GENETIC ALGORITHMS AND THEIR APPLICATIONS TO SYNTHETIC DATA GENERATION

THE THESIS SUBMITTED TO THE UNIVERSITY OF MANCHESTER FOR THE DEGREE OF DOCTOR OF PHILOSOPHY IN THE FACULTY OF HUMANITIES

YINGRUI CHEN

The University of Manchester
School of Social Science
Department of Social Statistics

2020

Contents

List of Figures

Word Count: 64450

ABSTRACT

Data synthesis is a statistical disclosure control technique that prevents the leakage of personal information from survey data. Rubin, who originally proposed this technique, treated the confidential data within a dataset as missing and then replaced those data using multiple imputation [103]. Most methods in data synthesis were then developed based on this principle. However, data synthesis is a multi-objective problem that aims to maximise information utility as well as minimising disclosure risks, and these methods have no explicit mechanism for balancing the objectives. This issue is the basis for the line of enquiry embodied in this thesis.

The need to optimise competing objectives suggests the possible use of iterative machine learning techniques for data synthesis, but - to date - investigations of this possibility have been limited. In the thesis, a new synthesis method using Genetic Algorithms (GAs) is introduced. GAs are evolutionary computational methods that simulate natural evolution. They allow candidates (which in this thesis are datasets) to compete, reproduce and mate in a pre-determined environment until one or more of them perfectly fits the environment (which is defined by a set of objectives). GAs were firstly used on binary strings and now they have variants that deal with different problems and data forms. In this thesis, a GA data synthesiser whose candidates are matrix and real-coded data is designed, and most of its parameters and hyper-parameters tested. A new information utility function to measure the overall divergence from synthetic data to the original data is used. The results of running the synthesiser on a real dataset are presented, which show that the GA approach successfully produced plausible synthetic data using a single utility objective and they were proved to be able to seek for a trade-off between information utility and disclosure risks during the process of synthesising. The overall conclusion is that GAs represent a significant opportunity for the practice of data synthesis.

Keywords: Genetic Algorithms, Data Synthesis, Data Privacy, Machine Learning

DECLARATION

The paper Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator [23] is jointly authored with another Manchester university student (Jennifer Taub). I understand that the paper will also be included in a thesis submitted by that student to this university. All the other work has not been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

COPYRIGHT

ACKNOWLLEDGEMENTS

I would like to express my sincere gratitude to my supervisor Prof. Mark Elliot: for his continuous support of my Ph.D. study and related research, for his patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

I would like to thank the rest of my thesis committee: Prof. Joseph Sakshaug, Prof. Natalie Shlomo, Dr Duncan Smith and Dr Julia Handl, thanks for your insightful comments and encouragement, which widens my research from various perspectives.

Thanks to everyone in the department, especially Jennifer Taub. It was great to work and party with all of you during the last four years.

Last but by no means least, I am eternally grateful to my wife Robyn, my mother Xingying, my father Jingsheng and all my great friends, with a special mention to Xinhuan, Xiawei, Li Zhou, Pan and Maomao. Thank you for always being there, loving me and supporting me when I needed you the most.

Chapter 1

Introduction

Since the advent of the Internet, people have been exchanging more information, and such sharing has become easier. This raises the ever-present possibility that such information would be shared beyond the original recipients and possibly with malevolent intent. Recent scandals like the Facebook confidentiality breach (that leaked data relating to millions of users to Cambridge Analytica [67]) have raised public concerns about personal data misuse.

Data controllers (such as Facebook) are organisations that make decisions about the collection and/or processing of personal data. They may include trade bodies and public services. Although the process by which such organisations collect and analyse personal data might be legitimate, for good purposes and under regulatory control, the risk of confidentiality breaches from sensitive information is always present when the data are published, disseminated or shared. Take, for example, respondents whose information is collected through surveys; this is perhaps the most general case in data protection. Leakage of their privacy usually happens on poorly anonymised data or through unauthorised access to the database. The breaches of privacy will not only harm respondents but discredit the reliability of data controllers with the public [87].

Data privacy controls act on two domains: regulatory and technological. The regular updating of data protection regulations seeks a balance between owner's controllers and respondents' rights. For example, the updated Data Protection Act 2018 and GDPR 2016/679 set out principles, rights and obligations for most processing of personal data for EU and EEA companies or other subjects that use personal data from the EU and EEA zones [16]. Technically, data controllers are responsible for applying appropriate technical processes and carrying out privacy impact assessments of data before processing it. As article 23 of GDPR states: "the controller shall implement appropriate technical and organisational measures in an effective way in order to meet the requirements of this

Regulation and protect the rights of data subjects [55]."

Anonymisation is one of the critical approaches to protect personal data that spans technical, regulatory, and organisational measures. A key technical approach to anonymisation is called statistical disclosure control (SDC). It protects respondents (data subjects)' confidentiality by removing, aggregating, distorting or modifying the identifiers that allow - either directly or in combination with other information - data subject re-identification and the disclosure of sensitive information. Common mechanisms used in SDC methods include:

1. Removing identifiers that are not necessary for research but confidential for respondents like name, addresses, postcode and phone numbers;
2. Aggregating variables to reduce the precision of actual information. For example, record the year of birth rather than the date of birth;
3. Hiding outliers; for example sensitive information like "annual salary" is often top- or bottom- bounded to avoid identifying high or low paid individuals [122].
4. Perturbing sensitive records in the original dataset so they cannot reveal the true information of respondents.

The SDC method that is the focus of this thesis is called *data synthesis*, which controls the risk of data subject identification from datasets by substituting real information from original data with artificial information but doing so in a manner that retains the datasets' statistical properties. It aims to provide publishable datasets derived from the original data with near-zero risk of linkage or re-identification. The research reported in this thesis studied data synthesis of microdata, which are matrix-format data whose columns represent variables and rows represent cases (usually individual respondents). Microdata is used as raw material for different kinds of data such as tables of counts, summary statistics and graphical visualisations. The job of data synthesis in microdata protection is to build a matrix similar to the original data but with very low disclosure risk.

The similarity between two datasets can be considered as a set of multivariate statistical properties, such as covariance, correlation and coefficients from regressions. Since datasets contain many variables, the similarity of their relations to the dataset's synthetic version cannot be easily captured through a single model or equation. For most orthodox approaches in data synthesis, data utility is not explicitly built into the generating mechanism and so whatever utility is produced is necessarily fully represented by the model used to generate the data [109]. As the parameters of synthesising models are fixed, data

utility and risk are routinely assessed or measured through post hoc measures. An alternative would be to embed the measures directly into the synthetic data generating process, which is core to the approach represented in this thesis. In principle, machine learning algorithms have the potential because they can involve utilities and risks as objectives in the model and are able to gradually optimise candidate solutions by interacting with the objectives in the process. Machine learning is a system that automatically learns and improves from experience without engaging in human decision making. The principle of machine learning is that candidates in a natural system can interact with a changing environment and evolve from feedback.

Genetic Algorithms (GAs) are machine learning algorithms that simulate evolutionary processes in biological systems. They have been successfully used in solving high-dimensional and complex problems [14]. A traditional GA works on binary strings, selection operators (broadly analogous to natural selection) and crossover and mutation operators (broadly analogous to natural reproduction processes) [62]. Although the idea for GAs was first inspired by natural evolution, the design of a GA does not necessarily follow the natural ways. Users are allowed to design crossover and mutation methods, to change parameters and implement hyper-parameters, to parallelise learning agents and to control population features. Critically, as the field has developed, more complex structures than a linear form for candidates has become supported. For example, matrix real-coded genetic algorithms (MRCGAs) is a variant that operate on matrices rather than strings, and which provides the potential for solving two-dimensional problems [124]. Since the format of microdata is a matrix (usually two dimensional), it is a reasonable starting point assume the MRCGAs are more suitable for data synthesis than binary GAs. How to practically apply this approach to data synthesis will be investigated in this thesis.

The thesis aims to demonstrate that the application of GAs to data synthesis is a promising development. The main objectives of the thesis include:

1. Design the framework for GA data synthesisers and produce application code using Python.
2. Choose suitable formats for inputs and outputs.
3. Choose or design measurable objectives for information utility and disclosure risk.
4. Choose or design suitable methods for operators and their corresponding parameters.
5. Evaluate the performance of the GA synthesiser and suggest routes to improving its efficiency and effectiveness (if necessary).

1.1 Summary of Chapters

The thesis consists of 7 chapters. The first three chapters narrate the fundamental knowledge that underpins the thesis. Chapter Statistical Disclosure Control introduces data privacy and statistical disclosure control (SDC) techniques, which includes data masking and data synthesis. The chapter also summarises some common synthesising models that fully and partially synthesise data, followed by statistical assessments of data utility and disclosure risks. Chapter Machine Learning, Natural Computation and Genetic Algorithms introduces machine learning algorithms and the relationship between machine learning and GAs at the beginning. It then summarises the theoretical background of GAs, including their operators, the common methods used in operators and the theories underpinning both linear and matrix GAs. Since data synthesis is a multi-objective optimisation problem, this chapter also introduces some multi-objective GA models. Chapter Model Design is the methodological core of the thesis, which describes how the GA synthesiser was designed and implemented. It firstly justifies the choice of full synthesis and MRCGA, followed by theoretical and experimental design which are necessary for the construction of GA synthesisers. Each section focuses on one of the essential parameter/operators (the initial population, selection mechanism, crossover and mutation), plus some critical issues including positional bias, schema Theory and adaptive parameters. Possible utility and disclosure measures are also formulated at the end of the chapter.

All experiments are in paper-format and are presented in Chapter Experiments and Results. The main target of these experiments was to investigate the suitable operator methods and parameters settings for GA synthesisers. The chapter also lists some 'bonus findings' on the relationship between data structure and utility. All experimental results are presented in seven papers, where information on the authors, their contributions and the destinations for each paper can be found in Section 1.2.

Chapter Summary and Model Integration delivers the summary of findings in the thesis. It combines all studies of components of GA synthesisers into a flowchart and gives a full description of GA synthesisers. The final chapter Impacts and Critical Analysis of the Model summarises the impacts and limitations of the research, including applying GA synthesisers to continuous or mixed data (data that forms with both continuous and categorical variables), the potential of use Pareto-GAs to explore the solution space, and advantages and disadvantages of the current model. This chapter also provides suggestions on future research for GA synthesisers.

1.2 Summary of Publications

The thesis contains eight papers/journal articles that have been published/are prepared for publication (7 of them are in Chapter Experiments and Results and the other is in the Appendix). Thanks to the contribution given by my supervisors and my colleague to the development of ideas, analysis and editing. They are co-authors for each paper to which they contributed. The following table (Table 1.1) provides briefs for each publication:

Paper title	Authors	Destination	State
A Genetic Algorithm Approach to Synthetic Data Production	Yingrui Chen, Mark Elliot, Joseph Sakshaug	European Conference on Artificial Intelligence, 2016	Published
Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data	Yingrui Chen, Mark Elliot, Joseph Sakshaug	UNECE Work Session on Statistical Data Confidentiality, 2017	Published
The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods	Yingrui Chen, Mark Elliot, Duncan Smith	Privacy in Statistical Database, 2018	Published
Matrix GA: building blocks in data synthesis	Yingrui Chen, Mark Elliot	ACM Transactions on Privacy and Security (TOPS)	Submitted
The Impact from Initial Population in GA Synthetic Data Generator	Yingrui Chen, Mark Elliot	Working paper	Complete
Impact of Full Contingency Table in Data Synthesis	Yingrui Chen, Mark Elliot	Transactions on Data Privacy	In Preparation
Exploring the Impact of Adaptive Parameters on a Genetic Algorithm Synthesiser	Yingrui Chen	Working Paper	Complete
Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator	Yingrui Chen, Mark Elliot, Jennifer Taub	Joint UNECE/Eurostat Work Session on Statistical Data Confidentiality, 2019	Published

Table 1.1: Summary of publications (in the order that they appear in the thesis)

My contributions to all the publications listed above include exploring the theoretical background of matrix GAs, proposing the idea of implementing them on data synthesis, constructing and coding the initial model, designing experiments, recording experimental results and drafting. Details of co-authors' contributions are listed below:

- For paper *A Genetic Algorithm Approach to Synthetic Data Production* (see Appendix 7.3) and paper Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data, the idea of implementing of matrix GAs was discussed together with Mark Elliot and Joseph Sakshaug, and all drafts were edited by them.
- Duncan Smith provided the idea and code of utility functions in *The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Method*. He and Mark also edited the draft of this paper.
- All experiment results in papers: Matrix GA: building blocks in data synthesis, The Impact from Initial Population in GA Synthetic Data Generator, Impact of Full Contingency Table in Data Synthesis and Exploring the Impact of Adaptive Parameters on a Genetic Algorithm Synthesiser are discussed with Mark and he shared his findings for the papers. He also provided editorial input on drafts of these papers.
- Jennifer Taub designed and coded the risk function in paper Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator and wrote all the relevant parts. She carried a post-hoc utility test of data. Mark Elliot supervised the whole process, suggested changes in the experiment design, coding and objective design, and edited the paper.

For all papers except paper The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods and paper Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator, I programmed and ran the experiments, produced the drafts and prepared the publishable versions. The overall intellectual architecture for the thesis is primarily mine with co-authors providing feedback and a sounding board for ideas.

Chapter 2

Statistical Disclosure Control

Microdata contains information that is collected from or assembled for individual population units [27]. It is a matrix with each column representing a variable and each row representing a data subject (a respondent or a case) and often underlies other data formats. Within SDC, variables in microdata are classified as (1) formal (or direct) identifiers, which are generally removed during the pre-processing of the original data (2) quasi (or indirect) identifiers, which do not themselves uniquely identify respondents but may create a unique identifier when combined with other quasi-identifiers, and (3) targets, which contain the sensitive information of respondents and thus are the *raison detre* for privacy protection [33].

A rational attacker can derive information that is explicitly contained in quasi-identifiers by a homogeneity attack or background knowledge attack [76] [79], thus reveal confidential information on one or more individuals. A homogeneity attack occurs when an equivalence class does not have enough diversity on a sensitive variable so that intruders can disclose sensitive information of a group of respondents in the dataset. An equivalence class in a dataset is a set of records that have the same values for a given set of variables (usually the quasi-identifiers). Even if exact values in sensitive variables are not revealed, something close to the correct value can harm respondents. For example, if the target information for an equivalence class is semantically similar then broad inferences can be made if everyone in an equivalence class has either gastric ulcers, gastritis or stomach cancer, the attacker can infer that all of the respondents in that equivalence class has a stomach-related disease). A background knowledge attack occurs when intruders build up a connection between the published quasi-identifiers and external information that may come from his own knowledge or external data, which enhances the certainty about the values of sensitive variables belonging to a group of records in the equivalence class of an individual [79].

To process a dataset whilst preventing personal information leakage, data owners have to anonymise the data they control before that processing takes place. However, a poorly conducted anonymisation can result in information loss. According to Matwin et al.. publishable data must be "very close" to the original data. Matwin does not define "very close", but we can reasonably interpret this as meaning that users should be able to obtain similar results by analysing the data as they would with the original data [79]. On the other hand, the data need to be sufficiently divergent from the original data to prevent attackers from obtaining sensitive information from it. The orthodox technology for controlling the balance between data utility and disclosure risks is called *statistical disclosure control*(SDC). In the SDC framework, disclosure risks are typically classified into (1) identification disclosure (also known as re-identification) and (2) attribute disclosure (also known as attribution or inference disclosure in [48]). Re-identification occurs when attackers can identify a natural person within released data, for example, through a background knowledge attack. Attribution occurs when attackers can determine the value of one or more confidential attributes of a group of respondents from released data, for example, through homogeneity attack [11].

SDC on microdata should be implemented in a way to prevent both re-identification and attribution, but it can be applied to either quasi-identifiers to reduce the possibility of re-identification or sensitive variables to reduce the possibility of correctly disclosing real information[1] [107]. [27] classified the methods used in SDC (which they called microdata protection) into four categories (see Fig. 2.1).

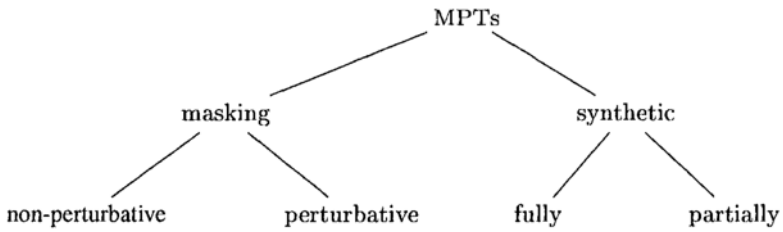


Figure 2.1: SDC (Microdata Protection) Techniques [27]

As there are two types of data: continuous and categorical, Domingo-Ferrer and Torra argued that there is no single SDC method that can be applied to both data types in a straightforward manner [34]. The corresponding model needs to be re-designed in order to be applied to different types of data. In the remainder of this chapter, some common SDC techniques will be reviewed. Measures of utility and disclosure risks for both re-identification and attribution on microdata will be introduced in the last section.

