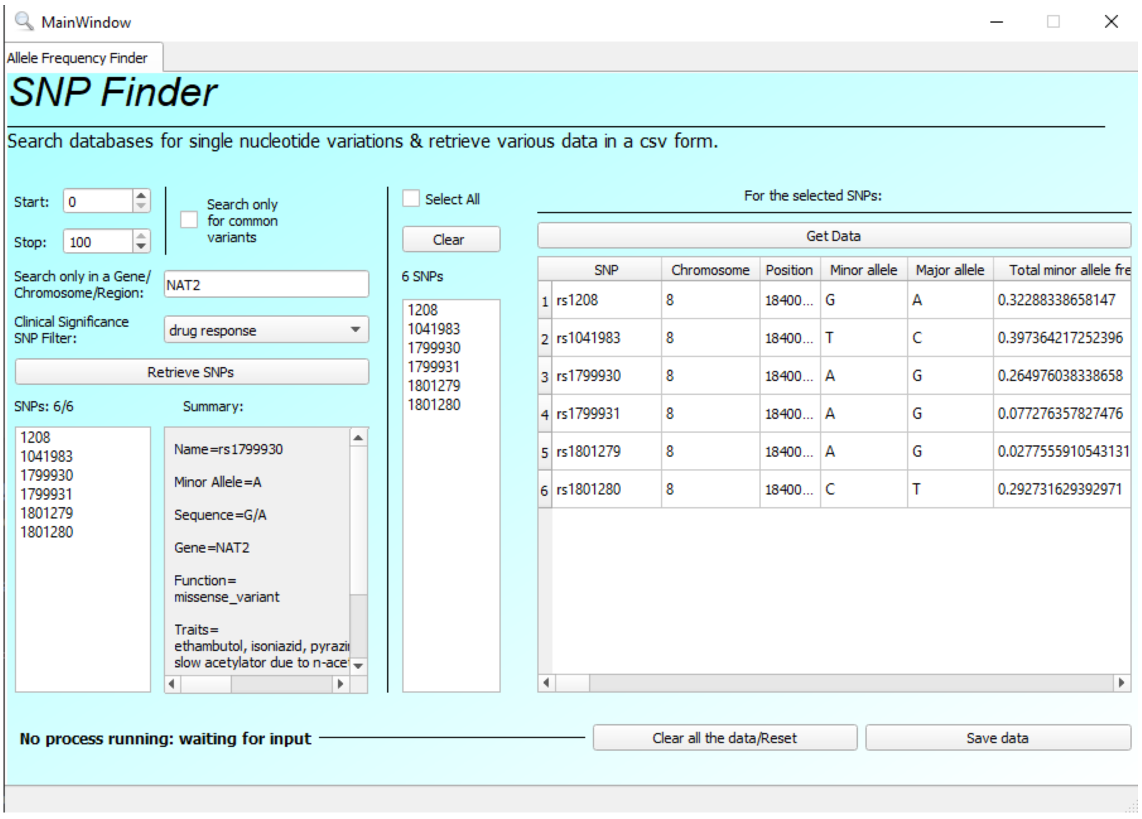
```r
  }
  if (population == 'South Asian'){
    hapmap <- (LDhap(snp,pop ='SAS',token ='user token'))
  }
  if (population == 'European'){
    hapmap <- (LDhap(snp,pop ='EUR',token ='user token'))
  }
  return(hapmap)
}

#Function to map the haplotypes to traits and clinical significance
traitablefunc <- function(haptable,row,df){
  y <- colnames(haptable)[0:(length(colnames(haptable)) -2)]
  z <- haptable[row,]
  v <- character()
  m <- character()
  l <- data.frame(z[0:(length(colnames(haptable)) -2)])
  for(snphap in y){
    dfrow <- which(df$SNP == snphap, arr.ind=TRUE)
    minoralleledf <- df[dfrow,][['Minor.allele']]
    majoralleledf <- df[dfrow,][['Major.allele']]
    allelehap <- z[[snphap]]
      if(allelehap == minoralleledf){
        w <- df[dfrow,][['Minor.allele.traits']]
        k <- df[dfrow,][['Clinical.Significance']]
        }
      if(allelehap == majoralleledf){
        w <- df[dfrow,][['Major.allele.traits']]
        k <- ''
        }
    v <- c(v,w)
    m <- c(m,k)
  }
  n <-v
  x <- data.frame(Traits = n,Clinical_Singificance = m,row.names = y)
  x1 <- as.data.frame(t(x))
  mylist <- list(one = x1 ,two = l)

  return(mylist)
```

Chapter 3

Results

In chapter 1, SNPs of the gene NAT2 were mentioned in the scope of population genetics for targeted drug therapy. Since the software can accept any user-defined search and produce results, in this chapter an example retrieval and analysis of these SNPs is detailed. It follows SNPop's pipeline, established in chapter 2 and showcased in figure (3.8).

3.1 Data obtained from Snpfinder



SNP Finder

Search databases for single nucleotide variations & retrieve various data in a csv form.

Start: 0

Stop: 100

Search only for common variants

Search only in a Gene/Chromosome/Region: NAT2

Clinical Significance SNP Filter: drug response

SNPs: 6/6

1208
1041983
1799930
1799931
1801279
1801280

Summary:

Name=rs1799930

Minor Allele=A

Sequence=G/A

Gene=NAT2

Function= missense_variant

Traits= ethambutol, isoniazid, pyrazi slow acetylator due to n-ace

6 SNPs

1208
1041983
1799930
1799931
1801279
1801280

For the selected SNPs:

	SNP	Chromosome	Position	Minor allele	Major allele	Total minor allele fre
1	rs1208	8	18400...	G	A	0.32288338658147
2	rs1041983	8	18400...	T	C	0.397364217252396
3	rs1799930	8	18400...	A	G	0.264976038338658
4	rs1799931	8	18400...	A	G	0.077276357827476
5	rs1801279	8	18400...	A	G	0.0277555910543131
6	rs1801280	8	18400...	C	T	0.292731629392971

No process running: waiting for input

Figure 3.1: Snpfinder, with SNP and their data retrieved

As shown in figure (3.1) to obtain these SNPs, the user defines the clinical significance filter of SNPs as drug response, the DNA region filter to the NAT2 gene and because the exact number of SNPs is unknown we input the max number (100) in the search window. *Snpfinder* informs the user that six such SNPs exist on the NAT2 genes and retrieves them.

3.1.1 SNP summary

Clicking on any SNP retrieved (e.g. rs1799930), returns a summary of its info on the GUI, as seen in figure (3.1), without saving any files.

Name	Minor Allele	Sequence	Gene	Function	Traits	Major Allele Traits	Clinical Significance
rs1799930	A	G/A	NAT2	missense variant	"ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr, metabolism/pk slow acetylator due to n-acetyltransferase enzyme variant"	None	drug response

Table 3.1: Clicked SNP info summary

3.1.2 CSV table output

If all six obtained SNPs are selected, *Snpfinder* outputs the CSV showcased below:

SNP	rs1208	rs1041983	rs1799930	rs1799931	rs1801279	rs1801280
Chromosome	8	8	8	8	8	8
Position	18400806	18400285	18400593	18400860	18400194	18400344
Minor allele	G	T	A	A	A	C
Major allele	A	C	G	G	G	T
Total minor allele frequency	0.32288338658147	0.397364217252396	0.264976038338658	0.077276357827476	0.0277555910543131	0.292731629392971
Total major allele frequency	0.67711661341853	0.602635782747604	0.735023961661342	0.922723642172524	0.972244408945687	0.707268370607029
African minor allele frequency	0.394856278366112	0.468229954614221	0.237518910741301	0.0287443267776097	0.102874432677761	0.291981845688351
African major allele frequency	0.605143721633888	0.531770045385779	0.762481089258699	0.97125567322239	0.897125567322239	0.708018154311649
European minor allele frequency	0.438369781312127	0.305168986083499	0.282306163021869	0.0228628230616302	0.00099403578528827	0.449304174950298
European major allele frequency	0.561630218687873	0.694831013916501	0.717693836978131	0.97713717693837	0.999005964214712	0.550695825049702
American minor allele frequency	0.373198847262248	0.291066282420749	0.171469740634006	0.112391930835735	0.00288184438040346	0.361671469740634
American major allele frequency	0.626801152737752	0.708933717579251	0.828530259365994	0.887608069164265	0.997118155619596	0.638328530259366

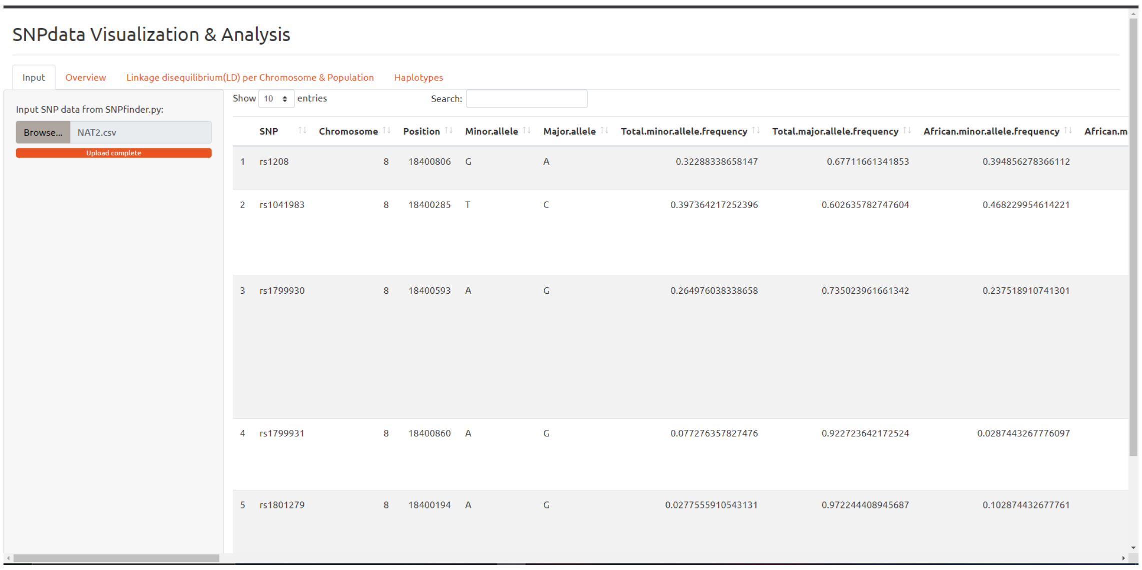
East Asian minor allele frequency	0.0396825396825397	0.439484126984127	0.255952380952381	0.179563492063492	0	0.0376984126984127
East Asian major allele frequency	0.96031746031746	0.560515873015873	0.744047619047619	0.820436507936508	1	0.962301587301587
South Asian minor allele frequency	0.362985685071575	0.428425357873211	0.359918200408998	0.0685071574642127	0	0.346625766871166
South Asian major allele frequency	0.637014314928425	0.571574642126789	0.640081799591002	0.931492842535787	1	0.653374233128834
Function	missense_variant	synonymous_variant	missense_variant	missense_variant	missense_variant	missense_variant
Gene	NAT2	NAT2	NAT2	NAT2	NAT2	NAT2
Minor allele traits		"|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr|"	"|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr, metabolism/pk||slow acetylator due to n-acetyltransferase enzyme variant|"	|slow acetylator due to n-acetyltransferase enzyme variant|	|slow acetylator due to n-acetyltransferase enzyme variant|	|slow acetylator due to n-acetyltransferase enzyme variant|
Major allele traits	|insulin resistance/response|					
Clinical Significance	'drug response'	'drug response'	'drug response'	'drug response'	'drug response'	'drug response'
Total heterozygous	0.372603833865815	0.428913738019169	0.367012779552716	0.129792332268371	0.0499201277955272	0.35702875399361
Total minor allele homozygous	0.136581469648562	0.182907348242811	0.0814696485623003	0.0123801916932907	0.00279552715654952	0.114217252396166
Total major allele homozygous	0.490814696485623	0.388178913738019	0.551517571884984	0.857827476038339	0.947284345047923	0.528753993610224
African heterozygous	0.447806354009077	0.452344931921331	0.360060514372163	0.0574886535552194	0.184568835098336	0.396369137670197
African minor allele homozygous	0.170953101361573	0.242057488653555	0.0574886535552194	0	0.010590015128593	0.0937972768532526
African major allele homozygous	0.381240544629349	0.305597579425113	0.582450832072617	0.942511346444781	0.804841149773071	0.509833585476551
European heterozygous	0.467196819085487	0.399602385685885	0.37375745526839	0.0457256461232604	0.00198807157057654	0.469184890656064
European minor allele homozygous	0.204771371769384	0.105367793240557	0.095427435387674	0	0	0.214711729622266
European major allele homozygous	0.328031809145129	0.495029821073559	0.530815109343936	0.95427435387674	0.998011928429423	0.31610337972167
American heterozygous	0.435158501440922	0.363112391930836	0.268011527377522	0.172910662824208	0.00576368876080692	0.423631123919308
American minor allele homozygous	0.155619596541787	0.109510086455331	0.037463976945245	0.0259365994236311	0	0.14985590778098
American major allele homozygous	0.409221902017291	0.527377521613833	0.694524495677233	0.801152737752161	0.994236311239193	0.426512968299712
East Asian heterozygous	0.0753968253968254	0.442460317460317	0.353174603174603	0.275793650793651	0	0.0714285714285714
East Asian minor allele homozygous	0.00198412698412698	0.218253968253968	0.0793650793650794	0.0416666666666667	0	0.00198412698412698
East Asian major allele homozygous	0.922619047619048	0.339285714285714	0.567460317460317	0.682539682539683	1	0.926587301587302
South Asian heterozygous	0.43558282208589	0.460122699386503	0.45398773006135	0.132924335378323	0	0.43558282208589
South Asian minor allele homozygous	0.14519427402863	0.198364008179959	0.132924335378323	0.00204498977505112	0	0.128834355828221

