

# Cancer-inspired Genomics Mapper Model for the Generation of Synthetic DNA Sequences with Desired Genomics Signatures

Teddy Lazebnik<sup>1\*</sup> and Liron Simon-Keren<sup>2</sup>

<sup>1</sup>Department of Cancer Biology, Cancer Institute, University College London, London, UK

<sup>2</sup>School of Mechanical Engineering, Tel Aviv University, Israel

\*Corresponding author: t.lazebnik@ucl.ac.uk

## Abstract

Genome data are crucial in modern medicine, offering significant potential for diagnosis and treatment. Thanks to technological advancements, many millions of healthy and diseased genomes have already been sequenced; however, obtaining the most suitable data for a specific study, and specifically for validation studies, remains challenging with respect to scale and access. Therefore, *in silico* genomics sequence generators have been proposed as a possible solution. However, the current generators produce inferior data using mostly shallow (stochastic) connections, detected with limited computational complexity in the training data. This means they do not take the appropriate biological relations and constraints, that originally caused the observed connections, into consideration. To address this issue, we propose cancer-inspired genomics mapper model (CGMM), that combines genetic algorithm (GA) and deep learning (DL) methods to tackle this challenge. CGMM mimics processes that generate genetic variations and mutations to transform readily available control genomes into genomes with the desired phenotypes. We demonstrate that CGMM can generate synthetic genomes of selected phenotypes such as ancestry and cancer that are indistinguishable from real genomes of such phenotypes, based on unsupervised clustering. Our results show that CGMM outperforms four current state-of-the-art genomics generators on two different tasks, suggesting that CGMM will be suitable for a wide range of purposes in genomic medicine, especially for much-needed validation studies.

**Keywords:** in silico sequence generation; biomarker detection; validation; bioinformatics deep learning model.

## 1 Introduction

Genetic data plays a central role in the new age of medicine, providing promising abilities in both diagnostics and treatment, thanks to advances in genetic sequencing technology [1–3]. However, current practices in medical genetics rarely allow robust genomics analysis, due to an insufficient amount of available data [4].

Multiple computational methods have been developed to analyze genomics datasets in order to explore germline and somatic genetic variants that are associated with phenotypic traits and diseases in a population [5–7]. While the scientific community, funding bodies, companies, and the public, recognize the importance of free and open access to genomics data for scientific research and medical progress [8, 9], in practice, there is insufficient or overly complicated sharing of genomics data and associated metadata [10]. Several projects have attempted to address this challenge. For example, the Personal Genome Project (PGP) [11] makes all its data available under open access. The Cancer Genome Atlas (TCGA) [12] and the International Cancer Genome Consortium (ICGC) [13] make different parts of their data available under two tiers, open and controlled access. Moreover, lack of data diversity (e.g. underrepresentation of rare diseases, of populations of non-European-descent and of ethnic minorities) hampers the development of many phenotype- and population-specific models required to make genomic medicine as egalitarian as possible.

To address these limitations, three main solutions have been developed. First, Data Simulators that simulate, *in silico*, genotype and phenotype data [14–17]. While these tools provide a solution for the lack of data problem, they do not adequately reflect the complexity of the data. Second, Data Generators generate the same type of

data using more advanced statistical methods to extract genomic signatures but largely result in only slightly altered versions of the input data [18–20]. Both methods lack mathematical formalization limiting the level of complexity they can handle. This is mainly due to the small parameter space considered and the limited size of the feasible operations these models utilize.

Another type of genomic generative method using machine learning (ML) approaches to extract complex patterns, which are later used to generate genome sequences that satisfy these patterns [21–23]. These methods can learn extremely complex patterns and signatures in the data, however, they are often inapplicable as they require a large input dataset to work properly (which is typically not available).

To tackle these issues, we developed CGMM, a novel mathematical framework for generating synthetic genome sequences, even from scarce input data. Unlike other models, that aim to capture signatures in a single set, our approach focuses on capturing the evolutionary and mechanistic dynamics of how mutations and variants modulate a control genome into a case genome. This way, our cancer-inspired genomics mapper model (CGMM), is able to exploit the availability of a large number of genomes as a control set and learn to map it to a small case set of genomes with the desired phenotype. The model is then able to apply the mapping it learned to convert independent control genomes into realistic case genomes.

Specifically, CGMM is built upon a genetic algorithm that optimizes the mapping of control genomics samples into case genomics samples. The genetic algorithm uses a fitness function that is inspired by the evolutionary mechanisms that drive cancer progression, including mutation, selection, and adaptation. The fitness function incorporates multiple features of the genomic data, including sequence alignment, quality scores, and structural variations, to accurately map the synthetic reads. Afterward, CGMM incorporates a machine learning component that uses a recurrent neural network to learn and predict the optimal mapping mutation steps. The neural network is trained on a large dataset of simulated mutation patterns to learn the relationship between the genomic features and the optimal mapping parameters.

We evaluated CGMM on two datasets, demonstrating its ability to reliably multiply the desired sample count, by generating synthetic genomes associated with the case set. We compare the resulting synthetic samples to those produced by four current state-of-the-art genome generative models, demonstrating that CGMM outperforms them.

The rest of the paper is organized as follows. In Section. 2, we provide an overview of the latest computational and mathematical models for *in silico* DNA generation, including the relevant information inspiring the construction of CGMM. Section. 3, formally introduces CGMM. In Section. 5, we describe two *in silico* experiments to evaluate the performance of CGMM compared to other genomics generator models. Finally, Section. 6 concludes the results and proposes possible future work.

## 2 Related Work

This section reviews the latest methods used to generate synthetic DNA sequences, as well as the natural evolution patterns of cancer cells and the advantages of GA in bioinformatics (which CGMM mimics and implements).

### 2.1 Generation of *in silico* DNA sequences

The generation of *in silico* DNA sequences based on diverse assumptions originated from previous findings of patterns in similar pathogens and samples. In particular, one defines a distribution of Single-nucleotide polymorphism (SNP) by processing available data or requesting the user to provide them, which are then randomly sampled to generate new sequences [24, 25]. Moreover, multiple simulators assume a linear and independent connection between the SNPs [16, 26]. For instance, [27] proposed multi-trait, multi-locus phenotype simulations in quantitative genetics, based on a linear mix of models in a mathematical framework that takes into consideration four components: genetic variant effects, infinitesimal genetic effects, non-genetic-covariate effects, and observational noise effects as properties of  $N$  samples and  $p$  traits. This approach allows the simulator to capture realistic covariate structures. However, the model is limited by its linear mathematical nature.