[1]This should be done very carefully because harm could happen even if inaccurate information is revealed about the respondent (but the intruder believes it to be accurate)

2.1 Data Masking

Masking techniques change the microdata and come in two forms: non-perturbative and perturbative. Non-perturbative masking techniques do not add noise to the original data. Instead, they reduce the detail in some or all of the quasi-identifiers and/or target variables. For example, they may withhold or remove sensitive information from quasi-identifiers (data suppression). They may also aggregate extreme values in a single category usually high or low (top/bottom recoding) or collapse all values into a smaller number of bands (generalisation) [24].

Perturbative masking techniques distort original records. There are some conventional ways to do this: adding noise to the original data, partitioning the data into k clusters and publishing the clusters' means, and *rank swapping* [36]. Rank swapping exchanges individual records whilst maintaining the marginal counts in variables. It was designed for ordinal and continuous variables. The target variable V_i is re-arranged in ascending order. Then for $p\%$ of the records, data are swapped for the equivalent data within another set of records usually chosen based on matching variables [24]. The idea of protecting marginal values within a restricted range $p\%$ is regularly used in SDC, and $p\%$ is usually chosen from 1 to 9 [34]. The post-randomisation method (PRAM) is an advance swapping approach to mask categorical variables. It changes the values of the variables in the original dataset using probabilities generated from a Markov matrix. This Markov matrix $\mathbf{A}$ is a $K \times K$ matrix, where K is the number of categories in the variable. Every entry a_{kl} in $\mathbf{A}$, where $k,l = 1,...,K$, indicates the probability of a record that originally was k but was transformed to category l, and for any k, $\sum_{l=1}^{k} a_{kl} = 1$[74][59].

2.2 Data Synthesis

Data synthesis is another branch of SDC. Instead of removing true information or adding noise to the real data, data synthesis produces an artificial dataset that does not contain the original records of respondents. It, therefore, aims to [2]:

- Preserve the same inferences as to the original data.
- Provide sufficient information include a reasonable number of records and variables to allow for proper multivariate analyses.

Researchers often distinguish between full and partial data synthesis [39]. Fully synthetic data aims to prevent actual values from being linked to any respondent, so, data owners

can focus on the quality of the published data instead of the re-identification risks [27]. Although there may still be attribution risks (as we will see later), the linkage of auxiliary information to external files by a data intruder will not cause the re-identification of respondents, since they cannot recognise if the information in the synthetic data corresponds to a real individual [2].

Rubin was the first to present how fully synthetic datasets can be generated via multiple imputation. He represented microdata generated from a survey sample as consisting of background variables Z and observed variables Y_{obs}, respectively, where Z is available for the entire population, but the observations of Y_{obs} are only known within the sample. Then all observations that included in the population but excluded from the Y_{obs} were Y_{nobs} and therefore treated as missing data. He then imputed Y_{nobs} through M multiple imputations (usually $3 \leq M \leq 10$) from known observations of Z and Y_{obs}, and drew samples from the imputed population. This principle is applicable for both continuous and categorical variables as long as a suitable model is adopted [103]. There are non-negligible disadvantages of full synthesis. Properties of the data that are not explicitly included in the generating model cannot be represented in the synthesised dataset unless they are dependent on the properties included in the generator. Thus, unforeseen analysis of the synthetic data might lead to results that differ from those obtained using the original data [78]. Synthesising a full dataset can be practically difficult as the generating model may require all the variables as input (so that their relationship can be captured) which may be computationally expensive.

Partially synthetic data, on the other hand, reduce model dependencies and computational workload because they do not attempt to synthesise every variable in the dataset. Data controllers intend to publish partially synthetic data with limited information on some quasi-identifiers in order to preserve respondents' privacy with reduced alterations to the collected survey data [73]. However, they have to control the risk of re-identification as the published data retains the true values of some variables [39]. Multiple imputation can also be implemented in partial synthesis based on the assumption that only some of the observations Y_{rep} from target variables need to be imputed. Reiter described how to apply multiple imputation to synthesise a partial data D: Suppose $I = (I_1, ..., I_n)$ where n is the size of Y_{obs}. $I_j = 1$ if observation j is selected to be replaced by a synthetic value otherwise $I_j = 0$. The values of $Y_{rep,i}$ could be generated by Bayesian posterior predictive distribution with prior knowledge $(Y_{rep,i}|D,Z)$ [98] but in the later research the method was proved to have less efficiency [100][120].

Multiple imputation produces multiple synthetic datasets, and it is necessary to have combining rules to when carrying out analyses to increase the level of analytical validity [120]. The classic combining rules were introduced by [95] and have been used for decades. In the rules, the inference of an *estimand of interest* Q (for example, a vari-

able's population mean) is estimated with the following steps: Each synthetic dataset $D_i, i = 1,,,M$, where M is the number of times of imputations, has q_i as an estimator of Q and v_i as the estimator of the variance of q_i. Analysts can obtain valid inferences for Q through the following equations:

$$\bar{q_M} = \sum_{i}^{M} \frac{q_i}{M}$$
$$\bar{b_M} = \frac{(q_i - \bar{q_M})^2}{M-1}$$
$$\bar{v_M} = \sum_{i}^{M} \frac{v_i}{M}$$

where $\bar{q_M}$ is the estimator of Q and $T = \frac{b_M}{M+\bar{v_M}}$ is the estimator of the variance of $\bar{q_M}$.

Multiple imputation was the original framework used for producing synthetic data, but it has proved time-consuming in many cases [100][120]. Therefore some synthesising models (synthesisers) deploy single imputation instead.

2.2.1 Synthesisers: Mechanisms and Models

This section briefly introduces some of the data synthesisers in use (for both continuous and categorical data). It aims to explore the common mechanism in these models.

Ideally, drawing a similar sample by adequately tuning the parameters of the exact distribution function of the actual data would be possible for continuous variables. However, this is not easy for multivariate situations. Work with classes of similar records tends to be more practical than work with the whole dataset. Therefore clustering is commonly used for synthesising continuous data. Cano, Ladra and Torra proposed a fuzzy c-regression model (FCRM) to synthesise continuous data. The model gives c regression models to predict the value of y_i (the synthetic version of x_i).

$$y_i = f_i(x_i, \beta_i) + \varepsilon_i, 1 \leq i \leq c.$$

The algorithm updates the constant coefficient β_i in different regression models to minimise the error vector ε [18]. Other clustering techniques include *Latin Hypercube sampling*, mixture model and *quantile regression*. Latin Hypercube sampling generates synthetic data from only uncorrelated variables in the original data then use restricted pairing algorithm to reproduces the rank correlation structure of the original data [30]. Mixture model uses different parameters to produce synthetic data from different clusters and modifies their covariance matrices to ensure there is no small cluster in the output to draw attention from intruders [90]. As for continuous data with skewed distributions, Pistner et al.. used *quantile regression* to generate a synthetic version [93].

Mateo-Sanz et al.., on the other hand, proposed a learning algorithm to to synthesise a continuous dataset iteratively [80]. They believed that retaining univariate means and the covariance matrix of a dataset can adequately preserve its other statistical qualities. Their synthesiser gradually optimises the covariance matrix of the synthetic data until it is equal that of the original data. The algorithm is not entirely stable because it produces different synthetic versions, and the utility of its output is acceptable if only the initial parameters of the algorithm were carefully selected [80].

To date, synthesisers for categorical data have been modelled based on Rubin's central idea [103]; they treat target variables as missing values and refill them using reasonable distributions or prediction/classification models, for example, multiple imputations. Reiter composed a CART model of synthetic data generation by estimating the value of each variable given the partitions of a predictor space. The classification and regression tree (CART) is a strong non-parametric prediction tool. It predicts outcomes by recursively determining the explanatory power and variance from input variables. He demonstrated that CART could generate synthetic data by estimating the conditional distributions of an outcome from multiple predictors and increases the synthesiser's effectiveness and accuracy in the fitting of non-linear relationships. However, Reiter only applied CART in the context of partial data synthesis [99].

More recently, Caiola and Reiter suggested using *random forest* for partially synthesising categorical data [17]. Random forest develops from CART, it grows a large number of trees based on different samples and/or predictor variables. This work inspired Drechsler [38], who then used *support vector machines* (SVMs) to synthesise categorical data. He stated that the multivariate predictors from CART models could form a hyperplane, which is optimised by maximising the margins between separated classes, as well as by reducing misclassification to an acceptable level. The SVM is trained with different misclassification levels, and its performance is measured based on how well the predicted target variable is in the test dataset. On the whole, deploying a classification method should start with binary variables from a set of predictors and then extend variables with more than two categories.

By assuming that each individual in the microdata belongs to one of a set of latent classes, Hu et al.. designed the DPMPM synthesiser for categorical data. The model assumes that individuals from the dataset in the same class share the same multinomial distribution over target variables [65]. Hu and Nobuaki then added a tuning parameter to transform the multinomial distribution to quasi-multinomial distribution, which they claim provides the model with agility in controlling utility and disclosure risk [66].

2.3 The Assessment of Utility and Risk of Microdata

2.3.1 Data Utility

There are two determinants to whether a synthetic dataset is useful. The dataset should contain a sufficient number of variables and records so that it is analytically interesting to users, and an acceptable level of accuracy across a reasonable range of analyses [81].

Information in census and social survey data are usually presented in a categorical format, i.e. data that take a finite number of (or countable numbers as) values. For any given set of categorical variables, the same information can - in principle at least - be encoded in a contingency table, which captures the between-variate structure of the data. For example, if we denote the jth column of a microdata set Y as $Y_{:,j}$, then the bi-variate contingency table constructed from distinct columns $Y_{:,j}$ and $Y_{:,k}$ is $CT(Y_{:,j}, Y_{:,k})$, and with entries $n_{r,c}$ is

$$n_{r,c} = \sum_{i=1}^{n} [Y_{i,j} = (I_j)_r \wedge Y_{i,k} = (I_k)_c] \tag{2.1}$$

where the square brackets are Iverson brackets and the levels of I_j and I_k are indexed $r \in [1..|I_j|]$ and $c \in [1..|I_k|]$, respectively. Meanwhile, the contingency table $CT(Y_{:,j}, Y_{:,k})$ is illustrated in Table 2.1.

$n_{1,1}$	$n_{1,2}$	$\cdots$	$n_{1,C}$
$n_{2,1}$	$n_{2,1}$	$\cdots$	$n_{2,C}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$
$n_{R,1}$	$n_{R,1}$	$\cdots$	$n_{R,C}$

Table 2.1: An illustrative $R \times C$ contingency table

One important statistic obtained from the contingency table is the Chi-square (χ^2). It is used in hypothesis tests of a model's goodness-of-fit and/or independence between variables.

$$\chi^2 = \sum_i \sum_j \frac{(n_{r,c} - e_{r,c})^2}{e_{r,c}},$$

where $e_{r,c}$ is the expected count for $n_{r,c}$ in Table 2.1. Aside from tests of goodness-of-fit and independence, χ^2 can also evaluate the homogeneity between two statistical datasets. Shlomo indicated that Chi-square and Cramer's V are two key statistics to measure information loss by cross-classifications of pairs of categorical attributes [107]. Cramer's V is a measure of association between a pair of nominal variables derived from χ^2 and can be

calculated from an $R \times C$ contingency table T,

$$\phi_C(T) = \sqrt{\frac{\chi^2 \backslash n}{min(R-1, C-1)}} \tag{2.2}$$

Modelling is another analytic tool used for categorical data analysis. Suppose a dataset has n random variables $X_1, X_2, \cdots, X_N$ corresponding to n observed values x_1, x_2, $\cdots, x_n$. A model to estimate unknown parameter Θ with respect to the joint probability

$$P(X_1 = x_1) \cap \cdots \cap (X_n = x_n)$$

can be written as [5].

$$f(x_1, \cdots, x_n | \theta), \theta \in \Theta. \tag{2.3}$$

The similarity between Θ in original and synthetic datasets is another measure for the utility of the synthetic dataset.

A real dataset usually contains continuous variables. Thus, the utility of continuous data is worth consideration when taking the application of GAs to the real world. Mateo-Sanz et al. suggested that synthetic versions of continuous data should retain at least the following properties of the original datasets in order to be analytically valid: (1) the mean and covariance, (2) marginal values for tabulations and (3) at least one distributional characteristic of the original data [81]. Modelling is also important for continuous data analysis. Besides keeping the high similarity of Θ in Equation 2.3 to the original data, a usable synthetic dataset should have similar results as the original data after being clustered by the same model [107].

2.3.2 Disclosure Risk

As indicated previously, here are two main types statistical disclosure risk in microdata: re-identification, which refers to the exposure of population units through the linkage of corresponding quasi-identifiers, and attribution, which refers to the leakage of target variable values for one or more equivalence classes.

2.3.2.1 Re-identification

Ciriani et al. observe that a record of a highly visible and unusual population unit, for example, respondents with high income, is more risky [27]. Duncan et al. indicated that for aggregated forms of microdata that contain the summarised information of the whole population in a table of counts, small cell counts in this table would draw more

attention from intruders [44]. Matwin also observed that skew in the distribution of an attribute increases the chance of re-identification [79]. An example involving all the three circumstances is called *population unique*

Definition 2.3.1 (Population Unique)**.** A population unique is a population unit that has unique values on a set of quasi-identifiers within that population [45].

Population unique has the highest risk in re-identification. Therefore, uniqueness is an essential measure of disclosure risk in microdata, especially for a dataset that partially synthesised or protected by non-perturbative techniques [27].

Re-identification becomes possible when small counts exist in the cross-tabulation of quasi-identifiers. Shlomo summarised the factors for assessing the levels of disclosure risks from social surveys [107]:

1. The probability that a sample unique is population unique (where re-identification occurs on a multi-dimensional contingency table from a set of quasi-identifiers): as quasi-identifiers usually are categorical, the identification risk amongst a set of variables can be assessed from their contingency tables. She defined the disclosure risk on a contingency table as a function of combinations of quasi-identifiers within the sample and population [107].

2. The probability that records from a set of quasi-identifiers that link to auxiliary information or an external file: the measurement of such a record linkage is based on an essential assumption that there exist external files containing quasi-identifiers that are the same as some of the variables presented in the released dataset and are also available to intruders [35]. The following section gives more details regarding the issues of record linkage.

3. How powerful the SDC model is: for example, does it consider all information confidential, such as full synthesis, or only protect selective information, like partial synthesis? Re-identification will be easier for intruders if the disclosure control model does not mask the data sufficiently.

2.3.2.2 Attribution

Technically, data has more risk if there exist small frequencies in some contingency tables derived from that data. The risk is higher if the table contains more cells with small values like 0 or 1 and low counts in population tables are more dangerous than those in sample tables [6][108]. Antal et al. [6] proposed the use of entropy and conditional entropy to

evaluate attribute risks contained in data. The entropy of a random variable X is:

$$H(X) = \sum_{j=1}^{K_X} Pr(X = k_j) \cdot \log Pr(X = k_j) \tag{2.4}$$

where K_X is the range of X, entropy increases when the distribution of X converges to a uniform distribution, meanwhile, the conditional entropy of two random variables X and Y is:

$$H(X|Y) = -\sum_{j=1}^{K_{X\wedge Y}} Pr(Y = k_j) \sum Pr(X = k_j|Y = k_j) \cdot \ln Pr(X = k_j|Y = k_j)$$

where $K_{X\wedge Y}$ is the common range of X and Y. Entropy enables the measurement of disclosure risk of variable X in population-based data. Suppose the corresponding frequency table is $F = (F_1, ..., F_K)$, the empirical distribution of X is $(\frac{F_1}{N}, ..., \frac{F_K}{N})$ where $N = \sum_{i=1}^{K} F_K$. Let D be the set of zeros in F and $|D|$ is size of D. Let $\mathbf{w}$ be a vector of weights: the weighted average of disclosure risk in X is:

$$R(F, \mathbf{w}) = \mathbf{w} \cdot (\frac{|D|}{K}, 1 - \frac{H(X)}{\ln K}, \frac{1}{\sqrt{N}} \ln \frac{1}{e\sqrt{N}})$$

Suppose E is the set of zeros in the perturbated frequencies of some of variable X and F is its original, *population-based* frequencies. The following equation 2.5 can access the attribution risk in perturbative data no matter of whether it is population-based or sample-based.

$$R(F, G, \mathbf{w}) = \mathbf{w} \cdot (\frac{|D|}{K}^{\frac{D\cup E}{D\cap E}}, (1 - \frac{H(X)}{\log K})(1 - \frac{H(X|Y)}{H(X)}), \frac{1}{\sqrt{N}} \ln \frac{1}{e\sqrt{N}}) \tag{2.5}$$

2.3.3 Privacy Models

Several privacy models attempt to provide a guarantee against privacy leakage, and such leakage is essentially synonymous with attribution. A privacy model can give the probability of whether a respondent in the protected dataset has property P. If there are p fraction of respondents having the property, then the intruder can estimate the fraction of respondents who truthfully have property P in the real data [8]. Sweeney proposed k-anonymity which provides a guarantee against smaller than acceptable counts in released datasets [118].