South Asian major allele homozygous	0.419222903885481	0.341513292433538	0.413087934560327	0.865030674846626	1	0.43558282208589

Table 3.2: Snpfidner CSV output

3.2 Data provided by SNPanalysis

Importing the CSV in *Snpanalysis* delivers further data, detailed in the following subsections.


SNPdata Visualization & Analysis

Input
Overview
Linkage disequilibrium(LD) per Chromosome & Population
Haplotypes
Input SNP data from SNPfinder.py:
Browse...
NAT2.csv
Upload complete
Show 10 entries
Search:

	SNP	Chromosome	Position	Minor.allele	Major.allele	Total.minor.allele.frequency	Total.major.allele.frequency	African.minor.allele.frequency	African.m
1	rs1208	8	18400806	G	A	0.32288338658147	0.67711661341853	0.394856278366112	
2	rs1041983	8	18400285	T	C	0.397364217252396	0.602635782747604	0.468229954614221	
3	rs1799930	8	18400593	A	G	0.264976038338658	0.735023961661342	0.237518910741301	
4	rs1799931	8	18400860	A	G	0.077276357827476	0.922723642172524	0.0287443267776097	
5	rs1801279	8	18400194	A	G	0.0277555910543131	0.972244408945687	0.102874432677761	

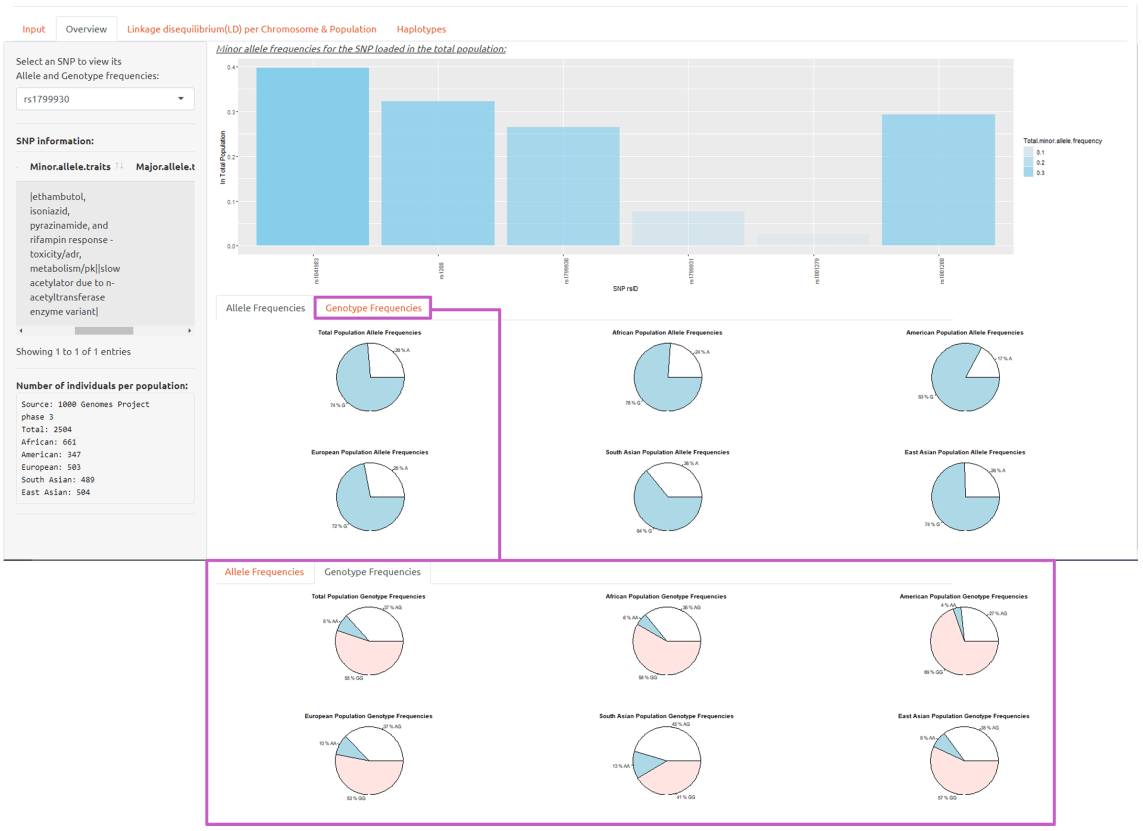
Figure 3.2: CSV imported in Snpanalysis

3.2.1 SNP frequency visualization

The overview tab (3.3) provides a bar chart, where each bar represents the frequency of each imported SNP in the total population: $frequnecy\ in\ total\ population$: $rs1041983 \approx 0.4 > rs1208 \approx 0.32 > rs1801280 \approx 0.29 > rs1799930 \approx 0.26 > rs1799931 \approx 0.07 > rs1801279 \approx 0.02$. Additionally, for each selected SNP population-specific allele and genotype frequencies are provided in pie charts, as well as phenotype and function data extracted from the CSV in a data-table. In (3.3) the SNP rs1799930 is selected:

SNP	Function	Gene	Minor allele traits	Major allele traits	Clinical Significance
rs1799930	missense_variant	NAT2	"|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr, metabolism/pk||slow acetylator due to n-acetyltransferase enzyme variant|"		'drug response'

Table 3.3: Selected SNP function and phenotype information


Input | Overview | Linkage disequilibrium(LD) per Chromosome & Population | Haplotypes

Select an SNP to view its
Allele and Genotype frequencies:

rs1799930

SNP information:

Minor.allele.traits	Major.allele.t
|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr, metabolism/pk||slow acetylator due to n-acetyltransferase enzyme variant|	

Showing 1 to 1 of 1 entries

Number of individuals per population:

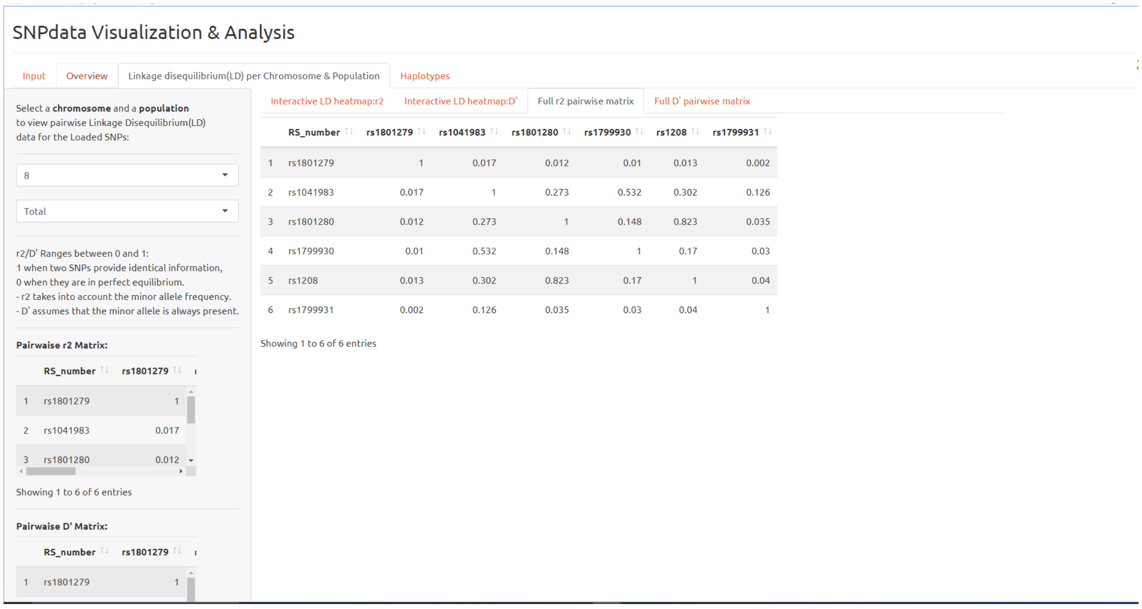
```
Source: 1000 Genomes Project
phase 3
Total: 2504
African: 661
American: 347
European: 503
South Asian: 489
East Asian: 504
```

Minor allele frequencies for the SNP loaded in the total population:
In Total Population
0.4
0.3
0.2
0.1
0.0
rs1041983
rs1208
rs1799930
rs1799931
rs1801279
rs1801280
SNP rsID
Total.minor.allele.frequency
0.1
0.2
0.3
Allele Frequencies
Genotype Frequencies
Total Population Allele Frequencies
26 % A
74 % G
African Population Allele Frequencies
24 % A
76 % G
American Population Allele Frequencies
17 % A
83 % G
European Population Allele Frequencies
28 % A
72 % G
South Asian Population Allele Frequencies
36 % A
64 % G
East Asian Population Allele Frequencies
26 % A
74 % G
Allele Frequencies
Genotype Frequencies
Total Population Genotype Frequencies
37 % AG
8 % AA
55 % GG
African Population Genotype Frequencies
36 % AG
6 % AA
58 % GG
American Population Genotype Frequencies
4 % AA
27 % AG
69 % GG
European Population Genotype Frequencies
37 % AG
10 % AA
53 % GG
South Asian Population Genotype Frequencies
45 % AG
13 % AA
41 % GG
East Asian Population Genotype Frequencies
35 % AG
8 % AA
57 % GG

Figure 3.3: Overview tab, visualization of allele and genotype frequency data

3.2.2 Linkage Disequilibrium analysis and visualization

Snpanalysis LD tab delivers matrices of the D' and r^2 (example figures: 3.4, 3.5) pairwise calculations for the imported SNPs, as well as the graphical representation of these matrices as heatmaps in the corresponding sub-tabs. All the LD population heatmaps produced by the pairwise calculations of rs1208, rs1041983, rs1799930, rs1799931, rs1801279 and rs1801280 are displayed in the figure (3.2.2). A gradient palette of purple = 0 to yellow = 1, is used to map the retrieved values. Furthermore, white squares correspond to unavailable LD data for the SNP pair in the corresponding population.



