Visualization of Longitudinal Phenotypes in the Norwegian Mother and Child Cohort Study

Christoffer Hjeltnes Støle

Autumn 2018

Master's Thesis
Department of Informatics
University of Bergen

Preface

The Norwegian Mother and Child Cohort Study (MoBa) is a pregnancy cohort study with over 100,000 children enrolled. Data was gathered through questionnaires mailed to the mothers, but also in the form of biological samples where more than 15,000 trios (mother, father, and child) have been genotyped so far.

Data collected by MoBa is sensitive and its access is therefore restricted to protect the privacy of the study participants. This can make it difficult (or even impossible) to access the data, not only for parents and the general public, but also for scientists and medical professionals. To solve this issue, it is necessary to provide access to the data in a manner that is high-resolution without compromising participant privacy.

The MoBa data is multidimensional and contains longitudinal information on several phenotypes (such as height and weight) for the children, as well as data on certain variables for the parents. Based on the recorded variables, the MoBa cohort can be divided into various subgroups that can be studied separately or compared with each other. Furthermore, the genotyping data can be viewed at different scales: (i) genetic variants can be considered individually, (ii) in the context of their genomic location, or (iii) the entire genome can be considered as a whole. Finally, a good presentation of the data has to account for and take advantage of the complexity of the MoBa data.

Hundreds of gigabytes of summary statistics can be generated from the genotyping data from MoBa. Depending on the use case, only a small subset of this data is relevant to present to a user at a given time point. In order to present these subsets to the user quickly upon request, a bioinformatics system that can find and dispatch data in a short amount of time must be implemented.

This thesis demonstrates how the issues related to large-scale sensitive data access and dissemination can be solved through a publicly available web application able to handle the associated data volumes efficiently.

The source code of a prototype web application is available at github.com/helse-data/mobavis, and a demo can be tested at helse-data.no/demo.

Acknowledgments

Dr. Harald Barsnes and Dr. Marc Vaudel were the supervisors for this master's thesis, and their input, together with the input from Prof. Dr. Stefan Johansson and Øyvind Helgeland, MD, was essential for the web application to receive its current design and function. They also anchored its development to an active and relevant scientific environment.

Yehia Farag provided guidance on how to develop the web application and set up a server that it could be run from. Dr. Dominik Kopczynski and Dr. David Bouyssié provided valuable feedback during the selection of database software.

The hosting of the web server is funded by Dr. Nils Henrichsen og hustru Anna Henrichsens legat and L. Meltzers Høyskolefond and Bergen University Fund.

List of figures and tables

Abbreviations

API	Application Programming Interface
AWS	Amazon Web Services
BMI	Body Mass Index
CSS	Cascading Style Sheets
GWAS	Genome-Wide Association Study
HDD	Hard Disk Drive
JSON	JavaScript Object Notation
LD	Linkage Disequilibrium
MAF	Minor Allele Frequency
MoBa	Norwegian Mother and Child Cohort study
RAM	Random Access Memory
SEM	Standard Error of the Mean
SNP	Single-Nucleotide Polymorphism
SNV	Single-Nucleotide Variant
SSD	Solid-State Drive
QQ-plot	Quantile-Quantile plot

1. Background

1.1. Scientific context

1.1.1. Visualization of health data

Numerous web pages provide online visualizations of health data, either as their main focus or as part of many types of data visualized. Examples include Our World in Data (ourworldindata.org), Gapminder (gapminder.org) and IHME (healthdata.org). These websites all have an international perspective, unlike the web application presented here, which deals exclusively with Norwegian data. In terms of scope, this places our initiative closer to websites like Wellbeing in Germany (gut-leben-in-deutschland.de). Additionally, online resources visualizing data from the UK Biobank are in development, including holtzyan.shinyapps.io/UKB_geo and big.stats.ox.ac.uk.

For the visualization of child growth standards, the World Health Organization has published weight-for-length charts (who.int/childgrowth/standards/weight_for_length_height), showing a percentile distribution for weight against height. Weight-for-length is the predominant method used to assess the amount of fat tissue in children under the age of two [1].

The developed web application is anticipated to have three main user groups:

- scientists (particularly geneticists)
- medical professionals
- parents

Each group has different prior relevant knowledge and interest in health data. Of these groups, the scientists likely differ the most from the two other groups. Genetic data is mainly the domain of scientists, and both medical professionals and parents likely have as their primary interest health data describing the general population in a manner not relying directly on genetic information (e.g. the median weight for all children of a certain age in a given population).

Optimizing the web application for different user groups can be done in different ways, for example by:

- creating separate user interfaces for the different user groups
- making assumptions about which part of the web application the different user groups are most likely going to use, and optimize the individual parts of the application for the specific user groups

1.1.2. Availability of health data

Health data from individuals is sensitive and must be confined to a secure environment for the protection of privacy. Only non-sensitive derivatives can be made freely accessible, e.g. as summaries in scientific publications. Since most scientists cannot access sensitive health data directly, but through rather complex procedures, the effective usefulness of the original data for research is to a large extent limited to the research focus of scientists with primary access.

Norwegian health data (and more generally Nordic health data) presently has several advantages. It has detailed information on the study participants, and data on the same individual from different data sources can be connected through national identification numbers. The collected health data is rich, of good quality, and the Norwegian population is genetically homogenous and well-educated. Finally, the response rate to health studies is high, and Norway was an early adopter of digital information systems.

1.1.3. Cohort studies

Cohort studies are studies where specific groups of people, referred to as cohorts, are followed uniformly, possibly over time, and different variables, such as mortality, are measured. Many different cohort studies exist, with differences both in the selection and number of participants, and in the variables analyzed [2]. Cohort studies are considered one of the most important elements of modern epidemiology [3].

The American epidemiologist Wade Hampton Frost introduced the term *cohort study* in 1935 [3], though the analysis of cohorts with the purpose of studying diseases had been conducted previously [4], including studies by the Norwegian physician and tuberculosis researcher Kristian Feyer Andvord, who may also have been the first to use the method [5].

Birth cohort studies follow participants from birth, with participation often continuing until adulthood. Biological samples, such as blood, are collected by many such studies [6–8]. A distinction can be made between pregnancy cohort studies and birth cohort studies, where the former starts the data collection already during pregnancy, while the latter collects data from birth and onwards [9]. Several countries have large birth cohort studies, including Norway [10], Denmark [11], China [12], and the Netherlands [13], were several thousand, to more than one hundred thousand, mother-and-child pairs are enrolled in the individual studies [14,15].

1.1.4. The Norwegian Mother and Child Cohort Study

The Norwegian Mother and Child Cohort Study (MoBa) is a pregnancy cohort study recruiting more than 100,000 pregnant women between 1999 and 2008 with the aim of studying causes of disease. In excess of 114,000 children were enrolled in the study, making it the world's largest study of its kind [14]. Starting in the year 2000, over 75,000 fathers were also recruited to the study through the mothers [16–18].

As the study progressed, 50 of the 52 Norwegian hospitals with more than 200 births per year recruited pregnant women to the study, making it a national study. A few smaller clinics and private practitioners were also involved in the study.

To collect data, questionnaires were sent out to the mothers at particular stages of the pregnancy and ages of the child, starting from week 15 of the pregnancy (fhi.no /en/studies/moba/for-forskere-artikler/questionnaires-from-moba). Around week 17, when the pregnant women attended a routine ultrasound examination at the hospital, blood and urine samples were also collected. If the father of the child was accompanying the mother, from year 2000 onwards, blood samples were also collected from him. At birth, blood samples were collected from the mother and the child; the latter from the umbilical cord.

Through the questionnaires, different variables were recorded, for both the children and the parents. Data on the pregnancy itself includes duration, whether it was a multiple birth, whether the parents smoked during the pregnancy, and data on the the amniotic fluid (the fluid surrounding the fetus during pregnancy). For the children, among the data recorded was the height and weight for 12 different ages (from birth to age eight), behaviour (e.g. crying and hyperactivity) and disease (e.g. common cold and diabetes). The questionnaires also asked for height and weight for both parents, though only at one time point for the fathers and before and after the pregnancy for the mothers. Furthermore, the fields of the questionnaire concerning the parents also included questions about their health, such as whether they had been diagnosed with various diseases, plus their mental health and diet. Variables not directly describing the health or body, such as income and drug use, were also covered.

1.1.5. DNA, genomes and genotyping

Genomics is the study of genomes - the complete DNA content of an organism, a copy of which is found within most or all cells of an organism. DNA (Figure 1) is an organic molecule acting as a set of instructions directing the machinery of cells. It consists of two strands intertwined to form a double helix. Each strand has a

sugar-phosphate backbone and a sequence of the four bases adenine (A), thymine (T), cytosine (C), and guanine (G), attached along its full length. RNA is a closely related polymer that has a different sugar-phosphate backbone and the base uracil (U) in place of thymine. In regular DNA, each base pairs up with another base on the opposite strand, forming a base pair. Only two types of base pairing are observed in normal DNA (base complementarity): A pairs only with T (U in the case of RNA) and C only with G.

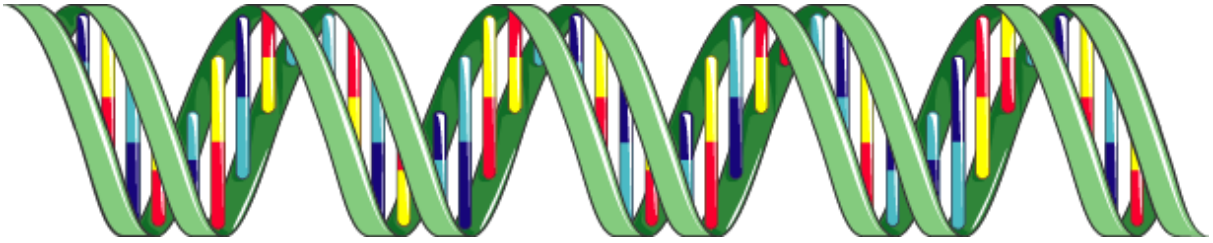

Figure 1: ***The structure of DNA.*** *In this figure, the sugar-phosphate backbone is shown in green and the four different bases are coloured yellow, red, cyan and dark blue.*
Figure by Servier Medical Art (smart.servier.com/smart_image/dna), licensed under CC BY 3.0 (creativecommons.org/licenses/by/3.0).

Genes are specific sequences of DNA that can encode proteins. Each gene consists of codons, which are DNA sequences three bases in length (Table 1). A given codon has a particular meaning in the genetic code, but multiple codons have the same meaning. Start codons mark where genes begin and stop codons where they end, and one of each is found within any given gene. The other codons encode one particular amino acid. More than one codon can encode the same amino acid: for example, the codons CGC and AGA both encode the amino acid arginine.

First base	Second base							
	U		C		A		G	
U	UUU	Phe	UCU	Ser	UAU	Tyr	UGU	Cys
	UUC		UCC		UAC		UGC	
	UUA	Leu	UCA		UAA	Stop	UGA	Stop
	UUG		UCG		UAG		UGG	Trp
C	CUU		CCU	Pro	CAU	His	CGU	Arg
	CUC		CCC		CAC		CGC	
	CUA		CCA		CAA	Gln	CGA	

	CUG		CCG		CAG		CGG	
A	AUU	**Ile**	ACU	**Thr**	AAU	**Asn**	AGU	**Ser**
	AUC		ACC		AAC		AGC	
	AUA		ACA		AAA	**Lys**	AGA	**Arg**
	AUG	**Met / Stop**	ACG		AAG		AGG	
G	GUU	**Val**	GCU	**Ala**	GAU	**Asp**	GGU	**Gly**
	GUC		GCC		GAC		GGC	
	GUA		GCA		GAA	**Glu**	GGA	
	GUG		GCG		GAG		GGG	

*Table 1: **The genetic code.** Each amino acid is in bold and referred to by the short form of its name. AUG encodes for methionine (Met) in addition to being a start codon. The code in this table refers to the codons as they are found on mRNA, where uracil replaces thymine.*

A protein is an organic molecule consisting of one or more chains of amino acids in an ordered sequence. The order of amino acids determines the structure and function of a protein. Before a protein can be made from a gene, a special type of molecule reads (transcribes) the sequence of the gene and creates an RNA molecule, known as messenger RNA (mRNA), carrying the gene sequence information. Finally, a large molecular complex known as a ribosome creates an amino acid chain based on the mRNA by adding amino acids one-by-one. The order of which amino acids are added by the ribosome is determined by the sequence of codons in the gene encoding the protein.

In all multicellular organisms, the DNA in a cell is organized in the form of chromosomes. Each chromosome is a highly compacted DNA molecule associated with various proteins. The human genome is organized into 23 chromosomes, where one copy of each is inherited from each parent. The chromosomes are further divided into somatic (chromosomes 1-22) and sex chromosomes (X and Y). While females have two copies of their sex chromosome (X), like they have for each somatic chromosome, males have one copy of the X chromosome and one copy of the Y chromosome.

The DNA of a cell can change, for instance by the alteration of a single base pair in a DNA molecule, by deletion of a sequence and loss or duplication of entire chromosomes. A change in the sequence of DNA is known as a mutation. The

alteration of a single base pair is a common form of mutation, and in any given population, the base pair at a given position of the DNA can vary between individuals. Mutations can have physical consequences, for example when one base pair is substituted with another within a codon. If the new codon encodes a different amino acid, the mutation is called a missense mutation. Since a protein resulting from a missense mutation has a different amino acid sequence compared to the original, it may function differently, or not at all.

Some mutations occur after a person was conceived (*de novo* mutations) and will most likely be present only in a few cells in the body, while other mutations are inherited from one or both parents. Mutations vary significantly in how common they are in a population. Of mutations occurring in a single base pair, the more common ones are known as single-nucleotide polymorphisms (SNPs, Figure 2). The four possible bases of a SNP are known as alleles, of which typically only two are common in a given population [19]. SNPs are one of the most common forms of genetic variation in the human genome [20], and the type of genetic variation considered in this thesis. Rare single-point mutations are often referred to as single-nucleotide variants (SNVs).

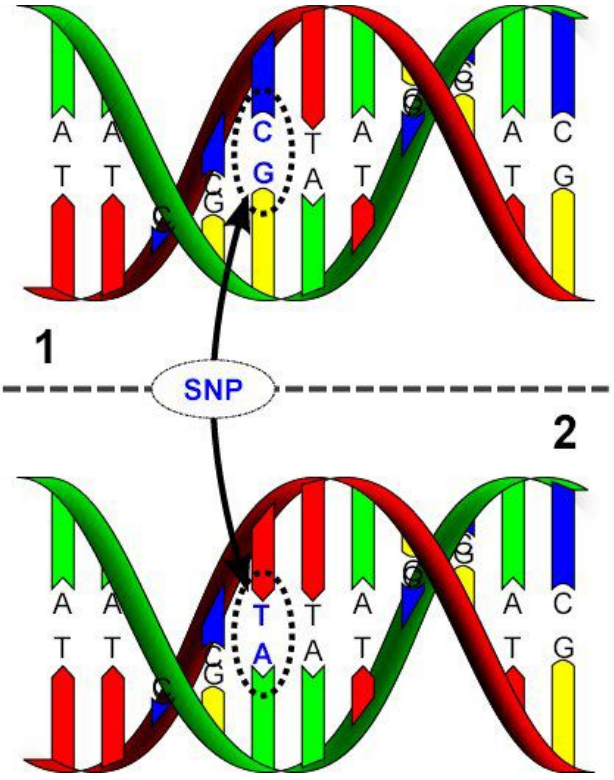

Figure 2: ***A single-nucleotide polymorphism (SNP).*** *In this figure, one of the DNA strands is shown in green and the other in red, and the individual bases are labeled with their one-letter abbreviations. The panels show the two different alleles of a SNP. When looking at the red strand, allele 2 has the base T at the same position where allele 1 has the base C.*

As required by base complementarity in DNA, the complementary bases on the green strands are also different for the different alleles.
(Figure by David Hall, published under the GNU Free Documentation License)

The genotype of an individual is their precise DNA sequence, which, for a given chromosome, is the particular base pair they have at each position. Since humans normally have two copies of each somatic chromosome, an individual's genotype for a given SNP comprises two sets of alleles: one allele on the copy of the chromosome inherited from the father and one allele on the copy of the chromosome inherited from the mother.

For a given SNP, the frequency of the least common base is called the minor allele frequency (MAF). More than 80 million SNVs in the human genome have been reported, regardless of MAF [21]. A reference genome has been constructed for humans [22], and the base of a SNP that is identical to the base found in the reference genome is referred to as the reference allele and is labeled A. The other base is referred to as the alternative allele and is labeled B.

Individuals with two copies of the same allele are referred to as homozygous and individuals with two different alleles are referred to as heterozygous. With this terminology, the three possible genotypes for a given SNP are:

- AA - the individual is homozygous for the reference allele
- AB - the individual is heterozygous
- BB - the individual is homozygous for the alternative allele

Linkage disequilibrium (LD) is a form of correlation between alleles in a genome. It accounts for the fact that the inheritance of genetic material from parents is non-random: genetic material is inherited chromosome-wise. Linkage disequilibrium can be measured by the square of the correlation coefficient (r^2) between SNPs in a given cohort [23]. A haplotype is a stretch of DNA on a single chromosome inherited from one parent, so all alleles of a haplotype are inherited together.

Meiosis is the process where precursor cells divide to form sperm or egg cells. During meiosis, genetic material is exchanged between chromosomes through the process of recombination. This breaks the inheritance linkage between neighbouring regions on a chromosome where the recombination occurred. Thus meiotic recombination affects the genetic diversity within a population by shuffling around genetic elements, creating novel genetic combinations [24]. The recombination rate is measured in units of centimorgans per megabase (cm/Mb), where 1 Morgan is the distance between two points on a chromosome for which the expected number of

genetic recombinations occuring on the sequence between them in a single meiosis event is equal to one [25].

Whereas DNA sequencing determines the full order of nucleotides in the DNA, genotyping establishes how a specific DNA sequence in the genome of an organism compares to a reference sequence through the evaluation of specific SNPs. On average, every 300th base pair in the human genome is a SNP[1], thus genotyping can be much more focused in scope compared to DNA sequencing while still covering many of the ways the DNA of an individual can differ from that of others.