Definition 2.3.2 (k-anonymity)**.** Suppose $Y(V_1, ..., V_m)$ is the real data, and QI is the set of quasi-identifiers for R. Y is k-anonymity if and only if each combination of values in QI appears in at least k occurrences.

Machanavajjhala et al. argued that k-anonymity was flawed: it was still possible - in some cases - to leak sensitive attribute information from a k-anonymous dataset because of the possible lack of conditional diversity in a sensitive attribute. They then introduced l-diversity to address this issue by determining the number of values ('well-presented' is the word used in their paper) from target variables conditioning on a given combination of quasi-identifiers [76].

Definition 2.3.3 (l-diversity)**.** A set of quasi-identifiers is l-diverse if it contains at least l diverse values for each sensitive variable. A dataset is l-diverse if every set of its quasi-identifiers is l diverse.

Although l-diversity needs no knowledge of the full distribution of quasi-identifiers and sensitive variables, and it allows privacy control when the dataset has linked, external files, in practice, the calculation of l-diversity on a dataset is complicated, especially if there are multiple sensitive variables. Moreover, its ability to prevent attribute disclosure is still insufficient. Intruders can learn information from skewed distributions or from equivalence classes that are categorically distinct but semantically similar. Therefore, the t-closeness was proposed [71].

Definition 2.3.4 (t-closeness)**.** An equivalence class has t-closeness if its distribution of the sensitive variable in this class has less than t distance[2] to the distribution of that variable in the whole dataset.

Barak et al. argued that none of the above models could measure information leakage derived from adversaries with arbitrary background knowledge, as they all assume the completeness of a chosen set of quasi-identifiers and this is impossible to verify with certainty in practice [8]. A more comprehensive risk measure is required to measure attribution risk.

There are some advanced privacy models like γ-amplification [50], v-dispersion [90] and differential privacy [46], with differential privacy being the most well-known and is used by the U.S. Census Bureau, Apple and Microsoft [54]. It assumes first that a privacy model M can be described as a mapping from a domain A to a range B over a probability simplex $\Delta(B)$[3].

[2]The distance between distributions can be calculated using the Kullback-Leibler distance, the earth mover's distance, or any similar distance measures.

[3] Given a discrete set B, the probability simplex $\Delta(B)$ is defined as

$$\Delta(B) = \{x \in \mathbb{R}^{|B|} : x_i \geq 0 \vee i \text{ and } \sum_{i=1}^{|B|} x_i = 1\}.$$

In differential privacy, data is defined as a histogram $x \in \mathbb{N}^{|\chi|}$ collected from a universe χ, where each entry x_i is the number of elements in x of type $i \in \chi$. Differential privacy uses the l_1 norm to measure how many records differ between two datasets x and y. The distance between $\chi \ni x_i$ and $\gamma \ni y_i$ by the l_1 norm is

$$||x-y||_1 = \sum_{i=1}^{\chi,\gamma} |x_i - y_i|.$$

Differential privacy guarantees that a privacy model, which involves randomness in its process, behaves similarly on a similar input dataset.

Definition 2.3.5 ((ε,δ)-differentially private)**.** The model M is (ε,δ)-differential private if for all $S \subseteq \text{Range}(M)$ and for all x,y, such as $||x-y||_1 \leq 1$,

$$Pr[M(x) \in S] \leq e^{\varepsilon} Pr[M(y) \in S] + \delta \tag{2.6}$$

When a privacy model M is $(\varepsilon,0)$-differentially private ($\delta = 0$), the output of a record is equally likely to be the output in different runs of the model. By contrast, if $\delta \neq 0$, then the output of a record is very unlikely to be the output in different runs of the model. Meanwhile, given an output $\xi \sim M(x)$, we could probably find a dataset X such that ξ looks more like an output from $M(Y)$ than from $M(X)$.

Differential privacy guarantees little or no disclosure to individuals unless the privacy model M is released. Datasets can be protected from arbitrary risks and possible linkage attacks, and the impact to individuals' privacy is negligible regardless of their presence in the dataset. Compared with other privacy-preserving models, differential privacy quantifies randomness impact and privacy loss. It allows any post-processing on the data with no leakage of information of more than $\exp(\varepsilon) \approx \exp(\varepsilon+1)$ factor [46]. One criticism of differential privacy is that although they provide a guarantee they do not seem to be able maintain the utility of the data at the same time. For example, it was confirmed that when applying differential privacy to an analysis data server, it is difficult to produce meaningful (statistical) inferences from the differentially private database [105].

The above privacy models allow data controllers to decide on a threshold of risk for a single and isolated dataset. However, the guarantee may not hold anymore if there is auxiliary information or external datasets available that have been collected from the same population, that can be linked to the isolated dataset and disclose more information of the respondents [118] (some of the linkages occur for analytical purposes). Once data controllers discover such an auxiliary dataset, they may conduct linkage between it and the single dataset they owned to assess if their dataset is still secure. The technique used to connect datasets is record linkage.

2.3.4 Record linkage

Record linkage is a technique that has many uses, one of which is a mechanism for simulating an attack on a dataset(also known as penetration test or a motivated intruder test)[61]. It is specifically used to evaluate re-identification risks in microdata [35].

Record linkage brings information together from two records that are believed to relate to the same entity. If the records perfectly match each other, then intruders would be able to construct the relationship between the released data and the corresponding external datasets [60]. Record linkage is not considered in data synthesis, especially full data synthesis, because it is unlikely to conduct record linkage as every record is 'synthetic' theoretically. Therefore, it is also briefly introduced in this section.

Often, the linkage between two datasets cannot be conducted deterministically, and we need to assess the probability that the two records belong to one entity. This is called probabilistic linkage a methodology that was initially formulated by [51]. Assume that A and B are the two populations to be compared. The records from A and B are denoted by $\alpha(a)$ and $\beta(b)$ that contain selected quasi-identifiers (e.g. name, age, sex, marital status) and errors (e.g. errors of reporting or failure to report, errors of coding). The two records can be compared using a comparison vector, which is defined as a function γ for $A \times B$. The set of all possible records of γ is the comparison space Γ.

$$\gamma[\alpha(a),\beta(b)] = \gamma_1[\alpha(a),\beta(b)],...,\gamma_k[\alpha(a),\beta(b)]. \tag{2.7}$$

The function $\gamma[\alpha(a),\beta(b)]$ computes three possible outcomes: (a,b) is matched, unmatched or possibly linked.

2.4 Chapter Summary

This chapter has introduced the theoretical background of key SDC techniques, including data synthesis. Unlike data masking, data synthesis produces a replacement for the original data that aims to carry no risk from it. The mechanism used by most data synthesisers is (multiply or singly) imputing the concerned records in the dataset. Since a particular model imputes these records, the quality of synthetic data, especially fully synthetic data, is highly dependent on this model. Measurements of that quality include data utility and disclosure risk. Therefore, the last section summarised conventional measures of utility and risks to synthetic data, including some privacy models.

Utility and risk are conflicting objectives so that improving one of them will worsen the other. So far, there is no synthesiser can optimise both objectives simultaneously or

converge to an appropriate trade-off. Compared with the common synthesiser approach, an iterative learning algorithm has the potential to explore the relationship between the conflicting objectives and to find the trade-off. Such an algorithm and its theoretical background will be introduced in the following chapter.

Chapter 3

Machine Learning, Natural Computation and Genetic Algorithms

Algorithms are the core of machine learning as they are a set of instructions transforming inputs into outputs. An algorithm can gradually train the machine learning model by learning from training data. Sometimes even the size of training data is limited compared to its entire information base. It still enables the model to explain parts of the base [4]. The majority of machine learning problems can be regarded as optimisation problems as they are finding the most suitable output through constraint and objective functions [49]. There are two questions to be considered in designing algorithms for a specific problem: how to compare the algorithm's prediction with reality and how to prove that the algorithm surpasses other algorithms on a given application. The first issue can be addressed by looking at errors from the training dataset, but these errors cannot be used to compare two algorithms because the more complex model with more parameters will always give fewer errors than a simple one. These are reasons that we usually need another dataset to validate the algorithm, namely the validating dataset. An algorithm generally run more than once in the validation set in order to obtain a generalised level of the expected error. From Ethem's view, this can reduce the effect of misleading cases and random factors [49].

Machine learning can be classified into three types: supervised learning, unsupervised learning and reinforcement learning. A supervised learning model M is defined by a set of parameters θ and optimised until the estimation error is minimised. Users label the examples found in the dataset to obtain specific mappings from inputs x to outputs y.

$$y = M(x|\theta)$$

Regression and classification are two typical problems in supervised learning. In these techniques, users first label the classes or variables to be analysed in the training data,

and then they allow the model to automatically optimise corresponding parameters until the estimation error reaches a minimum. An unsupervised learning model is designed when there is no prior expectation about the relationship between the input and the output and it is beyond the users' ability to supervise the algorithm. Unsupervised learning is used for finding hidden patterns and structures in unlabelled training data, but it is unable to identify explicit errors or rewards when evaluating a potential solution. Clustering is a typical unsupervised learning technique. When applying clustering, users give the criteria to evaluate the distance between cases within and between clusters, but they cannot tell which two cases share the same cluster before running the model [4].

With increasing system complexity, the model is unable to map the input to the output through one action but instead uses a sequence of multiple operators. The development of machine learning has inspired the development of more intelligent systems. An intelligent learning system can adapt to a changing environment without the need for supervision or manual adjustments, and the model learning is reinforced gradually from the goodness of policies it adopts. This learning technique is called reinforcement learning.

Sutton and Barto characterised reinforcement learning as 'any method that captures the most important aspects of the real problem through a learning agent interacting with its environment to achieve a goal'. This summarised the four key elements in a learning system: an environment, a policy, a reward function and a value function. An environment is a world in which the agent operates. It may vary from a simple function to a mathematical model. As the core of reinforcement learning, a policy maps the state of the system in the current environment to the action that assists the learning agent to maximise its rewards. It may vary from a simple function to a complicated searching process. A reward function evaluates the perceived state of the system after reacting to the environment. Unlike supervised learning, reinforcement learning is unable to determine how good an action is immediate. Instead, the rewards of a single action accumulate over time. A long-running estimation, or a value function, is then required to calculate the expected total rewards gained by a learning agent from a specific state. A state whose value is yielded as low in a reward function may be marked high in a value function or vice versa. A value function can be used as a prediction of rewards in the future, but it is a secondary criterion in the learning system, compared with reward functions [114]. Formulating the environment, identifying the policy and the value function is essential while constructing a reinforcement learning model from scratch. As when the environment is opaque, or the proper interaction with the environment can not be defined, it is common to simulate natural systems when they design a reinforcement learning algorithm.

3.1 Natural Computation and Biological Computing

Natural computation is a computational system inspired by natural systems and use them as 1) computational media, 2) simulators to explore potential knowledge and 3) references of algorithm design (Fig 3.1) [14]. Thus, research on natural computation is more like an investigation into different approaches to process information [102]. Examples of natural computation include but are not limited to neural computation inspired by brain functions, evolutionary methods inspired by Darwin's evolution theory, artificial systems inspired by natural life and cellular automata inspired by intercellular communication.

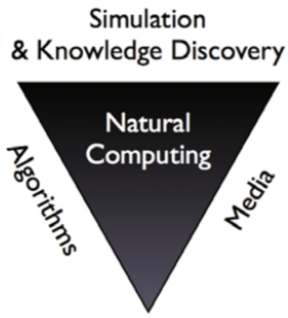


Figure 3.1: Three facets of natural computation [14]

Biological computing is a branch of natural computing. By simulating the process of organisms surviving from limited resources and predators, the multiple levels in a biological system suggest solutions to many problems, especially high-dimensional problems. A good simulation of a biological system has the following three characteristics: 1) a diverse population. Individuals in the population are not isolated but frequently exchange information about their current state to better adapt to the environment during the process of learning. Therefore diversity is considered the most important characteristic in setting up the population for a biological system. A population that can maintain diversity has a higher chance to survive in the environment, whereas a lack of diversity in the population could result in termination in local optima. 2) Survival is the primary objective rather than on optimisation. Individuals from the population can only be successful if they can adapt to the environment by taking advantages from limited resources whilst refrain predators. 3) The existence of a multilevel processor, such as the use of multiple levels and perhaps multi-timescales to stimulate individuals to evolve in multiple directions or adapt to a more complex environment [14].

A branch that uses the principles of natural evolution and genotypic variation to solve complicated optimisation problems in biological computing is called evolutionary computing. Although Barto et al. once criticised evolutionary computing for "ignoring much

of the useful structures of the reinforcement learning, overlooking individuals actions in their lifetime and neglecting the interaction with environments" in [9]. Recent research has proved that, with its development over multiple decades, modern evolutionary computing has improved its searching abilities and agility to meet different users' demands. It has become reliable and has even better exploration abilities than some popular reinforcement learning techniques, such as policy gradient methods. Moreover, evolutionary computing highly parallelise-able. Their run time significantly reduces when involving more agents to learn together [104][19]. Some algorithms in evolutionary computing are self-adaptable, which allows different strategies to work on different individuals in one generation simultaneously [102]. The relationship between evolutionary computing and reinforcement learning remains ambiguous. Barto and Sutton underlined the importance of policies on reinforcement learning. In their later work, the goal of reinforcement learning was sometimes identified as to solve sequential decision tasks by learning a policy that maps environmental states to particular actions [114]. In [9] they excluded evolutionary computing from reinforcement learning as it search for the best individual instead of the best policy of an individual to reach optima. However, they acknowledged that some evolutionary-style algorithms were applicable to maintain and evaluate a population of policies [114]. Discussing the relationship between evolutionary computing and reinforcement learning is beyond the scope of the thesis. If this issue is raised in the future, we shall call it evolutionary algorithms reinforcement learning methods, as what Moriaty et al. did in their paper [85].

Evolutionary computing can solve ill-structured global optimisation problems. Although they involve randomness in searching solutions, their overall efficiency is better than pure random search because they allow parameter control during the process and maintain good properties from candidate solutions [29]. The properties of evolutionary computing show potential as a solution to the problem of data synthesis. On the one hand, machine learning does not require accurate prior knowledge for the information base, which may not be available when dealing with a large and complex dataset. On the other hand, it can automatically eliminate inferior candidates in the solution space with user-determined objectives, which are also changeable during the procedure (also known as dynamic optimisation [15]). In computer science, evolutionary computation is a term used to describe a family of algorithms for global optimisation inspired by biological evolution. Its main algorithms include *evolutionary algorithms, evolution strategy, genetic programming and genetic algorithms*. This thesis applies a set of particular algorithms in evolutionary computing to data synthesis - genetic algorithms (GAs), which I shall now describe in more detail in the following sections. Compared with other algorithms in evolutionary computing, genetic algorithms simulate both crossover and mutation as reproduction strategies (whereas evolutionary algorithms and evolutionary strategies only deploy mutation) [25]

3.2 Genetic Algorithms (GAs)

GAs were first proposed in [62]. The algorithms search for the best solution by the following principles from natural selection [63][56]:

- The better the candidates adapt to the environment, the higher the chance they could survive and have offspring.
- The population becomes more adapted over time because only the fittest individuals can survive.

Compared with other algorithms, GA has particular features including parallel searching in the solution space, different selection method, random crossover and mutation operators, which offer more ways to increase the variation of candidates, thus resulting in a higher chance to find the global optimum from the searching space. However, there are as many failed applications in GAs as there are successful ones. Mitchell claimed that GAs could not solve all optimisation problems. He gave two conditions to check if GAs likely perform well in a specific problem. (1) The solution space of the problem must be large and ideally disconnected, i.e. there is no certain path for a search engine to explore the whole of the solution space in a limited time; otherwise, GAs will not be the most efficient solution. (2) The objectives are calculable and clear of errors; otherwise, the errors will be accumulated and be eventually irrecoverable [83] [1]. From the previous chapter, it is clear that data synthesis satisfies these conditions: (1) Even for a very modestly sized dataset, the solution space would have a non-practically large number of possible datasets as synthetic candidates. (2) The objectives in data synthesis are to minimise the divergence of the synthetic data from the original data while minimising its disclosure risks, both of which can be measured by precise functions.

A GA usually consists of the following components. They are briefly introduced here and will be thoroughly studied in the remaining chapter.

Candidates

A candidate refers to a single entity in the searching space of GAs. It was presented as binary strings, but a matrix format (to match the format of microdata) will be used in the thesis.

Population

[1] Although the conceptual definitions and functions of the objectives in the data synthesis application are clear, the operational choice of the objectives is complicated because there are many possibilities. Therefore, this in itself is an interesting research question. Nevertheless, once the decision was made the objectives are - in terms of their measurement properties - error-free.