SNPdata Visualization & Analysis
Input
Overview
Linkage disequilibrium(LD) per Chromosome & Population
Haplotypes
Interactive LD heatmap:r2
Interactive LD heatmap:D'
Full r2 pairwise matrix
Full D' pairwise matrix

Select a **chromosome** and a **population** to view pairwise Linkage Disequilibrium(LD) data for the Loaded SNPs:

8

Total

r2/D' Ranges between 0 and 1:
1 when two SNPs provide identical information,
0 when they are in perfect equilibrium.
- r2 takes into account the minor allele frequency.
- D' assumes that the minor allele is always present.

Pairwise r2 Matrix:

	RS_number	rs1801279
1	rs1801279	1
2	rs1041983	0.017
3	rs1801280	0.012

Showing 1 to 6 of 6 entries

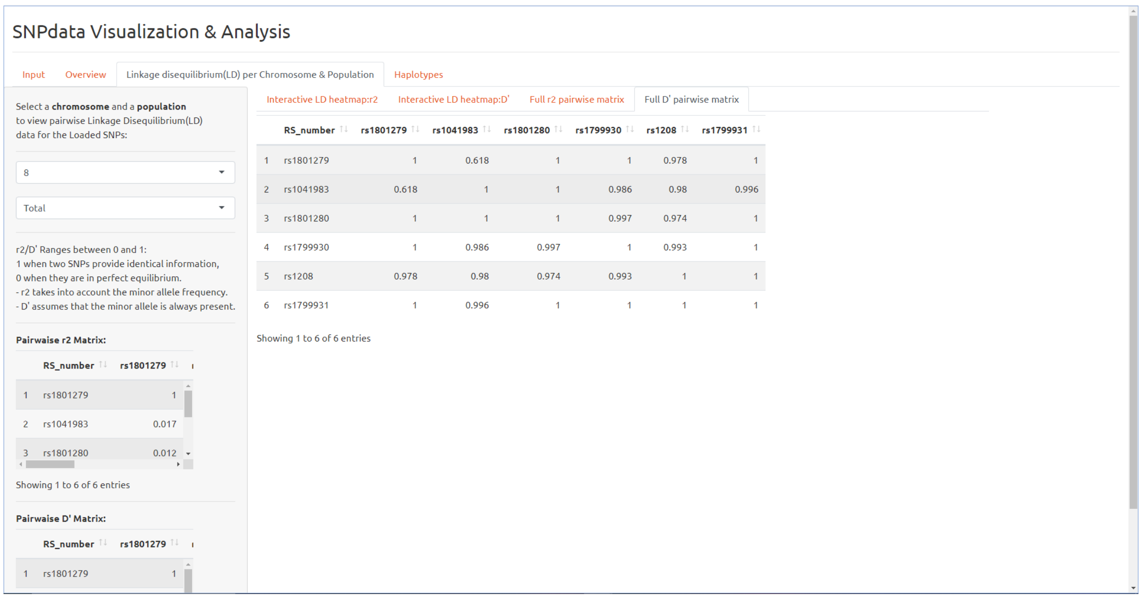
Pairwise D' Matrix:

	RS_number	rs1801279
1	rs1801279	1

	RS_number	rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931
1	rs1801279	1	0.017	0.012	0.01	0.013	0.002
2	rs1041983	0.017	1	0.273	0.532	0.302	0.126
3	rs1801280	0.012	0.273	1	0.148	0.823	0.035
4	rs1799930	0.01	0.532	0.148	1	0.17	0.03
5	rs1208	0.013	0.302	0.823	0.17	1	0.04
6	rs1799931	0.002	0.126	0.035	0.03	0.04	1

Showing 1 to 6 of 6 entries

Figure 3.4: LD tab, r^2 matrix sub-tab, Total Population



SNPdata Visualization & Analysis

Input
Overview
Linkage disequilibrium(LD) per Chromosome & Population
Haplotypes
Interactive LD heatmap:r2
Interactive LD heatmap:D'
Full r2 pairwise matrix
Full D' pairwise matrix

Select a **chromosome** and a **population** to view pairwise Linkage Disequilibrium(LD) data for the Loaded SNPs:

8

Total

r2/D' Ranges between 0 and 1:
1 when two SNPs provide identical information,
0 when they are in perfect equilibrium.
- r2 takes into account the minor allele frequency.
- D' assumes that the minor allele is always present.

Pairwise r2 Matrix:

	RS_number	rs1801279
1	rs1801279	1
2	rs1041983	0.017
3	rs1801280	0.012

Showing 1 to 6 of 6 entries

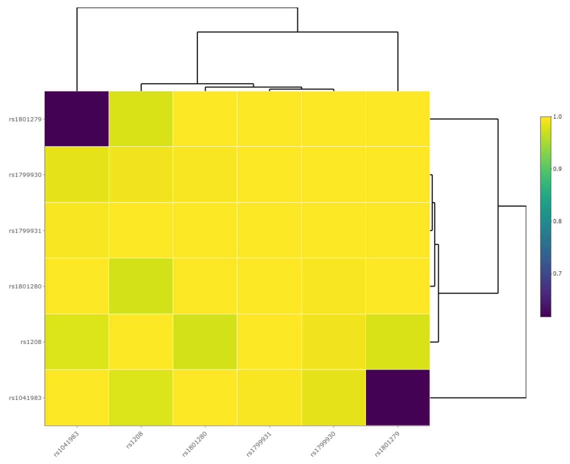
Pairwise D' Matrix:

	RS_number	rs1801279
1	rs1801279	1

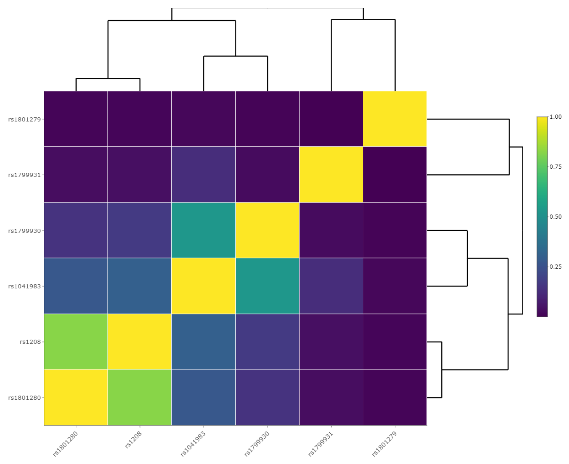
	RS_number	rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931
1	rs1801279	1	0.618	1	1	0.978	1
2	rs1041983	0.618	1	1	0.986	0.98	0.996
3	rs1801280	1	1	1	0.997	0.974	1
4	rs1799930	1	0.986	0.997	1	0.993	1
5	rs1208	0.978	0.98	0.974	0.993	1	1
6	rs1799931	1	0.996	1	1	1	1

Showing 1 to 6 of 6 entries

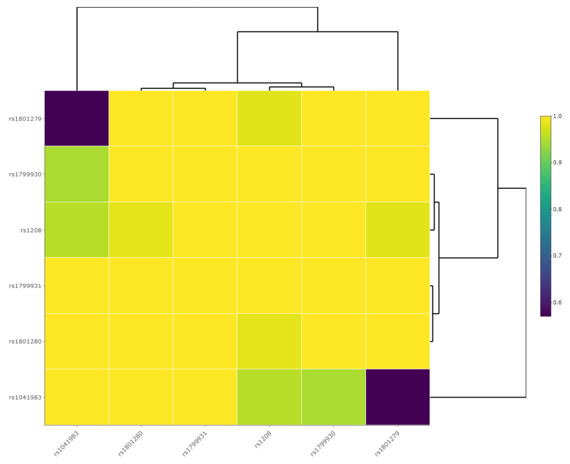
Figure 3.5: LD tab, D' matrix sub-tab, Total Population