Through linkage disequilibrium, the alleles of SNPs in the same haplotype can be imputed. For a given population, the stronger the linkage disequilibrium within a haplotype, the larger the probability that the genotype of a SNP will allow predicting the alleles within this haplotype. It is hence possible to infer the alleles of SNPs within a haplotype from the genotyping of haplotypic markers. Consequently, genotyping requires fewer resources and is less expensive compared to DNA sequencing while providing a good coverage of common variants. Unless specifically targeted by the genotyping array, rare and *de novo* variants are however not accessible. So far, a randomly selected subset of 15,000 trios (mother, father, and child) of the MoBa participants have been genotyped. For imputation, the IMPUTE2 software tool was used (mathgen.stats.ox.ac.uk/impute/impute_v2.html), and SNPs with an imputation score greater than 0.7 were considered of high imputation quality. In the resulting genotypes, more than 98.5% of the alleles are imputed, with the exact percentage varying from chromosome to chromosome.

A genome-wide association study (GWAS) evaluates the association between the genotype of individuals and their phenotype. The first GWAS results were published in 2005-2007 [26]. Such studies results in a list of effect sizes and *p*-values that describe the association of a given SNP with the phenotypic variable under consideration. In this thesis, the additive model is used for the association. In this model, the mean of a phenotype increases by *c* for individuals having a single copy of a so-called effect allele (AB) compared to those having none (AA), and by 2*c* for those having two copies of that allele (BB).

For the MoBa study, association *p*-values were not available until late in the thesis.

1.1.6. Visualization of GWAS data

1.1.6.1. Manhattan plots

Manhattan plots are scatter plots where the association *p*-values of individual SNPs are plotted against chromosomal coordinates (Figure 3). The *p*-values are

[1] ghr.nlm.nih.gov/primer/genomicresearch/snp

transformed with the formula $-\log_{10}(p)$, so the haplotypes with the most significant associations appear as vertical columns on the plot.

Although variations are found in the literature, 5×10^{-8} is widely accepted as genome-wide *p*-value significance threshold and is indicated by a horizontal line on the Manhattan plot. Another, lower threshold that can be encountered in the context of GWASs is the suggestive genome-wide significance threshold. It may e.g. be equal to 1×10^{-5}, and is also represented by a horizontal line in the plot. With the advent of very large GWASs, the usage of the suggestive threshold decreased and instead more stringent thresholds further controlling for multiple hypothesis testing are displayed on Manhattan plots (Figure 3).

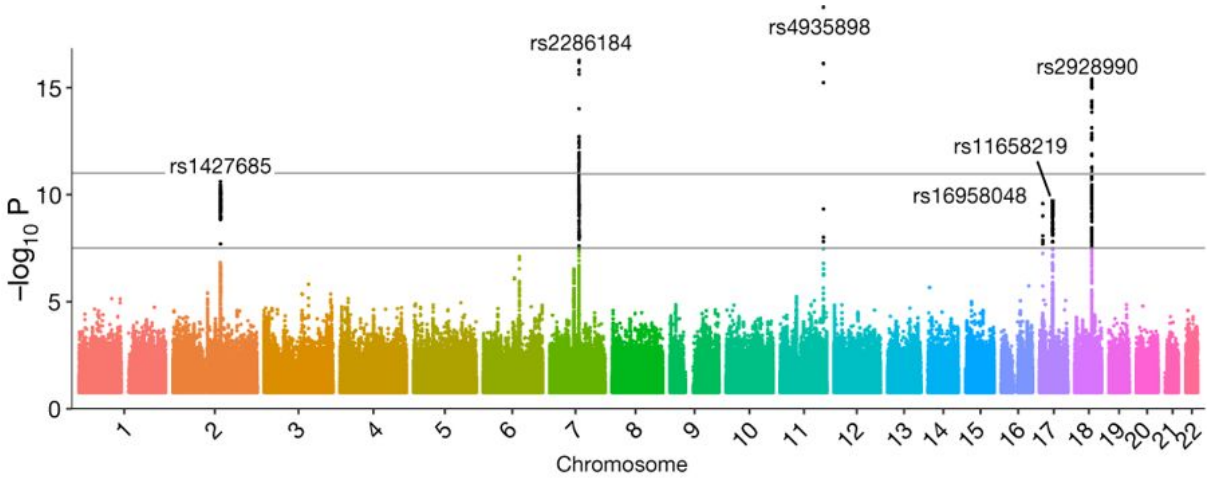

Figure 3: ***Manhattan plot.*** *The negative of the base-10 logarithm of association p-values of a GWAS are plotted against chromosomal coordinates. SNPs are coloured according to which chromosome they belong to. Several SNPs are mentioned by name in the plot. Two horizontal lines have been drawn for -log10(p) greater than 7.5 and 11, representing two significance thresholds for the study. Figure by Elliott et al. [27] with changes (label removed), licensed under CC BY 4.0 (creativecommons.org/licenses/by/4.0).*

1.1.6.2. Quantile-quantile plots

In quantile-quantile plots (QQ-plots), the quantiles of two distributions are plotted against each other (Figure 4). If the two distributions are similar, the points of the plot will be positioned along the diagonal.

For GWASs, QQ-plots are used for quality control: the quantiles of expected association *p*-values are plotted against the quantiles of observed *p*-values. Deviations from the diagonal for the majority of *p*-values indicates issues with the calibration of *p*-values. On the other hand, deviations from the diagonal for very small *p*-values are illustrative of significant association *p*-values.

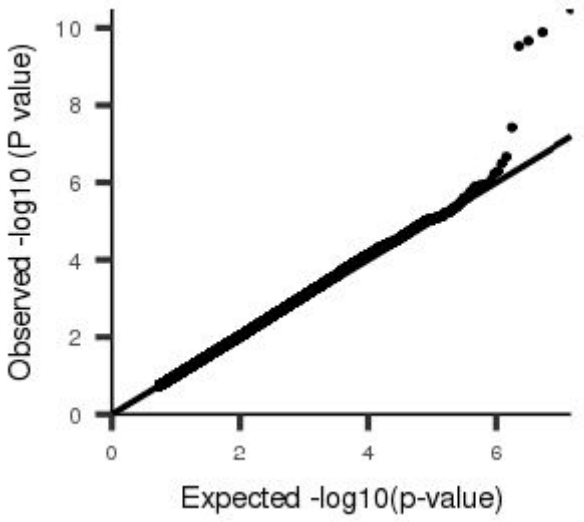

Figure 4: ***Example QQ-plot from the literature.*** *Observed association p-values are plotted against expected association p-values. Here, the p-values deviate from the diagonal at significant p-values only (observed $p < 10^{-7.5}$). Figure by Elliott et al. [27], without changes, licensed under CC BY 4.0.*

1.1.6.3. Regional plots

Like a Manhattan plot, a regional plot (or locus zoom plot) displays association *p*-values against chromosomal coordinates, but this time restricted to a locus (a specific region of the genome) centered around a given SNP of interest, highlighted with a diamond (Figure 5).

Where a Manhattan plot displays the entire genome at once, a regional plot thus provides a resolution in the *x*-axis allowing the distinction of individual SNPs. The level of linkage disequilibrium between SNPs in the locus and the SNP of interest are shown in color.

In addition, a regional plot shows known gene coordinates and the recombination rate in the chromosomal context of the SNP being studied.

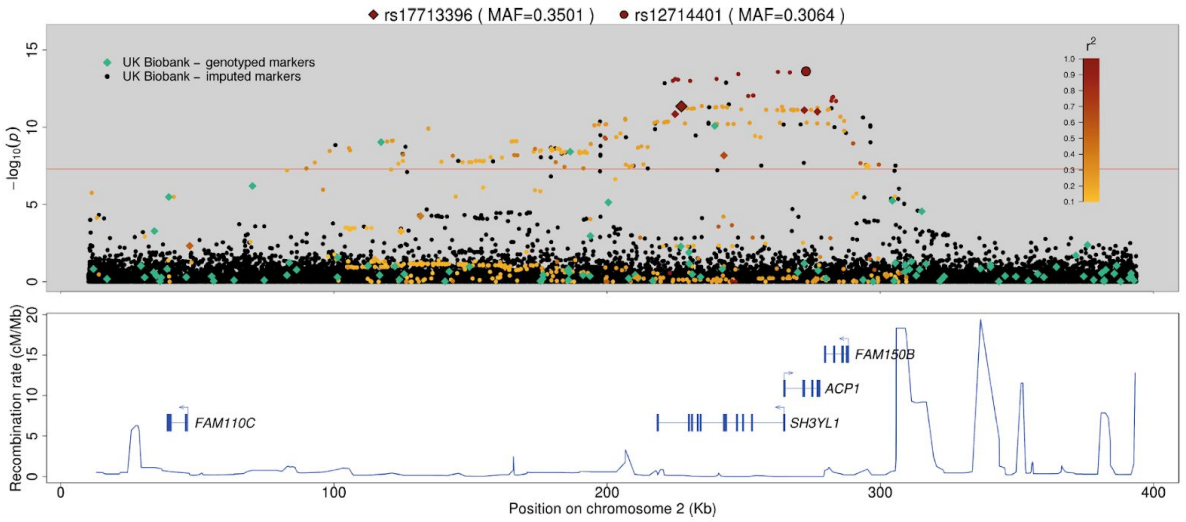

Figure 5: ***A regional or locus zoom plot for a region on human chromosome 2.*** *The association p-values are plotted in the upper panel against chromosomal coordinates. The LD relative to two reference SNPs (diamond) is represented by a colour scale from yellow to dark red. The recombination rate is plotted in the lower panel, where the coordinates of genes are also indicated by horizontal bars. Figure by Bycroft et al. [28], without changes, licensed under CC BY 4.0.*

1.1.7. Generated summary statistics of the MoBa data

Summary statistics on the MoBa data were generated for this master's project by the Johansson Group at the University of Bergen. They were generated on the HUNT Cloud (ntnu.edu/huntgenes/hunt-cloud), which provides a secure infrastructure for sensitive data. Annotation files, one for each chromosome, storing information on all SNPs included in the MoBa data were also generated. The annotation files are sorted by chromosomal coordinates and contain information on each SNP such as the chromosome it resides on, chromosomal coordinates and whether it was imputed or genotyped. Data that describes a number of individuals below ten was omitted to protect the privacy of the individuals.

The generated summary statistics can be divided into two categories: data with and data without genotype information. In the first category, the phenotypes are stratified, i.e. grouped, by SNP genotype. In the second category, the data comprises summary statistics for the phenotypes without any form of genotype information. The phenotypes stratified by genotype comprise a volume of several tens of gigabytes compressed, whereas the phenotype summary statistics (i.e. the data without genotype information) are less than 40 megabytes uncompressed. Both categories of data are stratified by sex.

The data on the children is longitudinal and covers the phenotypes height, weight and body mass index (BMI). The number of mothers answering the questionnaires

varied from age to age, and the number of children for which data was obtained at a given age are included for both categories of data.

In part due to time constraints, the generated summary statistics came in three different formats; one format for all the data with genotype information, and two formats for the phenotype summary statistics.

1.1.7.1. Phenotypes stratified by genotype

Generated data on the phenotypes stratified by genotype was output in the form of plain text files. All the genotyped SNPs and SNPs whose imputation has a quality score above a certain threshold (0.7) were included. Data on the parents and association *p*-values are not included. Many SNPs have a reference SNP ID number (rs number), and have names starting with *rs* followed by their rs number. For the purpose of this project, SNPs without rs numbers were given unique internal identifiers that included their chromosome, chromosome position, reference allele and alternative allele, all separated by underscores. For example, a SNP at position 154,729,900 on chromosome 1 with reference allele T and alternative allele G is named 1_154729900_T_G.

For each sex, age, and genotype, three summary statistics are generated for the phenotypes:

- median - the 50th percentile, the point where one half of the sample population with the given genotype falls above and one half falls below
- an inter-quantile based estimation of the (SEM) - $\pm\frac{1.58\,IQR}{\sqrt{n}}$ [29,30], the standard error of the mean of the phenotype of the individuals with the given genotype
- 2.5th and 97.5th percentiles - marks the area where 95% of the individuals of the cohort with the given genotype falls within

1.1.7.2. Phenotype summary statistics

The variables of the phenotype summary statistics (Table 2) come in two categories: continuous (e.g. pregnancy duration) or discrete (e.g. whether or not the birth was premature). For the the continuous variables, every fifth percentile from the 5th percentile to the 95th percentile was calculated (and for each age for the longitudinal variables). Conditioned statistics were also generated (Table 3): for each continuous variable (e.g. height), each participant was placed in several groups based on which percentile they belonged to for a set of other variables (such as weight), and summary statistics were calculated for each such group.

Data on the parents are also included in the phenotype summary statistics; including height, weight and BMI, but also information on income, smoking during pregnancy and drug use. Additional information includes the pregnancy duration and whether or not the child was born prematurely. For some variables, there was a checkbox on the MoBa questionnaire to indicate if something was the case (e.g. if the child was fed breast milk at an of age of 6 months), but there was no checkbox to indicate if it was not, such that it is unknown whether the the question was overlooked, or if "no" was the correct value. For variables that had a "no" option, the number of missing values was included in the statistics.

The phenotype summary statistics were generated in two batches: statistics for the continuous variables were generated early in the thesis, and statistics for the the discrete variables later in the thesis. The first batch contained variables describing the children and both parents, while the later batch mostly contained variables describing the mother.

Child	**Mother**	**Father**	**Pregnancy**
Height*	Height	Height	Duration
Weight*	Weight	Weight	Amniotic fluid[a]
BMI*	BMI	BMI	Premature birth[b]
Breast milk**			Caesarean section[c]

** from birth to age eight*

*** from birth to 18 months of age; either "yes" or "no value"*

[a] *whether the amount of fluid was less or more than normal, and whether it was malodorous or infected*

[b] *yes or no*

[c] *whether the Caesarean was elective, an emergency, unspecified, or "no value"*

Table 2: ***Sample of variables recorded by MoBa questionnaires and included in the generated summary statistics.*** *Discrete variables have grey backgrounds. Variables with longitudinal data are marked with one or more asterisks.*

Variable	**Condition variable**	**Condition**
Weight	Height at birth	< 5th percentile
		< 10th percentile

		< 25th percentile
		25th to 75th percentile
		> 75th percentile
		> 95th percentile
	Height at 6 weeks of age	...
	Height at 3 months of age	...
	...	...
Height	Height at birth	...

Table 3: ***Conditioned variables.*** *In this table, the height of children is used as an example of a conditioned variable. For each conditioned variable, a second variable is used as the condition. Highlighted in blue is the weight for the group of children whose height at birth was less than the 25th percentile.*

1.1.8. Storing and retrieving health data

To manage large amounts of data efficiently, some form of database system is needed. The speed of a database system depends on both software and hardware. Two common types of database architectures are relational databases and key-value stores. In relational databases, the data is stored in tabular form, and the user requests rows or columns. Key-value stores have their data stored in the form of pairs of keys and values, and the user requests a key to retrieve the associated value. Memory-mapped files are an alternative to database software. Here a file is loaded into the computer's memory, so that the file contents can be read directly from memory rather than from the computer's persistent data storage.

A database needs a storage medium, and different storage media have different read speeds and different data persistence. Three commonly used storage media today are random access memory (RAM), hard disk drives (HDDs) and solid-state drives (SSDs). RAM is faster than both SSDs and HDDs, but is a form of volatile memory and loses its data when powered off. RAM can be converted into a so-called RAM drive that acts as a regular storage medium as long as the computer is powered on. SSDs and HDDs are both non-volatile forms of memory and retain their data when powered off. SSDs are significantly faster than HDDs, but also cost significantly more per unit of storage capacity.

1.2. Main challenges

Data access and participant privacy

A bioinformatics framework that provides access to MoBa data should make the data available at a fine level of granularity to maximize its usefulness, without compromising the privacy of the MoBa participants. It should be intuitive and easy to use for parents, medical professionals, and scientists.

Data complexity

The informatics framework developed should account for the complexity of the MoBa data, including visualizing data on different subgroups of the MoBa cohort. It should present the genetic data at the different levels of the human genome: genome-wide level, locus level, and SNP level. Methods for the visualization of multiple longitudinal phenotypes simultaneously, which could help spot health trends, should be evaluated.

Data management

Large volumes of summary statistics were generated from the MoBa data, presenting challenges for its management and the responsiveness of the interface. Requested subsets of the data should be displayed by the informatics framework in a reasonable amount of time, not leaving the user waiting and discouraging them from further data queries.

1.3. Proposed solutions

Develop a prototype web application addressing the above problems as detailed below.

Data access and participant privacy

By relying on summary statistics, MoBa data are made accessible to the different user groups without compromising patient privacy. The summary statistics are not sensitive as they cannot be used to identify individuals represented by the data. By allowing the refinement of the summary statistics according to covariates, the data can furthermore be navigated at fine levels of granularity.

By presenting the data in a web application with an intuitive user interface, navigating multidimensional data is also simplified. The user interface is designed with all three user groups in mind.

Data complexity	Well-established standard methods for the visualization of genetic data at the genome-wide (Manhattan plots) and locus levels (regional plots) exist and are implemented for the web application. At the SNP level, the phenotypes stratified by genotype are plotted against age in a line chart. Summary statistics for subgroups of the MoBa cohort are made accessible through and visualized in the web application. Different visualization methods will be evaluated for their ability to show multiple longitudinal phenotypes in the same plot in a meaningful way.
Data management	Different database software and different types of data storage hardware are compared for their suitability for storage and retrieval of the generated MoBa summary statistics. Of particular importance is the effective read speed of the combined software and hardware system.

2. Methods and tools

2.1. Server

The web application is deployed on a Lightsail virtual private server provided by Amazon Web Services (AWS, aws.amazon.com) running Amazon Linux, managed through the Secure Shell client PuTTY. The server is hosted on SSDs, providing fast storage for the MoBa summary data. A web domain was purchased for the application and a Transport Layer Security certificate was obtained free of charge from the certificate authority Let's Encrypt (letsencrypt.org) to enable encrypted (HTTPS) communication with the web application. The summary statistics was uploaded to the LightSail instance using WinSCP, and is accessed via the file system on the instance. No sensitive data is stored on the server.