GAs use the term 'population' to refer the set of candidates in the current generation. For clear expression, the population in the context of statistics will be called *statistical population* in this thesis instead. The initial population is the first N input solutions in the GA. There is no requirement on the size of the initial population, but apparently, it should be greater than 2 to make the following process operable.

Objectives/Fitness function

Objectives define the environment of a GA. The measurement of how close a candidate to the algorithm's objectives is known as fitness functions in GAs.

Selection

In a selection operator, fitter candidates have more chances to exchange (part of) information with others and produce offspring, whereas inferior ones are eliminated. The selected candidates form a temporary *pool* where the later process (crossover and mutation) occurs. This step intends to increase the likelihood of candidates in current generation passing on desired properties to the next generation.

Crossover

Candidates in the pool have a randomly chosen part of its information swapped with its paired candidate under a pre-determined crossover rate p_c in a crossover operator. This step reinforces the exploration powers of GAs and increases variation in the population.

Mutation

After crossover, newborn candidates enter the mutation operator and mutate under a mutation rate p_m. The existence of mutation ensures that candidates do not develops from a uniform population, but it does not always advance the search for optima. When candidates are sufficiently mature, a high rate of mutation will break up their goodness and prevent further evolution. Therefore, the mutation rate is usually low, and most of the variations in GAs are from crossover. [2].

Replacement

After crossover, the old population is replaced with the new generation. Parent candidates usually do not take part in the next generation or compete with new candidates. Although Wei suggested that involving the last generation in competition might lead to fast convergence to the global optima [125], in this thesis, I shall follow the orthodox way which replaces parents with their offspring.

GA is an iterative process amongst selection, crossover, mutation and replacement (see Fig 3.2). The success of a GA algorithm depends on the details added to its op-

[2]Note that it is possible for mutation rates to change during the course of a GA process. With a common approach is to lower the mutation rate as the optimal state nears. This is a complex decision-making process itself. More detailed discussion of this topic will be presented in section 3.8 and chapter 4

erator/parameter settings. Thus, there is ideally a specific GA for every problem, and constructing one by directly copying from others is not suggested [83]. The remainder of this chapter will explore all operators/parameters introduced above, including its mechanisms and conventional methods.

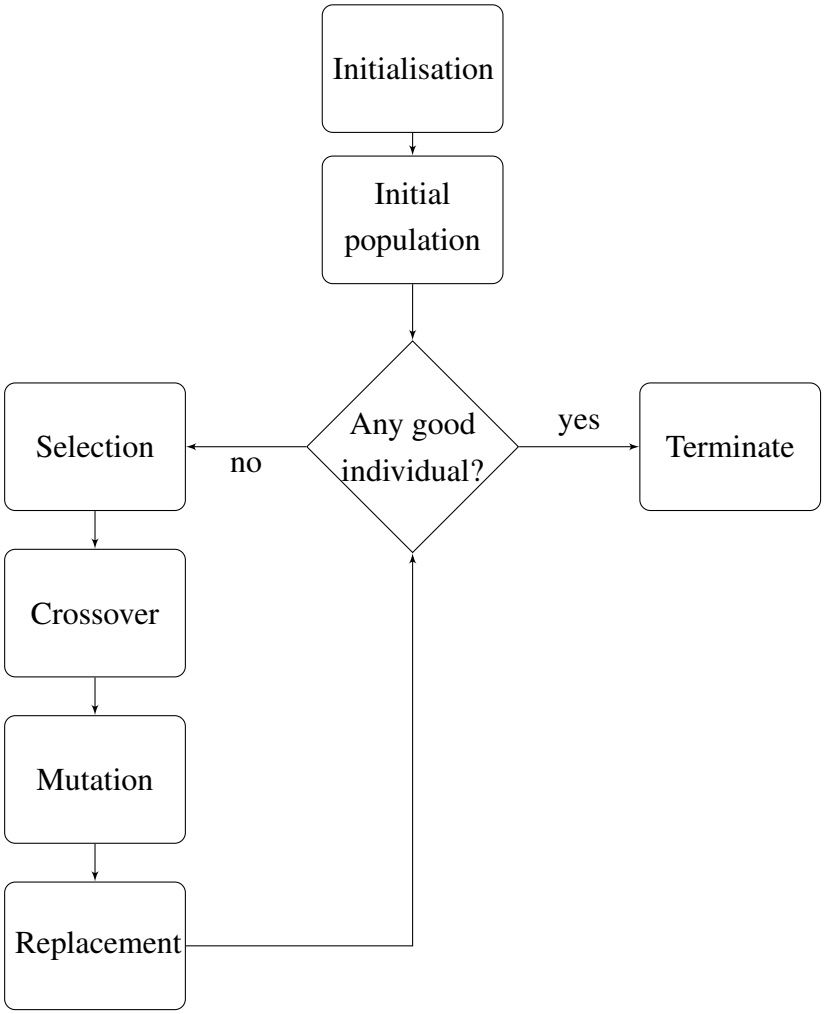


Figure 3.2: A flowchart of a GA

3.3 Initial Population

There are two characteristics which matter when designing the initial population for a GA: *size* and *diversity*. Gotshall and Rylander concluded that an arbitrarily large initial population might bring a large search space, but the impact becomes trivial once the population size exceeds a number. They clarified that the increase in population size also leads to an increase in the number of generations that converged [58]. As for diversity, most measures of population diversity are developed from binary-string GAs. For example, [70] defined the level of diversity as the number of non-identical candidates in the

population. [64] suggested calculating the number of equivalence classes (a class that a set of candidates in the population belongs to) as population diversity. Moreover, [126] proposed to use metrics, such as city-block distance and Euclidean distance, to measure distances between pairs of candidates in the population, which then enables to check the overall diversity of the population. Since the thesis use microdata as candidates, a different measure of population diversity should be applied and will be discussed later.

3.4 Selection

The selection operator aims to find fitter candidates and let them have offspring in the later process so that the next generation can carry their properties. Many studies have summarised selection methods in GAs, but I primarily refer to the technical report of Blickle and Thiele [13]. They thoroughly described how different selection methods work and how these affect the fitness distribution of the population in GAs. The framework of how to perform a selection operator is called a 'selection schema' in their work, but I decided to use the word 'method' instead in order to distinguish from the word 'schema' in 'schema theory'[3].

Suppose the number of different fitness values n is finite $f_1, ..., f_n$, $n \leq N$ in a current population of size N. Let $s(f_i)$ describe the number of frequencies in which the fitness value f_i appears. The fitness distribution of the current population and the selection method can be mathematically described as:

Definition 3.4.1 (Fitness distribution)**.** Suppose the function $s : \mathbb{R} -> \mathbb{Z}_0^+$ maps the fitness value $f \in \mathbb{R}$ to the number of candidates in the population that is under the same fitness value; then, s is the fitness distribution of the population.

Definition 3.4.2 (Selection)**.** A selection Ω is a function that transforms the fitness distribution s in the current generation to a new fitness distribution s^* with an optional list of parameters θ^k:

$$s^* = \Omega(s, \theta^k).$$

As most selection methods are probabilistic, their expected fitness distribution is defined as:

Definition 3.4.3 (Expected fitness distribution)**.** Let s^* denote the fitness distribution after applying Ω to the fitness distribution s; the expected fitness distribution $E(s^*)$ is

$$E(s^*) = E[\Omega(s, \theta^k)].$$

[3]The word 'schema' may still be used in some papers in Chapter Experiments and Results but it was clearly justified there.

As the fitness distribution is a discrete and takes values from $s(f_1), ..., s(f_{n-1}), s(f_n)$, $n \leq N$, the corresponding cumulative distribution is

Definition 3.4.4 (Cumulative (discrete) fitness distribution)**.** Suppose there are $n \leq N$ unique fitness values $f_1 < ... < f_{n-1} < f_n$ arranged in ascending order. Let $S(f_i)$ be the cumulative fitness distribution that outputs the number of candidates with a fitness value $\leq f_i$; then,

$$S(f_i) = \begin{cases} 0 & \text{if } i < 1 \\ \sum_{j=1}^{j=i} s(f_i) & \text{if } 1 \leq i \leq n \\ N & \text{if } i > n \end{cases}$$

$S(f)$ can be alternatively described as a cumulative *continuous* distribution given the range $f_0 < f \leq f_n$. The notations of $f_1, ..., f_{n-1}, f_n$ are retained as in the original case; $S(f)$ is re-written to a new continuous function $S'(f)$.

$$S'(f) = \int_{f_0}^{f_n} s(x)dx.$$

By these definitions, the mean and variance of population fitness can be calculated through:

Definition 3.4.5 (Average fitness)**.** Suppose the average fitness of the population is M:

$$M = \frac{1}{N}\sum_{f_0}^{f_n} f_i$$
$$M = \frac{1}{N}S'(f)$$

Definition 3.4.6 (Fitness variance)**.** Let σ^2 denote the fitness variance of the fitness distribution $s(f)$:

$$\sigma^2 = \frac{1}{N}\sum_{f_0}^{f_n} \frac{(f_i - M)^2}{N}$$
$$\sigma^2 = \frac{1}{N}\int_{f_0}^{f_n} s(f)(f - M)^2 df$$

Reproduction rate, loss of diversity, selection pressure and selection variance are the four properties that Blickle and Thiele believed to sufficiently measure how powerful a selection method is. These properties can be calculated through either a cumulative discrete distribution $S(f)$ or a cumulative continuous distribution $S'(f)$ of fitness values in the population. For most of the calculations, Blickle and Thiele provided both versions [13]. This thesis mainly focuses on the discrete distribution because the populations used for data synthesis will not be substantial ($N \leq 100$) in the later experiments and whether its fitness values can fit in any continuous distribution is unknown.

Reproduction rate computes the ratio of the number of candidates that have the same fitness value before and after selection. A selection has a proper method if its reproduction rate increases monotonically in f in maximisation problems and decreases monotonically in f in minimisation problems.

Definition 3.4.7 (Reproduction rate)**.** Suppose $R(f)$ denotes the reproduction rate of a particular fitness value f. Recall that $s(f)$ is the discrete fitness distribution of the population before selection, and $s^*(f)$ is the one after.

$$R(f) = \frac{s^*(f)}{s(f)} \text{ if } s(f) \neq 0$$

Loss of diversity L is the proportion of candidates in the population that is not selected by the selection operator. L should be sufficiently low; otherwise, there is a risk of premature convergence.

Definition 3.4.8 (Loss of Diversity)**.** For a proper method, given f_z as the fitness value that has the reproduction rate $R(f_z) = 1$,

$$L = \frac{1}{N}(S'(f_z) - S'^*(f_z)).$$

Selection pressure measures the degree to which a better candidate is acceptable. It was confirmed that the higher the selection pressure, the operator chooses more good candidates into the pool, and the quicker the GA converges [84].

Definition 3.4.9 (Selection Pressure)**.** The selection pressure I of a particular selection method Ω is defined as

$$I = \frac{M^* - M}{\sigma},$$

, where M and M^* are average fitness of populations before and after selection, and σ is the standard deviation of the previous population.

Selection variance determines the change in variance in the population after selection, which is usually used for comparing methods together with selection pressure.

Definition 3.4.10 (Selection variance)**.** The selection variance V is the ratio between the variance σ^* of the expected fitness distribution $E(s^*(f))$ of the population after selection and the variance of the fitness distribution $s(f)$ before selection.

$$V = \frac{\sigma^{*2}}{\sigma^2}$$

Selection controls the capability of a GA to retain good candidates and how efficient that process can be. Elitism is a characteristic in selection operators which prompts GAs

to reach optima. It aims to retain as much goodness from the fittest candidates as possible in the next generation. For example, Villalobos-Arias et al. used to moved a proportion of fittest candidates into an extra set and isolated them from the evolution process. Then they checked if there was a new candidate that dominates these candidates' fitness after each generation [123]. Wei proposed to involve all candidates that ever appeared into the competition to ensure that fitter candidates have had a chance to revive during the process [125]. However, elitism likely stuck the process into local optima due to the loss of diversity. The impact of elitism to a GA's efficiency is not yet clear. Thus I shall only compare common selection methods through the previous four properties in the thesis.

3.4.1 Fitness Proportional Selection

The probability that a candidate is selected is proportional to its fitness value in this method. Recall that M denotes the average fitness of the current population $= \frac{1}{N}\sum f_i$. The probability that an individual is selected is [26]:

$$p_i = \frac{f_i}{NM}, \tag{3.1}$$

and the expected fitness distribution after selection is [13]:

$$E(\Omega_P(f_i)) = s(f)\frac{f_i}{M}.$$

Roulette wheel selection (RWS) and stochastic universal sampling (SUS) are two general approaches in this method. RWS spreads the chance of selection to every candidate based on their fitness values. It simulates a roulette wheel with one pointer and a plate that the area of each candidate is given by dividing its fitness by the total fitness. SUS modifies the RWS: instead of spinning the roulette wheel N times, it spins once with N evenly distributed pointers [57].

3.4.2 Tournament Selection

In tournament selection, candidates are randomly selected with replacement and compete in a tournament with other t candidates. They are compared in terms of fitness, and only the winners will be selected [26]. According to Blickle and Thiele, the expected fitness distribution $\Omega_T(S,t)$ after tournament selection depends on the size of tournament t and the cumulative fitness distribution $S(f)$. Suppose f_{i-1} is the previous candidate of f_i, the expected fitness distribution is:

$$E(\Omega_T(S,t)(f_i)) = N(\frac{S(f_i)^t}{N} - \frac{s(f_{i-1})^t}{N})$$

3.4.3 Truncation Selection

Truncation selection only chooses a certain proportion of candidates in each generation. Given a threshold value T, $T \in (0,1)$, only $T\%$ best candidates from the current population can participate in crossover. The expected fitness distribution of this selection method is

$$E(\Omega_\tau(s,T)) = \frac{S(f_i) - (1-T)N}{T}, \text{ if } S(f_{i-1}) \leq (1-T)N \leq S(f_i).$$

With this selection method, the average fitness of the population increases while the previous fitness distribution $S(f)$ (or $S'(f)$ in the continuous version) is greater than $(1-T)N$.

3.4.4 Ranking Selection

In ranking selection, all candidates are first rearranged in ascending order by their fitness values (they should be assigned to different ranks even though they have the same fitness value) [3]. Mitchell believed that ranking selection could prevent premature convergence by reducing selection pressure and increasing fitness variance [83]. There are two ways to perform ranking selection: linear ranking selection and exponential ranking selection. In linear ranking selection, users need to specify a reproduction rate for the candidate with the highest rank (the best candidate) N as η^+ and the rate for the worst candidate 1 as η^-[4]. The expected fitness value $E(f_i)$ of the i-th candidate is given by:

$$E(f_i) = \eta^- + (\eta^+ - \eta^-)\frac{i-1}{N-1}$$

Therefore, the probability of the best individual to be selected is $\frac{\eta^+}{N}$, and for the worst individual, it is $\frac{\eta^-}{N}$. Let $p(i)$ present the probability that the i-th candidate is selected; $p(i)$ is linearly assigned to their ranks [13].

$$p(f_i) = \frac{1}{N}E(f_i) \tag{3.2}$$

When the population size is constant, $\eta^- \geq 0$ and $\eta^+ = 2 - \eta^-$ must be fulfilled to guarantee that the sum of probabilities for all candidates is 1. Given that η^- is predetermined, the expected fitness distribution after linear ranking selection is

$$E(\Omega_R(s,\eta^-)(f_i)) = s(f_i)\frac{(N\eta^- 1)}{N-1} + \frac{1-\eta^-}{N-1}(S(f_i)^2 - S(f_{i-1})^2)$$

As for exponential ranking selection, the probabilities that candidates are selected exponentially proportional to their ranks. A parameter called *exponent* $0 < c < 1$ is required

[4]The numbers will be altered to $N-1$ and 0 in program codes in order to adapt with Python, which is 0-indexed

and usually determined by the in-population variance. In a population with size N, the probability that i-th candidate is selected is:

$$p_i = \frac{c-1}{c^N-1} c^{N-i} \tag{3.3}$$

The constant coefficient $\frac{c-1}{c^N-1}$ guarantees $\sum_i^N p_i = 1$. The corresponding expected fitness distribution after exponential ranking selection is

$$E(\Omega_E(s,c,N)(f_i)) = N\frac{c^N}{c^N-1} c^{-S(f_i)}(c^{s(f_i)}-1).$$

3.5 Schema Theory

When Holland first proposed the idea of GAs, he described the work of GAs as parallel discovering, emphasising and recombining good *building blocks* of individuals in the population. He then formalised the notions of such building blocks as schema [62] [83]. Schema theory is well applicable on GAs but its formula so far only works on primitive settings that linear, binary and fixed-length candidates, fitness proportional selection, single-point crossover and single-point mutation.