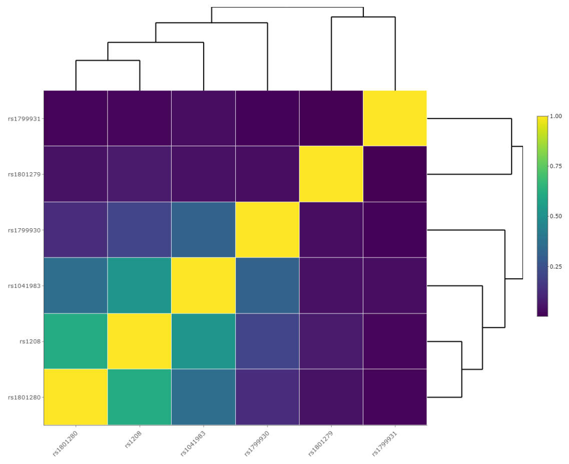
(a) D' Total population



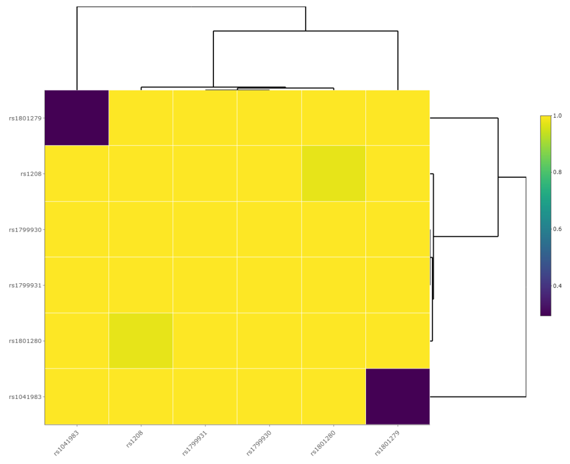
(b) r^2 Total population



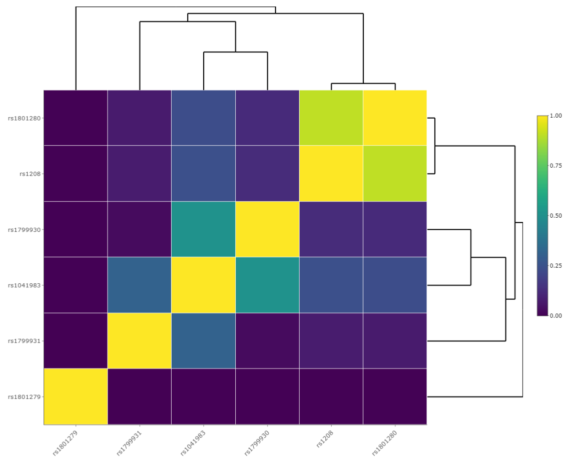
(c) D' African population



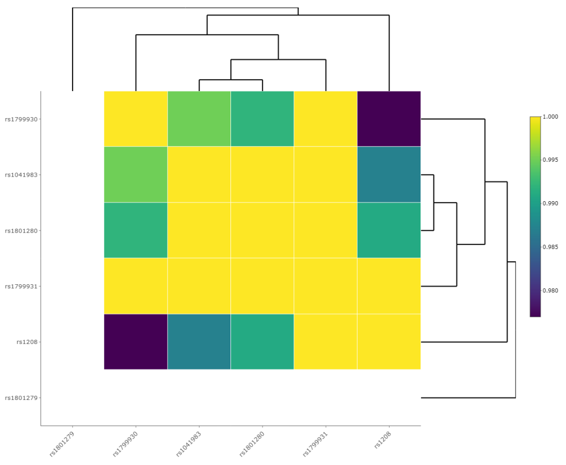
(d) r^2 African population



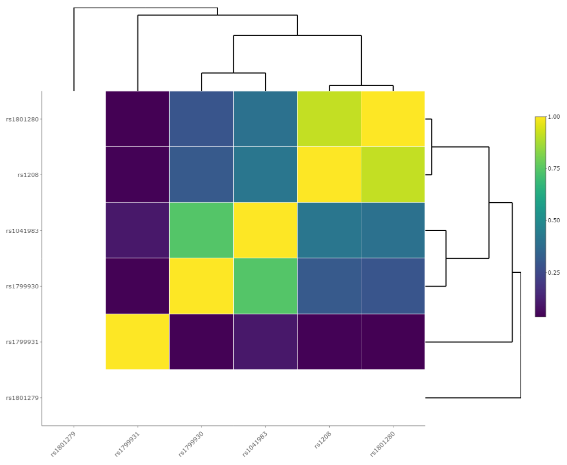
(e) D' American population



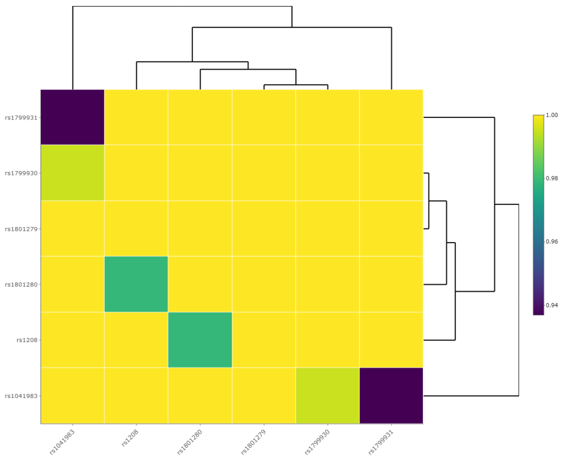
(f) r^2 American population



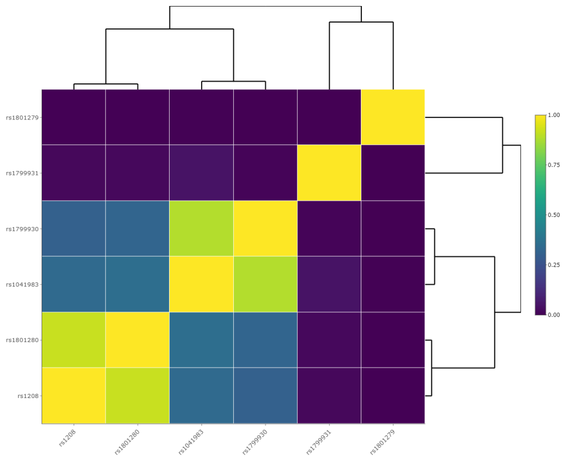
(g) D' South Asian population



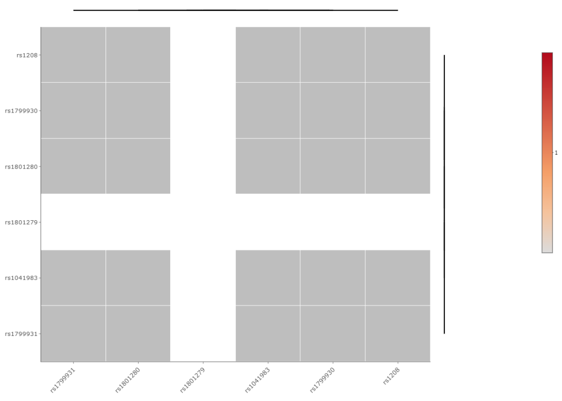
(h) r^2 South Asian population



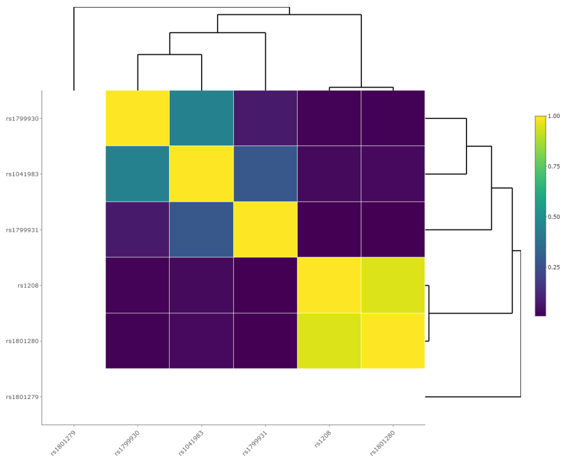
(i) D' European population



(j) r^2 European population



(k) D' East Asian population



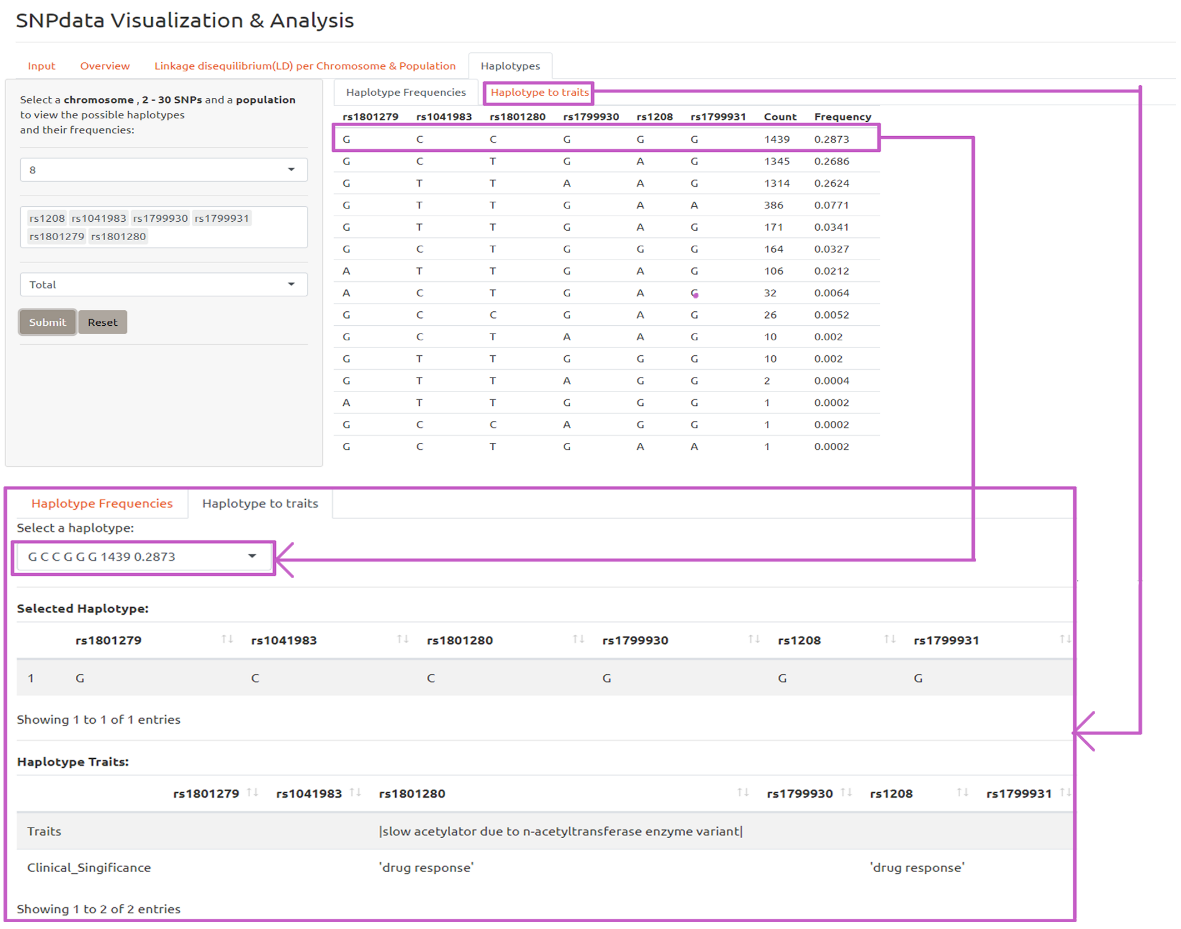
(l) r^2 East Asian population

Figure 3.6: Population specific LD heatmaps for the imported SNPs

3.2.3 Haplotype Analysis

As shown in figure (3.7), the haplotypes tab provides the frequencies of the imported SNPs haplotypes per population and the association of these haplotypes with phenotypes.

- For the six imported SNPs the haplotypes frequencies, retrieved in the 'haplotypes' sub-tab are presented in the table (3.4) for the African, American, European, South Asian, East Asian and Total population.
- For each haplotype with a frequency greater than 0.1%, the association with its allelic traits and clinical significance, retrieved in the 'haplotypes to traits' sub-tab is displayed in the tables (3.5, 3.6, 3.7, 3.8, 3.9 and 3.10).