The two server softwares Glassfish and Apache Tomcat 8 were used for the web application. Whereas Apache Tomcat was run on an online web server, Glassfish was run locally on the computer where the web application was developed, and made it possible to see how changes to the code affected the application without having to deploy it on the online server. During deployment of the web application, a WAR file is first built from the project source code and uploaded to the Lightsail instance. The WAR file contains all the code of the web application, but none of the data to be visualized. On the Lightsail instance, the WAR file is deployed on an

Apache Tomcat 8 running on the instance, and the web application is accessible over the Internet through HTTPS requests.

New versions of the web application are deployed by stopping the Tomcat server, deleting the files of the previous deployment and then restarting the server. It is possible to deploy a new version without restarting the Tomcat server, but this often leads to memory leaks on the server.

The internal routing of HTTPS requests in the Linux operating system was modified to make port numbers redundant in the URL of the web application, since the Tomcat server by default only listens to certain port numbers that differ from the standard for both HTTP and HTTPS. The settings of the Tomcat server and files specific to the web application were configured so that an attempt to establish an HTTP connection with the web application automatically results in an HTTPS connection instead.

2.2. Back end

The web application (see Figure 6 for an hierarchical overview) was written in the programming languages Java and JavaScript. Visualizations are generated by the JavaScript code, which is run in the web browser. Server-side, Tomcat runs the Java code, which operates the database system and dispatches the data from the server that the JavaScript code should display. It also sets up the user interface, which is ultimately rendered by the browser. Styling of the web application, i.e. the configuration of the appearance (such as colour) of individual user interface elements, was done using the Cascading Style Sheets (CSS) extension Sass.

JavaScript Object Notation (JSON) is used for the communication between the Java code and the JavaScript code, which stores data in the form of key-value pairs. A special Java class stores JSON objects, and special JavaScript script (known as a connector) registers when new JSON objects are stored in the Java class and passes on the JSON objects to other JavaScript code, such as the code for a plot.

For communication from the JavaScript side to the Java side, a Java class can create a JavaScript function callable from the JavaScript code that can pass information from the JavaScript code to the Java code. This way, the Java code can be notified of events such as the user clicking on a plot.

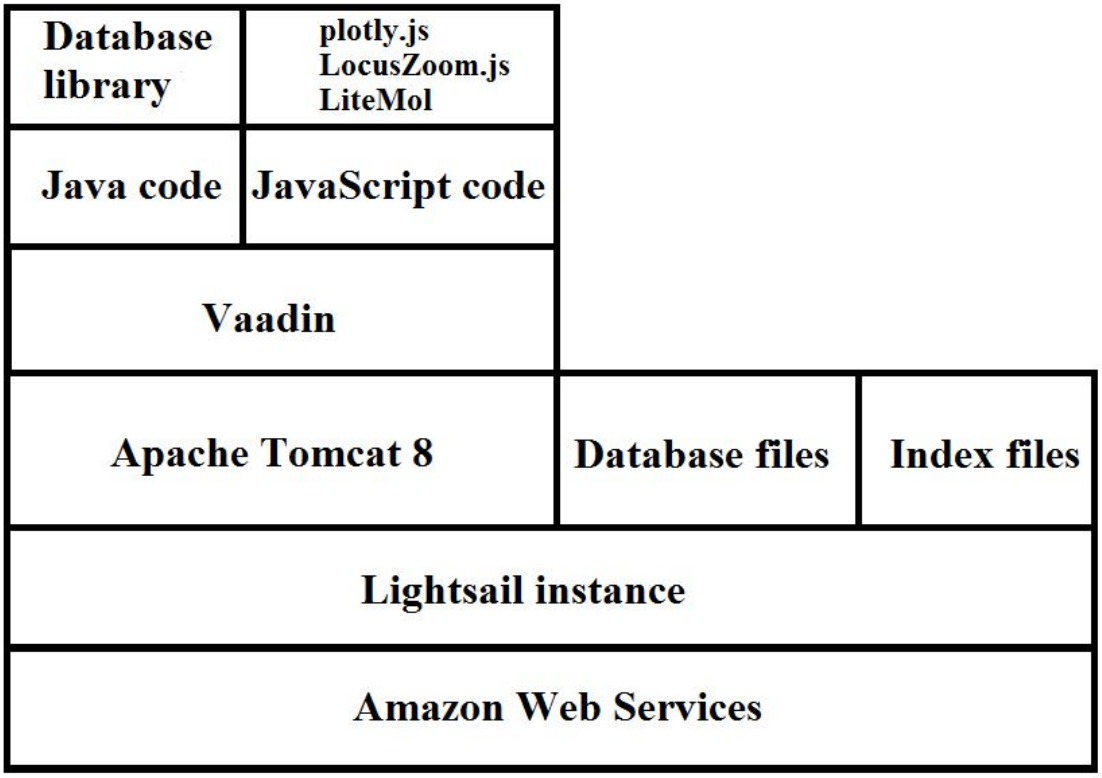

Figure 6: ***Hierarchical overview of the prototype web application and its server environment.*** *Amazon Web Services provides the hosting of the web server, a Lightsail virtual private server. The virtual private server hosts the database files and supporting index files and runs Apache Tomcat 8 in the same environment. Tomcat executes the web application, which has Vaadin as its fundament and is programmed in both Java and JavaScript.*

The source code for all of the web application was developed using the integrated development environment (IDE) NetBeans, including the styling. Apache Tomcat and Glassfish are both integrated within NetBeans, such that either software can be launched directly from within the IDE, which also builds the WAR files deployed on Tomcat 8 on the AWS server or on the GlassFish server locally. Software dependencies were handled using Maven, also integrated within NetBeans.

Appearance and performance of the web application was mainly tested in Google Chrome, and occasionally in Firefox and Opera. The fundament of the web application is provided by Java framework Vaadin (Figure 6). It is the Vaadin code that is initiated by the web server and that must itself initiate the rest of the code of the web application, either directly or indirectly. JavaScript code can readily be integrated with Vaadin and extend the functionality of the framework. This makes it possible to include existing JavaScript libraries directly in Vaadin applications, including plotting libraries and other visualization tools.

The Java code is highly modular, and each visualization (such as for the SNP level or the Manhattan plot) is given its own Java class. An instance of a controller class storing the SNP currently selected by the user is passed around to the different visualization instances, so that the selected SNP is not reset when a different visualization is selected. The controller object also allows the code in a visualization class to change which visualization is currently active and displayed to the user.

The Java database libraries RocksDB (rocksdb.org, a key-value store) and SQLite (sqlite.org, a relational database) were evaluated for their suitability for the web application, particularly for the genotyping data. For both SQLite and RocksDB, database files containing genotyping data from MoBa were created, with one file per chromosome. The SQLite files were provided early in the thesis by the Johansson Group to provide a starting point for the thesis. Later, a new version of the genotyping data were provided in plain text files, from which RocksDB database files were created. The SQLite and RocksDB database files covered similar size ranges, with the SQLite files ranging from 4.88 to 31.4 GB in size, and the RocksDB files from 3.81 to 17.6 GB. Because of the time and effort it would take to create RocksDB and SQLite based on the same underlying data, such files were not created, and the database files containing the genotyping data were used for performance comparison within the web application, but not for benchmarking of the database libraries.

For all benchmarking, a HP Envy 700-306no stationary computer running Windows 8.1 64 bit with 8 GB RAM, an Intel Core i5 4460 processor (four cores with a speed of up to 3.2 GHz), and an NVIDIA GeForce GTX 770 graphics card with 2048 GB GDDR5 memory, was used.

Database benchmarking was done outside of the web application with randomly generated data. For the random data, a set of 100,000 pairs of keys and values in the form of random text strings were generated. The size of the keys and values of the randomly generated strings were intended to be similar to those of the IDs of SNPs and the values stored, respectively. RocksDB and SQLite databases with different numbers of entries were created from the same subset of the randomly generated data, ranging in size from 1-10,000 entries (the latter corresponding to 230-250 megabytes of database file size) were used for benchmarking.

For use within the application, both database libraries were integrated directly within the Vaadin code, and do not run as separate applications on the web server. They are both supported by an index system that identifies which chromosome a given SNP resides on, so that the database library can query the correct database file (Figure 7), if the SNP is requested by its *rs*ID.

The index system consists of a master index that points to the other index files, with each index file storing which chromosome a range of SNPs belong to, a range defined by their rs numbers. All indexes are numerically sorted, meaning that a given index file for instance can store which chromosome each SNP with an rs number between 20,000 (rs20000) and 40,000 (rs40000) resides on. In practice, since not every SNP with an rs number in a certain range is likely to be present in the MoBa data, the actual ranges of the indices are irregular in the current implementation, necessitating a master index.

If the user inputs a chromosome and a position instead of the ID of the SNP, the annotation file for the input chromosome is read line by line until either a SNP with a matching position is found, or the position of the SNP on the line of the file currently being read is beyond the input position. If a matching SNP is retrieved from the annotation file, its ID is used to retrieve its data from the database system as previously described. If a SNP lacks an *rs*ID and is requested by its unique internal identifier (for example 21_28849120_G_C), the chromosome is included in its ID and the database system already knows which database file to query.

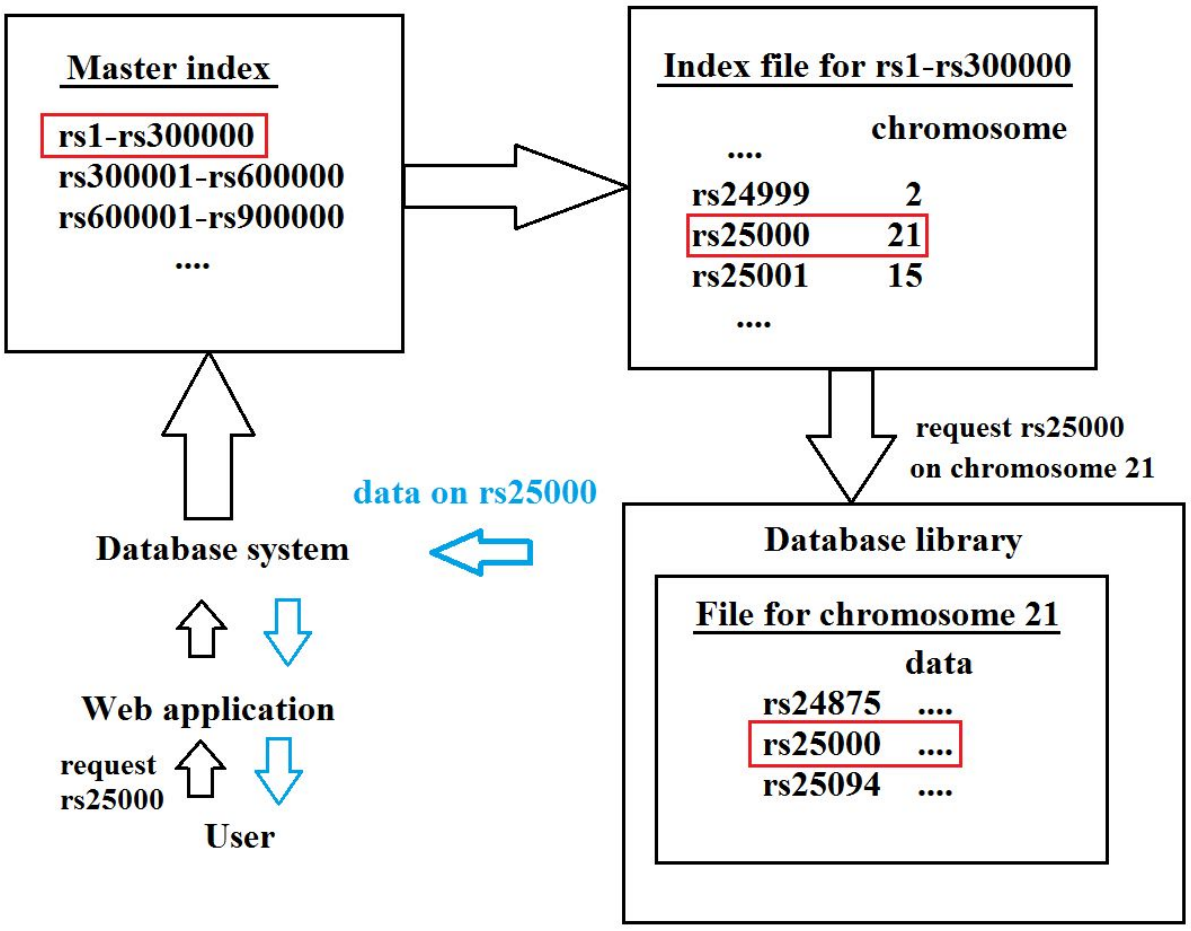

Figure 7: ***A request in the database system.*** *When a user requests a particular SNP through the user interface (rs25000 in this case), the database system of the web application searches the numerically sorted master index for the correct index file. The index file, also numerically sorted, is then searched for the name of the SNP itself. The index file provides the chromosome the SNP is located on and hence which database file the database library should query. Data stored on the SNP in the database is then returned for utilization by the rest of the web application and ultimately presented to the user. Unlike in this figure, the ranges of the index files in the actual web application are not predictable.*

Two methods were used to store and retrieve the phenotype summary statistics. The first method was used for the early batch of summary statistics containing continuous variables, while the later method was used for the later batch containing discrete variables. The first method is the simplest, and stores data in plain text files that are parsed by the application, which then loads the parsed data in-full into memory. The second method is more complicated and relies on memory-mapped files. An Application Programming Interface (API) to query the memory-mapped files was made available by the Johansson Group, and implemented allowing the web application to query these files. Time did not permit the evaluation of whether one of the methods were preferable for storing MoBa data for the web application. Since the

data stored in the two different formats also is different, both methods are in use by the current implementation web application.

For the visualization at the SNP level, the API of the dbSNP database [20] is automatically queried when a SNP is selected by the user through the interface. The request returns a JSON object that is parsed and searched for information of interest, such as which locus the SNP resides in by dbSNP's definition.

2.3. User interface

The user interface was built using elements provided by the Vaadin framework. Vaadin provides a range of user interface elements (referred to as components), such as buttons, drop-down lists, and input fields. Several component can contain other components, a property making the Vaadin interface highly modular as components can be programmatically added, removed, or substituted on the fly as needed.

In a Vaadin application, components are organized by a type of components known as layout components, which organize other components in the user interface by placing them side by side vertically or horizontally, or in a grid. The tab sheet component (Figure 8) organizes content in the form of tabs, with each tab holding another component. In the web application presented here, a tab sheet rather than a layout component acts as the root component and fills the page in the web browser. The data with genotype information and the phenotype summary statistics are given two separate tabs (the outer tabs) in this tab sheet. Each of these two tabs in turn contain a new tab sheet, with each of their tabs (the inner tabs) having a particular visualization component as its content. Each visualization component consists of one or more plots and a tailored interface that lets the user manipulate the plots and select which data to visualize.

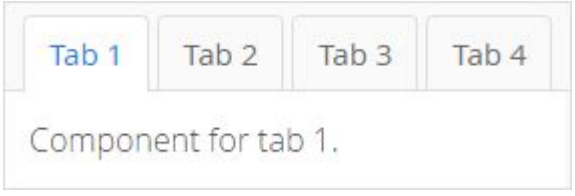

Figure 8: ***The tab sheet component in Vaadin.*** *Each tab contains another component, such as another tab sheet.*

No particular design choices were made for the user interface in terms of optimizing it for the three user groups, though the two outer tabs do to some extent emulate two separate user interfaces: the tab for data with genotype information is most relevant for scientists, while the tab for phenotype summary statistics is most relevant for medical professionals and parents.

As (i) permission was not obtained from the owners of MoBa to make summary statistics from the study available, and (ii) the genotyping data was unpublished, a landing page with a password feature was implemented to prevent unauthorized access to the data through the AWS server.

2.4. Visualizing data in the web browser

All visualizations in the web application are rendered in the web browser by JavaScript libraries. Thus, only the data to be visualized is sent from the server to the client, and not any graphics. Several JavaScript libraries are used: the regional plot is implemented using the library LocusZoom.js [31] and LiteMol [32] is used to allow the user to visualize the 3D structure of proteins within the application. All other visualizations are implemented with the use of the open-source JavaScript library plotly.js (plot.ly).

An important aspect of the JavaScript libraries used is that they are interactive, so that the user can quickly and intuitively adjust the way the data is being presented, and interact with it. Plotly.js was chosen among five free interactive JavaScript plotting libraries considered. It supports zooming and panning, it does not depend on other JavaScript libraries and provides an extensive selection of plot types. It is built on top of d3.js (d3js.org) and stack.gl (github.com/stackgl), and is under active development.

As association *p*-values from MoBa were not available until late in the thesis, *p*-values from other studies were instead used for the development of the Manhattan plot and the regional plot. For the regional plot, *p*-values from a type 2 diabetes GWAS performed by the DIAbetes Genetics Replication And Meta-analysis (DIAGRAM) Consortium was used [33]. The data from this study was store online in a format that meant it could be requested directly from LocusZoom.js; none of this data was stored on the server. LocusZoom.js supports using own data sources, such that the library is ready to use association *p*-values from MoBa.

The development of the Manhattan plot used a dataset from the GIANT BMI Exome Array containing more than 246,000 SNPs [34]. This dataset was stored in a plain text file on the server and parsed and loaded into memory by the web application. Using MoBa *p*-values for the Manhattan plot is a simple matter of storing them in a plain text file with the same format used for the GIANT BMI Exome Array data.

Given the large number of SNPs expected to have their association *p*-values plotted in a single Manhattan plot (up to 10 million or more), the plot performance for large number of data points is important. Therefore, the Manhattan plot was created with

the WebGL option for the Plotly scatter plot, which allows plotting more data points with better performance, rather than the regular Scalable Vector Graphics (SVG) option.