In the GA context, a schemata is a template consisting of ones, zeros and asterisks, in which asterisks represent wildcards. For example, consider the schemata $H = 1**0*$: any binary string that has 1 and 0 at the first and fourth positions is called an instance of H. The number of fixed bits is the schemata's order, $o(H)$, and the distance between the first and last fixed bits is the schemata's distance, $\delta(H)$ (in the example, $o(H) = 2$ and $\delta(H) = 4-1 = 3$). By involving Schema Theory, the evaluation of the overall fitness of a population can be re-considered to implicitly evaluate the average fitness values of any instances of a schemata in it. Schema Theory demonstrates that the number of instances from schema with low order and short length but whose average fitness is greater than the mean fitness value increases exponentially during the process. Assume that the number of instances of H at time t is $m(H,t+1)$, $\hat{u}(H,t)$ is the average fitness observed at time t, $\bar{f}(t)$ is the average fitness of the population at time t, and p_c and p_m are the crossover probabilities of a single-point crossover and the mutation probability of a single-point mutation, respectively. The expected number of instances of H of the next generation $E(m(H,t+1))$ is [83]

$$E(m(H,t+1)) \geq \frac{\hat{u}(H,t)}{\bar{f}(t)} m(H,t)(1-p_c\frac{\delta(H)}{l-1}[(1-p_m)^{o(H)}]).$$

Schema theory was argued limited in usage as it is only valid in certain types of GAs, and Goldberg and Jon also questioned if the formula is practical, as its application is based

on an infinite population, it might not always work on finite populations due to sampling error from initial populations [56].

Inspired by Schema theory, Holland develop another hypothesis that GA works well when instances of low-order, short schema with high fitness would be recombined to instances of high-order, long schema that confers even higher fitness, which is called the *building block hypothesis* [62]. However, this hypothesis was soon overturned by Forrest and Mitchell, who indicated that building blocks have disruptive effects. They highly likely cause premature convergence when two or more non-overlapping schema appear simultaneously in the population [52]. Moreover, Baum et al. showed that building blocks did not make GAs exceed other searching algorithms in some problem, which confirms that there is no explicit connection between the hypothesis and the effectiveness of GAs [10]. Forrest and Mitchell also stated that, although the hypothesis was proven in later experiments to have no positive effects on GAs. GAs indeed learn or search by building blocks as the process is to combine lower-order schema with higher-order ones with higher fitness values and eventually take over the entire population. However, the effectiveness of this process depends on the landscape of the solution space in the particular problem [52]. There is no study that if building block hypothesis still holds when candidates in GAs become multi-dimensional or in data synthesis, and this will be investigated later this thesis.

3.6 Crossover

In binary GAs, crossover takes place on random strings with a user-determined probability (crossover rate p_c). It is the major operator that provides variation to the population [111][83]. The simplest crossover method is a single-point crossover, which exchanges (all) bits after a random position between two paired strings. Reeves concluded that a single-point crossover likely broke higher-ordered schema [97]. Mitchell also confirmed that a single-point crossover has limited ability to generate new schema. For example, it is unable to combine $11*****1$ and $****11**$ to a new form of $11**11*1$ [83]. The situation that the disruption and recombination of schema depends on the position of the bit in the string is called positional bias. A single-point crossover has been proven to have the maximum positional bias [116]. An alternative that reduces positional bias is a two-point crossover, which is defined as [56]:

Definition 3.6.1 (two-point crossover)**.** Suppose any 2 strings have length l; an integer position k along the string is chosen randomly between $[1, l-1]$. Then, two new strings are created by swapping all characters between $k+1$ and l from the original strings.

Compared with a single-point crossover, a two-point crossover can disrupt and re-

combine higher-ordered schema, but it does not eliminate positional bias. Researchers therefore proposed Parameterised uniform crossover (PUC) that contain no positional bias. PUC exchanges each bit of the string by a user-determined probability p_0, and it disrupts all schema equally regardless of their order and length. Spear and De Jong praised PUC for the following reasons: (1) It has no positional bias. (2) It has an adjustable parameter p_0, so the force of disruption and recombination is under the user's control; for example, PUC can be less disruptive than a two-point crossover when $p_0 = 0.1$. (3) PUC explores the solution space more powerfully than a two-point crossover. However, Spear and De Jong did not adequately discuss the shortage of PUC [111]. Sirivas and Patraik indicated that although PUC has no positional bias, its disruption power to schema is higher than those of the single and two-point crossovers [116]. Moreover, statement (2) is proved based on an extreme case in which one parent has only 1s and another one has only 0s [111]. Meanwhile, Mitchell indicated that the statement (3) does not hold if the solution space in GA contains disconnected sub-spaces and PUC is no better than a two-point crossover in this situation in [84]. Since GAs are parallel-able, both two-point crossover and PUC can be adapted to a matrix-format candidate with parallel sub-processors, but they are not the only two crossover methods considered in the thesis. A set of suitable crossover methods for a GA data synthesiser will be described in the following chapter.

3.7 Mutation

Mutation can prevent the process of GAs from pre-mature convergence. It may perform using the same mechanism as the crossover method like a single-point, two-point or parametric uniform crossover. Depending on the method the operator adopts, positional bias also exists in mutation. However, since mutation is the secondary variation provider to the population compared with crossover, its rates (p_m), especially the adaptability in its rates, are mainly studied in the thesis.

3.8 Adaptive GAs

GAs require pre-setting parameters like the population size, the crossover and mutation rates. They can be fixed or adaptive depending on the complexity of the problem. GAs with adaptive parameters are also known as adaptive GAs. Since GAs are sensitive to parameters, controlling their values allows timely adjustments to the search process. [41] confirmed that adaptive GAs can improve the performance - when compared with fixed-parameter GAs - by:

- maintaining population diversity.
- improving the speed of convergence as well as preventing premature convergence.

Korejo et al. sorted adaptive GAs into deterministic adaption, adaptive adaption and self-adaption. Deterministic adaption tunes the parameters according to some deterministic rules. Adaptive adaption changes the parameters executing the model so that users can get feedback from the searching space [69]. Since the searching space depends on the original data in GAs, I shall only explore self-adaption in GA synthesisers in the thesis. Self-adaption is a set of functions equipped in GAs to enable the model to change parameters by itself. Thierens classified approaches of self-adaption into dynamic parameter control and adaptive parameter control. The former tunes parameters according to the number of generations or the convergence level of the current population, so the parameters for all candidates in the same generation are equal. The later adapts parameters according to individual fitness, in which candidates have different parameters in the same generation [121]. As there is no research on adaptive GAs in data applications or on matrix GAs, the optimum choice of adaptation method is not well defined for this application. Therefore, in the thesis, I shall be exploring the potential of using self-adaptive parameters in GA data synthesisers using simple self-adaptive functions.

3.9 Multi-objective GAs

Objectives define optimal solutions and how the solutions are measured (to qualify to be optimal). For a real-world problem, there is invariably more than one objective to be considered, and they sometimes contradict one another in that they have distinct optimisation goals and possess distinct search spaces (including in this application the utility and risk objectives of data synthesis). Fig 3.3 summarises common method used in solving MOOPs.

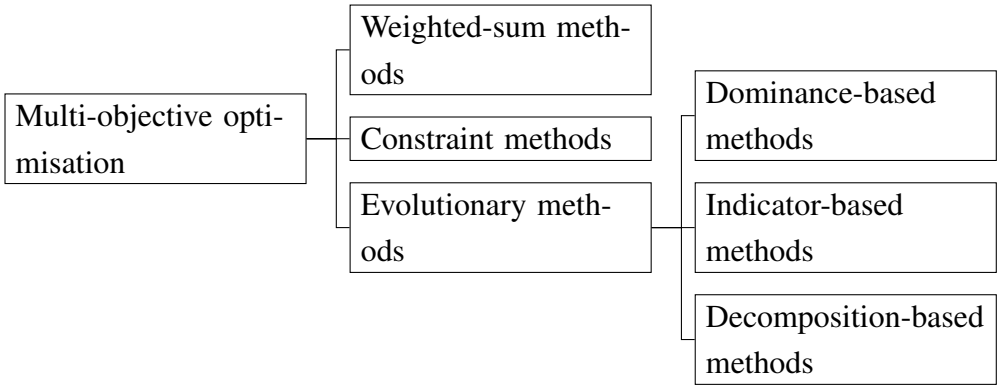


Figure 3.3: Classification of multi-optimisation methods [32]

One of the classical approaches is to compose these objectives into a single function with determined weights (commonly, these objectives must be normalised before being formulated into a single function to avoid an objective with a larger numerical variance dominating the less-varied one). However, the selection of weights is easy to get wrong, and even a small difference in weighting can lead to different outcomes. Alternatively, the constraint method can be used to alleviate the difficulties in the weighted-sum method. It suggests simplifying multi-objective optimisation problems (aka MOOPs) by optimising one of the objectives while setting constraints to the others with pre-determined values [32]. Similar to the weighted-sum method, it depends on users' preference for the selection of constraints and objectives and could return different solutions to the same problem.

As simultaneous optimisation is usually unlikely in MOOPs, it is impossible to get the ideal solution. Alternatively we may aim to look for a set of Pareto-optimal solutions instead. Pareto solutions refer to a set of solutions that are non-dominated by (in other words, no better or worse than) one another, and there is always a certain amount of sacrifice in some objectives to achieve a certain level of the other objectives whilst moving the Pareto solution from one solution to another [75][68]. The concept of 'dominance' is defined as:

Definition 3.9.1 (Dominance)**.** In a MOOP, a solution $\mathbf{x}^{(\mathbf{1})}$ is said to dominant the other solution $\mathbf{x}^{(\mathbf{2})}$ if:

1. $\mathbf{x}^{(1)}$ is no worse than $\mathbf{x}^{(2)}$ in all objectives.
2. $\mathbf{x}^{(1)}$ is strictly better than $\mathbf{x}^{(2)}$ in at least one objective.

Pareto solutions for a MOOP are usually part of the frontier of its solution space. It is believed that evolutionary computing, known for its exploratory power, has great potential in finding them. GAs, as an important branch in evolutionary computing, have many algorithms developed for solving MOOPs (see [68][106][86][75]). Since many of them were designed in 80s and 90s which might be obsolete, in this section I shall only talk about those still used nowadays and related to the three methods.

3.9.1 Dominance-based Methods

The commonalities of dominance-based GAs are: (1) they follow the principle that fitter candidates have more chance to survive and, (2) they use an explicit fitness function to measure population diversity [32].

3.9.1.1 Niched Pareto GAs (NPGA)

Horn et al. proposed an alternative selection method and a fitness evaluation function for GAs in order to search for Pareto-optimal solutions [64]. The selection operator in NPGAs has two parts: Pareto domination tournaments and non-domination tournaments. The first part helps find dominantly optimal candidates, and the second one maintains diversity in the population during the process. Pareto domination tournaments compare two randomly selected candidates with a comparison set of random individuals from the same population; if one candidate dominates and the other does not, the dominant one enters the following process. If none or both dominate the comparison set compared with, then the two enter the non-dominant tournament. The winner in non-dominant tournaments is the one that has the least number of individuals in its niche for diversity purpose. This strategy is known as *equivalence class sharing*.

3.9.1.2 Non-dominated Sorting GAs-II (NSGA-II)

The idea of the original version of NSGA combined ranking selection method to emphasise good candidates and a niche method to maintain population diversity. The algorithm first ranks (sorts) non-dominant candidates in the population then assigns them dummy fitness to compete for [117]. Due to computational inefficiency in performing non-dominated sorting to all candidates in the population, Deb et al. then adjusted the sorting algorithm and involved parents into competition [31][32]. Once the sorting is done, a niche strategy called crowding-sorting is deployed to select non-dominant candidates with less surrounding neighbours [32], which resembles equivalence class sharing.

3.9.2 Indicator-based Methods

Indicator-based evolutionary algorithms (IBEA) - which are also implementable in GAs - significantly alter the measurement of fitness in a MOOP. Ziztler and Künzli introduced the concept of the indicator I, which is used to preserve dominance among candidates while transferring their fitness to measurable values [129].

Definition 3.9.2 (Indicator)**.** $I : \Omega \times \Omega : \mathbb{R}$ is an indicator if for candidates $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}$ in the population:

1. $\mathbf{x}^{(1)} \succ \mathbf{x}^{(2)} \implies I(\{\mathbf{x}^{(1)}\}, \{\mathbf{x}^{(2)}\}) < I(\{\mathbf{x}^{(2)}\}, \{\mathbf{x}^{(1)}\})$
2. $\mathbf{x}^{(1)} \succ \mathbf{x}^{(2)} \implies I(\{\mathbf{x}^{(3)}\}, \{\mathbf{x}^{(1)}\}) < I(\{\mathbf{x}^{(3)}\}, \{\mathbf{x}^{(2)}\})$

For example, one way in [129] is to use a pre-determined parameter ε to give the minimum distance by which a Pareto set approximation can be translated in each front (for example, B) in the objective space such that another approximation in a front (for example, A) is weakly dominated.

$$I_\varepsilon(A,B) = min\{\forall \mathbf{x}^{(2)} \in B \exists \mathbf{x}^{(1)} \in A : \mathbf{x}_i^{(1)} - \varepsilon \leq \mathbf{x}_i^{(2)} \text{ for each objective } i\}$$

3.9.3 Decomposition-based Methods

The algorithm most representative of decomposition-based methods is MOEA/D (decomposition-based multi-objective evolutionary algorithms). Thanks to the parallel-processing feature in evolutionary algorithms, it decomposes a MOOP into a set of scalar-objective sub-problems (with sub-populations) and solves them simultaneously. MOEA/D assumes that for each Pareto solution there exists a weight vector such that: the solution is the optimal solution of a sub-problem in the MOOP and optimal solutions of all sub-problems are Pareto solutions of the MOOP [127]. Zhang introduced the Tchebycheff approach to decompose a MOOP. This involves an external population which serves as the mid-transit point where the selected candidates from sub-populations were preserved and returned for reproducing. This mechanism enables information exchange between sub-populations [127] [128].

3.10 Matrix Real-Coded GAs

Although GAs were initially designed to operate on binary-coded strings, they can be applied to higher-dimensional, real-coded problems. Matrix real-coded GAs (MRCGAs) are believed to reflect more phoneme structures of problems. It has higher exploration power and optimality in complex problems. The application of MRCGAs showed remarkable agility in algorithm and operator design, and it can simplify a complicated problem to a single-level optimisation problem [124][112][94].

Sun et al. adopted MRCGA to solve a thermal unit commitment problem: a mixed-integer non-linear programming problem consisting of discrete and continuous variables. They represented all information in a candidate as 3.4.

$$G_k = [V_1, V_2, \cdots, V_t, \cdots, V_T] = \begin{bmatrix} R_1 \\ R_2 \\ \vdots \\ R_i \\ \vdots \\ R_N \end{bmatrix} = \begin{array}{c} \begin{matrix} 1 & 2 & \cdots & t & \cdots & T \end{matrix} \\ \begin{bmatrix} C_{1,1} & C_{1,2} & \cdots & C_{1,t} & \cdots & C_{1,T} \\ C_{2,1} & C_{2,2} & \cdots & C_{2,t} & \cdots & C_{2,T} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ C_{i,1} & C_{i,2} & \cdots & C_{i,t} & \cdots & C_{i,T} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ C_{N,1} & C_{N,2} & \cdots & C_{N,t} & \cdots & C_{N,T} \end{bmatrix} \end{array} \begin{matrix} 1 \\ 2 \\ \vdots \\ i \\ \vdots \\ N \end{matrix}$$

Figure 3.4: Multi-Matrix Candidate Representation I [112]

In their work crossover and mutation occurred on a random vector of parents and that the portion of elements they swap is pre-determined [112]. MRCGA had shown an improvement compared with results from other algorithms and it was therefore adjusted (for this very problem later) by adapting operators [92] or amending the way to encode candidates [7] [101]. For example, in [92] the authors noticed that 'the ends of the chromosome tend to remain unaltered during crossover' (aka positional bias) in the original design. They therefore renewed the crossover method to 'annular crossover', in where each vector in the candidate is represented as a ring 3.5.

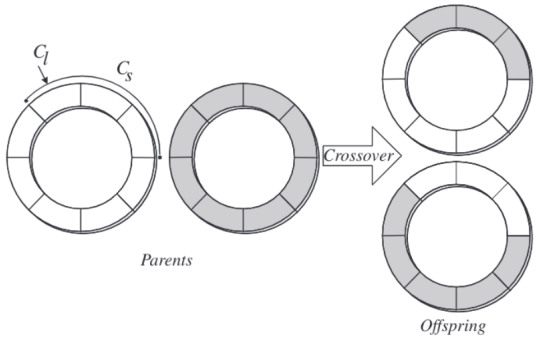


Figure 3.5: Vectors of candidates are represented as rings in annular crossover [92]

Moreover, a new 'random key' used to generate candidates in the initial population was proposed in [101]. It allows the algorithm works on only feasible solutions in order to improve the effectiveness and efficiency of the whole process[5].

Pongcharoen et al. used MRCGA to optimise the stage transportation problem in a logistic chain [94]. In their research, the candidates in GA are represented as multiple matrices (Fig 3.6). The crossover mechanism they adopted is less stochastic; it either swaps the entire sub-matrix randomly selected between parents, or swaps selected elements between a randomly selected sub-matrix.