Such assumptions are known to be invalid in practice since biologically, the SNPs are not randomly sampled across the genome and unnecessarily have a linear and independent relationship between them [28–30]. For example, [31] show a non-linear accumulation of structural and numeric chromosomal aberrations as well as of gene mutation during cancer progression. Therefore, several models relax these assumptions to obtain a more accurate representation of the genomics dynamics [32–34].

[35] proposed an epistasis simulation pipeline that can generate dichotomous, categorical, and quantitative phenotypes which can simulate non-linear interaction in the distribution of SNPs while understanding observation bias. The simulation is constructed from four sequential computational processes: generating a genotype corpus, subsampling of SNPs sets, applying the epistasis model to the sampled SNPs sets, and subsampling the set of individuals. One limitation of the model is that it requires the user to declare the set of SNPs one wants to simulate specifically.

[36] presented a simulator that aims to balance the realism and computation speed for genotype data. In particular, the simulator is based on randomly sampling genetic markers as quantitative trait nucleotides (QTNs) from the whole genome or user-specified genomic area in either a normal or a geometry distribution. The proposed model allows dynamic changes in the QTNs due to the previous allocation of QTNs in different areas of the genome. This dynamic constraint allocation allows complex, non-linear dynamics but requires knowing them during the design phase rather than extracting them from the inputted data.

[37] developed a simulator that simulates GWAS phenotype data based on user-supplied genotype data for a population and considers several biological properties such as heritability, dominance, population stratification, and epistatic interactions between SNPs. The authors divided the interaction between the SNPs into a causal sampling of the one-dimensional distribution of SNPs, and the latter in the pipeline enriched these dynamics by computing the inter-SNPs interactions. Afterward, the simulator introduces phenotype habitability in the population and population stratification to obtain case and control phenotype genomes.

## 2.2 Natural mutation process in cancer cells

Cancer is a complex disease that arises from the accumulation of genetic and epigenetic alterations in cells that drive uncontrolled proliferation and invasion [38]. The evolution of cancer cells from healthy ones is a multi-step process that involves the acquisition of various mutations in genes that regulate critical cellular functions, including cell cycle progression, DNA damage repair, and cell death [39, 40]. Once a cancer cell acquires the initial mutations, it undergoes clonal expansion and further accumulates additional mutations, leading to the emergence of subclones with distinct phenotypes and genetic profiles [41]. Thus, from the cancer cells' population point of view, the amount of mutations is an ever-increasing quantity, if not treated. During the mutation process, the healthy cells that become cancer cells are bounded in the way they mutate, as unwanted mutations that are detectable by the immune system would be removed from the body [42, 43]. In a similar manner, cells that mutate at a high enough rate in a short period of time are also usually detected by the immune system and removed from the body [44]. In total, cancer cells increase their mutation rate during the process while being bounded in each mutation step. In our model, we utilize this evolutionary approach in the form of an accelerated-bounded mutation operator as part of a genetic algorithm scheme.

## 2.3 Genetic algorithms for bioinformatics

A genetic algorithm (GA) is an optimization method inspired by the evolution theory [45]. In particular, GA simulates the process of "evolving" through natural selection, where solutions (also referred to as "genes") that obtain a better score from the fitness function are considered more adapted. Therefore, these genes have a higher probability to pass their information to the next generation. This evolution jump between one generation to the other is performed by three stochastic processes: mutation [46], crossover [47], and feasibility test [48].

GAs have been used in multiple fields such as engineering [49], medicine [50], and economy [51]. [48] tackled the task of process planning optimization that is based on sequences of machines and the operations they perform. The authors used GA to obtain feasible processes for a start and later found the optimal process from the set of feasible processes. [47] explored and analyzed the usage of GA with various constraints in process route sequencing and astringency. The authors reconstructed the GA, including the establishment of the coding strategy, the evaluation operator, and the fitness function, showing that the new GAs can meet the requirement of sequencing work and can meet the requirement of astringency.

In the context of genomics data, several bioinformatic challenges have been tackled using GA. For instance, [52] show promising results studying different genetic algorithm operators for the assembly of Deoxyribonucleic acid (DNA) sequence fragments into a consensus sequence corresponding to the parent sequence, from a parent clone whose sequence is unknown. Additionally, [53] analyzed gene expression for several cancer types, including ovarian, prostate, and lung cancer. The authors proposed an integrated gene-search algorithm for gene expression data analysis, that is based on the GA approach with correlation-based heuristics for data

preprocessing. This algorithm obtained 89% accuracy in classifying 44 DNA samples into three types of cancer.

Overall, GA has demonstrated promising results in bioinformatics research [54–57], and especially in genomics-related tasks [58, 59]. Thus, GA poses a promising computational approach to the task of generating synthetic DNA sequences.

### 3 Model Definition

The usage of CGMM, like many other data-driven models, is divided into two phases: training and inference. During the training phase, CGMM solves several optimization tasks to obtain the weights of inner machine learning components that construct CGMM. Afterward, CGMM accepts new, previously unseen, samples as part of the inference phase and makes its prediction. A schematic view of CGMM is provided in Fig. 1, outlining the main computational steps in both the training and inference processes.

**During the training phase**, CGMM performs four main steps. The first step is acquiring the necessary inputs of a *control* set and a *case* set of Variant Call Format (VCF) file format. The control and case set samples are divided in some user-defined ratio into two, where one part is for the training phase, and the second part is left for the inference phase. In addition, the user may provide SNPs associated with the case group, to be used during the training of CGMM. Afterward, according to some pairing approach (II) chosen at the discretion of the user, CGMM generates pairs of *control* and *case* samples from the training dataset. Roughly divided, there are three possible pairing approaches one can take: 1) A *brute-force* approach where each sample in the *control* set is paired with all the samples in the *case* set. This approach results in the most amount of pairs and, therefore, with the largest training dataset. However, at some point, the marginal contribution of each new pair becomes insignificant, which leads to unnecessary computations. 2) A individual-centric approach, where pairs of the *control* and *case* samples are obtained from the same individual. This enables the preservation of personal traits in the resulting synthetic DNA sequence containing the *case* mutations. 3) A reduced *brute-force* approach, where one can use any subset of the pairs group generated by the *brute-force* approach. In particular, as the end goal is to learn pairing from the control to the case sets, one can take  $z \in \mathbb{N}$  random *case* samples, in a uniform manner, for each *control* case. While this approach introduces another level of randomness to the model, appropriate usage of it removes the disadvantages of the *brute-force* approach while preserving its advantages. Therefore, this approach is the default of CGMM.