The performance of the WebGL option was further benchmarked with randomly generated mockup data to see how many data points it could maximally handle while retaining usable interactivity and without taking more than a handful of seconds to render. 3D scatter plots were also created with plotly.js to test the possibility of having Manhattan plots for multiple phenotypes in the same interactive plot. For the longitudinal phenotype summary statistics, an input form was implemented that enables the user to input data to overlay the MoBa data in the plot. Extra CSS styling was added to allow the highlighting of user input fields, used to indicate problems with input from the user (such as a negative age).

Three general methods were considered for the visualization of two longitudinal phenotypes simultaneously: (i) using parameterization, (ii) plotted per age, and (iii) plotted against all ages. With parameterization, the individuals studied can be divided into different groups, for example by genotype. At each age point, one or more statistics (e.g. median) are calculated for the groups for the two phenotypes and the resulting statistics are used as *x* and *y* coordinates on the plot (Figure 9).

For a single age, binning is used on one axis while percentiles are used on the other (Figure 10). Non-overlapping ranges are created for the binned phenotype, and for each bin, the median and lower and an upper percentiles are calculated for all the individuals in that bin. When showing all ages at once, without parameterization, a specific subgroup of individuals is followed at a time (Figure 11). The group is defined by its statistics for one phenotype (such as how their height compares to the rest of the population), and the development of another (such as weight) is followed longitudinally in the plot. Due to time constraints, no summary statistics were generated for any of the methods, nor were visualizations created in the web application.

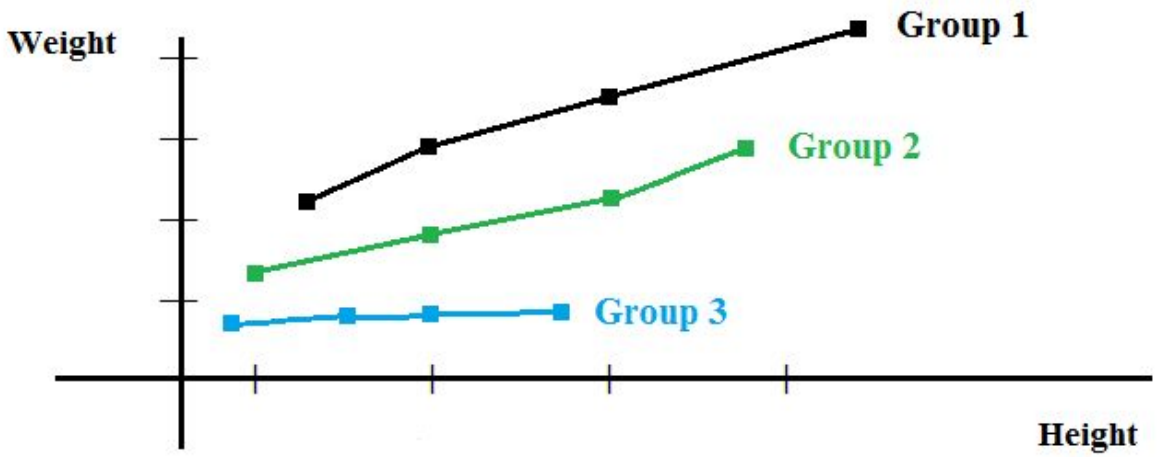

Figure 9: ***Visualizing two phenotypes simultaneously using parameterization.*** *The plot follows the longitudinal development of three groups of individuals, for example grouped by the three different genotypes of a SNP. For each line, four data points correspond to four different ages. The four data points could for instance have the group median for height and weight as x- and y-coordinates, respectively.*

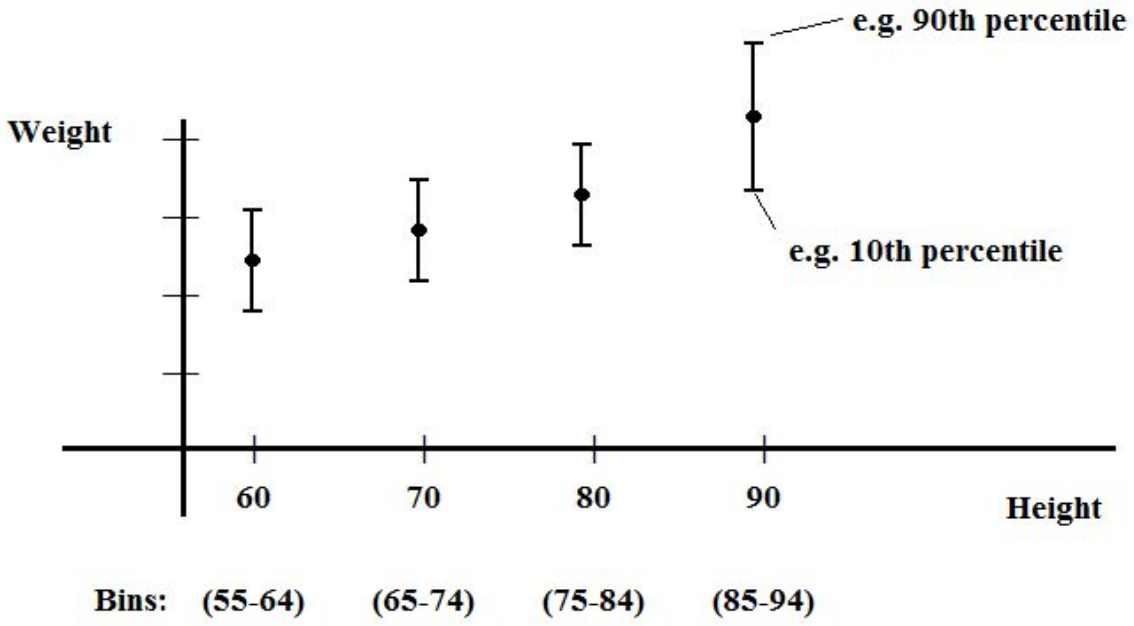

Figure 10: ***Visualizing two phenotypes simultaneously using binning and percentiles for a given age.*** *In this plot, the heights are binned while weights are are represented by percentiles. The vertical bars represent the range of the weight between the 10th and 90th percentiles at a given height, the central dot is the median, and the upper and lower horizontal bars represent the upper and lower percentiles considered, respectively. Only phenotypes at a given age are considered with this method.*

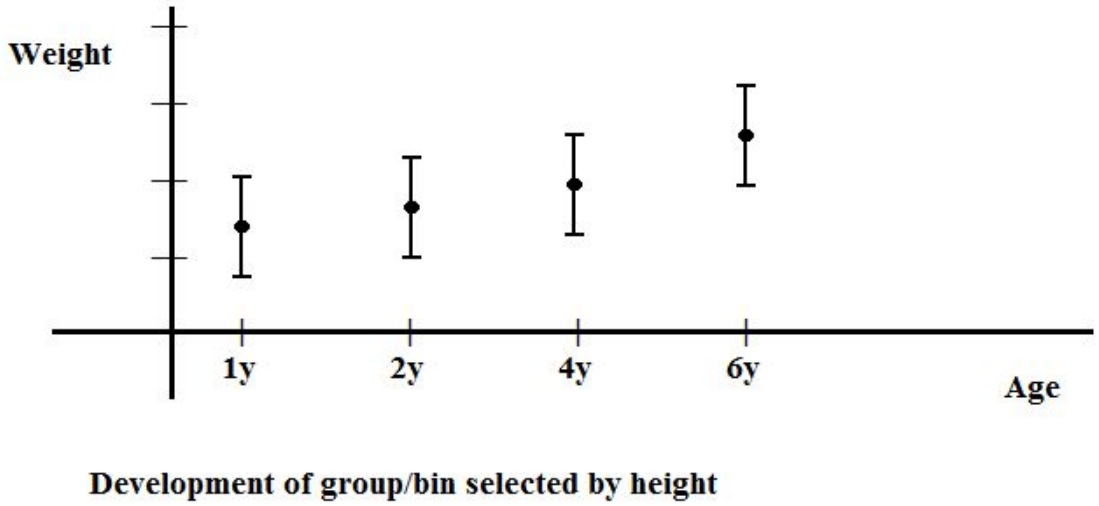

Figure 11: ***Visualizing the longitudinal development of two phenotypes simultaneously.*** *One group or bin at a time is followed by the plot. The group could for example be the shortest 10 percent of the children, which is a group whose members are likely to change from age to age. Alternatively, it could be the shortest 10 percent of the children at age two, a group whose membership is constant.*

3. Results

A prototype web application able to visualize data from MoBa without compromising participant privacy was developed, organized in two layers of tabs. All visualizations are interactive, allowing the user to zoom and pan. RocksDB and SQLite were compared for the purpose of storing large amounts of genotyping data from MoBa, with RocksDB being selected as the preferred option.

3.1. Database benchmarking

Within the application, it appeared that if an SQLite database file is not queried for a few hours or so, the next query can take 20 seconds or more. This was not observed with RocksDB, whose query times are a few seconds at most. When benchmarked outside of the application, four trends emerge:

1) For databases with less than 4-5000 entries (around 100 megabytes of data), the median RocksDB performance is often better (less than a fourth of that of SQLite for 1,000 entries) than SQLite; but beyond this, it could be much worse, with a median query time of more than 88 times that of SQLite for 10,000 entries (Figure 12). At the same time, the shortest query times for any

database size are delivered by RocksDB. In general, RocksDB tends to have much greater variation in the query times than SQLite.

2) The first query to an SQLite database is usually much slower than following queries; sometimes by a factor of several thousands. For RocksDB, no similar pattern was observed.
3) The creation of SQLite databases can be order of magnitudes slower than the creation of RocksDB databases (Figure 13). In the benchmarking session shown in Figure 12, SQLite is almost 250 times slower than RocksDB at 10,000 entries, taking more than 20 minutes to create compared to less than five seconds for RocksDB.
4) The sizes of the databases are similar, with SQLite database files being slightly larger in the size range studied, and the difference appears to be growing with larger numbers of entries (Figure 14).

A query time of tens of seconds is well beyond what can be considered acceptable for an interactive web application like this, even if it just for the first query to a database file in several hours. RocksDB was therefore chosen over SQLite as the preferred database software for the prototype web application.

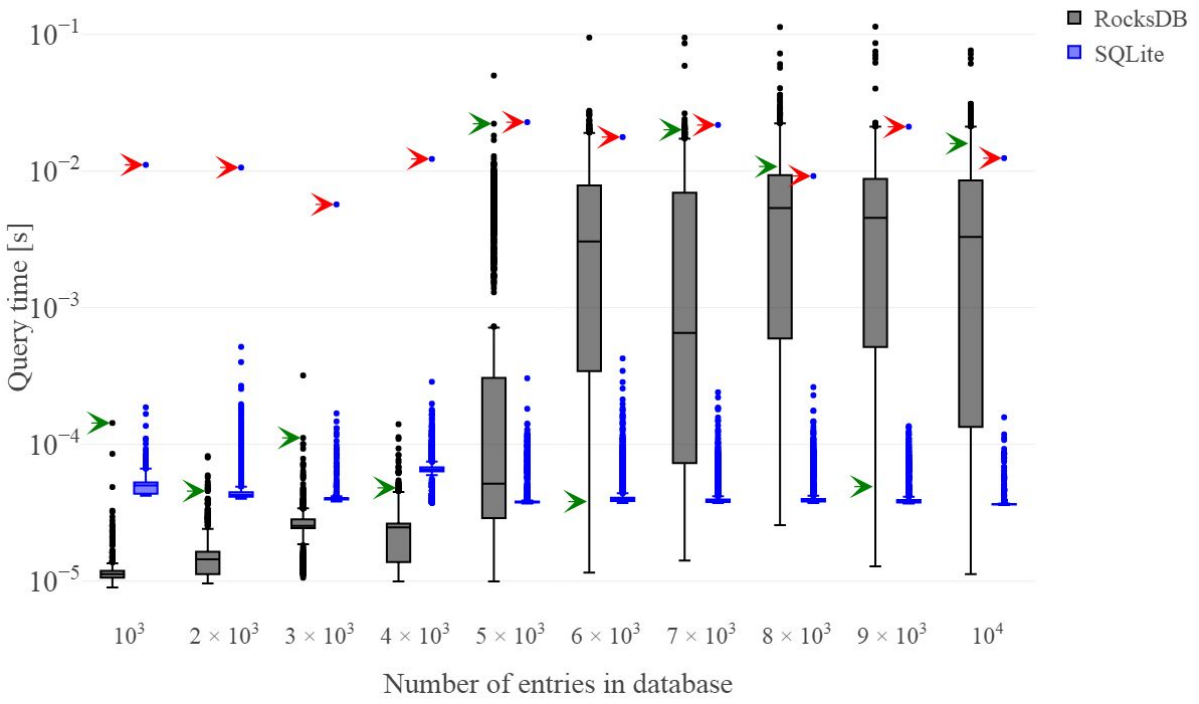

Figure 12: ***Box plot of query times for RocksDB and SQLite.*** *Query times measured in seconds are plotted against the number of entries in the queried database as box plots. For databases with less than 6,000 entries, all keys in the database were requested exactly once. For larger databases, 5,000 randomly selected keys were requested once. Results for SQLite are shown in blue and RocksDB in black. The first query to the database during the*

benchmarking session is indicated by arrowheads, with green heads for RocksDB and red heads for SQLite. A logarithmic scale is used on the y-axis. The horizontal bar in a box represents the median, the lower and upper limits of the box, the first and third quartiles, respectively. The horizontal bars at either end of the vertical bars extending from a box are whiskers. For all the boxes, the lower whisker corresponds to the minimum value and the upper whisker to the upper fence. The upper fence is located at a distance from the box equal to 1.5 times the length of the box. Outliers are drawn as individual points outside of the boxes.

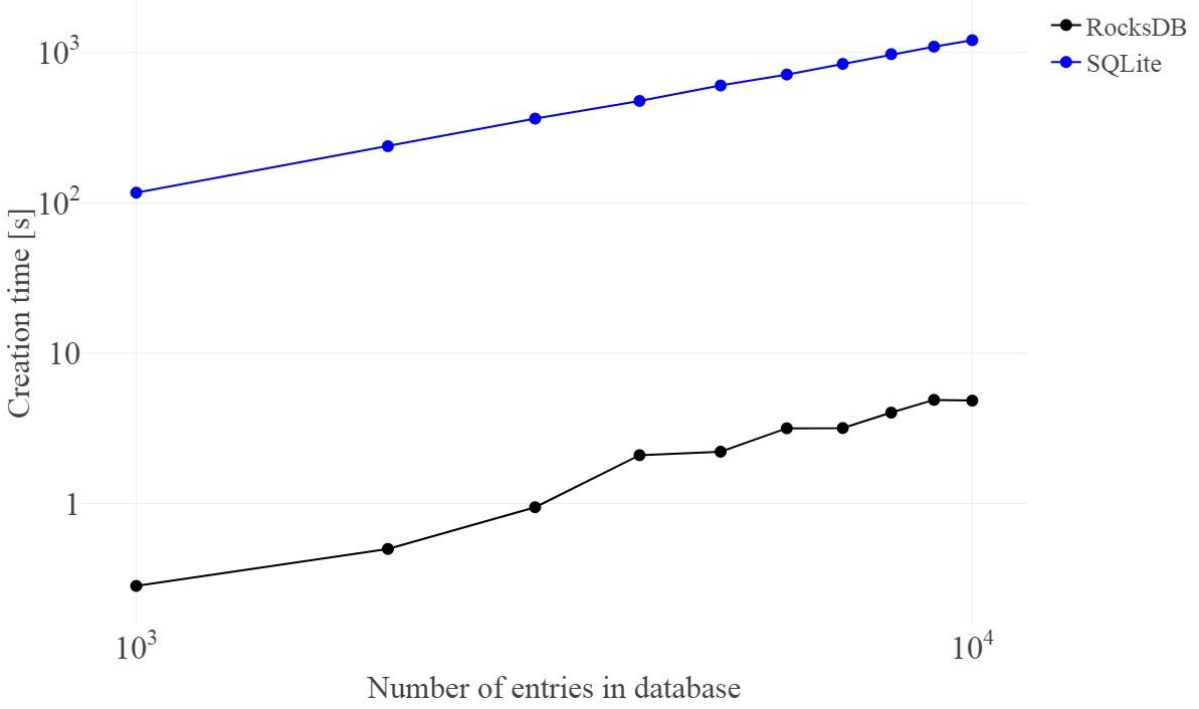

Figure 13: ***Database creation times for RocksDB and and SQLite****. Creation times from a single benchmarking session measured in seconds are plotted against the number of entries in the created databases. A logarithmic scale is used on both axes.*

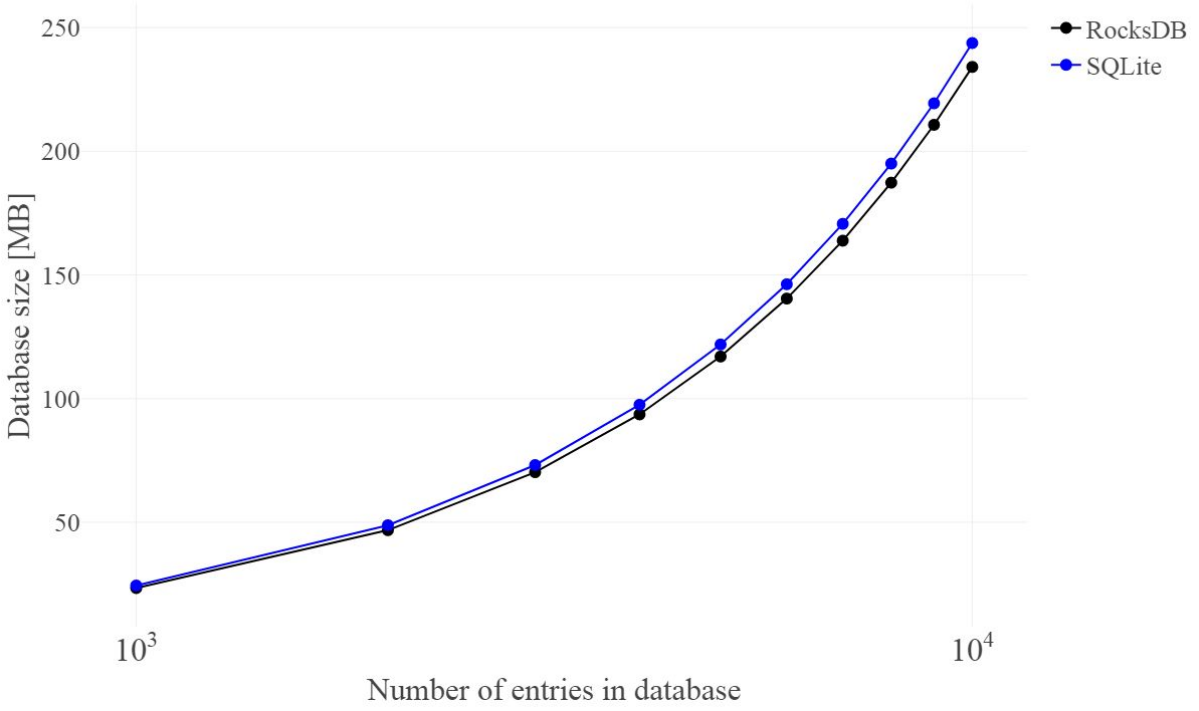

Figure 14: ***Database sizes for RocksDB and SQLite.*** *Database sizes in megabytes are plotted against the number of entries in the databases. A logarithmic scale is used on the x-axis.*

3.2. User interface

The first page displayed to the user in the web application, is the landing page with a password feature (Figure 15). If they user provides the correct password, the tab sheet-based user interface is made available (Figure 16 and 17). This interface has four tabs for visualization of data with genotype information: (i) a Manhattan plot, (ii) a regional plot, (iii), a plot for phenotypes stratified by genotype, and (iv) a visualization for information describing the SNPs available in the data (such as how many SNPs on each chromosome). For the phenotype summary statistics, three tabs are available: (i) for the children, (ii) for the mothers, and (iii) for the fathers. A closeable welcome message appears at the bottom of the web application when the user first enters the application. The message provides some background information on MoBa and indication to users about what data will most likely interest them. Note that the layout and style of the user interface was not the primary focus of this work, and remain to be optimized.