[5]The authors did not mention whether and how the new approach affects population diversity

$$M_{IxJ}M_{JxK}M_{KxL}=\begin{bmatrix} x_{11}\ x_{12}\ldots x_{1J} \\ x_{21}\ x_{22}\ldots x_{2J} \\ :\ :\ \ldots\ : \\ x_{I1}\ x_{I2}\ldots x_{IJ} \end{bmatrix}\begin{bmatrix} x_{11}\ x_{12}\ldots x_{1K} \\ x_{21}\ x_{22}\ldots x_{2K} \\ :\ :\ \ldots\ : \\ x_{J1}\ x_{J2}\ldots x_{JK} \end{bmatrix}\begin{bmatrix} x_{11}\ x_{12}\ldots x_{1L} \\ x_{21}\ x_{22}\ldots x_{2L} \\ :\ :\ \ldots\ : \\ x_{K1}\ x_{K2}\ldots x_{KL} \end{bmatrix}$$

Figure 3.6: Multi-Matrix Candidate Representation II [94]

Wallet et al. first proposed that instead of swapping elements in a linear structure, parents could swap a rectangular sub-matrices (matrix crossover).

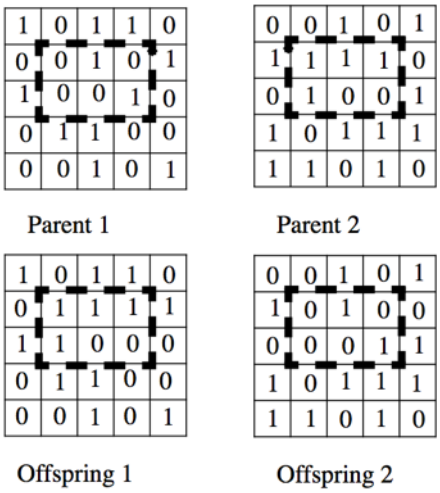


Figure 3.7: An example of matrix crossover

They also found that matrix crossover has more symmetry compared with a linear crossover in the disruption power because a rotation of the matrix representation of the candidates has no effects on the efficiency of the algorithms. In their experiments, matrix crossover performs better than the other when the candidate matrix is small (10×10). However, their performances became similar when the candidates' matrix size increased to 20×20 [124].

3.11 Chapter Summary

This chapter narrated the relations between GAs, natural computing and machine learning. It gave background knowledge for GAs by exploring details in conventional settings used in different components and introducing three methods of GAs to find solutions for an MOOP. All will be used to support the model design of the GA synthesiser in the next chapter.

The MRCGA is believed to be an adaptable variant in GAs for generating synthetic data. The abundant literature only gives some outlines about the approach and its possible applications in some problems. The applications of MRCGAs to data synthesis is not investigated yet and their efficiency compared with orthodox synthesisers is not known. In the next chapter, I shall discuss how to design a GA that is specifically used for generating synthetic data.

Chapter 4

Model Design

Navarro-Arribas and Torra confirmed the potential of GAs for privacy protection. They described data privacy as an optimisation problem with two opposing constraints: information loss and disclosure risk; and proposed using GAs to improve privacy protection of databases and/or divergence from the original data in parameters in statistical models [88]. Moreover, as stated in the previous chapter, the three conditions that Mitchell described as an ideal solution space for GAs apply to this use case [83].

1. The solution space of the problem is sufficiently large. This condition applies to data synthesis because there will be many comparable datasets generated in the process.

2. The solution space is not smooth, and there is no possible way to evaluate the gradients or track the trace of optima. This is true for data synthesis because the solution space contains practically infinite datasets that are measured by multiple fitness functions that we are unable to picture its solution space.

3. The objectives can be calculated through clear and quantitative formulae. This also holds because the original dataset provides very clear inferences to the synthetic data and there are various privacy models available for measuring the risk.

Since a single SDC approach should only be used on one type of data [34] and records in census and social survey data is usually presented in categorical format. The type of data that will be used in the thesis is single-level, categorical microdata. However, as GAs use a different mechanism from other synthesisers, there might be potentials of using GAs in continuous and even mixed data synthesis. The issue will be discussed in the last chapter: Impacts and Critical Analysis of the Model later.

The thesis will use full synthesis as the SDC approach. Although, as indicated in the

previous chapter, full synthesis cost more computational workload than partial synthesis, there are reasons to choose it. First, since there exists no universal rule of personal privacy, respondents may define their confidential variables differently from data controllers. For example, some respondents may not want to disclose how many cars they owned but the number of cars is usually treated as an innocuous variable in data protection [44]. Second, full synthesis does not contain real values of any variable theoretically, which reduces interest from data intruders because they are less likely to derive true information from the released dataset. Adopting full synthesis simplifies the process of selecting variables for synthetic data and allows more focus on utility and risk over the whole dataset. Although full synthesis eliminates disclosure risk and reduces intruders' interests to the dataset, its utility more depends on the data generating model compared with partial synthesis. Therefore the model ought to retain all possible statistical properties in all variables in order to promise the validity of substantive analyses [37].

MRCGA is used to synthesise data in the thesis as its format of candidates can more appropriately display the multivariate structure of microdata [124]. Although Holland once argued that real-code candidates exhibit poorer performance than binary code, It is unreal to expect that binary strings represent the complicated information contained in a multivariate dataset, and Mitchell disproved his statement due to a lack of evidence [84].

Issues to be addressed in model design for the GA used for data synthesis (aka the GA synthesiser in the following contents) include:

1. Although allowing users to set parameters increases a model's agility, it may be not user-friendly especially for users who have no background knowledge of GAs or the problem the GA is used to deal with [110]. Therefore, selecting appropriate operators and parameter tuning is important when designing a GA. In the sections and the following chapter, I shall set up different trials to identify suitable settings of operators and parameters for the synthesiser.

2. GA's efficiency for a particular problem also depends on its operators/parameters, which will be investigated in the following content.

3. The objectives of the GA for generating synthetic data will only take the minimal set of all potential objectives. For example, the utility objectives of the GA can be all the statistical properties of the original dataset. However, if two statistical properties are highly related, then the model only takes one into account.

The four major components in GAs were introduced in the last chapter: the initial population, selection, crossover and mutation. The ideal process for GA synthesiser is; first, users input a one-level, categorical microdata as the original data into the synthesiser and

give the size of the population N. The synthesiser then generates N synthetic datasets (candidates) as the initial population using a simple method, for example, sampling from univariate distributions of variables in the original data. These candidates are evaluated in the selection operator by fitness functions. Selected candidates enter the pool where crossover and mutation with a given crossover and mutation rates occur. They then form the new generation, and their parents are eliminated. The GA synthesiser repeats the above steps until there is at least one candidate in the population that reaches an acceptable level of fitness.

4.1 Initial Population

Diversity is an important feature in populations in GAs. A population with higher diversity is more likely to jump out of local optima and to converge to an optimal solution [14]. However, the challenge of applying GA in data synthesis is that no clear standard can be adopted for measuring the diversity in sets of datasets. It is probably true that populations generated from different statistical approaches will have a different level of diversity. It may be the case that the initial population with the higher diversity will be generated from a 'freer' model, like sampling from uniform or univariate distributions of all variables from original data. On the contrary, a more strict model, like CART or other known synthesisers, is likely to produce more similar candidates to the initial population.

On the other hand, the fitness of the initial population may also impact on the efficiency of the whole GA process. The initial population can be generated from some observed statistical properties of the original data such as means, frequencies, equivalence class structures, covariance or Chi-squared statistics [38]. Generating initial populations that have those properties will reduce the running time of the whole process compared with pure random data because properties embedded in the initial population can be retained during the process of the GA synthesiser unless it objectives are mistakenly set.

Although size is another main characteristic in populations, it is not easy to set a proper value for it because there is no stable relationship found between the size and the efficiency of GAs (not like selection or crossover). In [58] Gotshall and Rylander claimed that a large initial population might bring a more extensive search space, but they also indicated that a large initial population might lead to an increase in the number of generations that converged. Moreover, the impact from the population size to the efficiency of GA synthesisers may vary from one dataset to another, or different operator/parameter settings used (It can be checked from the following content of the thesis in which the number of generations that converged decreased as the initial population size increased when using a specific selection method in the GA synthesiser). Last but not least, the

GA synthesiser adopts datasets as candidates; it would massively increase the computational workload using a large population size. Therefore, in the later experiments, 100 will be the number set as the size of initial populations, in order to limit the cost of time in running the synthesiser.

4.2 Selection Methods

In the previous chapter, I summarised some common selection methods used in GAs in table 4.1. These methods are classified into two categories based on their mechanisms: value-proportional methods and ranking-proportional methods. Value-proportional methods select candidates according to their fitness values; the chance that a candidate can survive and has its offspring is proportional to its fitness. Ranking-proportional methods select candidates based on the rank of their values; the one in higher rank is more likely to survive[1].

Value-proportional selection schemes	Ranking-proportional selection schemes
Roulette Wheel Selection	Tournament selection
Stochastic Universal Selection	Linear ranking selection
	Exponential ranking selection

Table 4.1: Common selection methods in GAs

As there is no decision in the previous chapter on which methods will work best for this use case, I shall carry out both theoretical and experimental assessments of the common selection methods in the remaining section.

4.2.1 Theoretical Comparison of Selection Methods

Reproduction rate (Def: 3.4.7), loss of diversity L (Def: 3.4.8), selection pressure I (Def: 3.4.9), and selection variance V (Def: 3.4.10) are four properties used to examine selection methods in binary GAs. Reproduction rate tells whether a selection method is favouring fitter candidates. Generally speaking, a reasonable method should have at least 1 as the reproduction rate of good candidates and at most 1 for bad ones. It is unable to tell how much elitism the method has because there is no explicit boundary between good and bad candidates in a generation. So it will not be used for comparison in this section. Loss

[1]Truncation selection belongs to none of these categories, and it is less probabilistic but more depends on users' setting on the proportion of selected candidates. Therefore it will not be discussed in the thesis any more

of diversity gives the proportion of candidates that not selected by the selection operator, which is calculated by the corresponding reproduction rate. Selection pressure is the most frequently used property [3] [26]. It measures the change in the average fitness of the current generation before and after selection. Similar to selection pressure, selection variance computes the ratio between fitness variance before and after selection. Compared with loss of diversity, selection variance takes account fitness values of both emerging and eliminated candidates into account, which is suitable for changing populations.

Blickle and Thiele assumed that fitness values of the population were sampled from standardised Gaussian distribution $G(0,1)$. They gave a comparison of the selection methods in [13] and some of the values of the three properties in different selection method with a reasonable selection of parameters were recorded in Table 4.2.

Selection Method	Loss of Diversity L	Selection Pressure I	Selection Variance V
Linear ranking $(\eta^-1 = 0.0, \eta^+ = 2.0)$	0.25	0.5642	0.6817
Linear ranking $(\eta^-1 = 0.3, \eta^+ = 1.7)$	0.175	0.3949	0.844
Exponential ranking $(c = 0.5, N = 100)$	0.9244	N/A	N/A
Exponential ranking $(c = 0.9, N = 100)$	0.6816	N/A	N/A
Tournament $(t = 2)$	0.25	0.5642	0.6817
Tournament $(t = 3)$	0.3849	0.8463	N/A
Roulette Wheel	N/A	N/A	N/A
Stochastic Universal	N/A	N/A	N/A

Table 4.2: L, I and V in different selection methods with a reasonable choice of their parameters. N/A indicates that there is no analytic method to calculate the particular property or not given in the work.

Unfortunately, their work is not sufficient to draw a conclusion on which selection method is preferable in this thesis. Regardless that there are only a few data available due to some of the properties of a certain method cannot be analytically calculated, the GA they conducted their theories on is a maximisation GA whose average fitness of the population is supposed to increase by generations. Moreover, sampling fitness values from $G(0,1)$ means negative fitness is allowed. Data synthesis is rather a minimisation GA because there are at least two objectives from synthetic data to be minimised: 1) the divergence to the original data, 2) disclosure risks. In order to carry a better theoreti-

cal comparison for the use case between these methods, some settings from Blickle and Thieles work needs to be re-designed:

(a) In order to avoid non-analytic properties and give more relatable simulation to our use case, the new comparison is carried out on a sample of values drawn from the uniform distribution $U(0,1)$, which gives normalised and non-negative fitness values.

(b) The size of population for data synthesis is presumably small (100 is used in this thesis). Therefore the comparison is applied on discrete distribution rather than continuous distribution. The expected values of the interesting properties for all selection methods are estimated from 1000 runs of selection on these methods.

(c) The GA synthesiser is a minimisation algorithm whose average fitness of the population is supposed to decrease by generations. The formula of selection pressure is changed to

$$I = \frac{M - M^*}{\sigma}$$

, given M and M^* are average fitness of the population before and after selection.

The table below (Table. 4.3) shows properties of selection methods with the same parameters above from the renewed design. Exponential ranking selections have the lowest figures in selection variance, and the one with $c = 0.9$ has also the highest loss of diversity compared with others. Tournament selection with $t = 3$ has the most selection pressure whereas most of ranking selections have unsatisfied values in these properties. Tournament selection with $t = 2$, RWS and SUS have acceptable values among all the properties.

Selection Method	Loss of Diversity	Selection Pressure	Selection Variance
Linear ranking ($\eta^-1 = 0.0, \eta^+ = 2.0$)	0.4323	0.0294	0.5832
Linear ranking ($\eta^-1 = 0.3, \eta^+ = 1.7$)	0.3985	0.0206	0.7691
Exponential ranking ($c = 0.5$)	0.9283	0.2630	0.0010
Exponential ranking ($c = 0.9$)	0.7228	-0.1028	0.0580
Tournament (t=2)	0.4316	0.5820	0.5803
Tournament (t=3)	0.5089	0.8493	0.3296
Roulette Wheel	0.2654	0.5954	0.5262
Stochastic Universal	0.4375	0.5956	0.5203

Table 4.3: L, I and V in different selection methods with a reasonable choice of their parameters on 1000 populations with 100 candidates whose fitness values are drawn from $U(0,1)$.

Compared with the previous table (Table 4.2), this table here provides more information for the interesting properties in these selection methods. However, it cannot be strong evidence for the final decision on which selection method will be used because uniform distribution is a hypothetical condition in the GA synthesiser. A further investigation should be carried on a real dataset.

4.2.2 Experimental Comparison of Selection Methods

Due to the complexity of mechanisms behind the selection methods, the initiator of improper convergence is hard to be detected by only running the selection operator for once. By running some trials on a toy dataset with various population sizes, I conjectured two possible reasons for that selection operators cause early convergence.

First, The selection operator only senses better candidates, so it selects a small subset of candidates. It happens under two conditions: either the population size is too small or the probability of some candidates to be selected is much higher than others in the same population. Second, The selection operator is insusceptible to good candidates, so it selects only a limited amount of them into crossover. For example, in a linear ranking selection, the increase of population size results in closer selection probabilities between two neighbouring ranks. When setting the population size as 10, the probability of the best candidate selected reduced to 0.2, followed by 0.1778, 0.1556..., where the fittest candidate is likely to be abandoned in the selection operator.

In order to obtain more evidences for choosing the best selection method in this thesis, a toy dataset with 100 cases and 4 variables is used to carry out an experimental comparison of previously-introduced selection methods. The model that used to run these methods is a simple GA synthesiser that adopts matrix crossover with a fixed crossover rate 1.0 (see the illustration of this crossover in Fig 4.3) and has no mutation operator. The fitness function to determine the fitness of each candidate in this synthesiser is the mean divergence of pairwise Cramer's V values to the original dataset (see the formula of this objective in 4.7.1.1). For each method, I shall run the model 10 times with population sizes 10, 50, and 100 and record the average number of generations when the process reaches convergence. Convergence is considered to have occurred on the generation since when the average fitness value no longer changes in next 20 generations. As the model eliminates the impact from the mutation operator, the whole population will start to converge to the fittest candidate once the convergence reached.

4.2.2.1 Linear Ranking Selection

In linear ranking selection, all candidates are assigned to different ranks even though they may have the same fitness value. The best candidate is ranked as N, and the least fitted one is ranked as 0[2]. Let $p(i)$ present the probability of candidate i being selected, $p(i)$ is proportional to their ranks [13].

$$p(i) = \frac{1}{N} E f_i$$

η^- is a pre-determined parameter in linear ranking that determines to what extent users would like the worst candidate in each generation to survive. η^+ is another parameter that derived from η^- by $\eta^+ = 2 - \eta^-$. Theoretically, η^- takes any value in $[0.0, 2.0]$, but this allows the probability of the best and worst candidates share the same probability when $\eta^- = 1.0$. Actually, the selection probability distribution becomes uniform in this situation as every candidate now has the same probability to be selected. While the value of η^- increases from 1.0 to 2.0, the worse candidates would have a higher chance to survive than the fitter ones. Linear ranking selection is a mild method even the most strict condition is applied ($\eta^- = 0, \eta^+ = 2$), and its selection pressure does not change much as η^+ or η^- change(Table. 4.3). Therefore, the values $\eta^- = 0, \eta^+ = 2$ are used for comparison in this section.