Once pairs of control and case samples are obtained, CGMM implements the second training step, using the *Reversed bioprocess genetic algorithm* (RBGA) to generate a set of possible mutation processes from the *control* to the *case* samples. Formally, the RBGA is an instance of a GA algorithm with several modifications. Overall, GA follows the same computational framework with four main operators: stop condition, selection, crossover, and mutation. First, A GA would generate a random population of solution candidates. Then, the stop condition operator indicates the GA to stop once met. Otherwise, the GA would execute the other three operators in an iterative manner. The selection operator is the stage in which individual members are chosen from a population for the next iteration. This operator aims to stochastically use a fitness function to find and generate a new population of solutions that, on average, are better suited to a task in hand. The crossover operator is used to combine the information of two members to generate new members, mimicking the reproduction and biological crossover. Finally, the mutation operator introduces random changes to the members of the population. There are multiple implementations for each one of these operators that directly define the GA. For the case of RBGA, we set the fitness function to be the MASH metric [60] between the member of the population and the *case* sample. Following this decision, the stop condition operator is set to be  $\frac{1}{N} \sum_{i \in N} MASH(p_i, C) \leq \xi$ , where  $N \in \mathbb{N}$  is the size of the GA’s population which denoted by  $P$ ,  $p_i \in P$  is each member in the current GA’s population,  $C$  is the *case* sample, and  $\xi \in \mathbb{R}^+$  is a threshold value set by the user. The selection operator (also known as the *next-generation* operator) is implemented to be the “*tournament with royalty*” [47]. We choose this selection operator as multiple studies show it performs better than other operators and is able to converse in complex and multi-dimensional optimization tasks such as ours [61]. For the crossover operator, we choose at random (in a uniformly distributed manner) pairs of members from the GA’s population and a pivot point, generating two new members for the GA’s population. These members replace the original two by crossing the two parts of each from the original members, as defined by the random pivot point [46]. Lastly, we propose a novel mutation process inspired by the way cancer mutates. Similar to how cancer cells (representing *case* samples) evolve from healthy cells (representing *control* samples), CGMM removes, adds, or edits SNPs in the *control* DNA sequence, while maintaining similarity to its origin. To do so, we adopt the decreasing learning rate approach from the adaptive gradient descent algorithm [62], while also bounding the

total distance the changes can cause in a single mutation step. Precisely, let  $\mu \in \mathbb{R}^+$  and  $\sigma \in \mathbb{R}^+$  be the mean and standard deviation of a normal distribution stochastic variable  $\zeta \sim N(\mu, \sigma^2)$ . In addition, let us define a discount factor  $\lambda \in (0, 1)$ . Now, the accelerated-bounded mutation operator operates as follows. For the  $i_{th}$  generation of the GA, we choose a random number of mutations  $\zeta_i$  that distributed  $N(\mu \cdot \lambda^i, \sigma^2 \cdot \lambda^i)$ . Then, we apply  $\zeta_i$  changes to the member reflected by random SNPs. If the MASH metric distance between the member after the mutation and the sample before the mutation is larger than a pre-defined threshold  $\beta \in \mathbb{R}^+$ , the mutated member is considered infeasible, and the process repeats. Moreover, at this stage, if the user provided SNPs associated with the case group, the distribution of these SNPs in the case group is calculated and used as the distribution function for the mutation operator (replacing the uniform distribution function).

After a predefined number of pairs (or all of them, if not set otherwise) produce the mutation processes, CGMM initiates the third step of the training phase, in which a model of AutoEncoder (AE) with a self-attention layer is trained. Formally, the AE accepts a mutation process in the form of a list of numbers, which indicate the index in the genome array and one of the 12 possible SNP mutations. The AE is generated using the AutoKeras automatic deep learning library, to find the near-optimal AE architecture [63] for the specific inputs inserted. To do so, the AutoKeras framework aims to minimize the mean square error between the input and the reconstructed samples, using Bayesian optimization on the size of the encoding layer [64]. Once the AE model is obtained, each step in the mutation processes generated during the RBGA is converted into an encoded vector. This results in a set of vectors with the same dimension (i.e., the AE’s encoding layer’s dimension), describing a possible mutation process that links a specific pair of *control* and *case* samples.

In the fourth and final step of the training phase, CGMM uses a recurrent neural network (RNN) architecture, with a long-short term memory layer (LSTM), to predict the next step in a possible mutation process between pairs of *control* and *case* samples. This RNN component of CGMM is denoted as the Next Mutation Predictor (NMP). The NMP works as a time-series predictor, similar to the way Large Language Models learn to predict the next word in a sentence [65]. In particular, it is generated using the model architecture proposed by [66], as it showed promising accuracy on the next-word prediction task while keeping the model’s size relatively small, and therefore, less computationally expensive. Like the work of [66], the NMP receives a special token indicating the mutation process is over every time it reaches the end of an encoded representation vector. This indicates to the NMP that the specific mutation process has ended and prevents it from attempting to predict the next step in the process indefinitely.

Finally, the trained NMP and the AE models are saved for later usage in the inference phase.

**During the inference phase**, CGMM performs three main steps, this time on the input of  $\alpha \in \mathbb{N}$  *control* samples that were set aside for inference. In the first step, CGMM implements the Encoder part of the AE generated during training, to encode the samples into encoded vectors. In the second step, the encoded vectors are passed to the NMP model, where their respective mutation process into *case* samples are predicted. Finally, the resulting predictions are decoded in the Decoder part of the AE, converting them back into VCF format sequences. This results in a set of  $\alpha$  synthetic *case* samples.