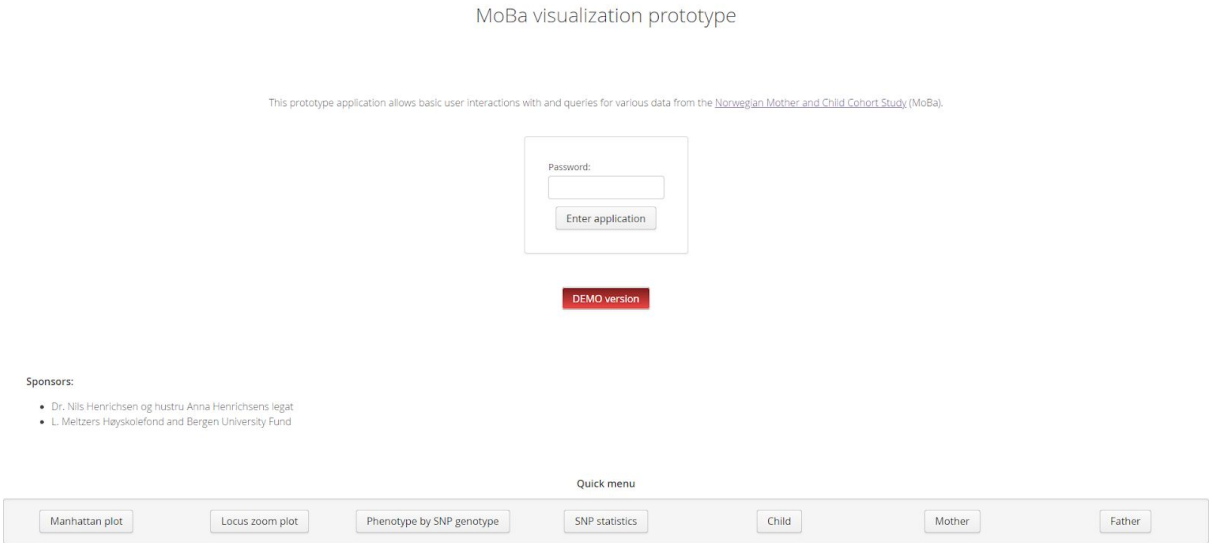

Figure 15: ***The landing page for the web application with a password feature.*** *The red button beneath the input field for the password redirects the web browser to the URL of the publicly available demo version. At the bottom of the page, a set of buttons take the user directly to a visualization, provided they have entered the correct password.*

Figure 16: ***The appearance of the application in the web browser.*** *The different elements of the web application visible here are: 1) the outer tabs, 2) the inner tabs for the selected outer tab, 3) a drop-down list for the selection of the sex to visualize data from (females currently selected), 4) a drop-down list for the phenotype to visualize (height currently selected), a drop-down list for the condition category (non selected) and a drop-down list for the condition, given a selected condition category (not active as no condition category selected), 5) a button that opens a pop-up letting the user enter data, 6) a button that opens*

a pop-up displaying the data underlying the plot, 7) a checkbox that specifies whether the MoBa summary statistics should be displayed in the plot, and 8) the plot visualizing the selected data. Elements 1-7 are implemented with Vaadin and element 8 with plotly.js. The plot shows the percentile distribution of height (cm) for females from birth to eight months of age.

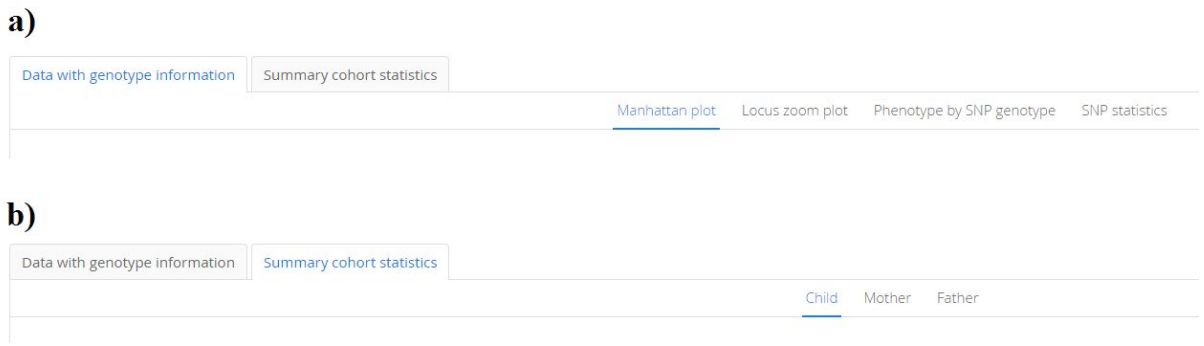

Figure 17: ***The tab layout of the user interface.*** *The tab for the genetic data is selected in a) and the tab for non-genetic data in b).*

3.3. Visualizations with genotype information

3.3.1. Manhattan plot

When the tab for the Manhattan plot (Figure 18) is first loaded, a button takes the place of the plot in the interface. The plot takes a few seconds to load, so through the button-dependent loading, the web application avoids unnecessary slowdowns if the user accidentally selects the tab for the Manhattan plot, is just exploring the web application, or is otherwise interested in looking at the interface surrounding the plot without necessarily being interested in looking at the plot itself. Only after this button is clicked, is the Manhattan plot rendered, which takes four to eight seconds. Once loaded, the interactivity of the Manhattan plot is comparable to less complex Plotly plots: when zooming or panning the plot, the response is practically instant, with the delay measured in a small fraction of a second.

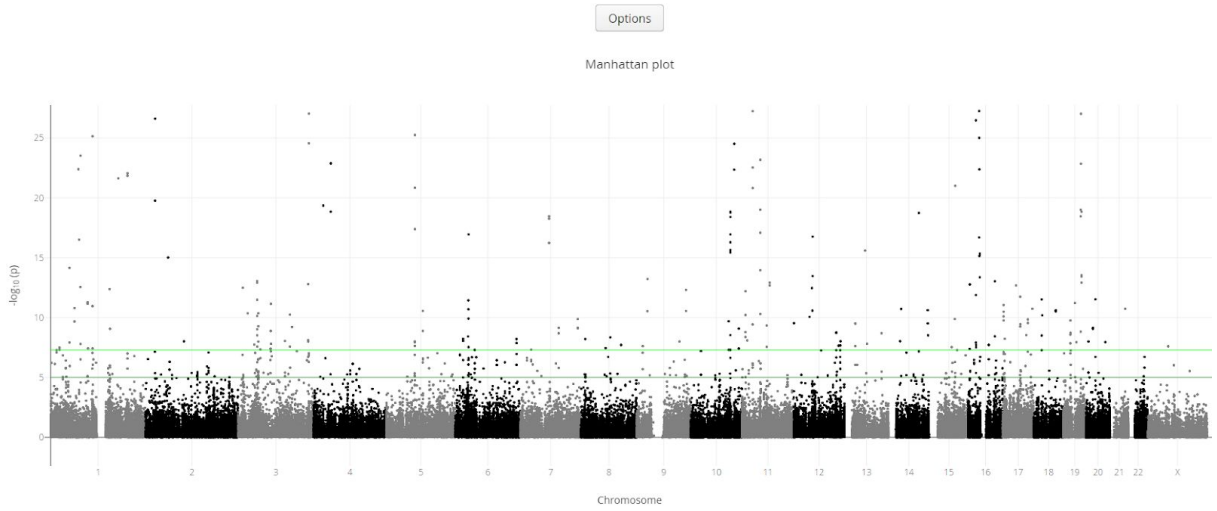

Figure 18: ***The Manhattan plot of the web application.*** *In this figure, the negative of the base-10 logarithm of the association p-values with BMI are plotted against chromosomal coordinates for variants on the 1-22 and X chromosomes. The plot is zoomed in, so SNPs with an association p-value less than* 10^{-29} *are not shown. Colours of the individual SNPs alternate between grey and black depending on which chromosome they reside on. Two significance thresholds are represented in the plot by green horizontal lines: the upper line is the genome-wide significance threshold (*$p = 5 \times 10^{-8}$*) and the lower line the suggestive significance threshold (*$p = 1 \times 10^{-5}$*).*

The implementation has the options to hide and show the genome-wide and the suggestive significance thresholds. When the user hovers the mouse pointer over a specific SNP, information on the SNP in question appears in a box on top of the plot. The user can click on any SNP on the plot and select it, which will result in a pop-up with information on the SNP selected as well as the option to navigate to the regional plot with the SNP selected, or to see how the phenotypes vary by the genotype for this SNP (Figure 19).

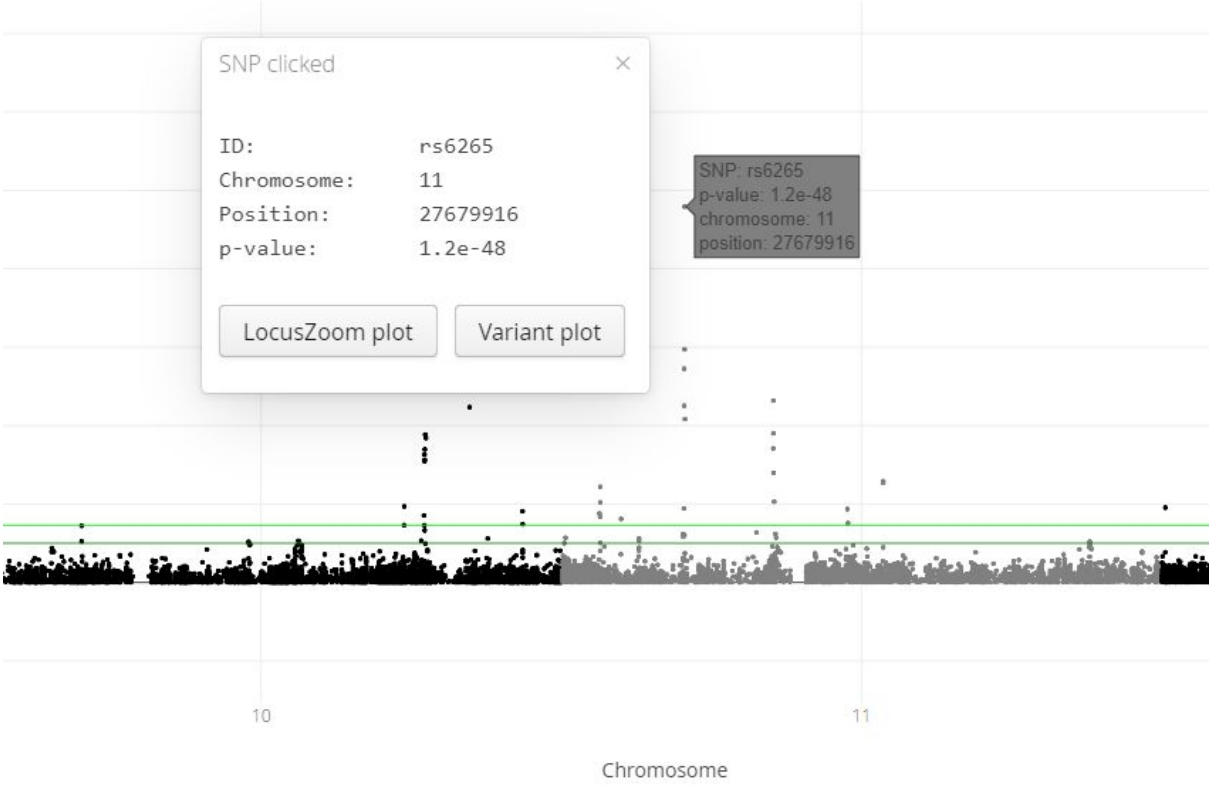

Figure 19: ***Interacting with the Manhattan plot.*** *The same data is plotted as in Figure 18. SNPs from chromosomes 10-12 are shown, as are the genome-wide and suggestive significance thresholds. Two boxes are overlaid on the plot; the left box is a pop-up in response to the user clicking on the SNP rs6265 (implemented with Vaadin), while the right is a popup (implemented with plotly.js) in response to the user hovering over the same SNP. Both boxes display the ID, chromosome, position and association p-value of the SNP. The left box also contains two buttons that will take the user to regional plot (left button) or the plots with phenotypes stratified by genotype (right button) with the clicked SNP as the selected SNP.*

During testing in the Google Chrome web browser, the plotly.js scatter plot could handle several million data points, although it would sometimes enter debugging mode due to the slow rendering times (often more than ten seconds) with this many points. The rendering times appeared to even out around four million data points (Figure 20), with the observed median decreasing from three to four million points, perhaps due to the large fraction of points overlapping in the plot, some quirk in the code for the plotting library, or both. The further increase in rendering time observed at five million points could be due to the increasing amount of data for the library to handle, regardless of the degree of point overlapping.

The zooming and panning functions, though useable, felt noticeably slower when a few million points were plotted. Rendering times in Firefox and Opera were similar to the rendering times in Chrome. The 3D scatter plot option in plotly.js is less capable

of handling large number of data points and starts to struggle with a few hundred thousand data points. Nonetheless, 3D plots with multiple Manhattan plots containing over 200,000 points (Figure 21) were successfully created, with rendering times around eight seconds, and retaining interactivity.

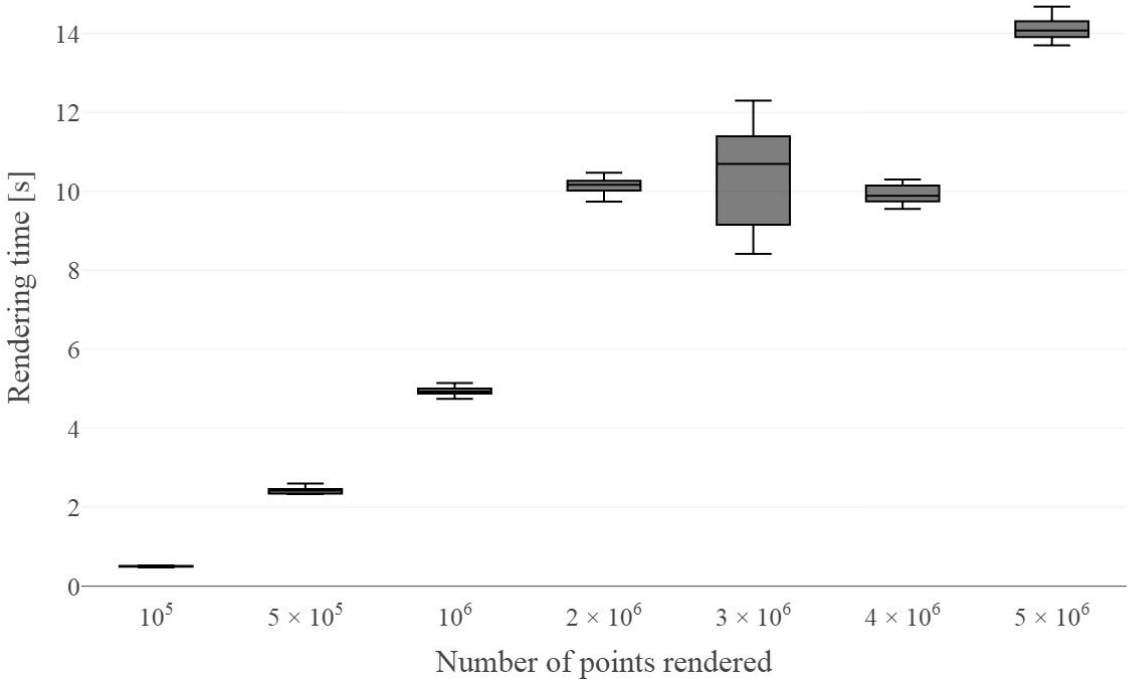

Figure 20: ***Box plot of rendering times of plotly.js Manhattan plots.*** *The time it took in seconds to render the Manhattan plot is plotted against the number of points rendered as box plots. For all boxes, the horizontal bar in the box is the median, and the upper and lower whiskers correspond to the maximum and minimum values, respectively. The tests were performed ten times for each number of points in the Google Chrome web browser.*