SNPdata Visualization & Analysis

Select a **chromosome** , **2 - 30 SNPs** and a **population** to view the possible haplotypes and their frequencies:

8

rs1208 rs1041983 rs1799930 rs1799931 rs1801279 rs1801280

Total

rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931	Count	Frequency
G	C	C	G	G	G	1439	0.2873
G	C	T	G	A	G	1345	0.2686
G	T	T	A	A	G	1314	0.2624
G	T	T	G	A	A	386	0.0771
G	T	T	G	A	G	171	0.0341
G	C	T	G	G	G	164	0.0327
A	T	T	G	A	G	106	0.0212
A	C	T	G	A	G	32	0.0064
G	C	C	G	A	G	26	0.0052
G	C	T	A	A	G	10	0.002
G	T	T	G	G	G	10	0.002
G	T	T	A	G	G	2	0.0004
A	T	T	G	G	G	1	0.0002
G	C	C	A	G	G	1	0.0002
G	C	T	G	A	A	1	0.0002

Select a haplotype:

G C C G G G 1439 0.2873

Selected Haplotype:

	rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931
1	G	C	C	G	G	G

Showing 1 to 1 of 1 entries

Haplotype Traits:

	rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931
Traits			|slow acetylator due to n-acetyltransferase enzyme variant|			
Clinical_Singificance			'drug response'		'drug response'	

Showing 1 to 2 of 2 entries

Figure 3.7: Haplotypes tab, haplotype frequencies and haplotype traits, Total population

(a) Haplotype frequencies, Total population

Haplotype	Count	Frequency
G_C_C_G_G_G	1439	0.2873
G_C_T_G_A_G	1345	0.2686
G_T_T_A_A_G	1314	0.2624
G_T_T_G_A_A	386	0.0771
G_T_T_G_A_G	171	0.0341
G_C_T_G_G_G	164	0.0327
A_T_T_G_A_G	106	0.0212
A_C_T_G_A_G	32	0.0064
G_C_C_G_A_G	26	0.0052
G_C_T_A_A_G	10	0.002
G_T_T_G_G_G	10	0.002
G_T_T_A_G_G	2	0.0004
A_T_T_G_G_G	1	0.0002
G_C_C_A_G_G	1	0.0002
G_C_T_G_A_A	1	0.0002

(b) Haplotype frequencies, African population

Haplotype	Count	Frequency
G_C_C_G_G_G	382	0.289
G_T_T_A_A_G	305	0.2307
G_T_T_G_A_G	161	0.1218
G_C_T_G_A_G	148	0.112
G_C_T_G_G_G	129	0.0976
A_T_T_G_A_G	104	0.0787
G_T_T_G_A_A	38	0.0287
A_C_T_G_A_G	31	0.0234
G_T_T_G_G_G	10	0.0076
G_C_T_A_A_G	9	0.0068
G_C_C_G_A_G	4	0.003
A_T_T_G_G_G	1	0.0008

(c) Haplotype frequencies, American population

Haplotype	Count	Frequency
G_C_C_G_G_G	247	0.3559
G_C_T_G_A_G	228	0.3285
G_T_T_A_A_G	119	0.1715
G_T_T_G_A_A	78	0.1124
G_C_T_G_G_G	12	0.0173
G_C_C_G_A_G	4	0.0058
G_T_T_G_A_G	4	0.0058
A_C_T_G_A_G	1	0.0014
A_T_T_G_A_G	1	0.0014

(d) Haplotype frequencies, East Asian population

Haplotype	Count	Frequency
G_C_T_G_A_G	525	0.5208
G_T_T_A_A_G	258	0.256
G_T_T_G_A_A	181	0.1796
G_C_C_G_G_G	38	0.0377
G_T_T_G_A_G	4	0.004
G_C_T_G_G_G	2	0.002

(e) Haplotype frequencies, European population

Haplotype	Count	Frequency
G_C_C_G_G_G	436	0.4334
G_T_T_A_A_G	283	0.2813
G_C_T_G_A_G	240	0.2386
G_T_T_G_A_A	22	0.0219
G_C_C_G_A_G	16	0.0159
G_C_T_G_G_G	5	0.005
A_T_T_G_A_G	1	0.001
G_C_T_A_A_G	1	0.001
G_C_T_G_A_A	1	0.001
G_T_T_G_A_G	1	0.001

(f) Haplotype frequencies, South Asian population

Haplotype	Count	Frequency
G_T_T_A_A_G	349	0.3569
G_C_C_G_G_G	336	0.3436
G_C_T_G_A_G	204	0.2086
G_T_T_G_A_A	67	0.0685
G_C_T_G_G_G	16	0.0164
G_C_C_G_A_G	2	0.002
G_T_T_A_G_G	2	0.002
G_C_C_A_G_G	1	0.001
G_T_T_G_A_G	1	0.001

Table 3.4: Hapolotype frequencies per population for the imported SNPs

	rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931
Haplotype 0.2873	**G**	**C**	**C**	**G**	**G**	**G**
Traits			|slow acetylator due to n-acetyltransferase enzyme variant|			
Clinical Significance			drug response			
Haplotype 0.2686	**G**	**C**	**T**	**G**	**A**	**G**
Traits					|insulin resistance/response|	
Clinical Significane					drug response	
Haplotype 0.2624	**G**	**T**	**T**	**A**	**A**	**G**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr, metabolism/pk| |slow acetylator due to n-acetyltransferase enzyme variant|	|insulin resistance/response|	
Clinical Significane		drug response		drug response	drug response	
Haplotype 0.0771	**G**	**T**	**T**	**G**	**A**	**A**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr			|insulin resistance/response|	|slow acetylator due to n-acetyltransferase enzyme variant|
Clinical Significane		drug response			drug response	drug response
Haplotype 0.0341	**G**	**T**	**T**	**G**	**A**	**G**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr			|insulin resistance/response|	
Clinical Significane		drug response			drug response	drug response
Haplotype 0.0327	**G**	**C**	**T**	**G**	**G**	**G**
Traits						
Clinical Significane						
Haplotype 0.0212	**A**	**T**	**T**	**G**	**A**	**G**
Traits	|slow acetylator due to n-acetyltransferase enzyme variant	|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr			|insulin resistance/response|	
Clinical Significane	drug response	drug response			drug response	

Table 3.5: Haplotypes to traits association, Total population

	rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931
Haplotype 0.289	**G**	**C**	**C**	**G**	**G**	**G**
Traits			|slow acetylator due to n-acetyltransferase enzyme variant|			
Clinical Significance			drug response			
Haplotype 0.2307	**G**	**T**	**T**	**A**	**A**	**G**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr, metabolism/pk| |slow acetylator due to n-acetyltransferase enzyme variant|	|insulin resistance/response|	
Clinical Significane		drug response		drug response	drug response	
Haplotype 0.1218	**G**	**T**	**T**	**G**	**A**	**G**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr			|insulin resistance/response|	
Clinical Significane		drug response			drug response	
Haplotype 0.1121	**G**	**C**	**T**	**G**	**A**	**G**
Traits					|insulin resistance/response|	
Clinical Significane					drug response	
Haplotype 0.0976	**G**	**C**	**T**	**G**	**G**	**G**
Traits						
Clinical Significane						
Haplotype 0.0787	**A**	**T**	**T**	**G**	**A**	**G**
Traits	|slow acetylator due to n-acetyltransferase enzyme variant	|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr			|insulin resistance/response|	
Clinical Significane	drug response	drug response			drug response	
Haplotype 0.0287	**G**	**T**	**T**	**G**	**A**	**A**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr			|insulin resistance/response|	|slow acetylator due to n-acetyltransferase enzyme variant|
Clinical Significane		drug response			drug response	drug response
Haplotype 0.0234	**A**	**C**	**T**	**G**	**A**	**G**
Traits	|slow acetylator due to n-acetyltransferase enzyme variant				|insulin resistance/response|	
Clinical Significane	drug response				drug response	

Table 3.6: Haplotypes to traits association, African population

	rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931
Haplotype 0.3559	**G**	**C**	**C**	**G**	**G**	**G**
Traits			|slow acetylator due to n-acetyltransferase enzyme variant|			
Clinical Significance			drug response			
Haplotype 0.3285	**G**	**C**	**T**	**G**	**A**	**G**
Traits					|insulin resistance/response|	
Clinical Significane					drug response	
Haplotype 0.1715	**G**	**T**	**T**	**A**	**A**	**G**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr, metabolism/pk| |slow acetylator due to n-acetyltransferase enzyme variant|	|insulin resistance/response|	
Clinical Significane		drug response		drug response		
Haplotype 0.1124	**G**	**T**	**T**	**G**	**A**	**A**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr			|insulin resistance/response|	|slow acetylator due to n-acetyltransferase enzyme variant|
Clinical Significane		drug response			drug response	drug response
Haplotype 0.0173	**G**	**C**	**T**	**G**	**G**	**G**
Traits						
Clinical Significane						

Table 3.7: Haplotypes to traits association, American population

	rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931
Haplotype 0.289	**G**	**C**	**C**	**G**	**G**	**G**
Traits			|slow acetylator due to n-acetyltransferase enzyme variant|			
Clinical Significance			drug response			
Haplotype 0.2813	**G**	**T**	**T**	**A**	**A**	**G**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr, metabolism/pk| |slow acetylator due to n-acetyltransferase enzyme variant|	|insulin resistance/response|	
Clinical Signifi-cane		drug response		drug response	drug response	
Haplotype 0.2836	**G**	**C**	**T**	**G**	**A**	**G**
Traits					|insulin resistance/response|	
Clinical Signifi-cane					drug response	
Haplotype 0.0219	**G**	**T**	**T**	**G**	**A**	**A**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr			|insulin resistance/response|	|slow acetylator due to n-acetyltransferase enzyme variant|
Clinical Signifi-cane		drug response			drug response	drug response
Haplotype 0.0159	**G**	**C**	**C**	**G**	**A**	**G**
Traits			|slow acetylator due to n-acetyltransferase enzyme variant|		|insulin resistance/response|	
Clinical Signifi-cane			drug response		drug response	

Table 3.8: Haplotypes to traits association, European population

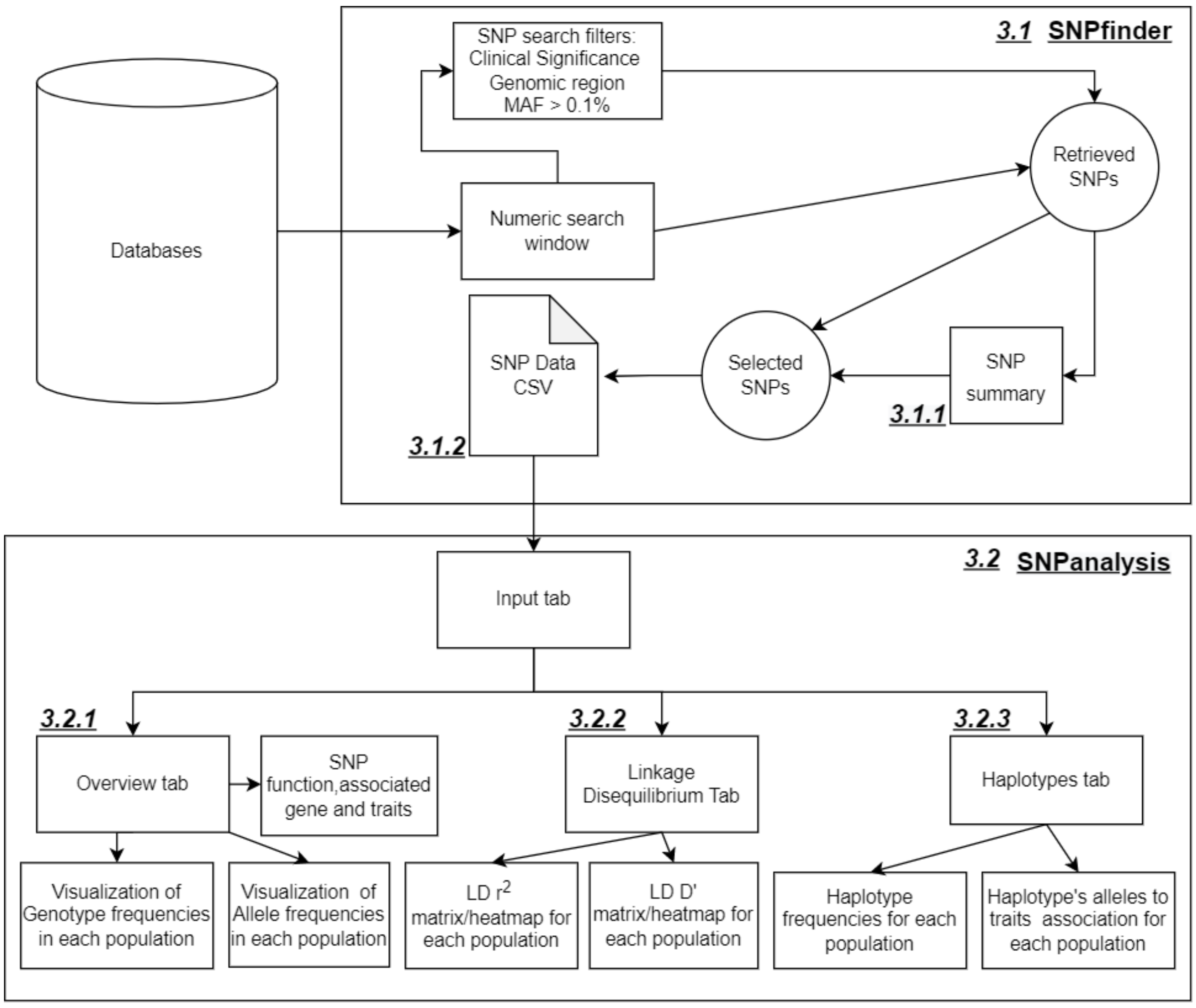
	rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931
Haplotype 0.3569	**G**	**T**	**T**	**A**	**A**	**G**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr, metabolism/pk| |slow acetylator due to n-acetyltransferase enzyme variant|	|insulin resistance/response|	
Clinical Significance					drug response	
Haplotype 0.3436	**G**	**C**	**C**	**G**	**G**	**G**
Traits			|slow acetylator due to n-acetyltransferase enzyme variant|			
Clinical Significane			drug response			
Haplotype 0.2836	**G**	**C**	**T**	**G**	**A**	**G**
Traits					|insulin resistance/response|	
Clinical Significane					drug response	
Haplotype 0.0219	**G**	**T**	**T**	**G**	**A**	**A**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr			|insulin resistance/response|	|slow acetylator due to n-acetyltransferase enzyme variant|
Clinical Significane		drug response			drug response	drug response
Haplotype 0.0159	**G**	**C**	**C**	**G**	**A**	**G**
Traits			|slow acetylator due to n-acetyltransferase enzyme variant|		|insulin resistance/response|	
Clinical Significane			drug response		drug response	