[2]Python is 0-indexed, therefore all formula in the thesis are altered to adapt this environment while necessary

4.2.2.2 Exponential Ranking Selection

Candidates' probabilities of being selected raises exponentially in exponential ranking selection. The power of this method is determined by parameters $0 < c < 1$. In N (N is population size) ranks, the individual probability of being selected is:

$$p_i = \frac{c-1}{c^N - 1} c^{N-i}$$

Same as linear ranking selection, better candidates in exponential ranking have higher ranks and the worst candidate is ranked as 1. The constant coefficient $\frac{c-1}{c^N-1}$ guarantees $\sum_i^N p_i = 1$.

The following figure (Fig 4.1) tells that c has to be reasonably large otherwise the probability of the best candidate to be selected is much higher, and it will takeover the population in no time. I took the value of c as 0.7 in the later experiments, which gives a reasonable distribution to all candidates in the population.

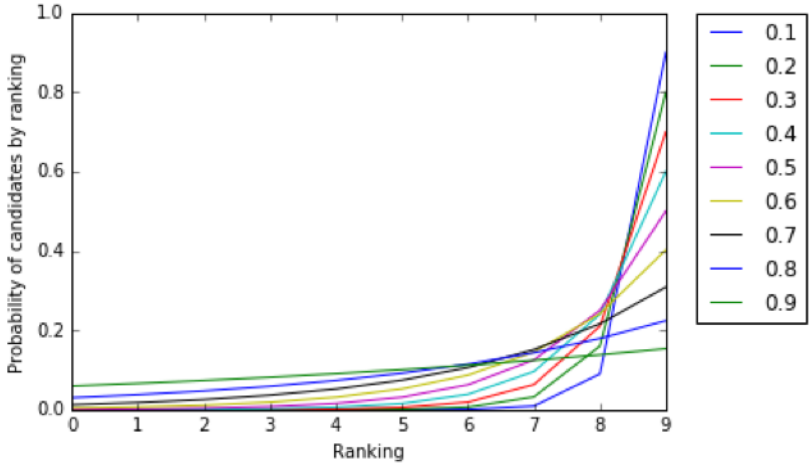


Figure 4.1: Selection probabilities of 10 candidates vs their ranks from 0 (the worst) to 9 (the best) with different c values in exponential ranking

4.2.2.3 Tournament Selection

Tournament selection holds N tournaments for N candidates in the population, where only the winner from each tournament is selected into crossover. The size of tournaments t is the number of randomly chosen candidates that enter one tournament [26]. Since the population size in the experiment is not large and tournament selection with $t = 2$ has less loss of diversity and better selection variance (Table 4.3). I only compare tournament selection with $t = 2$ here with others in this section.

4.2.2.4 Fitness-Proportional Selection

In fitness-proportional selection methods, the probability that a candidate is selected is proportional to its fitness value. Recall that M denotes the average fitness of the current population $M = \frac{1}{N}\sum f_i$, the probability that an individual is selected is:

$$p_i = \frac{f_i}{NM} \tag{4.1}$$

Fitness-proportional selection is designed for maximisation problems. Thus it requires a proper transformation in a minimisation problem. The most common way is to subtract the fitness value of a individual from the maximum value among them then the new values are displayed in a reversed order.

$$f_i' = f_{max} - f_i \tag{4.2}$$

For example, suppose the fitness values of initial population with size 4 are $[0.4730, 0.3613, 0.5803, 0.2381]$. Their converted fitness values are $[0.1100, 0.2190, 0.0000, 0.3485]$ that are subtracted from the maximal value 0.5803. As the method always eliminates the worst candidate, there are two alternatives: (1) use the reverse values of current fitness values; (2) assume that there is a maximum value ψ and take the differences between it and original fitness values as the converted fitness values. (2) should be used with cautions because, once the assumption of the maximum value fails, i.e. the generator generates a candidate whose fitness value is higher than ψ, he converted fitness value of this candidate would be negative, so does its selection probability.

According to Eq 4.1, the probability of candidates in a minimisation problem to be selected in approach (1) is:

$$p_i = \frac{1}{f_i NM}, M = \frac{1}{N}\sum_{i=1}^{N} f_i^{-1} \tag{4.3}$$

, meanwhile in approach (2) the probablity is:

$$p_i = \frac{\psi - f_i}{NM}, M = \frac{1}{N}\sum_{i=1}^{N}(\psi - f_i) \tag{4.4}$$

After transformation, these formulas still holds the sum of p_i as 1.0. The following figure illustrates how the two transformation works. Suppose a set of 100 candidates whose fitness values are randomly sampled from the uniform distribution $U(0,1)$. The following figure (Fig 4.2) shows probabilities calculated through the two approaches and the original fitness values of these candidates.

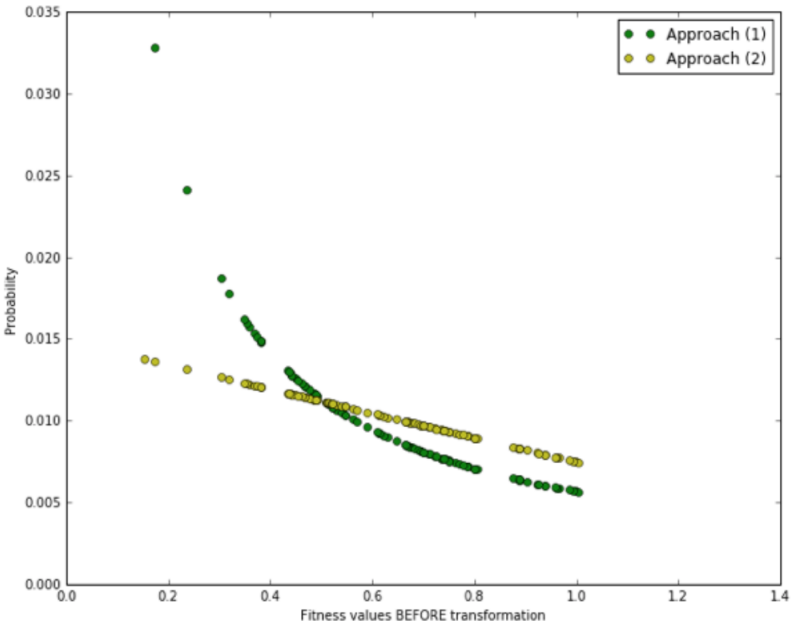


Figure 4.2: Selection probabilities over the (previous) 100 candidates between two approaches

In the result, approach (1) gives much higher probabilities to better candidates, which is likely to cause pre-mature convergence like exponential ranking selection. However, it is not saying that approach (1) is ill-designed because, unlike ranking selection, the probability from fitness-proportional selection depends on the value of fitness itself. In exponential ranking selection, a candidate from a particular rank always has fixed probability to be selected no matter to what extent it is better than others. One whose divergence to the original dataset is 0.02 may only have slightly higher chance to be selected compared to one whose divergence is 0.3 in the same scale. Compared with ranking selection, fitness-proportional selection methods using the two transformation approaches may avoid unnecessary elimination of fitter candidates.

Probabilities given by transformation approach (2) are more evenly distributed, which avoids generating more bias during the process of selection. The issue of approach (2) is the choice of ψ, the the value is hard to be determined without any prior knowledge. A small ψ results negative fitness values, but a large ψ attenuates differences between candidates. It can be solved by normalising fitness values so they can take 1.0 as the upper bound. In the following experiment I adopt approach (2) and $\psi = 1.0$ to compare the two fitness proportional selection methods: roulette wheel selection (RWS) and stochastic universal selection (SUS). Most researches use RWS as the fitness-proportional selection method, that simulates a roulette wheel with a single pointer and run it for N times to obtain the right population size for the next generation. SUS is a roulette wheel with N

evenly distributed pointers, and it only runs once to get the desired population size.

4.2.2.5 Experimental Comparison Results

These selection methods were tested through a simple GA synthesiser with population sizes 10, 50 and 100 on a tiny dataset ($100 \times r$). The following table (Table 4.4) records the mean number generations taken to converge in 10 trials and the mean value of best fitness values for every trial after converge. In the experiment, the performance of SUS is found differently from the others, in where the best fitness value is kept for several generations then suddenly drops. SUS takes much longer to converge and caused unnecessary uncertainties to the model so it will not be used or discussed any more.

	Population size	10	50	100
Linear Ranking	Mean number of generations	10	50	80
	Mean value of fitness	0.025	0.012	0.008
Exponential ranking	Mean number of generations	6	2	N/A
	Mean value of fitness	0.04	0.018	N/A
Tournament	Mean number of generations	10	48	77
	Mean value of fitness	0.024	0.016	0.007
RWS	Mean number of generations	31	119	219
	Mean value of fitness	0.042	0.017	0.015

Table 4.4: Different selection methods and their mean numbers of generations taken to converge and the mean fitness values of the best candidate in 10 trials with different population sizes. (N/A indicates that the model took 1 generation to converge.)

When comparing the speed of convergence and the best candidates after convergence, tournament selection and RWS are suitable selection methods for the GA synthesiser. The two give a higher chance to the best candidate in each generation to survive and reasonably distribute selection probabilities to other candidates. As RWS was initially designed for maximisation problems, all fitness values were converted using approach (2)

with $\psi = 1.0$ in the experiment. However, it is found that fitness values of candidates in the later process were very close to 0.0 and their differences were indistinguishable when setting $\psi = 1.0$, which may result in inefficiency and unnecessary cost of time. Therefore the method is not yet practical in real-world data synthesis, and tournament selection with $t = 2$ seems so far the most suitable selection method for the GA synthesiser.

4.3 Crossover Methods

Crossover is the main resource in GAs to provide variation, but there is a lack of investigation on crossover for matrix GAs and GA synthesisers. In the thesis, I propose some of the possible methods that can be applied to data synthesis. These methods are classified into three categories: Matrix crossover, Parallelised crossover and Uniform crossover, based on different levels of entity of a dataset they take and the shape of blocks that they swap between paired candidates. It can be assumed that microdata has three levels of entities. Level 1 regards the whole dataset as an entity without splitting its variables or cases. Level 2 assumes that data is formed of a number of variables/cases, and they can be independently processed in GAs. Level 3 has every single cell (element) in the dataset as an entity. A series of experiments are carried out to compare some potential crossover methods and find the most suitable crossover method for GA synthesisers. These methods are listed in this section and the results were presented in Chapter Experiments and Results.

4.3.1 Matrix Crossover

Matrix crossover was first proposed in [124]. In this use case, it takes the whole dataset as a single entity (level 1) where variables or cases are not considered. During the process, the method chooses a rectangle shape sub-matrix with random sizes and swaps it with the paired individual. Fig 4.3 shows how matrix crossover works.

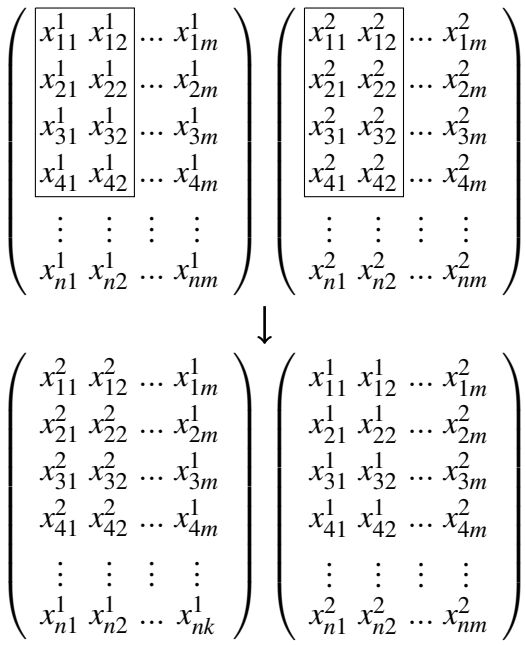


Figure 4.3: X^1 and X^2 before and after matrix crossover

4.3.2 Parallelised Crossover

By assuming that all level 2 entities (variables or cases) in the dataset are independent, they can be allocated to parallel sub-processors that the total amount equals to the number of level 2 entities. In other words, the model runs crossovers and mutations independently to different entities at the same time in one generation.

4.3.2.1 Variable Parallelised Crossover

Variable parallelised crossover (it was once called parallelised crossover in some papers in Chapter Experiments and Results) randomly chooses a sub-matrix within a variable (column) then swaps it with the corresponding position in the same variable from the paired dataset. Fig 4.4 illustrates parallelised crossover between a pair of candidates X^1 and X^2:

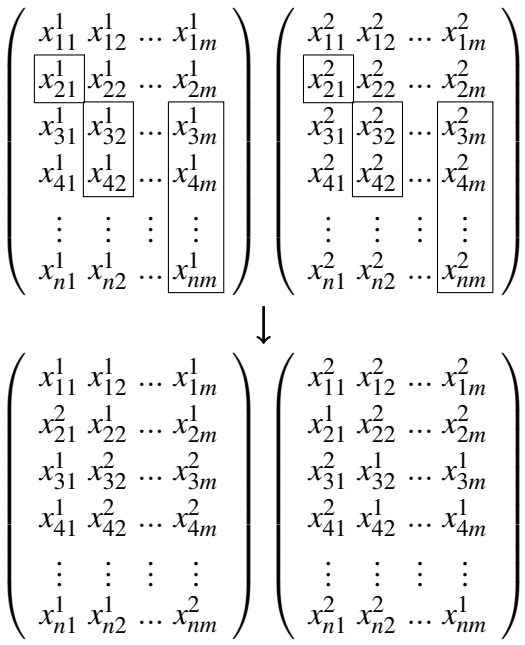


Figure 4.4: X^1 and X^2 before and after variable parallelised crossover

4.3.2.2 Case Parallelised Crossover (CPC)

Case parallelised crossover (CPC) is similar to variable parallelised crossover instead takes rows as entities for a dataset (Fig 4.5).

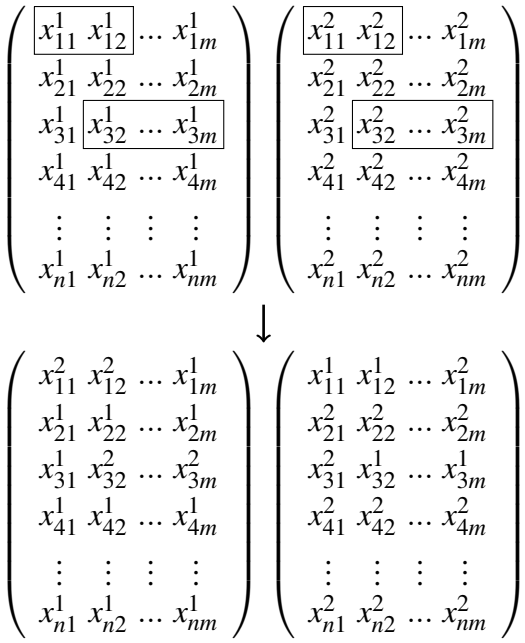


Figure 4.5: X^1 and X^2 before and after CPC

4.3.2.3 Round-CPC

Round-CPC has a similar mechanism as annular crossover in [92]. It first selects two random endpoints from a case, then it swaps elements either between or out of the two endpoints by 50/50 chance, which gives the elements at the margin or centre of each row (case) equal chance to be swapped (Fig 4.6).

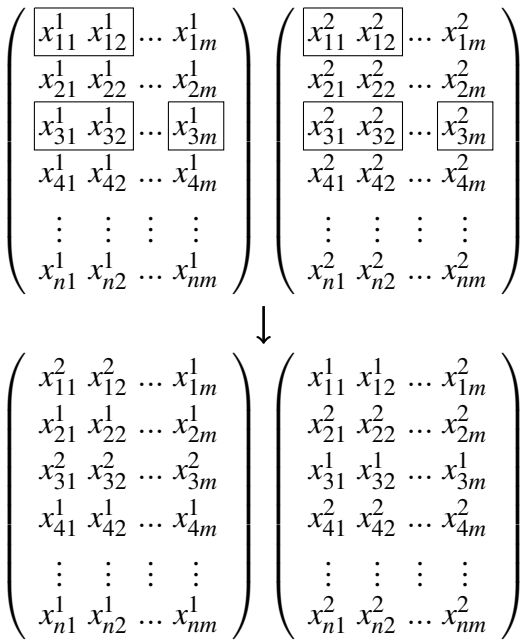


Figure 4.6: X^1 and X^2 before and after round-CPC

4.3.2.4 Whole-CPC

Instead swaps elements between or out of two random end points, whole-CPC exchanges the entire case (Fig 4.7).