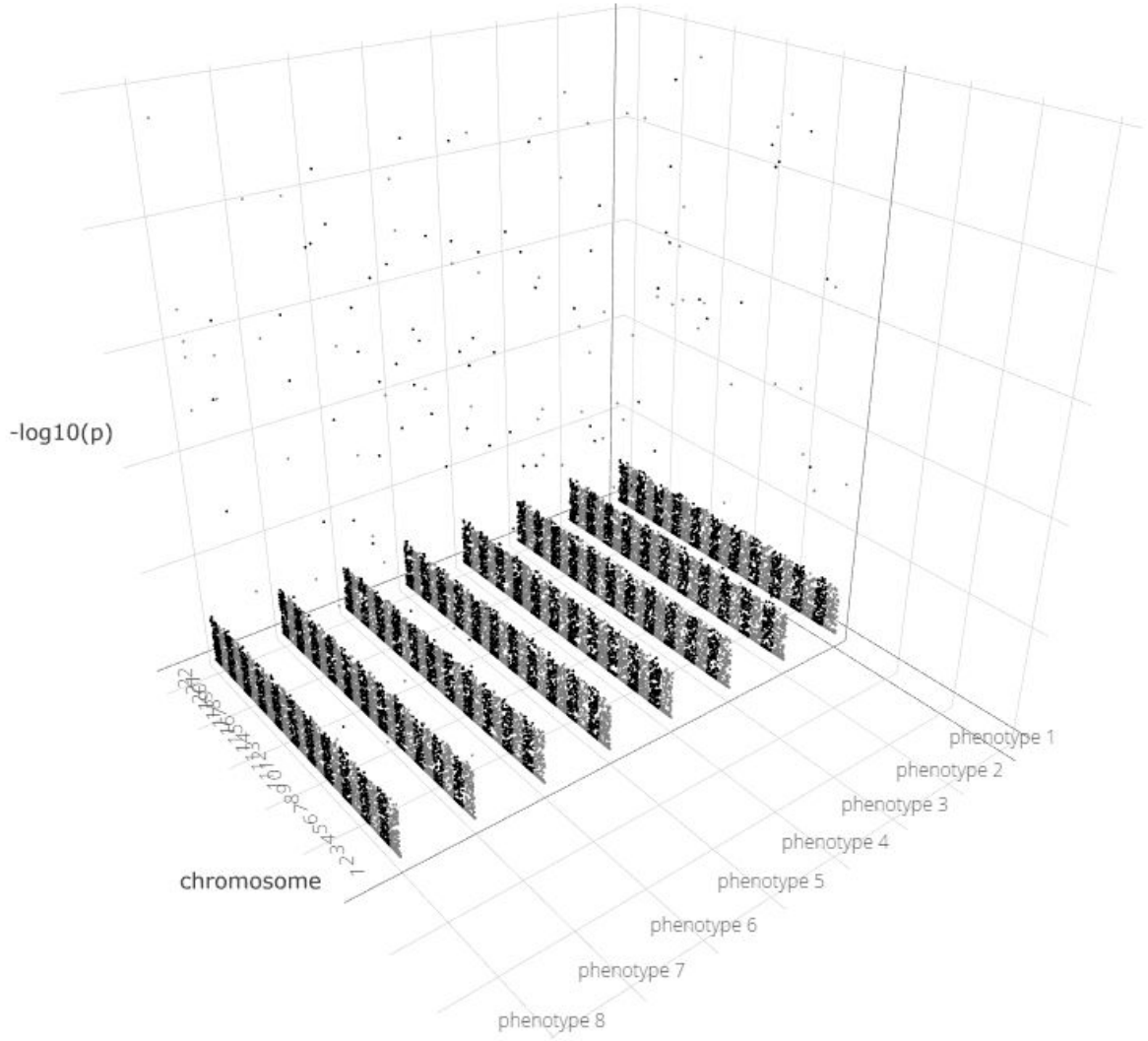

Figure 21: ***Manhattan plots for multiple phenotypes in a single interactive 3D scatter plot.*** *Randomly generated association p-values on the z-axis are plotted against chromosomal coordinates and a set of eight phenotypes. 200,112 points in total, or 25,014 points per phenotype, are plotted, distributed across 22 chromosomes.*

3.3.2. Regional plot

LocusZoom.js was integrated successfully with Vaadin in the web application (Figure 22), although some issues with the size of the plot were encountered. LocusZoom.js has built-in support for zooming and panning. A drop-down list for the chromosomes and a text field for the chromosome position were added to the user interface to provide additional means for the user to navigate the regional plot. A warning is displayed upon entering a negative chromosome position or a position that exceeds the length of the selected chromosome. The currently selected SNP is highlighted by a red cross, which is the only configuration altered from the default LocusZoom.js setup.

LocusZoom.js creates a plot consisting of two panels. The upper panel contains a scatter plot of the association *p*-values plotted against chromosomal coordinates, combined with a line chart for the recombination rate. In the lower panel, coordinates of known genes overlapping the region are shown. LocusZoom.js by default seems to set the SNP with the lowest *p*-value as the reference SNP for which the LD is calculated, but allows the user to set a different reference SNP by clicking on it in the scatter plot.

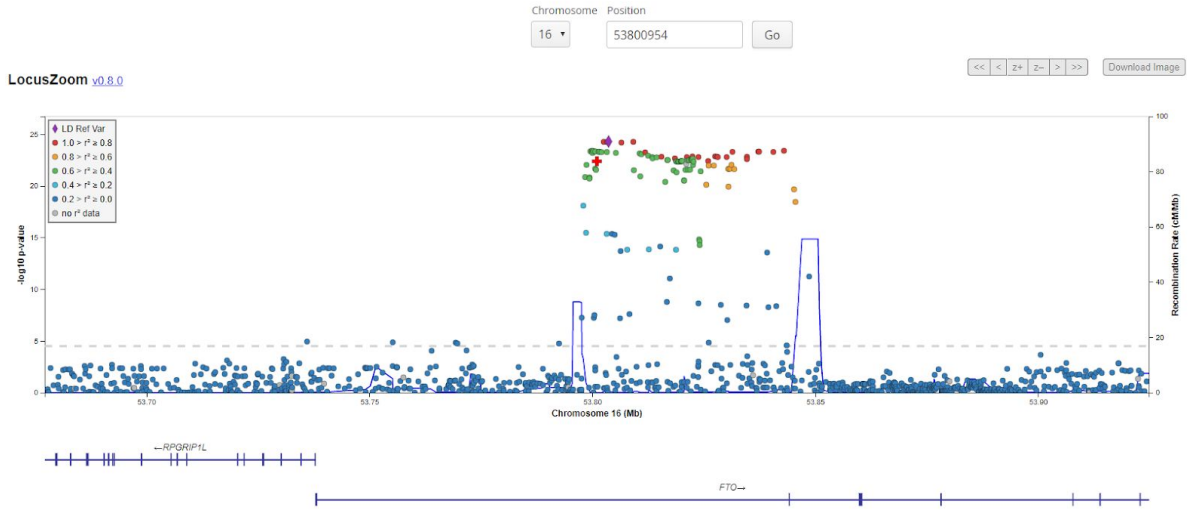

Figure 22: ***The regional plot of the web application.*** *Chromosomal coordinates are on the x-axis and transformed p-values on the left y-axis. The recombination rate (centimorgans per megabase) is shown as a blue line and uses the y-axis on the right. A red cross marks the SNP selected by the user. The colours of the individual SNPs correspond to the strength of the linkage disequilibrium calculated relative to the reference SNP (purple diamond). The area beneath the scatter plot shows the coordinates of known genes in this region on the chromosome, marked by blue horizontal lines with bars. In this case, the two genes RPGRIP1L (left) and FTO (right) are found in the region. Arrows next to the gene names indicate which of the two DNA strands the gene in question resides on, and in this case, the two genes reside on opposite strands as indicated by opposite arrows. The chromosome and position input fields above the scatter plot are Vaadin elements allowing the selection of a SNP and are not part of the LocusZoom.js library.r*

3.3.3. Phenotype stratified by genotype

In order to visualize a phenotype stratified by SNP genotype, the user can enter the SNP to study in an input field that also allows selection from a drop-down list. The input field accepts input in three formats: (i) *rs*ID number or the unique internal ID format, and (ii) the chromosome and position separated by a colon. Phenotypes can be selected from a drop-down list, and the data used in the plots is available in a pop-up window through a button.

This visualization instance is the only one that requests data from the database system of the web application, as the regional plot and Manhattan plot only use non-MoBa data. If a user selects a SNP in the regional plot, the chromosome and position is used to query the database system (input format (ii) above), while if it is selected in the Manhattan plot, the *rs*ID number is used (format (ii)). The query happens only when and if the phenotypes stratified by genotype tab is selected by the user. When the user selects a SNP in the web application, the controller object introduced previously is updated, such that other visualization instances can retrieve the identity of the currently selected SNP from the object.

The data retrieved for the requested SNP is visualized in two separate plots (Figure 23), one for females and one for males. The range of the *y*-axis is the same for the female and male plots by default, and the estimation of the mean is drawn as a dashed line while the estimation of the SEM and the 2.5^{th} and 97.5^{th} percentiles are drawn as ribbons. Below each phenotype plot, a bar chart shows the number of children with each genotype for which data is available at a given age (Figure 24). The age is selected by a slider beneath each bar chart, and dragging this slider also highlights the three medians on the graph corresponding to the selected age.

Information on the selected SNP is shown underneath the plot options and includes the ID of the SNP, the chromosome it resides on and at which position. From dbSNP the name of the associated locus (as defined by dbSNP) is shown in short form. The full name of the locus is shown when the user places the mouse pointer over the short form.

The settings for the plots are placed to the right and control which of the three statistics are shown and whether or not to show the bar charts for the number of children. A button placed above these settings opens a pop-up with more options, including whether *x*-axis labels should be evenly distributed or have a position on the *x*-axis correspond to the age.

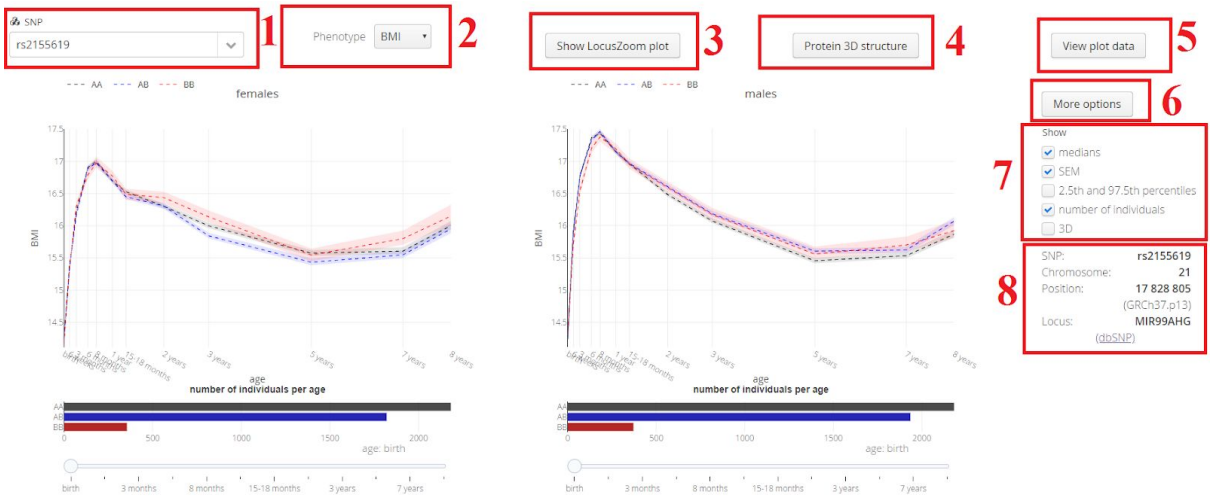

Figure 23: ***Stratification of phenotypes by SNP genotype.*** *In addition to the plots, shown here is: 1) the input field for the SNP (rs2155619 entered), 2) a drop-down list for the selection of the phenotype (BMI selected), 3) a button that launches the regional plot in a pop-up, 4) a button that launches LiteMol in a pop-up, 5) a button that opens a pop-up displaying the data underlying the plot, 6) a button opens a pop-up for additional plot options, 7) a group of checkboxes for adjusting the plot settings for both plots, and 8) information on the selected SNP. The left bar chart and line chart correspond to females, and the right bar chart and line chart to males. Both line charts show the BMI distribution for the sex in question from birth to age eight stratified by genotype, with the mean estimation represented by a dashed line and the SEM by a ribbon. The 2.5th and 97.5th percentiles are deselected in the plot options and not drawn. For all charts, AA is plotted in red, AB in grey and BB in red. Each horizontal bar chart beneath the line charts shows the number of individuals for which data exist at the selected age, which is at birth for both bar charts in this figure.*

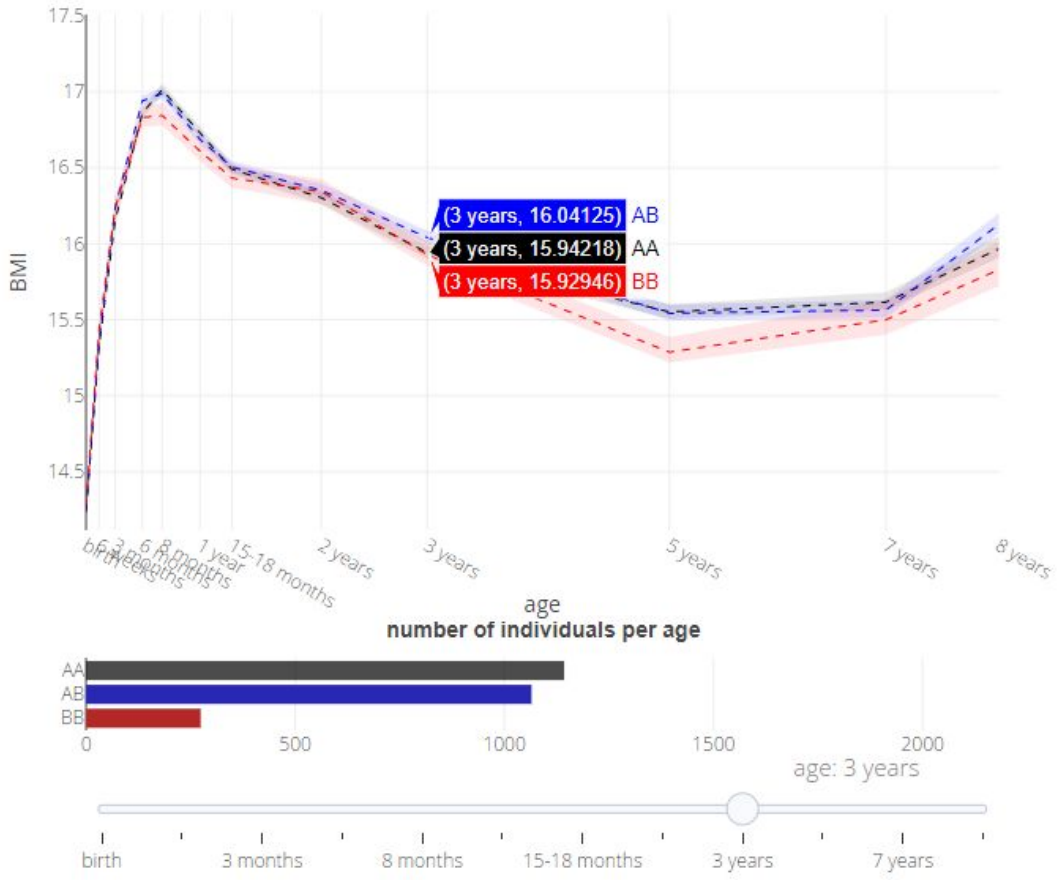

Figure 24: ***Dragging the slider for the bar chart visualizing the number of children.*** *In the figure, the slider is set to age three, and the median for the three genotypes at that age are displayed in separate boxes overlaid on the plot.*

The emulation of three physical dimensions that 3D graphics provide allow the coordinates of a plot to represent three variables instead of the two variables possible with a regular 2D visualization in the *xy*-plane. 3D rendering of the two phenotype plots were therefore implemented and are available through the plot settings (Figure 25). As with the 2D version, the phenotype is plotted against age using the *x*- and *y*-axes, while the number of individuals is now plotted on the *z*-axis, instead of the bar charts accompanying the 2D versions. The 3D plots are interactive just like the 2D plots, the user can manipulate the plots independently of each other by rotating them, zoom in or out, or pan.

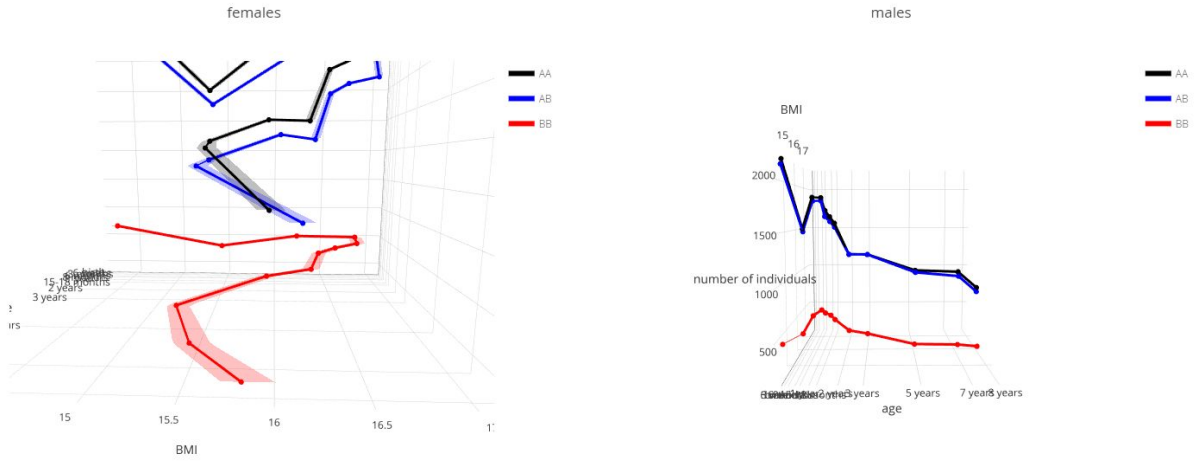

Figure 25: ***3D versions of the phenotype stratification plots.*** *The 3D versions are used with the same interface as the 2D versions, with separate plots for females (left) and males (right). The number of individuals is plotted on the z-axis and the phenotype (BMI in this case) against age on the two other axes, as with the 2D versions. The right plot is rotated relative to the left plot.*

LiteMol is accessible through a button above the phenotype stratification plots and opens up in a separate window in the user interface (Figure 26). For SNPs located in the codon of a gene, alternative alleles of the SNP can yield different amino acids in the protein encoded by the gene. LiteMol can highlight individual amino acids in the protein sequence in the 3D model, and thus the amino acid affected by the SNP selected by the user. This makes it possible to speculate on possible consequences of a non-synonymous variant from the 3D model. LiteMol does not automatically search the PDB database for relevant protein models, and in the current implementation of the web application, the user instead has to enter the name of the model they wish to render.