Table 3.9: Haplotypes to traits association, South Asian population

	rs1801279	rs1041983	rs1801280	rs1799930	rs1208	rs1799931
Haplotype 0.5208	**G**	**C**	**T**	**G**	**A**	**G**
Traits					|insulin resistance/response|	
Clinical Significance					drug response	
Haplotype 0.256	**G**	**T**	**T**	**A**	**A**	**G**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr, metabolism/pk| |slow acetylator due to n-acetyltransferase enzyme variant|	|insulin resistance/response|	
Clinical Significane		drug response		'drug response'	drug response	
Haplotype 0.1796	**G**	**T**	**T**	**G**	**A**	**A**
Traits		|ethambutol, isoniazid, pyrazinamide, and rifampin response - toxicity/adr			|insulin resistance/response|	|slow acetylator due to n-acetyltransferase enzyme variant|
Clinical Significane		drug response			drug response	drug response
Haplotype 0.0377	**G**	**C**	**C**	**G**	**G**	**G**
Traits						
Clinical Significane						
Haplotype 0.0159	**G**	**C**	**C**	**G**	**A**	**G**
Traits			|slow acetylator due to n-acetyltransferase enzyme variant|		|insulin resistance/response|	
Clinical Significane			drug response		drug response	

Table 3.10: Haplotypes to traits association, East Asian population

3.3 SNPop data retrieval, visualization and analysis pipeline



3.1 SNPfinder
Databases
SNP search filters: Clinical Significance Genomic region MAF > 0.1%
Numeric search window
Retrieved SNPs
Selected SNPs
SNP summary
3.1.1
SNP Data CSV
3.1.2
3.2 SNPanalysis
Input tab
3.2.1
Overview tab
SNP function,associated gene and traits
Visualization of Genotype frequencies in each population
Visualization of Allele frequencies in each population
3.2.2
Linkage Disequilibrium Tab
LD r2 matrix/heatmap for each population
LD D' matrix/heatmap for each population
3.2.3
Haplotypes tab
Haplotype frequencies for each population
Haplotype's alleles to traits association for each population

Figure 3.8: SNPop package data pipeline

Chapter 4

Discussion and Conclusions

4.1 Comparison with the existing pipeline for SNP data across populations

As discussed in chapter 1, tools to overcome the barrier of non-programmatic SNP data in bulk access exist. In this section, the BioMart to LDlink pipeline will be compared to SNPop's, figure (3.8). In order to properly identify SNPop's strengths and limitations the same SNP search criteria, mentioned in the results chapter (chapter 3) are used.

4.1.1 Comparison of data retrieval, BioMart & SNPfinder

BioMart is utilized for the retrieval of SNPs responsible for drug response existing at the NAT2 gene, with the following data-set, search filters and attributes. **Data-set:** Human Short Variants (SNPs and indels excluding flagged variants) (GRCh38.p13) **Filters:** variant set name: 1000 Genomes 3 - All, clinical significance: drug response, gene stable ID: ENSG00000156006 (The Ensembl stable ID for NAT2). **Attributes:** Variant name, variant source, chromosome, chromosome /scaffold position start (bp), minor allele (ALL) ,ancestral allele, global minor allele frequency (all individuals), clinical significance, phenotype description, associated variant risk allele .

Assessing the CSV output from $SNPfinder$, table (3.2) and BioMart, table (4.1.1), both software are accurate and return the same six SNPs: rs1208, rs1041983, rs1799930, rs1799931, rs1801279, rs1801280, along with almost identical association between SNP alleles and traits. $SNPfinder$ does not return the phenotype description: 'Slow acetylator due to N-acetyltransferase enzyme variant' for rs1208, the reason is that $SNPfinder$ associates SNP alleles with traits and since this particular trait has no specific allelic correlation, as shown in table (4.1.1), its not reported in the CSV output, table (3.2). BioMart offers an extensive selection of further important filters and attributes missing from $Snpfinder$, shown in tables (1.3.3, 1.5), most importantly the options to search SNPs associated with a specific phenotype, variant consequence and to return the p-value of phenotypes, as well as the impact on the protein's amino acid. However, when considering SNP frequency data across populations, BioMart allows only one 'variant set name' selection per search. This means that, if the user needs the frequencies of SNPs across population, separate searches should be done for each of them. Excluding allele frequency data, this results in file outputs with identical information. Furthermore $SNPfinder$ outputs the genotype frequencies of SNPs for each population, a feature unavailable in BioMart. A final fundamental difference is

Variant name	Variant source	Chromosome	Position start (bp)	Minor allele (ALL)	Ancestral allele	Global minor allele frequency (all individuals)	Phenotype description	Associated variant risk allele
rs1208	dbSNP	8	18400806	G	A	0.3229	Insulin resistance/response	A
rs1208	dbSNP	8	18400806	G	A	0.3229	Slow acetylator due to N-acetyltransferase enzyme variant	
rs1041983	dbSNP	8	18400285	T	C	0.3974	ethambutol isoniazid pyrazinamide and rifampin response - Toxicity/ADR	T
rs1799930	dbSNP	8	18400593	A	G	0.265	ethambutol isoniazid pyrazinamide and rifampin response - Toxicity/ADR Metabolism/PK	A
rs1799930	dbSNP	8	18400593	A	G	0.265	Slow acetylator due to N-acetyltransferase enzyme variant	A
rs1799931	dbSNP	8	18400860	A	G	0.07728	Slow acetylator due to N-acetyltransferase enzyme variant	A
rs1801279	dbSNP	8	18400194	A	G	0.02776	Slow acetylator due to N-acetyltransferase enzyme variant	A
rs1801280	dbSNP	8	18400344	C	T	0.2927	Slow acetylator due to N-acetyltransferase enzyme variant	C

Table 4.1: BioMart CSV output

the data-set of these software, Biomart has a selection of data-sets. Its Human SNP data-set includes deletions and insertions an unlike $SNPfinder$ it's not limited to SNPs with frequency data provided by the 1000 Genomes, as shown in figure (2.2). However the limited data-set of $SNPfinder$ was a conscious decision, made for two reasons:

- Since *SNpop* is an application package for the retrieval, small scale analysis and visualization of SNP population data and the 1000 Genomes project offers the biggest catalogue of variation data across populations [36].

- To reduce additional unnecessary response time, by searching for SNPs with no catalogued population data.

4.1.2 Comparison of data importation, Biomart to LDlink & SNPfinder to SNPanalysis

As discussed in section (2.5.1) and shown in figures (3.2, 3.8), $SNPanalysis$ is coded to recognize and read the CSV output of $SNPfinder$, table (3.2) as it is, without any need for file modifications. LDlink's web interface has an option for file input, shown in figure (4.1). However, it does not accept the CSV output of BioMart, table (4.1.1). As a workaround, the user can copy and paste only the rsIDs numbers of the SNPs from the CSV and paste them in the 'RS Numbers or Genomic Coordinates' input. This option though is non-intuitive and creates the need to come back to BioMarts CSV output to cross-reference its data with the data produced from LDlink, in order to retrieve complete results.

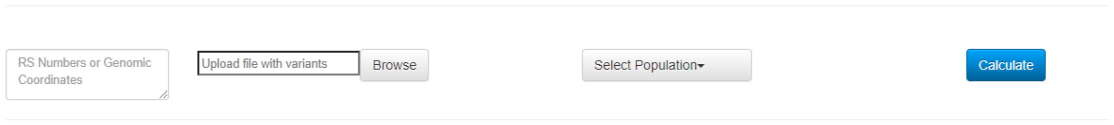
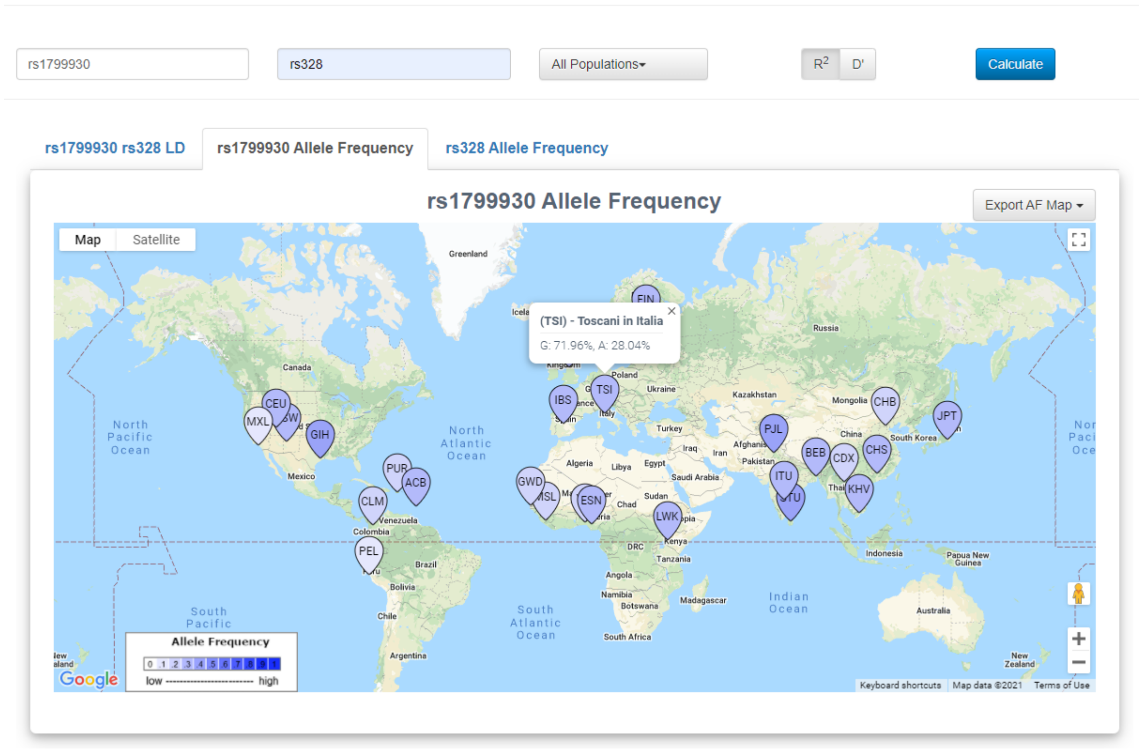