## 4 Experimental design

We tested CGMM on three tasks, to demonstrate its capability to reliably increase the readily available DNA sequences of the *case* set. To do so, we collected genomics sequences from public repositories and divided them into the desired *case* set, which we needed to enrich with synthetic data, and the control set of DNA sequences, that we used for this task. Specifically, for the first two experiments (Exps. 1 and 2), we used 300 samples from the Personal Genomics Project (PGP), from three countries: the United Kingdom (120 samples), Canada (110 samples), and the United States (70 samples). The samples were manually divided into 200 European-dominant DNA sequences, acting as the control set, and 100 African-dominant DNA sequences, acting as the *case* set. In order to make sure the tagged ancestry is valid, we verified that the tags of all the samples correspond both to a PCA-based ancestry classification and to the ancestry self-reported by the examined individuals [67]. For the final experiment (Exp.3), we used 90 control samples from PGP corresponding to healthy individuals from the United Kingdom, and 30 *case* samples were obtained from Genomic Data Commons (GDC), corresponding to individuals with skin melanoma.

To demonstrate the balance between the conversion rate of control samples into *case* samples, and the absolute number of successfully converted control samples, we used a 50%-50% training and testing ratio for Exp. 2, compared to a 70%-30% ratio for Exps. 1 and 3. The resulting sample counts for each phase are presented in Table. 1.

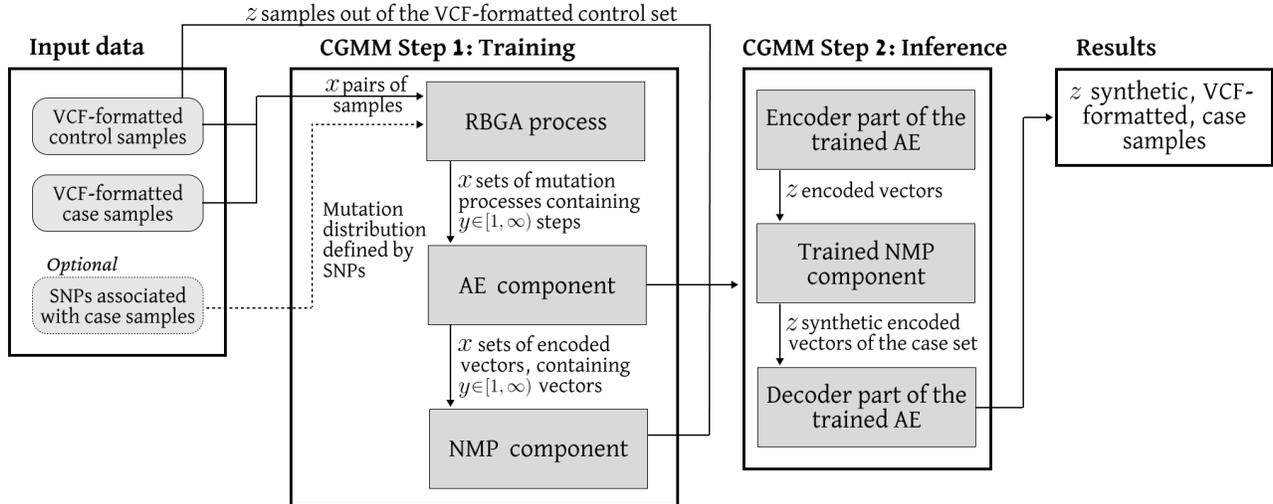


Figure 1: A schematic view of the model’s training and inference phases. First, the input data is divided, leaving a specified part of the *control* samples ( $z$  samples) for inference. The rest of the input data proceeds into the training phase, where: 1) The evolution processes linking each pair of *control* and *case* samples are estimated (RBGA process). 2) Each step in the process is encoded to a smaller, *latent space* (AE component). 3) A predictive model is trained to predict the next encoded step of an evolution process (NMP component). During the inference phase, the  $z$  *control* samples are used to generate encoded vectors of the first step in an evolution process (Encoder part). Then, the trained NMP component predicts a series of mutation steps, until reaching the desired genomic signatures. Finally, the last predicted step, representing an encoded, synthetic, *case* sample, is decoded into VCF format (Decoder part).

	<b>Exp.1</b>		<b>Exp.2</b>		<b>Exp.3</b>	
	<i>Control</i>	<i>Case</i>	<i>Control</i>	<i>Case</i>	<i>Control</i>	<i>Case</i>
Train	140	70	100	50	60	20
Test	60	30	100	50	30	10

Table 1: The sample count of *control* and *case* samples used during the training and inference phases. For inference, only the *control* samples are needed for CGMM’s operation, whilst the listed *case* samples are used to evaluate the reliability of the results.

Once the dataset was divided, the model was trained using the hyperparameters listed in Table. 3, provided in the Appendix.

To demonstrate that CGMM is capable of learning the correct relation and genomics signatures associated with each *case* set, we repeated each experiment twice: once without any knowledge integration, then with outside knowledge in the form of ancestry-related SNPs from ForenSeq™ DNA [68] (Exp.1-2), or in the form of skin-melanoma-related SNPs from [69].

To evaluate the performance of CGMM, we used hierarchical clustering following the CLINK algorithm [70], and computed the distances between samples using the MASH metric [60]. While there are many other metrics, such as the Jensen-Shannon divergence (JSD) [71] and the Kullback-Leibler (KL) divergence metrics [72], we decided to use MASH due to its popularity and well-established validation in bioinformatics. We then report the conversion rate, meaning the percentage of *control* samples that were classified as *case* samples out of the total *control* samples used during inference. The results of CGMM for each experiment are compared to those obtained by four other models, considered as state-of-the-art genomics generators [27, 36, 37, 73].

## 5 Results

Table. 2 presents the conversion rate of *control* samples into reliable *case* samples, as achieved by the five examined models. As expected, the results indicate that the more training samples obtained, the easier it

becomes for the models to detect the genomic signatures associated with each *case* set. Therefore, Exp.1 leads to the highest conversion rates for all models (neglecting cases where SNPs are included). However, once SNPs’ knowledge is integrated, the effect of the training sample count reduces significantly. Additionally, in all experiments CGMM is the least affected by the addition of SNPs knowledge, gaining only a  $15.6\% \pm 10.2\%$  increase in conversion rates for Exp.1-3, compared to a  $36\% \pm 2.8\%$  increase observed in the rest of the models.