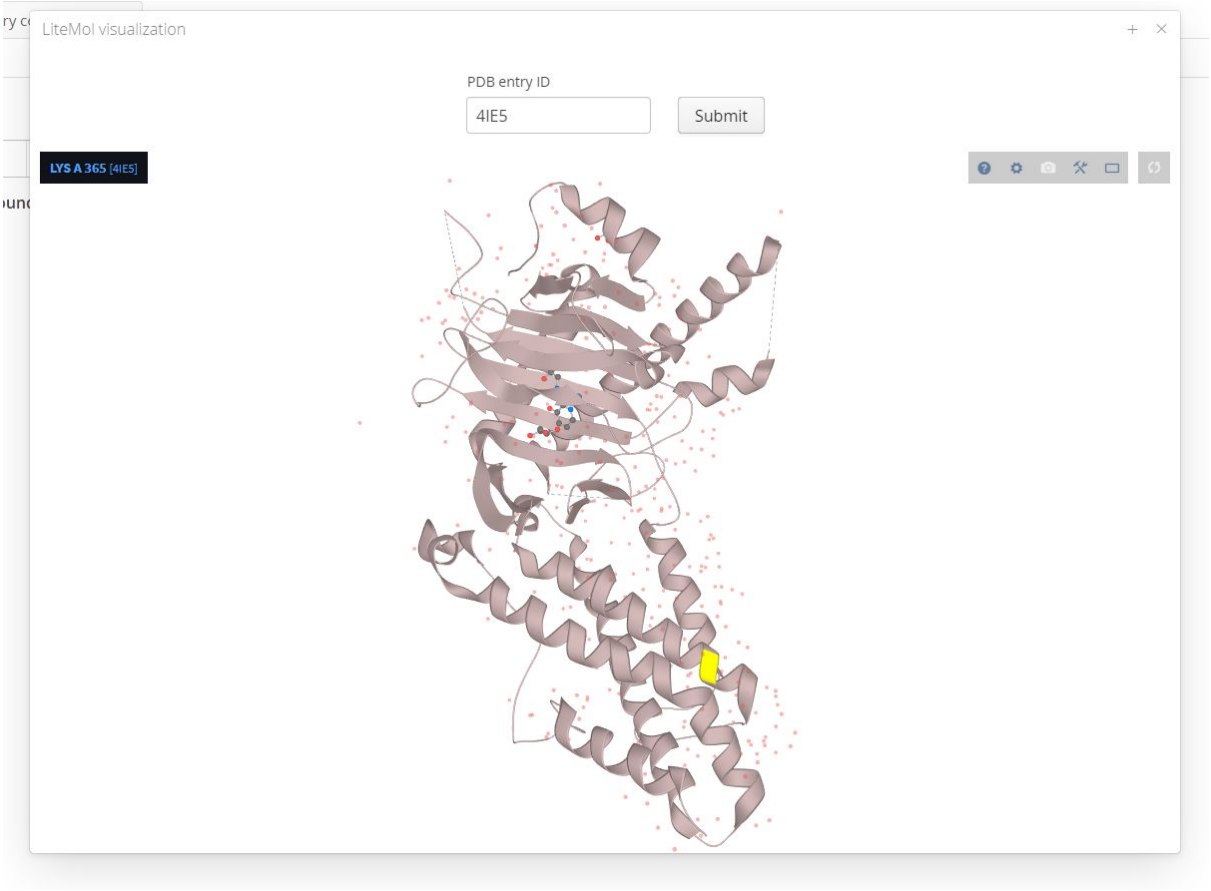

Figure 26: ***Visualizing protein 3D structure with LiteMol in the web application.*** *PDB model 4IE5 of the FTO protein is visualized by LiteMol in this figure. SNP rs758583500 is located in a codon of the gene encoding FTO and has an alternative allele that leads to the amino acid arginine appearing instead of lysine at position 365 (Lys 365) in the protein (see ncbi.nlm.nih.gov/projects/SNP/snp_ref.cgi?geneId=79068). The lysine is located in an alpha helix (here represented by a corkscrew shape), and is highlighted in yellow in the 3D model.*

3.3.4. SNP statistics

Statistics on the data with genotype information, generated separately for each chromosome, are displayed by bar charts in a separate tab (Figure 27). The statistics include the number of available SNPs per chromosome, how many are genotyped or imputed, the number of alleles that are the same or different from the reference genome, and whether imputed SNPs have an imputation score that meets the quality threshold. Except from the first, all the data can be visualized either by absolute numbers or fractions.

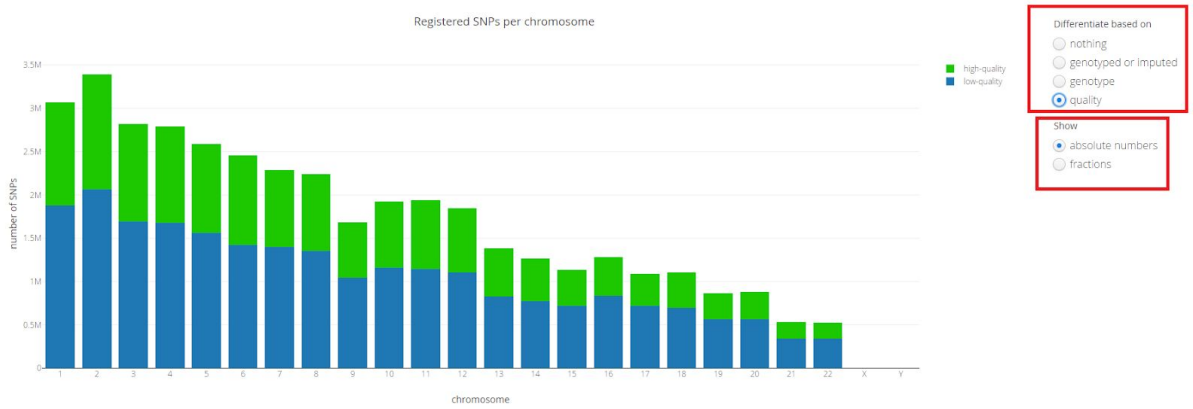

Figure 27: ***Visualization of statistics on SNPs available in the data.*** *This particular bar chart shows the number of imputed SNPs per chromosome of high quality. To the right of the bar chart, two groups of radio buttons (red rectangles) set the content of the plot. The upper group selects which statistic to plot and the lower group whether absolute numbers or fractions should be used.*

3.4. Visualizations for the phenotype summary statistics

3.4.1. Data on the children

The phenotype summary statistics are presented in a user interface similar to that used for the SNP level visualization. For children, only data for one sex at a time is plotted (Figure 28). The percentiles are drawn in shades of blue with darker shades representing percentiles closer to the median. Phenotype and sex are both selected from drop-down lists. Two other drop-down lists enable the specification of subgroups of the cohort. The first drop-down list specifies which variable the subgroup is based on and the second drop-down list the precise subgroup (Figure 29). As with the SNP level plots, the underlying data is available via a pop-up.

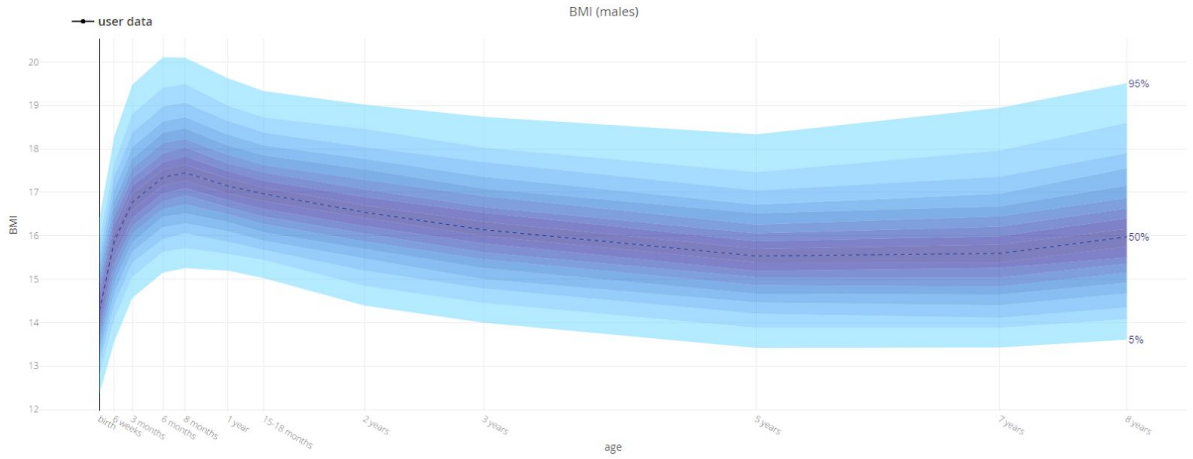

Figure 28: ***Plotting of phenotype summary statistics.*** *This plot shows the BMI (kg/m^2) distribution in terms of percentiles from birth to age eight of male children. The dashed line is the median, and percentiles closer to the median are represented by darker shades of blue.*

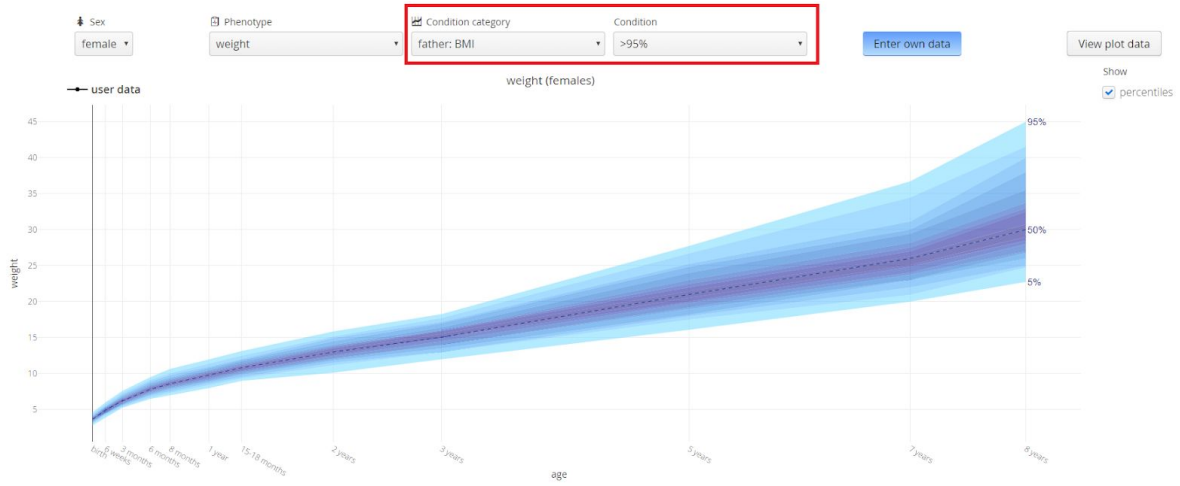

Figure 29: ***Plotting of summary statistics of cohort subgroups.*** *The user interface is the same as in Figure 16, only with a condition category and condition value selected (red rectangle). In this case, the selected condition category is the BMI of the father and the condition value ">95%". This selection corresponds to the girls whose fathers had a BMI above the 95th percentile, and their weight distribution in kilograms from birth to age eight is plotted.*

A feature that separates the summary cohort statistics visualization from SNP level visualization is the ability for the user to enter their own data (Figure 30). The user enters data in a pop-up accessible through the button with a light blue colour gradient. A black line overlaid on the summary cohort statistics represents the user input and is updated immediately in response to input from the user. The user input is checked for whether the input age is correct and supports input in the form of days, weeks, months and years, in addition to birth. An input field with invalid input that the user is interacting with is highlighted by a yellow border, while a red border highlights other input fields with invalid input. A short-lived pop-up appears if the user enters invalid input and informs them of why the input is invalid.

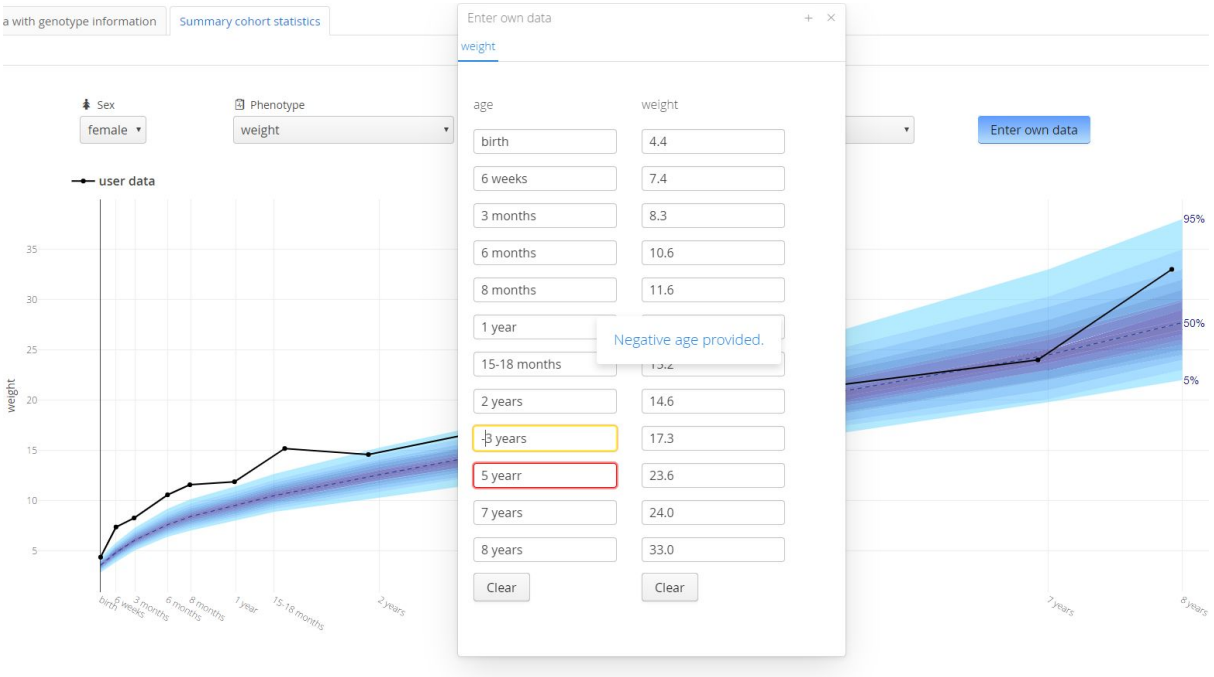

Figure 30: ***Plott cohort summary statistics with own data and warnings.*** *The user interface is the same as in the previous figures for the visualization of phenotype summary statistics. The pop-up over the plot is an input form for users. A message overlaid on the pop-up warns the user that a negative age has been entered in one of the input fields, namely the field highlighted in yellow, which is also where the mouse cursor of the user is. The input field below is highlighted in red and has an invalid age unit. The plot in the background displays a weight distribution in kilograms for females from birth to age eight, and the overlaid black line is the plotted user input.*

3.4.2. Data on the parents

Two different visualizations were implemented for the variables describing the parents. The first (Figure 31) visualizes the continuous variables in the form of a line chart, while the second tab visualizes the discrete variables in the form of bar charts (Figure 32). All other discrete variables (such as whether the birth was premature) are also visualized through such bar charts.