Figure 4.1: LDlink's web interface, input options

4.1.3 Comparison of data analysis and visualization, LDlink & SNPanalysis

$SNPanalysis$ utilizes functions to analyze and visualize data from the LDlinkR package, specifically the tools: LDhap and LDmatrix [20], along with data from $SNPfinder's$ CSV, detailed in section (2.5). The software produces comprehensive results regarding the visualization of allele and genotype frequencies of SNPs, their LD patterns and their haplotypes frequencies/trait association, detailed in section (3.2). The first fundamental difference between $SNPanalysis$ and LDlink lies in the fact that the latter allows the user to retrieve data of not only the five main populations, available in $SNPanalysis$, but also their sub-populations from the 1000 Genomes data-set. Regarding the abilities of the tools provided, $SNPfinder's$ SNP alleles and genotypes frequency visualization tool, found in the overview tab, figure (3.3), offers increased functionality in comparison to LDlink's alternative: LDpop (4.2), which does not include genotype frequency data or the SNPs alleles traits, shown in $SNPfinder's$ table (3.2.1). Considering the SNP LD data retrieval and visualization, since the D' and r^2 matrices of $SNPanalysis$,showcased in figures(3.4, 3.5), are obtained using the LDlinkR: LDmatrix package, the results are identical, with insignificant differences on the visualization side. As for haplotype retrieval and analysis LDlink's: LDhap tool provides haplotype frequencies across population, exactly as LDlinkR's does, shown in table (3.4). $SNPanalysis$ extends the results of LDlinkR by mapping each haplotype's SNP alleles to their respective traits, showcased


rs1799930 | rs328 | All Populations | R² D' | Calculate

rs1799930 rs328 LD | rs1799930 Allele Frequency | rs328 Allele Frequency

rs1799930 Allele Frequency

(TSI) - Toscani in Italia
G: 71.96%, A: 28.04%

Figure 4.2: LDlink, LDpop

in tables (3.5, 3.6, 3.7, 3.8, 3.9 and 3.10). This addition is crucial for understanding the impact certain SNPs have across populations. Finally LDlink offers two additional tools, not available in *SNPanalysis*, LDassoc and and LDexpress, providing SNP association p-value results and association with gene expression in multiple tissue types respectively.

4.2 Evaluation of findings

4.2.1 Snpfinder's CSV output

SNPs retrieved:

The arylamine n-acetyltransferase gene nomenclature committee assigns official symbols to 88 NAT2 alleles. NAT2*4 is the wild type allele, other alleles differ by having distinct combinations of SNPs [19]. Specifically, the SNPs responsible for alleles with impact on drug response are rs1799930, rs1799931, rs1801280, rs1801279, rs1041983, rs1208, rs72554616, rs1805158 and rs56054745 [35][19]. The last three SNPs are not returned by *SNPfinder*, as seen in the CSV output, table (3.2). This is not a software error as BioMart returns the same results. The reason for the omission of these SNPs is the fact that, unlike the six previous SNPs, rs72554616, rs1805158 and rs56054745 have no reported clinical significance in ClinVar. As a result *SNPfinder's* clinical significance: drug response filter does not include them in the output.

SNP data:

The returned SNP data for rs1799930, rs1799931, rs1801280, rs1801279, rs1041983, rs1208 are reliable, the SNP alleles, associated gene, chromosome position, traits and clinical significance of SNPs, as well as their genotype and allele frequencies across populations, when cross-referenced, match the expected data available in databases dbSNP, ClinVar and Ensembl.

4.2.2 SNPanalysis data

SNP Data visualization:

The 'Overview' tab, shown in figure (3.3), provides an accurate visualization of SNP allele and genotype frequnecies across populations, since the frequencies outputted in the CVS are reliable, as stated in section (4.2.1). Additionally, the LD r^2 and D' data of the 'Linkage Disequilibrium' tab, showcased in figure (3.2.2) were cross-referenced and found reliable using the Ensembl's REST API call: $GETld/ : species/pairwise/ : id1/ : id2$ which computes and returns LD values between the given variants across populations.

Haplotype analysis:

It's important to assess the significance of the added 'haplotypes to traits' functionality. The haplotypes (with population frequency >1%) to traits data for the drug response SNPs existing on NAT2, obtained from $SNPanalysis$ in chapter 3 and showcased in tables (3.5, 3.6, 3.7, 3.8, 3.9 and 3.10), written as NAT2 alleles, following the n-acetyltransferase gene nomenclature are shown in table (4.2). Regarding the acetylation speed of NAT2 produced enzymes,individuals with allelic diplotypes: NAT2*4, NAT2*12A and NAT2*13A are reported as rapid acetylators, while individuals with NAT2*5C, NAT2*6A, NAT2*7B, NAT2*14A, NAT2*14B, NAT2*5D are slow acetylators [35][19]. Additionally, SNP's rs1208, A allele, is associated with increased severity of Insulin Resistance [16]. Finally individuals with the allele NAT2*6A have increased increased risk of toxic liver disease when treated with isoniazid, pyrazinamide and rifampin in people with tuberculosis [4]. These reported findings, when cross referenced with the data in tables (3.5, 3.6, 3.7, 3.8, 3.9 and 3.10) match and prove that the association of haplotypes with traits produced by $SNPanalysis$ is accurate. However, additional traits for the haplotypes exist, but they are not returned by $SNPanalysis$ since it is limited by the phenotype data available in ClinVar and Ensembl. Regarding the frequencies of haplotypes, they match with the data presented by a worldwide population survey with variants of NAT2 from 128 population samples, these data are mentioned in section (1.2.4) [25]. One exception occurs again, for the same reason stated in previous sections, the study reports the NAT2 allele: NAT2*5B in the place of NAT2*5C in table (4.2).

(a) NAT2 allele frequencies, Total population

frequency	allele
0.2873	NAT2*5C
0.2686	NAT2*4
0.2624	NAT2*6A
0.0771	NAT2*7B
0.0341	NAT2*13A
0.0327	NAT2*12A
0.0212	NAT2*14B

(b) NAT2 allele frequencies, African population

frequency	allele
0.289	NAT2*5C
0.2307	NAT2*6A
0.1218	NAT2*13A
0.1121	NAT2*4
0.0976	NAT2*12A
0.0787	NAT2*14B
0.0287	NAT2*7B
0.0234	NAT2*14A

(c) NAT2 allele frequencies, American population

frequency	allele
0.3559	NAT2*5C
0.3285	NAT2*4
0.1715	NAT2*6A
0.1124	NAT2*7B
0.0173	NAT2*12A

(d) NAT2 allele frequencies, East Asian population

frequency	allele
0.5208	NAT2*4
0.256	NAT2*6A
0.1796	NAT2*7B
0.0377	NAT2*12A
0.0159	NAT2*5D

(e) NAT2 allele frequencies, European population

frequency	allele
0.289	NAT2*5C
0.2813	NAT2*6A
0.2836	NAT2*4
0.0219	NAT2*7B
0.0159	NAT2*5

(f) NAT2 allele frequencies, South Asian population

frequency	allele
0.3569	NAT2*6A
0.3436	NAT2*5C
0.2836	NAT2*4
0.0219	NAT2*7B
0.0159	NAT2*5D

Table 4.2: NAT2 Allele frequencies across populations

- NAT2*5C consists of T>C (rs1801280) and A>G (rs1208) [19].
- NAT2*5B consists of T>C (rs1801280) and A>G (rs1208) and C>T (rs1799929) [19].

SNP rs1799929 is a synonymous variant with no clinical significane, as reported in dbSNP and ClinVar. This results again to the exclusion of it in *SNPfinder's* results and in *SNPanalysis* haplotypes. However, as discussed the haplotypes to the reported traits association is still accurate, since the excluded SNP has no affect on the phenotype.

4.3 Conclusions

SNPop meets the expectations of a software for the collection, visualization and small scale analysis of catalogued data regarding single nucleotide polymorphisms across populations. Its able to obtain desired SNPs and their traits. Then by using allele and genotype frequnecy, as well as LD and phased haplotype data retrieved, it's able to seamlessly investigate their association between them in five main populations. The easy to follow pipeline, presented in figure (3.8), along with the GUI of *SNPfinder* and *SNPanalysis* makes this task a user-friendly experience, which was crucial for this thesis.

- **Strengths:** BioMart and LDlink provide increased general functionality on their own, however in regards to population genetics, BioMart requires multiple requests and the outputs need to be modified to fit LDlink's input. SNPop combines the collection of important SNP data for population analysis in one intuitive package. Additionally it provides genotype frequencies, as well as the association with haplotypes to traits across populations, which are unavailable in LDlink and Biomart.

- **Limitations:** Much like BioMart $SNPfinder$ is limited by data available in databases, this results in possible omitted SNPs and/or traits. Consequently, while SNPop can paint an accurate picture for the phenotype associated with a haplotype across populations. It's results shouldn't be taken for granted but as a direction for the user's further research on the topic.

- **Recommendation:** An addition that would greatly increase SNPpop functionality is the inclusion of the option to select specific sub-populations for each main population. This can be achieved by modifying the code of the data collection API functions in $SNPfinder$ as well as the LDlinkR functions in $SNPanalysis$, showcased in chapter 2.

Bibliography

[1] Michael J. Bamshad et al. “Human Population Genetic Structure and Inference of Group Membership”. In: *The American Journal of Human Genetics* 72.3 (Mar. 2003), pp. 578–589. ISSN: 00029297. DOI: 10.1086/368061. URL: https://linkinghub.elsevier.com/retrieve/pii/S0002929707605746.

[2] Anthony J. Brookes. “The essence of SNPs”. In: *Gene* 234.2 (July 1999), pp. 177–186. ISSN: 03781119. DOI: 10.1016/S0378-1119(99)00219-X. URL: https://linkinghub.elsevier.com/retrieve/pii/S037811199900219X.

[3] Annalisa Buniello et al. “The NHGRI-EBI GWAS Catalog of published genome-wide association studies, targeted arrays and summary statistics 2019”. In: *Nucleic Acids Research* 47 (D1 Jan. 8, 2019), pp. D1005–D1012. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gky1120. URL: https://academic.oup.com/nar/article/47/D1/D1005/5184712.

[4] Sze Ling Chan et al. “Association and clinical utility of NAT2 in the prediction of isoniazid-induced liver injury in Singaporean patients”. In: *PLOS ONE* 12.10 (Oct. 16, 2017). Ed. by Selvakumar Subbian, e0186200. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0186200. URL: https://dx.plos.org/10.1371/journal.pone.0186200 (visited on 01/03/2022).