Test configuration	CGMM	G2P [36]	Zhou et al. [73]	PhenotypeSimulator [27]	GEPSi [37]
$E_1$	78.3%	38.3%	31.6%	40.0%	56.6%
$E_1$ with SNPs	85.0%	71.6%	68.3%	83.3%	81.6%
$E_2$	73.3%	31.6%	26.6%	38.3%	41.6%
$E_2$ with SNPs	86.6%	68.3%	73.3%	81.6%	78.3%
$E_3$	46.6%	23.3%	16.6%	36.6%	30.0%
$E_3$ with SNPs	73.3%	53.3%	46.6%	66.6%	70.0%

Table 2: Comparison of the conversion rate of *control* samples into *case* samples, as achieved by CGMM and four state-of-the-art models.

Fig. 2 demonstrates the increase in the sample count of the resulting *case* dataset of DNA sequences in Exp.1-2. Although both experiments started with the same baseline count of samples (blue), and Exp.2 led to smaller conversion rates for all models (see Table. 2), models achieved a more optimal result in Exp.2 than in Exp.1. In other words, they all resulted in a larger sum of reliable synthetic DNA sequences (either with SNPs knowledge (yellow) or without (red)).

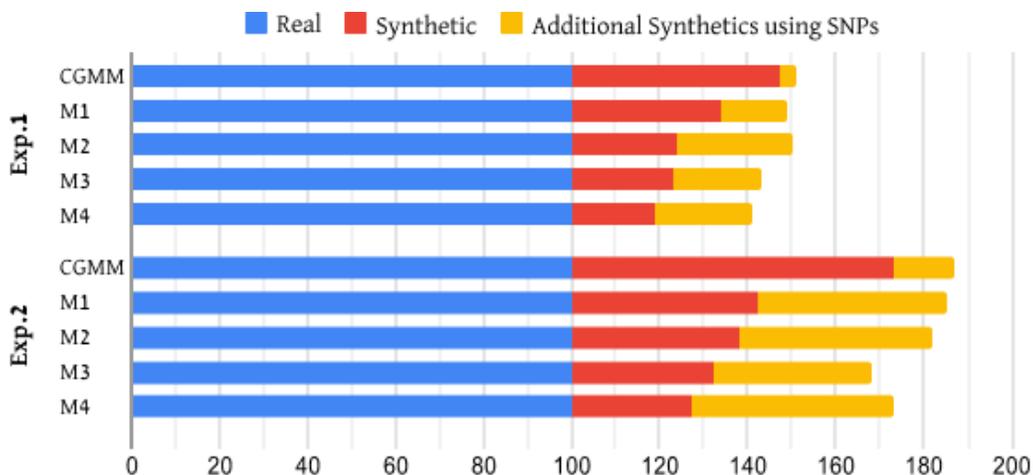


Figure 2: The resulting sample count of the *case* set DNA sequences, comprised of the original, real samples (blue), the synthetic samples produced without any SNP knowledge (red), and the additional synthetic sample count that was achieved with knowledge integration (yellow). The results of CGMM are compared to those achieved by GEPSi (M1), PhenotypeSimulator (M2), G2P (M3), and Zhou et al. (M4).

## 6 Discussion and conclusion

In this study, we proposed a novel cancer-inspired genomics mapper model (CGMM) that combines a genetic algorithm with a unique cancer-like mutation process, a deep learning AutoEncoder, and a next-word prediction model, to capture the non-linear dynamics linked to the mutation process between two genomics samples. Using our CGMM model, we were able to generate synthetic genomes while maintaining the personal traits of the input samples. This was done by learning the possible mutation evolution process between the pairs of samples and bounding each evolutionary step with logical assumptions that mimic the evolution of cancer cells. Thus, CGMM is able to receive an input genome and mutate it until some desired property is achieved. Therefore,

our computational approach differs from current synthetic genomics models, which try to capture the unique statistical properties on the genomics level, given a dataset of genomes with a desired property (such as ancestry or pathogen). Opposed to them, we aim to capture the statistical mutation dynamics between a dataset of *control* samples, that are more readily available, and a dataset of *case* samples, that possess the desired property we want. This way, CGMM is able to take advantage of genome samples that are found in abundance in order to reliably enrich a different, smaller dataset of genome samples, with synthetic samples.

In order to evaluate CGMM’s performance compared to other synthetic genome generation models [27,36,37,73], we conducted three experiments that aim to generate genomes with a desired property (Exp. 1-3). For that, we used two datasets, one of European and African-dominant ancestries, and the other of healthy individuals and individuals with skin melanoma. In all experiments, as shown in Table. 2, CGMM outperformed the other models, generating a higher conversion rate of *control* samples into reliable, synthetic *case* samples.

We repeated each experiment, integrating outside knowledge in the form of SNPs. The results show that the integration of knowledge further improves the performance of CGMM, however, at a much smaller rate than that of all other models (as seen in Table. 2 and Fig. 2). This leads to comparable results of CGMM and the rest of the state-of-art models if knowledge is integrated. Thus we argue that: 1) The results demonstrate that CGMM, opposed to the other models, is capable of learning the correct relation between the *control* and *case* set. Hence, the integration of knowledge has a relatively low effect on it. 2) CGMM seems to be able to capture SNP signatures partially. This is a promising outcome as with a proper adoption, CGMM can be used for SNP signature detection as well. 3) Without knowledge of SNP signatures, CGMM may achieve better results than the other models, even on small datasets. This is because the mutation path approach of CGMM can extend the training set to a much larger cohort, by pairing the *case* samples with multiple *control* samples. This, in turn, increases the number of mutation processes available for the training of CGMM, easing the deduction of synthetic *case* samples.

Additionally, Fig. 2 shows that one has to balance the need for a large training set (which aids in achieving a higher conversion rate), and the need to preserve samples for the generation of synthetic ones. Thus, we show that although a higher conversion rate is achieved in Exp.1, a larger, absolute increase in sample count is achieved in Exp.2 (i.e., CGMM generated in Exp.2 31 more reliable synthetic samples than in Exp.1 although it was trained on a smaller dataset of samples).

This study has several limitations that should be addressed in future work to improve further the performance and usability of CGMM. First, the AutoEncoder component in CGMM is designed to optimize an encoding task, as obtained by using the auto deep learning approach, but does not utilize any domain knowledge that can be helpful [74]. Second, the current method is not explainable, which may limit its usage in the clinical domain [75,76]. Third, due to the lack of freely available genomics data, the experiments were conducted on relatively small datasets. While these experiments represent important usage configurations, a test on a statistically larger dataset can allow obtaining a more robust evaluation of CGMM.