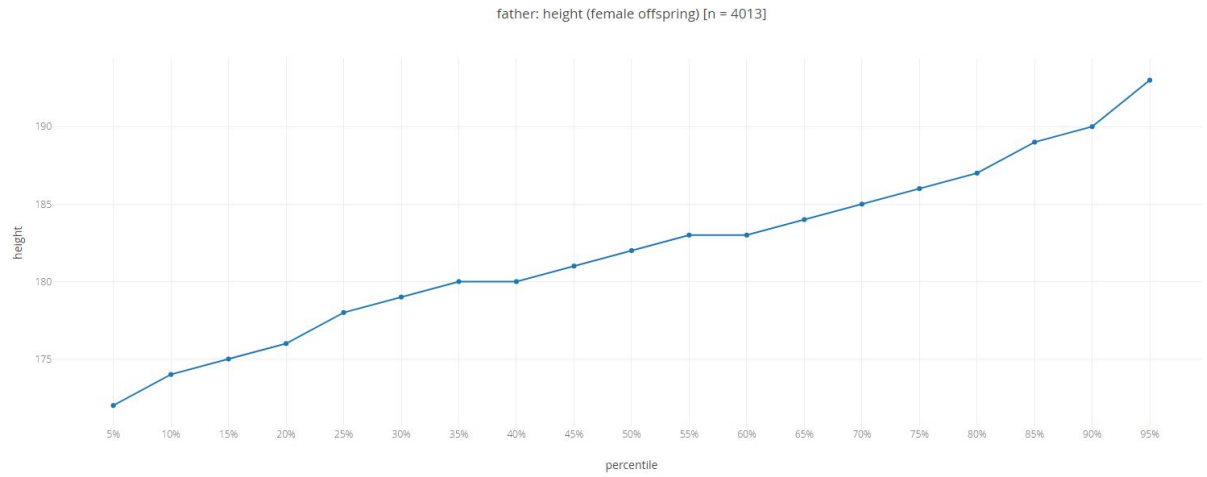

Figure 31: ***Visualization of continuous variables describing the parents.*** *This plot shows the percentile distribution of height in centimetres for fathers who had daughters. The number in the title indicates that the distribution is based on data from 4013 fathers.*

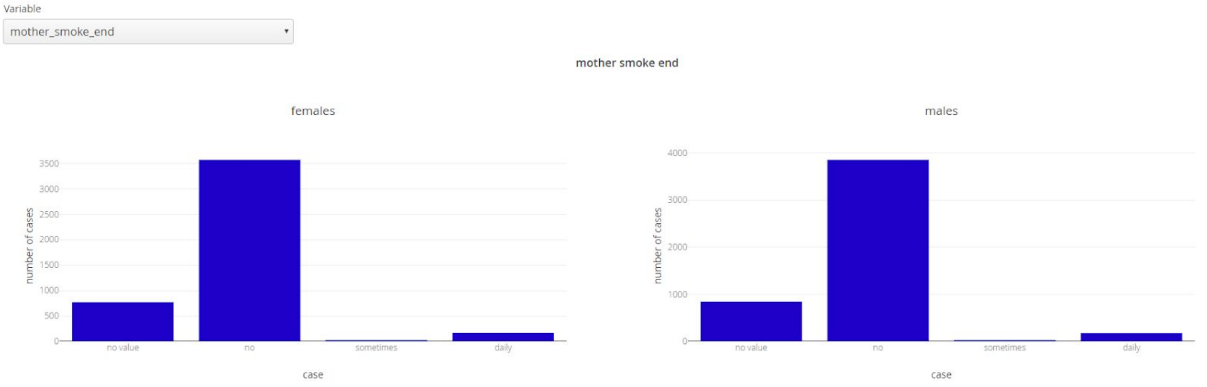

Figure 32: ***Visualization of discrete variables describing the parents.*** *This visualization shows the smoking habits of the mother at the end of the pregnancy. The left bar chart corresponds to mothers of boys and the right to mothers of girls. From left to right on each bar chart, the columns represent the following cases at the end of the pregnancy: data missing, mother did not smoke at all, mother smoked sometimes, and mother smoked daily. The drop-down list above the left bar chart specifies which variable to visualize.*

4. Discussion and future work

4.1. Running, expanding and adapting the web application

Since the web application and its source code is highly modular, adding new visualizations or additional features, or adapting it to different forms of data or different cohort studies is a relatively simple process. The current process of adding new or updating existing data to the web application server is also relatively

straightforward, particularly because the data is non-sensitive and do not require special solutions relating to data security.

There is still room to make the data deployment more efficient; the step of transferring newly generated summary statistics may be possible to automate, so that the statistics are automatically transferred to the web application server after generation. Potentially, users of the web application could also request the generation of specific summary statistics not currently stored on the server through the web application user interface. If user-requested statistics are stored on the server only for a limited amount of time, the user could be allowed to fine-tune the summary statistics they want to use without prohibitive data storage capacities being required on the server-side.

By using a commercial cloud hosting service such as AWS to host the web application and its data, hardware maintenance and operation is entirely outsourced, as are several aspects of software operation. One downside is that the Amazon Linux operating system used by default by LightSail does not offer any graphical user interface, so the server has to be operated solely through a command line interface. The server system requires little maintenance, though certain software updates have to be installed as the operator of the virtual server. Each Let's Encrypt certificate lasts for 90 days, so a new certificate has to be obtained regularly; but this process is automatable.

4.2. Web application interface and navigation

The current implementation of the user interface has the disadvantage that some parts of the application may be simultaneously interesting for very different user groups, making optimization of the user interface difficult. Implementing separate user interfaces has the disadvantage that more navigation is necessary before the user can browse the data in a user interface optimized for them, unless the group the user belongs to can be known or guessed in advance (e.g. by which precise URL the user followed to the application).

It would be desirable to look more closely at the option to develop two or three separate user interfaces for the three different user groups, given their large differences in interest and background knowledge; particularly scientists versus parents. A user interface optimized for the parents could, for example, give much less prominence to the visualization of the genetic data, given both the inability of most parents to understand the terminology and their probable lack of genotype information both for themselves, their children or other relatives.

A similar analysis also holds for many medical professionals. At the same time, future expansion of the use of genotyping in personalized and precision medicine [35,36] and an increase in the use of private genotyping services like 23andMe (23andme.com) could mean that both parents and medical professionals (and people more generally) could find the genetic data both more interesting and understandable.

During further development and refinement of the web application, medical professionals should be involved so that feedback can be received on the user-friendliness of the web application interface, the intuitiveness of the visualizations, and what data they want to have visualized. Ideally, parents without scientific or medical background would also be involved for the same reasons.

With the current implementation of the user interface, it is possible to refine navigation with simple steps, for example by introducing a navigational guide that in some detail explains to users where they can find different data, and that could have different versions tailored to different user groups. The guide could be clearly visible in the user interface also linked to from the welcome message, and be in the form of a pop-up or a separate menu.

Currently, the web application does not support the use of URLs to navigate or otherwise send information to the application. It is not possible to link to a specific visualization, or to set visualization or application settings through the URL, something that could be very useful for users. Back and forward navigation in the web browser is also not supported. It could therefore be worthwhile to look into ways of supporting some or all of these features relating to web application URLs.

4.3. Visualization strategies for the web application

While 3D visualization allows three variables of a dataset to be represented by three coordinates in the same plot, 3D visualizations on 2D displays (such as a regular computer monitor) may not be suitable for the visualization of many forms of data, as it can be difficult to read values off the plot. However, interactive 3D plots, like those provided by plotly.js, that let the user rotate the scene should be of lesser concern [37]. This means that the concerns are less relevant for the 3D versions of the phenotype stratified by genotype plots and the multiple Manhattan plots in 3D. For the latter, the alternation between grey and black makes it easier to read off which chromosome a SNP belongs to, and with the introduction of a colour scale where each colour corresponds to a single chromosome, this could be made even easier.

The methods to visualize multiple phenotypes simultaneously should be reviewed further and implemented in the web application, and summary statistics generated

from the raw data for this purpose. For the method considering a single age at a time, a slider can be implemented to let the user select the age. The slider would also make it possible for the user to follow the development longitudinally by dragging the slider. An option to make the plot automatically iterate over the ages, i.e. making it animated, could also be implemented.

For the two methods to visualize multiple phenotypes simultaneously relying on groups (parameterization and following a single group longitudinally), care needs to be taken when defining the groups. Certain definitions will yield groups with likely non-constant membership, for example definitions in terms of a phenotype like height, or a diagnosis. As an example, for the group of the 10 percent shortest children at the current age, some in the group may at a later age grow to become taller than children not in the group, and get replaced by them as members of the group. For a group of individuals with a certain diagnosis, reasons for non-constant group membership include additional individuals getting the diagnosis at a later age due to a later onset of symptoms or disease, and the changing of the diagnosis criteria.

The 3D visualization of protein structure should be properly integrated with the application: either the application should itself select the most relevant model to visualize for a given protein, or it should present a list of models the user can choose from. A basic QQ-plot should also be added to the application to allow the users to see quality control information for the genotype data. For the regional plot, it would be more useful if by default the SNP used as the reference for the calculation of the linkage disequilibrium is the SNP the user has selected through the user interface.

4.4. Computing power available to the application and scaling of datasets

The fact that the graphics used in the visualizations are all generated client-side in the user's web browser rather than server-side means that a lot of the computing is decentralized and does not place any computing load on the server. One downside to this approach is that different users are likely to use computer units of different computing power, so the limits of data volume and visualization complexity that the user's computer unit is capable of can vary a lot from user to user. Another potential downside is that the web browser may not be able to utilize as much of the available computing power on the user's computer unit as a dedicated standalone visualization application may be able to do.

For the first issue, an alternative is to generate all graphics server-side and stream them to the user [38]. For the second issue, a dedicated desktop application could be created to supplement the web application. Part of the motivation for creating a

web application is to make the data easily accessible without having to install or download any extra software. However, if the available computing power increases sufficiently with a desktop application to enable useful visualizations that are not possible in a web browser on most computer units, it could potentially be worthwhile.

The current implementation of the Manhattan plot does not scale well to the many millions of SNPs that can be commonly found in GWASs. However, by setting an upper limit for the number of SNPs included in the Manhattan plot and by only including the most statistically significant SNPs, the current implementation should be able to handle much larger datasets while providing a Manhattan plot that is still both interactive and displays the most interesting SNPs in the GWAS data. Without interactivity, most of the less significant SNPs would not be possible to discern on a Manhattan plot anyway, given its compressed *x*-axis relative to the amount of points plotted. It might still be worthwhile to benchmark other plotting libraries to see if they can handle large amounts plot points better; including non-interactive ones, which could potentially have interactive functionality overlaid. A similar analysis holds for the 3D scatter plot with multiple Manhattan plots.

4.5. Alternative implementations of the database system

Although the current implementation of the database system relying on RocksDB is relatively fast, it could be faster still; and it would be worthwhile to continue evaluating database software options. This may include SQLite, which showed some promising benchmarking results, although it would have to be used in a manner where the occasionally extremely slow initial query of a database is not an issue (such as through creating many smaller databases), or eliminated completely.

The current server implementation uses SSDs as storage media. Another method to speed up the database system is by loading the database files into the RAM, for example by creating a RAM drive. A major downside to this is that the RAM available on the hosting service is likely to be a bottleneck due to the large volumes of data generated from MoBa. Given the volatile nature of RAM, non-volatile storage media would also be needed on the server to store the data for whenever the instance is powered off, or the RAM cleared. However, if all the data is loaded into the RAM, and only as the server system is started up or rebooted, it should be adequate to use cheaper HDDs over more expensive SSDs for the non-volatile storage. Another solution is to differentiate the data based on how probable it is that it is requested by the user, and only load the data that is the most likely to be requested into the RAM.

If the database architecture is otherwise unchanged, it might be better to skip the master index for the database system and simply create index files that contain every SNP with an rs number in a certain range, where the range of each index file

would be of the same size. This would give a varying number of SNPs per index file and less precise control over the number of SNPs per index file, but neither point should be particularly problematic.

4.6. Potential of the web application

Given the current state of access to Norwegian health data in general, and not just to the MoBa data, the developed web application may help to alleviate this larger issue of data access both directly and indirectly. Directly, the web application can be adapted to other cohort studies (or other forms of health data), and a similar methodology can be used to generate summary statistics for the application to visualize. Indirectly, the web application may inspire others to develop new systems that provide public access to Norwegian health data. Examples of cohort studies the web application can be adapted to include the Hordaland Health Studies (HUSK, husk-en.w.uib.no) and the Nord-Trøndelag Health Study (HUNT, ntnu.edu/hunt).

For the indirect potential in particular, the exposure of the application is crucial. This application should have a certain potential, simultaneously interesting three different groups of people (parents, medical professionals and scientists) that at any given time represents a significant fraction of the Norwegian population. By providing high-quality data on the health of Norwegian children, a future refined version of the application could potentially become a part of the toolbox of many groups of Norwegian medical professionals. When they interact with the health services, parents could in that case learn about the application as a tool they can use to compare the development of their children with the general population. The resulting, significant, exposure of the application to the general public should help raise the expectations for access to Norwegian health data, providing an actual and useful example of a public web portal to Norwegian health data.

5. Conclusion

A prototype web application that can visualize data from the Norwegian Mother and Child Cohort Study without compromising participant privacy has been developed. It enables the users to visualize data in interactive plots on subgroups of the cohort, and data with genotype information data on three genomic scales: genome-wide level, regional level and SNP level. The developed framework is able to retrieve and visualize all requested data within a few seconds at most, and has a user interface organized in a way that makes navigation easy and intuitive. Finally, the web application prototype can readily be modified and implemented for other cohort studies, and may both directly and indirectly work to reduce the difficulty with accessing Norwegian health data for scientists, medical professionals and parents.

The documented source code of the web application is available at github.com/helse-data and a running demo is available at helse-data.no/demo.

6. References

1 Roy, S.M. *et al.* (2016) Infant BMI or Weight-for-Length and Obesity Risk in Early Childhood. *Pediatrics* **137** (5)

2 Wijmenga, C. and Zhernakova, A. (2018) The importance of cohort studies in the post-GWAS era. *Nat. Genet.* 50 (3), 322–328

3 Doll, R. (2004) Cohort studies: history of the method. In *A history of epidemiologic methods and concepts* (Morabia, A., ed), pp. 249–250, Birkhäuser Basel

4 Comstock, G.W. (2001) Cohort analysis: W.H. Frost's contributions to the epidemiology of tuberculosis and chronic disease. *Soz.-Präventivmed.* 46 (1), 7–12

5 Naess, O. and Schiøtz, A. (2008) Commentary: Kristian Feyer Andvord's studies on the epidemiology of tuberculosis and the origin of generation cohort analysis. *Int. J. Epidemiol.* 37 (5), 923–932

6 Campbell, A. and Rudan, I. (2011) Systematic review of birth cohort studies in Africa. *J. Glob. Health* 1 (1), 46–58

7 Townsend, M.L. *et al.* (2016) Longitudinal intergenerational birth cohort designs: A systematic review of australian and new zealand studies. *PLoS ONE* 11 (3), e0150491

8 Lucas, P.J. *et al.* (2013) How are European birth-cohort studies engaging and consulting with young cohort members? *BMC Med. Res. Methodol.* 13, 56

9 Woolcott, C.G. and Dodds, L. Birth Cohort Studies (2014). Oxford Bibliographies. doi:10.1093/OBO/9780199756797-0075

10 Magnus, P. *et al.* (2016) Cohort Profile Update: The Norwegian Mother and Child Cohort Study (MoBa). *Int. J. Epidemiol.* 45 (2), 382–388

11 Olsen, J. *et al.* (2001) The Danish National Birth Cohort - its background, structure and aim. *Scand. J. Public Health* 29 (4), 300–307

12 Qiu, X. *et al.* (2017) The Born in Guangzhou Cohort Study (BIGCS). *Eur. J. Epidemiol.* 32 (4), 337–346

13 Jaddoe, V.W.V. *et al.* (2010) The Generation R Study: design and cohort update 2010. *Eur. J. Epidemiol.* 25 (11), 823–841

14 Pearson, H. (2012) Children of the 90s: Coming of age. *Nature* 484 (7393), 155–158

15 Cyranoski, D. (2018) Gigantic study of Chinese babies yields slew of health data. *Nature* 559 (7712), 13–14

16 Nordhagen, R. and Lie, K.K. (2014) The Norwegian Mother and Child Cohort Study (MoBa) – its birth and early development. *Nor J Epidemiol* 24 (1-2)

17 Schreuder, P. and Alsaker, E. (2014) The Norwegian Mother and Child Cohort Study (MoBa) – MoBa recruitment and logistics. *Nor J Epidemiol* 24 (1-2)

18 Paltiel, L. *et al.* (2014) The biobank of the Norwegian Mother and Child Cohort Study – present status. *Nor J Epidemiol* 24 (1-2)

19 Bush, W.S. and Moore, J.H. (2012) Chapter 11: Genome-wide association studies. *PLoS Comput. Biol.* 8 (12), e1002822

20 Sherry, S.T. *et al.* (2001) dbSNP: the NCBI database of genetic variation. *Nucleic Acids Res.* 29 (1), 308–311

21 1000 Genomes Project Consortium *et al.* (2015) A global reference for human genetic variation. *Nature* 526 (7571), 68–74
22 International Human Genome Sequencing Consortium (2004) Finishing the euchromatic sequence of the human genome. *Nature* 431 (7011), 931–945
23 Mangin, B. *et al.* (2012) Novel measures of linkage disequilibrium that correct the bias due to population structure and relatedness. *Heredity* 108 (3), 285–291
24 Silva-Junior, O.B. and Grattapaglia, D. (2015) Genome-wide patterns of recombination, linkage disequilibrium and nucleotide diversity from pooled resequencing and single nucleotide polymorphism genotyping unlock the evolutionary history of Eucalyptus grandis. *New Phytol.* 208 (3), 830–845
25 Frommlet, F. *et al.* (2016) *Phenotypes and Genotypes*, 18, Springer London.
26 Visscher, P.M. *et al.* (2012) Five years of GWAS discovery. *Am. J. Hum. Genet.* 90 (1), 7–24
27 Elliott, L.T. *et al.* (2018) Genome-wide association studies of brain imaging phenotypes in UK Biobank. *Nature* 562 (7726), 210–216
28 Bycroft, C. *et al.* (2018) The UK Biobank resource with deep phenotyping and genomic data. *Nature* 562 (7726), 203–209
29 Chambers, J.M. *et al.* (1983) *Graphical Methods for Data Analysis (Wadsworth & Brooks/Cole Statistics/Probability Series)*, (1st edn) Duxbury Press.
30 McGill, R. *et al.* (1978) Variations of box plots. *The American Statistician* 32 (1), 12
31 Pruim, R.J. *et al.* (2010) LocusZoom: regional visualization of genome-wide association scan results. *Bioinformatics* 26 (18), 2336–2337
32 Sehnal, D. *et al.* (2017) LiteMol suite: interactive web-based visualization of large-scale macromolecular structure data. *Nat. Methods* 14 (12), 1121–1122
33 Morris, A.P. *et al.* (2012) Large-scale association analysis provides insights into the genetic architecture and pathophysiology of type 2 diabetes. *Nat. Genet.* 44 (9), 981–990
34 Speliotes, E.K. *et al.* (2010) Association analyses of 249,796 individuals reveal 18 new loci associated with body mass index. *Nat. Genet.* 42 (11), 937–948
35 Dugger, S.A. *et al.* (2018) Drug development in the era of precision medicine. *Nat. Rev. Drug Discov.* 17 (3), 183–196
36 Schork, N.J. (2015) Personalized medicine: Time for one-person trials. *Nature* 520 (7549), 609–611
37 Wilke, C.O. (2019) Don’t go 3D. In *Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures,* O’Reilly Media
38 Sawicki, B. and Chaber, B. (2013) Efficient visualization of 3D models by web browser. *Computing* 95 (S1), 661–673