[5] Laura Clarke et al. “The international Genome sample resource (IGSR): A worldwide collection of genome variation incorporating the 1000 Genomes Project data”. In: *Nucleic Acids Research* 45 (D1 Jan. 4, 2017), pp. D854–D859. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkw829. URL: https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkw829.

[6] Francis S. Collins, Michael Morgan, and Aristides Patrinos. “The Human Genome Project: Lessons from Large-Scale Biology”. In: *Science* 300.5617 (Apr. 11, 2003), pp. 286–290. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1084564. URL: https://www.science.org/doi/10.1126/science.1084564 (visited on 11/15/2021).

[7] *dbSNP Summary.* URL: https://www.ncbi.nlm.nih.gov/SNP/snp_summary.cgi (visited on 11/06/2021).

[8] Pierre-Luc Germain, Emanuele Ratti, and Federico Boem. "Junk or functional DNA? ENCODE and the function controversy". In: *Biology & Philosophy* 29.6 (Nov. 2014), pp. 807–831. ISSN: 0169-3867, 1572-8404. DOI: 10.1007/s10539-014-9441-3. URL: http://link.springer.com/10.1007/s10539-014-9441-3.

[9] Sara Goodwin, John D. McPherson, and W. Richard McCombie. "Coming of age: ten years of next-generation sequencing technologies". In: *Nature Reviews Genetics* 17.6 (June 2016), pp. 333–351. ISSN: 1471-0056, 1471-0064. DOI: 10.1038/nrg.2016.49. URL: http://www.nature.com/articles/nrg.2016.49.

[10] Kevin L Howe et al. "Ensembl 2021". In: *Nucleic Acids Research* 49 (D1 Jan. 8, 2021), pp. D884–D891. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkaa942. URL: https://academic.oup.com/nar/article/49/D1/D884/5952199.

[11] Iris E. Jansen et al. "Genome-wide meta-analysis identifies new loci and functional pathways influencing Alzheimer's disease risk". In: *Nature Genetics* 51.3 (Mar. 2019), pp. 404–413. ISSN: 1061-4036, 1546-1718. DOI: 10.1038/s41588-018-0311-9. URL: http://www.nature.com/articles/s41588-018-0311-9 (visited on 12/08/2021).

[12] N L Kaplan, R R Hudson, and C H Langley. "The "hitchhiking effect" revisited." In: *Genetics* 123.4 (Dec. 1, 1989), pp. 887–899. ISSN: 1943-2631. DOI: 10.1093/genetics/123.4.887. URL: https://academic.oup.com/genetics/article/123/4/887/5998829 (visited on 01/03/2022).

[13] Manfred Kayser and Peter de Knijff. "Improving human forensics through advances in genetics, genomics and molecular biology". In: *Nature Reviews Genetics* 12.3 (Mar. 2011), pp. 179–192. ISSN: 1471-0056, 1471-0064. DOI: 10.1038/nrg2952. URL: http://www.nature.com/articles/nrg2952.

[14] Kenneth Reitz. *Requests: HTTP for Humans™ — Requests 2.27.0 documentation.* URL: https://docs.python-requests.org/en/latest/ (visited on 01/03/2022).

[15] Chava Kimchi-Sarfaty et al. "A "Silent" Polymorphism in the *MDR* 1 Gene Changes Substrate Specificity". In: *Science* 315.5811 (Jan. 26, 2007), pp. 525–528. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1135308. URL: https://www.science.org/doi/10.1126/science.1135308 (visited on 01/03/2022).

[16] Joshua W. Knowles et al. "Identification and validation of N-acetyltransferase 2 as an insulin sensitivity gene". In: *Journal of Clinical Investigation* 125.4 (Apr. 1, 2015), pp. 1739–1751. ISSN: 0021-9738. DOI: 10.1172/JCI74692. URL: http://www.jci.org/articles/view/74692 (visited on 01/03/2022).

[17] Melissa J. Landrum et al. “ClinVar: public archive of interpretations of clinically relevant variants”. In: *Nucleic Acids Research* 44 (D1 Jan. 4, 2016), pp. D862–D868. ISSN: 0305-1048, 1362-4962. DOI: 10.1093/nar/gkv1222. URL: https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkv1222.

[18] Mitchell J. Machiela and Stephen J. Chanock. “LDlink: a web-based application for exploring population-specific haplotype structure and linking correlated alleles of possible functional variants: Fig. 1.” In: *Bioinformatics* 31.21 (Nov. 1, 2015), pp. 3555–3557. ISSN: 1367-4803, 1460-2059. DOI: 10.1093/bioinformatics/btv402. URL: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btv402 (visited on 12/10/2021).

[19] Ellen M. McDonagh et al. “PharmGKB summary: very important pharmacogene information for N-acetyltransferase 2”. In: *Pharmacogenetics and Genomics* 24.8 (Aug. 2014), pp. 409–425. ISSN: 1744-6872. DOI: 10.1097/FPC.0000000000000062. URL: https://journals.lww.com/01213011-201408000-00007 (visited on 01/03/2022).

[20] Timothy A. Myers, Stephen J. Chanock, and Mitchell J. Machiela. “LDlinkR: An R Package for Rapidly Calculating Linkage Disequilibrium Statistics in Diverse Populations”. In: *Frontiers in Genetics* 11 (Feb. 28, 2020), p. 157. ISSN: 1664-8021. DOI: 10.3389/fgene.2020.00157. URL: https://www.frontiersin.org/article/10.3389/fgene.2020.00157/full (visited on 11/02/2021).

[21] Linus Pauling et al. “Sickle Cell Anemia, a Molecular Disease”. In: 110 (1949), p. 6.

[22] *PyQt5 Reference Guide — PyQt v5.15 Reference Guide*. URL: https://www.riverbankcomputing.com/static/Docs/PyQt5/ (visited on 11/02/2021).

[23] R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2021. URL: https://www.r-project.org/.

[24] RStudio, Inc. *shiny: Easy web applications in R.* 2021. URL: http://shiny.rstudio.com.

[25] Audrey Sabbagh et al. “Arylamine N-Acetyltransferase 2 (NAT2) Genetic Diversity and Traditional Subsistence: A Worldwide Population Survey”. In: *PLoS ONE* 6.4 (Apr. 6, 2011). Ed. by John Relethford, e18507. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0018507. URL: https://dx.plos.org/10.1371/journal.pone.0018507.

[26] F. Sanger, S. Nicklen, and A. R. Coulson. “DNA sequencing with chain-terminating inhibitors”. In: *Proceedings of the National Academy of Sciences* 74.12 (Dec. 1, 1977), pp. 5463–5467. ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.74.12.5463. URL: http://www.pnas.org/cgi/doi/10.1073/pnas.74.12.5463.

[27] Eric Sayers. “The E-utilities In-Depth: Parameters, Syntax and More”. In: (), p. 21.

[28] Eric Sayers. “The E-utilities In-Depth: Parameters, Syntax and More”. In: (), p. 21.

[29] B. S. Shastry. “SNP alleles in human disease and evolution”. In: *Journal of Human Genetics* 47.11 (Nov. 2002), pp. 0561–0566. ISSN: 1434-5161, 1435-232X. DOI: 10.1007/s100380200086. URL: http://www.nature.com/articles/jhg200293 (visited on 11/15/2021).

[30] S. T. Sherry. “dbSNP: the NCBI database of genetic variation”. In: *Nucleic Acids Research* 29.1 (Jan. 1, 2001), pp. 308–311. ISSN: 13624962. DOI: 10.1093/nar/29.1.308. URL: https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/29.1.308.

[31] Montgomery Slatkin. “Linkage disequilibrium — understanding the evolutionary past and mapping the medical future”. In: *Nature Reviews Genetics* 9.6 (June 2008), pp. 477–485. ISSN: 1471-0056, 1471-0064. DOI: 10.1038/nrg2361. URL: http://www.nature.com/articles/nrg2361.

[32] Damian Smedley et al. “BioMart – biological queries made easy”. In: *BMC Genomics* 10.1 (Dec. 2009), p. 22. ISSN: 1471-2164. DOI: 10.1186/1471-2164-10-22. URL: https://bmcgenomics.biomedcentral.com/articles/10.1186/1471-2164-10-22.

[33] Matthew Stephens, Nicholas J. Smith, and Peter Donnelly. “A New Statistical Method for Haplotype Reconstruction from Population Data”. In: *The American Journal of Human Genetics* 68.4 (Apr. 2001), pp. 978–989. ISSN: 00029297. DOI: 10.1086/319501. URL: https://linkinghub.elsevier.com/retrieve/pii/S0002929707614244.

[34] Zachary D. Stephens et al. “Big Data: Astronomical or Genomical?” In: *PLOS Biology* 13.7 (July 7, 2015), e1002195. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.1002195. URL: https://dx.plos.org/10.1371/journal.pbio.1002195.

[35] *Supplementary Table 2: Distribution and annotation of NAT genes in sequenced genomes of protists (a)*. URL: http://nat.mbg.duth.gr/Human%20NAT2%20alleles_2013.htm (visited on 01/03/2022).

[36] The 1000 Genomes Project Consortium et al. “A global reference for human genetic variation”. In: *Nature* 526.7571 (Oct. 1, 2015), pp. 68–74. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature15393. URL: http://www.nature.com/articles/nature15393.

[37] The ENCODE Project Consortium. “An integrated encyclopedia of DNA elements in the human genome”. In: *Nature* 489.7414 (Sept. 2012), pp. 57–74. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature11247. URL: http://www.nature.com/articles/nature11247.

[38] The pandas development team. *pandas-dev/pandas: Pandas*. 2021. URL: https://doi.org/10.5281/zenodo.3509134.

[39] the RACI consortium et al. “Genetics of rheumatoid arthritis contributes to biology and drug discovery”. In: *Nature* 506.7488 (Feb. 2014), pp. 376–381. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature12873. URL: http://www.nature.com/articles/nature12873 (visited on 12/08/2021).

[40] Clare Turnbull et al. “The 100 000 Genomes Project: bringing whole genome sequencing to the NHS”. In: *BMJ* (Apr. 24, 2018), k1687. ISSN: 0959-8138, 1756-1833. DOI: 10.1136/bmj.k1687. URL: https://www.bmj.com/lookup/doi/10.1136/bmj.k1687.

[41] Bram Verstockt, Kenneth GC Smith, and James C Lee. “Genome-wide association studies in Crohn’s disease: Past, present and future”. In: *Clinical & Translational Immunology* 7.1 (2018), e1001. ISSN: 20500068. DOI: 10.1002/cti2.1001. URL: https://onlinelibrary.wiley.com/doi/10.1002/cti2.1001.

[42] Yumi Yamaguchi-Kabata et al. “Distribution and Effects of Nonsense Polymorphisms in Human Genes”. In: *PLoS ONE* 3.10 (Oct. 14, 2008). Ed. by Alan Christoffels, e3393. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0003393. URL: https://dx.plos.org/10.1371/journal.pone.0003393 (visited on 01/03/2022).

[43] Andrew Yates et al. “The Ensembl REST API: Ensembl Data for Any Language”. In: *Bioinformatics* 31.1 (Jan. 1, 2015), pp. 143–145. ISSN: 1460-2059, 1367-4803. DOI: 10.1093/bioinformatics/btu613. URL: https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btu613 (visited on 01/03/2022).