## Declarations

## Funding

This research received no specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

## Conflicts of interest/Competing interests

The authors have no financial or proprietary interests in any material discussed in this article.

## Data availability

The data used is publicly available, and the sources are cited in the text.

## Author Contributions

Teddy Lazebnik: Conceptualization, data curation, formal analysis, investigation, methodology, software, visualization, supervision, writing - original draft, and writing - review & editing.

Liron Simon-Keren: Formal analysis, visualization, and writing - original draft.

## Acknowledgements

The authors wish to thank Olga Chervova, Vitaly Voloshin, Ismail Moghul, and Stephan Beck for their help in this project.

## References

- [1] C. M. Fraser. A genome to celebrate. *Science*, 371(6529):545, 2021.
- [2] A. J. Gates, D. M. Gysi, M. Kellis, and A-L. Barabasi. A wealth of discovery built on the human genome project — by the numbers. *Nature*, 590:212–215, 2021.
- [3] F. S. Alkuraya. How the human genome transformed study of rare diseases. *Am J Hum Genet*, 97(2):199–215, 2015.
- [4] J. Berg, M. Houry, and J. Evans. Deploying whole genome sequencing in clinical practice and public health: Meeting the challenge one bin at a time. *Genet Med*, 13:499–504, 2011.
- [5] S. L. Poon, M. N. Huang, Y. Choo, J. R. McPherson, W. Yu, H. L. Heng, A. Gan, S. S. Myint, E. Y. Siew, L. D. Ler, L. G. Ng, W-H. Weng, C-K. Chuang, J. S. P. Yuen, S-T. Pang, P. Tan, B. T. Teh, and S. G. Rozen. Mutation signatures implicate aristolochic acid in bladder cancer development. *Genome Medicine*, 7:38, 2015.
- [6] L. B. Alexandrov and M. R. Stratton. Mutational signatures: the patterns of somatic mutations hidden in cancer genomes. *Current Opinion in Genetics & Development*, 24:52–60, 2014.
- [7] S. Nik-Zainal, L. B. Alexandrov, D. C. Wedge, P. Van Loo, C. D. Greenman, K. Raine, D. Jones, J. Hinton, J. Marshall, L. A. Stebbings, A. Menzies, S. Martin, K. Leung, L. Chen, C. Leroy, M. Ramakrishna, R. Rance, K. W. Lau, L. J. Mudie, I. Varela, D. J. McBride, G. R. Bignell, S. L. Cooke, A. Shlien, J. Gamble, I. Whitmore, M. Maddison, P. S. Tarpey, H. R. Davies, E. Papaemmanuil, P. J. Stephens, S. McLaren, A. P. Butler, J. W. Teague, G. Jonsson, J. E. Garber, D. Silver, P. Miron, A. Fatima, S. Boyault, A. Langerod, A. Tutt, J. W. M. Martens, S. A. J. R. Aparicio, A. Borg, A. V. Salomon, G. Thomas, A-L. Borresen-Dale, A. L. Richardson, M. S. Neuberger, P. A. Futreal, P. J. Campbell, and M. R. Stratton. Mutational processes molding the genomes of 21 breast cancers. *Cell*, 149(5):979–993, 2012.
- [8] E. Birney, T. J. Hudson, E. D. Green, C. Gunter, and et al. Prepublication data sharing. *Nature*, 461:168–170, 2009.
- [9] M. Walport and P. Brest. Sharing research data to improve public health. *Lancet*, 377:537–539, 2011.
- [10] K. Powell. The broken promise that undermines human genome research. *Nature*, 590:198–201, 2021.
- [11] O. Chervova, L. Conde, J. A. Guerra-Assuncao, and et al. The personal genome project-uk, an open access resource of human multi-omics data. *Scientific Data*, 6:257, 2019.
- [12] J-S. Lee. Exploring cancer genomic data from the cancer genome atlas project. *BMB Reports Online*, 49(11):607–611, 2016.
- [13] J. Zhang, J. Baran, A. Cros, J. M. Guberman, S. Haider, J. Hsu, Y. Liang, E. Rivkin, J. Wang, B. Whitty, M. Wong-Erasmus, L. Yao, and A. Kasprzyk. International Cancer Genome Consortium Data Portal—a one-stop shop for cancer genomics data. *Database*, 2011, 2011.
- [14] D. Marbach, T. Schaffter, C. Mattiussi, and D. Floreano. Generating realistic in silico gene networks for performance assessment of reverse engineering methods. *Journal of Computational Biology*, 16(2):229–239, 2009.

- [15] T. Mailund, M. H. Schierup, C. N. Pedersen, P. J. M. Mechlenborg, J. N. Madsen, and L. Schausser. CoaSim: A flexible environment for simulating genetic data under coalescent models. *BMC Bioinformatics*, 6:252, 2005.
- [16] P. F. O’Reilly, L. J. M. Coin, and C. J. Hoggart. invertFREGENE: software for simulating inversions in population genetic data. *Bioinformatics*, 26(6):838–840, 2010.
- [17] A. Pinna, N. Soranzo, I. Hoeschele, and A. de la Fuente. Simulating systems genetics data with SysGenSIM. *Bioinformatics*, 27(17):2459–2462, 2011.
- [18] P-R. Loh, G. Tucker, B. K. Bulik-Sullivan, B. J. Vilhjalmsson, H. K. Finucane, M. Salem, R. D. I. Chasman, P. M. Ridker, B. M. Neale, B. Berger, N. Patterson, and A. Price. Efficient bayesian mixed-model analysis increases association power in large cohorts. *Nature Genetics*, 47(3):284–290, 2015.
- [19] S. Yang, J. Wen, S. T. Eckert, Y. Wang, D. J. Liu, R. Wu, R. Li, and X. Zhan. Prioritizing genetic variants in GWAS with lasso using permutation-assisted tuning. *Bioinformatics*, 36(12):3811–3817, 2020.
- [20] J. Mbatchou, L. Barnard, J. Backman, A. Marcketta, J. A. Kosmickim, A. Ziyatdinov, C. Benner, C. O’Dushlaine, M. Barber, B. Boutkov, L. Habegger, M. Ferreira, A. Baras, J. Reid, G. Abecasis, E. Maxwell, and J. Marchini. Computationally efficient whole-genome regression for quantitative and binary traits. *Nature genetics*, 2021.
- [21] E. Sawyer, M. Banuelos, R. F. Marcia, and S. Sindi. A neural network approach for anomaly detection in genomic signals. *2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 968–971, 2020.
- [22] M. Nicolau, R. Tibshirani, A-L. Borresen-Dale, and S. S. Jeffrey. Disease-specific genomic analysis: identifying the signature of pathologic biology. *Bioinformatics*, 23(8):957–965, 2007.
- [23] S-J. Wang and R. Simon. Use of genomic signatures in therapeutics development in oncology and other diseases. *Pharmacogenomics Journal*, 6(8):166–173, 2006.
- [24] Li. C. and M. Li. Gwasimulator: a rapid whole-genome simulation program. *Bioinformatics*, 24(1):140–142, 2008.
- [25] Su. Z., J. Marchini, and P. Donnelly. HAPGEN2: simulation of multiple diseases snps. *Bioinformatics*, 27(16):2304–2305, 2008.
- [26] H. F. Porter and P. F. O’Reilly. Multivariate simulation framework reveals performance of multi-trait GWAS methods. *Scientific Reports*, 7:38837, 2017.
- [27] H. V. Meyer and E. Birney. Phenotypesimulator: A comprehensive framework for simulating multi-trait, multi-locus genotype to phenotype relationships. *Bioinformatics*, 34(17):2951–2956, 2018.
- [28] P. C. Phillips. Epistasis-the essential role of gene interactions in the structure and evolution of genetic systems. *Nature Reviews*, 9(11):855–867, 2008.
- [29] D. B. Fogel and J. W. Atmar. Comparing genetic operators with gaussian mutations in simulated evolutionary processes using linear systems. *Biological Cybernetics*, 63:111–114, 1990.
- [30] D. I. Podolskiy, A. V. Lobanov, G. V. Kryukov, and V. N. Gladyshev. Analysis of cancer genomes reveals basic features of human aging and its role in cancer development. *Nature communications*, 7:12157, 2016.
- [31] R. Gao, A. Davis, T. O. McDonald, E. Sei, X. Shi, Y. Wang, P. C. Tsai, A. Casasent, J. Waters, H. Zhang, F. Meric-Bernstam, F. Michor, and N. E. Navin. Punctuated copy number evolution and clonal stasis in triple-negative breast cancer. *Nature Genetics*, 48:1119–1130, 2016.
- [32] I. Y. Zhbannikov, K. G. Arbeev, and A. I. Yashin. cophesim: a comprehensive phenotype simulator for testing novel association methods. *F1000Research*, 6(1294), 2017.
- [33] M. Shi, D. M. Umbach, A. S. Wise, and C. R. Weinberg. Simulating autosomal genotypes with realistic linkage disequilibrium and a spiked-in genetic effect. *BMC Bioinformatics*, 19, 2018.

- [34] W. Yang and Gu. C. C. A whole-genome simulator capable of modeling high-order epistasis for complex disease. *Genetic Epidemiology*, 37(7):686–694, 2013.
- [35] D. B. Blumenthal, L. Viola, M. List, J. Baumbach, P. Tieri, and T. Kacprowski. EpiGEN: an epistasis simulation pipeline. *Bioinformatics*, 36(19):4957–4959, 2020.
- [36] Y. Tang and X. Liu. G2P: a Genome-Wide-Association-Study simulation tool for genotype simulation, phenotype simulation and power evaluation. *Bioinformatics*, 35(19):3852–3854, 2019.
- [37] D. A. Reidenbach, A. Lal, L. Slim, O. Mosafi, and J. Israeli. GEPSi: A python library to simulate GWAS phenotype data. *bioRxiv*, 2021.
- [38] R. A. Weinberg. How cancer arises. *Scientific American*, 275(3):62–70, 1996.
- [39] H. M. Temin. Evolution of Cancer Genes as a Mutation-driven Process1. *Cancer Research*, 48(7):1697–1701, 1988.
- [40] A. F. Rubin and P. Green. Mutation patterns in cancer genomes. *Proceedings of the National Academy of Sciences*, 106(51):21766–21770, 2009.
- [41] A. Makohon-Moore and C. A. Iacobuzio-Donahue. Pancreatic cancer biology and genetics from an evolutionary perspective. *Nature Reviews Cancer*, 16:553–565, 2016.
- [42] M. J. Fusco, H. J. West, and C. M. Walko. Tumor Mutation Burden and Cancer Treatment. *JAMA Oncology*, 7(2):316–316, 2021.
- [43] C. U. Blank, J. B. Haanen, A. Ribas, and T. N. Schumacher. The cancer immunogram. *Science*, 352(6286):658–660, 2016.
- [44] N. Bellomo and M. Delitala. From the mathematical kinetic, and stochastic game theory to modelling mutations, onset, progression and immune competition of cancer cells. *Physics of Life Reviews*, 5(4):183–206, 2008.
- [45] J. H. Holland. Genetic algorithms. *Scientific American*, 267(1):66–73, 1992.
- [46] L. Davis. Applying adaptive algorithms to epistatic domains. *Proceedings of the international joint conference on artificial intelligence*, pages 162–164, 1985.
- [47] Z. W. Bo, L. Z. Hua, and Z. G. Yu. Optimization of process route by genetic algorithms. *Robotics and Computer-Integrated Manufacturing*, 22:180–188, 2006.
- [48] M. Salehi and A. Bahreinejad. Optimization process planning using hybrid genetic algorithm and intelligent search for job shop machining. *Journal of Intelligent Manufacturing*, 22(4):643–652, 2011.
- [49] L. Bo and L. Rein. Comparison of the luus-jaakola optimization procedure and the genetic algorithm. *Engineering Optimization*, 37(4):381–396, 2005.
- [50] A. Ghaheri, S. Shoar, M. Naderan, and S. S. Hoseini. The applications of genetic algorithms in medicine. *Oman Med J.*, 30(6):406–416, 2005.
- [51] J. Zhao and M. Xu. Fuel economy optimization of an atkinson cycle engine using genetic algorithm. *Applied Energy*, 105:335–348, 2013.
- [52] R. J. Parsons, S. Forrest, and C. Burks. Genetic algorithms, operators, and dna fragment assembly. *Machine Learning*, 21:11–33, 1995.
- [53] S. Shah and A. Kusiak. Cancer gene search with data-mining and genetic algorithms. *Computers in Biology and Medicine*, 37(2):251–261, 1995.
- [54] S. Katoch, S. S. Chauhan, and V. Kumar. A review on genetic algorithm: past, present, and future. *Multimedia Tools and Applications*, 80:8091–8126, 2021.

- [55] D.S. Weile and E. Michielssen. Genetic algorithm optimization applied to electromagnetics: a review. *IEEE Transactions on Antennas and Propagation*, 45(3):343–353, 1997.
- [56] T. Bhoskar, O. K. Kulkarni, N. K. Kulkarni, S. L. Patekar, G. M. Kakandikar, and V. M. Nandedkar. Genetic algorithm and its applications to mechanical engineering: A review. *Materials Today: Proceedings*, 2(4):2624–2630, 2015.
- [57] D. B. Hibbert. Genetic algorithms in chemistry. *Chemometrics and Intelligent Laboratory Systems*, 19(3):277–293, 1993.
- [58] S.K. Pal, S. Bandyopadhyay, and S.S. Ray. Evolutionary computation in bioinformatics: a review. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 36(5):601–615, 2006.
- [59] B. Chowdhury and G. Garai. A review on multiple sequence alignment from the perspective of genetic algorithm. *Genomics*, 109(5):419–431, 2017.
- [60] B. D. Ondov, P. Treangen, T. J. Melsted, A. B. Mallonee, N. H. Bergman, S. Koren, and A. M. Phillippy. Mash: fast genome and metagenome distance estimation using minhash. *Genome Biology*, 17:132, 2016.
- [61] A. Shukla, H. M. Pandey, and D. Mehrotra. Comparative review of selection techniques in genetic algorithm. In *2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, pages 515–519, 2015.
- [62] D. Dvinskikh, A. Ogaltsov, A. Gasnikov, P. Dvurechensky, A. Tyurin, and V. Spokoiny. Adaptive gradient descent for convex and non-convex stochastic optimization. *arXiv*, 2020.
- [63] H. Jin, F. Chollet, Q. Song, and X. Hu. Autokeras: An automl library for deep learning. *Journal of Machine Learning Research*, 24(6):1–6, 2023.
- [64] J. Wu, X-Y. Chen, H. Zhang, L-D. Xiong, H. Lei, and S-H. Deng. Hyperparameter optimization for machine learning models based on bayesian optimization. *Journal of Electronic Science and Technology*, 17(1):26–40, 2019.
- [65] F. Almeida and G. Xexeo. Word embeddings: A survey. *arXiv*, 2019.
- [66] A. F. Ganai and F. Khursheed. Predicting next word using rnn and lstm cells: Stastical language modeling. In *2019 Fifth International Conference on Image Information Processing (ICIIP)*, pages 469–474, 2019.
- [67] O. Chervova, L. Conde, J. A. Guerra-Assuncao, I. Moghul, A. P. Webster, A. Berner, E. L. Cadieux, Y. Tian, V. Voloshin, T. F. Jesus, R. Hamoudi, J. Herrero, and S. Beck. The personal genome project-uk, an open access resource of human multi-omics data. *Scientific Data*, 6:257, 2019.
- [68] J. D. Churchill, S. E. Schmedes, J. L. King, and B. Budowle. Evaluation of the illumina® beta version forenseq™ dna signature prep kit for use in genetic profiling. *Forensic Science International: Genetics*, 20:20–29, 2016.
- [69] J. Alexandrov, L. B. Kim, N. J. Haradhvala, M. N. Huang, A. W. T. Ng, Y. Wu, A. Boot, K. R. Covington, D. A. Gordenin, E. N. Bergstrom, S. M. A. Islam, N. Lopez-Bigas, L. J. Klimczak, J. R. McPherson, S. Morganella, R. Sabarinathan, D. A. Wheeler, V. Mustonen, G. Getz, S. G. Rozen, M. R. Stratton, and PCAWG Consortium. The repertoire of mutational signatures in human cancer. *Nature*, 578:94–101, 2020.
- [70] D. Defays. An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366, 1977.
- [71] M. L. Menendez, J. A. Pardo, L. Pardo, and M. C. Pardo. The jensen-shannon divergence. *Journal of the Franklin Institute*, 334(2):307–318, 1997.
- [72] J. M. Joyce. *Kullback-Leibler Divergence*, pages 720–722. Springer Berlin Heidelberg, 2011.
- [73] J. Zhou, P. Zhong, and T. Zhang. A novel method for alignment-free dna sequence similarity analysis based on the characterization of complex networks. *Evolutionary Bioinformatic*, 12:229–235, 2016.
- [74] T. Lazebnik and S. L. Keren. Knowledge-integrated autoencoder model. *arXiv*, 2023.

- [75] Y. A. Veturi, W. Woof, T. Lazebnik, I. Moghul, P. Woodward-Court, S. K. Wagner, T. A. Cabral de Guimaraes, M. D. Varela, B. Liefers, P. J. Patel, S. Beck, A. R. Webster, O. Mahroo, P. A. Keane, M. Michaelides, K. Balaskas, and N. Pontikos. Syntheye: Investigating the impact of synthetic data on ai-assisted gene diagnosis of inherited retinal disease. *Ophthalmology Science*, page 100258, 2022.
- [76] T. Lazebnik and S. Bunimovich-Mendrazitsky. Decision tree post-pruning without loss of accuracy using the sat-pp algorithm with an empirical evaluation on clinical data. *Data & Knowledge Engineering*, 145:102173, 2023.

## Appendix

The proposed model’s hyperparameters descriptions and values have been used as part of the presented experiments. These values are chosen throughout the trial and error process during the development of CGMM for a subset of the samples used in the presented Exp 1.

Parameter definition	Value
Pairing approach [1]	Subset approach with 10 pairs for each <i>case</i> samples
Number of generations in RBGA [1]	100
Initial mutation rate in the genetic algorithm [1]	$2.5 \cdot 10^{-4}$
Mutation rate decrease factor in the genetic algorithm [1]	$1 \cdot 10^{-5}$
Royalty rate in the genetic algorithm [1]	0.05
The threshold value, $\zeta$ , for the stop condition [1]	$1.5 \cdot 10^3$
Bounding factor, $\beta$ , for the accelerated-bounded mutation operator [1]	$4.5 \cdot 10^3$
AutoEncoder’s encoding layer dimension [1]	8192

Table 3: The description of the model’s hyperparameters and their values.