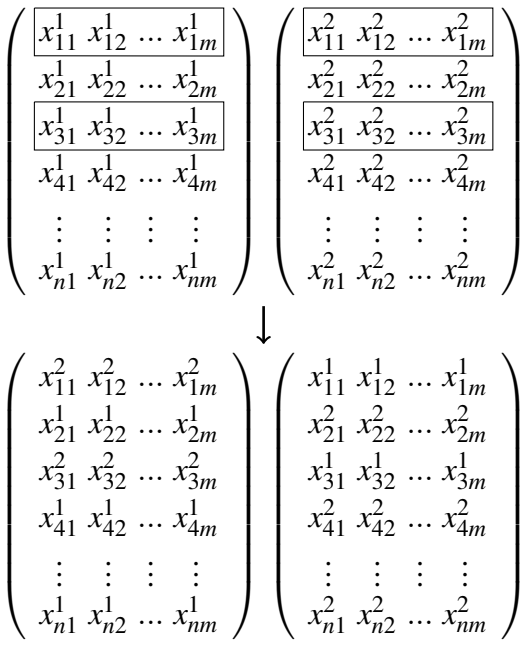


Figure 4.7: X^1 and X^2 before and after whole-CPC

4.3.3 Parametric Uniform Crossover (PUC)

PUC is applied to each cell (level 3 entity) of the dataset with a pre-determined probability p_0. Fig 4.8 illustrates how PUC swaps elements between two datasets:

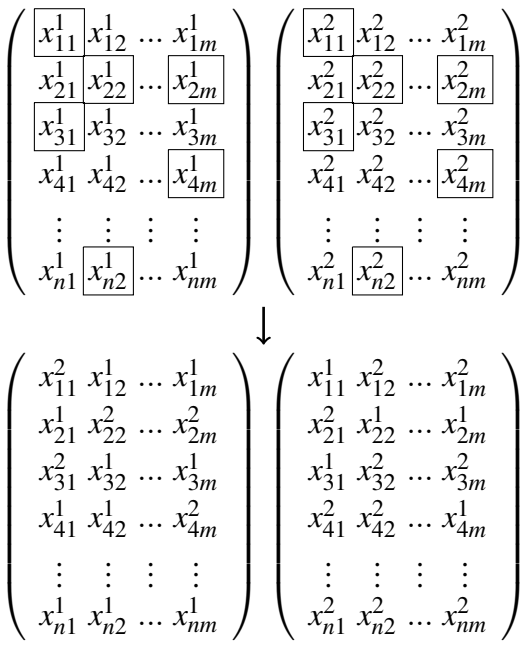


Figure 4.8: X^1 and X^2 before and after PUC

4.4 Mutation Methods

Mutation will not be mainly discussed in this thesis because 1) it usually contributes less variation than crossover operators to the next generation and 2) Any features of crossover (such as positional bias, cost of computational loads, impact to the synthesisers' efficiency) can also be found in the mutation with the same mechanism. The mutation operator used most in the thesis simulates PUC, and it is called uniform mutation (Fig 4.9). It changes the value of a randomly selected cell with a mutation rate p_m in the dataset by uniformly sampled the values taken by the corresponding variable (column) where the cell belongs to.

$$\begin{pmatrix} \boxed{x^j_{11}} & x^j_{12} & \dots & x^j_{1m} \\ x^j_{21} & \boxed{x^j_{22}} & \dots & \boxed{x^j_{2m}} \\ \boxed{x^j_{31}} & x^j_{32} & \dots & x^j_{3m} \\ x^j_{41} & x^j_{42} & \dots & \boxed{x^j_{4m}} \\ \vdots & \vdots & \vdots & \vdots \\ x^j_{n1} & x^j_{n2} & \dots & x^j_{nm} \end{pmatrix} \quad \begin{pmatrix} x^{j^*}_{11} & x^j_{12} & \dots & x^j_{1m} \\ x^j_{21} & x^{j^*}_{22} & \dots & x^{j^*}_{2m} \\ x^{j^*}_{31} & x^j_{32} & \dots & x^j_{3m} \\ x^j_{41} & x^j_{42} & \dots & x^{j^*}_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ x^j_{n1} & x^j_{n2} & \dots & x^j_{nm} \end{pmatrix}$$

Figure 4.9: X before and after mutation

4.5 Positional Bias and Schema Theory

The experiments carried out in the next chapter not only compare the above crossover methods but also help to determine the characteristics of a good crossover method for GA synthesisers. The experiments aim to address two questions: (1) does positional bias in crossover method impact the efficiency of GA synthesisers? And (2) what are schema and how 'blocks' are building in GA synthesisers?

Both matrix crossover and parallelised crossover (except round-CPC and whole-CPC, the purpose of designing both methods is for dropping positional bias in parallelised crossover) are based on the idea of two-point crossover. For example, in variable-parallelised crossover, the element with row index i will be swapped if, and only if, one of the selection points has index not greater than i while the other has index greater than i. Thus the swap probability is the hypergeometric probability:

$$P(\min(a,b) \leq i < \max(a,b)) = i(n-i+1)\binom{n+1}{2}^{-1} \tag{4.5}$$

, where a and b are the indices of a pair of (distinct) randomly chosen crossover points.

It is trivial to show that this is a monotonically increasing function of i where $i < \frac{n}{2}$ and a monotonically decreasing function of i where $i > \frac{n}{2}$. For matrix crossover that selects rectangle blocks (4 endpoints) the probability of an endpoint with index (i, j) being swapped is a product of hypergeometric probabilities. PUC, round-CPC and whole-CPC, on the other hand, contains no positional bias and can provide us with an implicit evaluation of the effect of positional bias to GA synthesisers.

GAs were described as a process of discovering, emphasising and recombining good 'schema' or 'building blocks' in individuals over generations (see section 3.5) and such process was observed frequently in binary GAs [82][52][63]. Whether the process also exists when synthesising data using matrix GAs is unknown yet, and it is worth exploring by comparing the performance of crossover methods from different levels of entity.

4.6 Adaptive GAs

There are arguments on involving adaptive parameters in GAs. First, it might increase computational workload unnecessarily, especially when using a fixed mutation rate in GAs sometimes is sufficient to give satisfactory outputs. This was also confirmed by Galaviz and Xuri, who observed that adaptive GAs did not always outperform fixed GAs in some cases [53]. Second, Adaptive parameters, especially those determined by candidates' fitness, might make the whole process more unpredictable [121]. However, Libelli and Alba described the disadvantages of **not** using adaptive parameters using an example: taking mutation rate p_m as an instance, a high, constant value of p_m can break the goodness of near-optimal candidates whereas a low, constant value of p_m retains unwanted information in bad candidates [72]. There are three parameters in GAs can be adaptive: the crossover rate, the mutation rate and the population size [69]. It showed that the population size has no stable effect to the efficiency of GAs [58]. Therefore, in the thesis, I mainly study adaptive parameters in two operators: crossover and mutation, on their rates p_c and p_m, and their impact to GA synthesisers.

4.6.1 Adaptive Crossover Rates

Srinivas and Patnaik proposed that p_c can be adapted based on the fitness value of the corresponding candidate [115]. They used the following formula to adjust crossover rates: Suppose f_c is the larger fitness value of the two paired candidates, p_c of this pair of candidates can be:

$$p_c = \frac{k_c(f_{\max} - f_c)}{f_{\max} - \bar{f}}, 0 < k_c < 1.0 \tag{4.6}$$

The formula is originally designed for maximisation problems, but fitness in data synthesis is a proxy for the divergence from the original dataset which should be minimised. Therefore, the formula therefore is converted to:

$$p_c = \frac{k_c(d_c - d_{\min})}{\bar{d} - d_{\min}}, 0 < k_c < 1.0$$

where $\bar{d}$ is the mean of divergence in the pool, $d_{\min}$ is the minimum divergence in the current generation and d_c is the smaller divergence between the paired candidates. This method is fairly practical as it assumes a linear dependence between p_c and fitness values. However, there is no clear rule to decide the value of k_c. Fig 4.10 compares the impact from different k_c to p_c over fitness values of 20 candidates and, as it presents, the value of p_c returned by the formula can be greater than 1.0 because p_c is not a probability that sums to unity over all candidates but is specific to each candidate and such a value then needs to be adjusted to 1.0 heuristically (see the horizontal dash line in the figure). Moreover, the formula prevents the best candidate from crossover, which may limits the

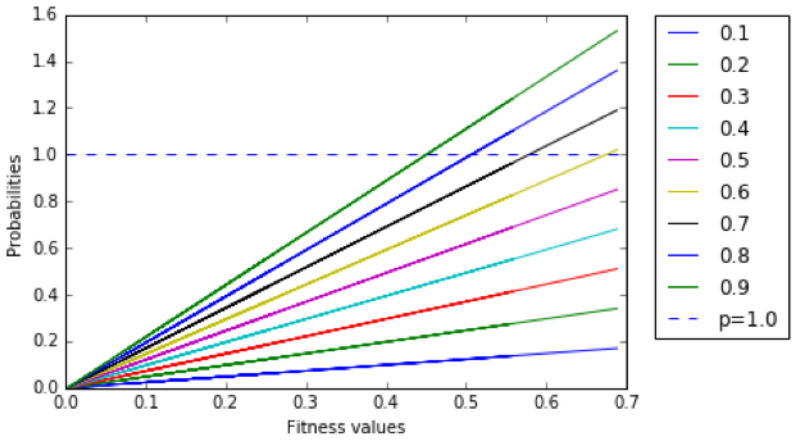


Figure 4.10: The impact of k_c on p_c for different fitness values

exploration power of GAs when the population is not sufficiently mature. Experiments are required to decide whether to involve adaptive crossover rates and to determine the impact of adaptive crossover rates.

4.6.2 Adaptive Mutation Rates

Suppose the fitness value of the candidate is f_i, and the average fitness value in its generation is $\bar{f}$. p_m can be defined by the following principles [72]: If f_i is 'better' (higher in maximisation problems and lower in minimisation problems) than $\bar{f}$, then p_m should keep low to keep the method from the good individual. If f_i is 'worse' than $\bar{f}$, then p_m should be reasonably high to generate new species into the pool. Thierens introduced two

approaches to adjust p_m during the GA process [121]. In the first approach, p_m of a particular candidate is determined by its fitness at the current stage. For example, a simple adaptive function for p_m can be:

$$p_m = \begin{cases} 0 & \text{if } f_i \leq \bar{f} \\ \lambda |\bar{f} - f_i| & \text{if } f_i > \bar{f} \end{cases}, \lambda \in [0, f_{max} - \bar{f}]$$

, where $\bar{f}$ is the mean value of fitness and λ is a pre-determined parameter to bound the value of p_m less than 1. The function forces the least fit candidate to mutate so it may not performs well at the later process of GAs, where all candidates in the population have well developed. Furthermore, it requires more computational workloads to calculate fitness values one more time for all children in the pool. Since with a proper setting, the population in GAs always becomes fitter over generations, the alternative approach in adaptive p_m is to let p_m decreases by the number of generations t. A general formulae of $p_m(t)$ is:

$$p_m(t) = (2 + \frac{l-2}{T-1}t)^{-1}$$

, where T is the number of generations that the GA has run so far [121].

Self-adaptive p_m is generally favourable because it offers users' control over p_m and provides an evolutionary advantage to all individuals in the generation [72]. However, Bingu argued that the relationship between fitness values and parameters were complex so that it was better to be described linguistically [12]. Moreover, since there is no universal rule of parameter control in self-adaptive GAs for either linear or matrix GAs, it is reckless to apply a random adaptive function on p_m and p_c without experiments. Therefore, I shall consider to only explore the effects of simple adaptive functions like linear functions in the thesis.

4.7 Objectives and Evaluation Tests

Objective design is equivalent to fitness function design in GAs. It sets up the environment and controls how candidates should evolve during the process. Objective design is also important in data synthesis because it determines which statistical properties the synthetic data will carry. There are two reasonable principles to follow while designing objectives for a GA synthesiser: (1) the objective should be universally applicable to all categorical data, and (2) the objective should be quantitatively measurable. This section gives a set of objectives specifically designed and used for GA synthesisers in the thesis (please note they are not used in every experiment).

4.7.1 Data Utility Objectives

There are some assumptions in this section and applied to the whole thesis. First, it assumes that the synthetic data X has the same size as the original data Y so n is the total number of cases, and m is the total number of variables in both datasets. Second, it assumes that a variable in synthetic data X has the same categories as its corresponding variable in the original dataset. Suppose that $CT(Y, j, k)$ is an $R \times C$ contingency table (see Table 2.1) between a pair of variables Y_j, Y_k and R, C are the numbers of categories of Y_j and Y_k respectively (so does in synthetic data X but it has the same numbers of categories in X_j and X_k as Y_j and Y_k), r, c are a pair of indices that determines the position in a contingency table CT, and $CT_{r,c}$ indicates the value of counts in that particular cell.

4.7.1.1 Objective 1: To minimise the mean absolute difference in Cramer's V between the synthetic and the original datasets across all pairs of distinct variables

Respecting variable associations in the original data is essential to produce high-quality synthetic data. A measure of association between categorical (nominal) variables is Cramer's V. Cramer's V (see Equation 2.2) is calculated from the contingency table of these variables. The first objective is to minimise the mean absolute difference in Cramer's V between the synthetic dataset X and the original dataset Y across all pairs of their distinct variables, and therefore the fitness function is:

$$F_1(X,Y) = \binom{m}{2}^{-1} \sum_{j=1)}^{m-1} \sum_{k=j+1}^{m} |\Phi_c(CT(X_{:,j}, X_{:,k})) - \Phi_c(CT(Y_{:,j}, Y_{:,k}))| \tag{4.7}$$

4.7.1.2 Objective 2: To minimise mean of the sum of differences between counts of a bi-variate contingency table in the synthetic and the original datasets

Objective 1 is not sufficient because two datasets that are similar in terms of Cramer's V may contain different data structure. So the second objective is designed based on the absolute difference between all distinct bi-variate contingency tables, which captures a different element of the bi-variate structure to Cramer's V (and is scalable to dimensionality beyond bi-variate). The divergence measure between two contingency tables is set as the mean of the sum of the element-wise absolute differences.

$$\Delta CT(X,Y,j,k) = \frac{1}{n} \sum_{r}^{|I_j|} \sum_{c}^{|I_k|} |CT(X_{:,j}, X_{:,k})_{r,c} - CT(Y_{:,j}, Y_{:,k})_{r,c}|$$

Therefore, the fitness function of objective 2 is defined as the mean of absolute differences between the bi-variate contingency tables of the synthetic dataset X and the original

dataset Y across all pairs of their distinct variables.

$$F_2(X,Y) = \binom{m}{2}^{-1} \sum_{j=1)}^{m-1} \sum_{k=j+1}^{m} \Delta CT(X,Y,j,k) \tag{4.8}$$

where the square brackets are Iverson brackets and the levels of I_j and I_k are indexed $r \in [1..|I_j|]$ and $c \in [1..|I_k|]$, respectively.

4.7.1.3 Objective 3: To minimise Jensen-Shannon Divergence between counts of ζ-dimensional contingency tables in the synthetic and the original datasets

As for higher dimensional contingency tables, Jensen-Shannon divergence D_{JS} is used to measure the divergence between them. D_{JS} gives normalised outputs between from 0 (no divergence) to 1. It was initially designed to measure the divergence between two multi-dimensional discrete probability distributions P and Q:

$$D_{KL}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)}$$

where D_{JS} symmetries KullbackLeibler divergence D_{KL}. By given $M = \frac{1}{2}(P+Q)$,

$$D_{JS}(P||Q) = \frac{1}{2} D_{KL}(P||M) + \frac{1}{2} D_{KL}(Q||M)$$

It takes the square root of right hand side from above equation for more observable outputs in the thesis:

$$D_{JS}(P||Q) = (\frac{1}{2} D_{KL}(P||M) + \frac{1}{2} D_{KL}(Q||M))^{\frac{1}{2}}$$

By introducing D_{JS}, the divergence measure of a pair of bi-variate (2-dimensional) contingency tables in the synthetic dataset X and the original dataset Y in Objective 2 is:

$$\Delta(X,Y,\{j,k\}) = D_{JS}(\frac{1}{n} CT(X_{:,j}, X_{:,k}) || \frac{1}{n} CT(Y_{:,j}, Y_{:,k}))$$

Meanwhile, analogous measures in higher (ζ) dimensional contingency tables is:

$$F_3(X,Y) = \binom{m}{\zeta}^{-1} \sum_{S \in P_\zeta([1..m])} \Delta(X,Y,S) \tag{4.9}$$

, where $P_\zeta(Z)$ denotes the members of the power set of Z of size ζ.

In Chapter Experiments and Results, most experiments use full contingency table. Assume that each variable in the dataset takes values from finite sets I_j so that $I = \bigtimes_{j=1}^{m} I_j$ denotes the possible configurations of all m variables, a full contingency table is an m-dimensional table containing a count for each member of I. However, using a full contingency table as the only utility objective increases the risk of re-identification because the synthesiser will eventually stop at the original data. Therefore conflicting risk objectives are required.

4.7.2 Disclosure Risk Objectives

4.7.2.1 Identification Disclosure Risk

A protecting model must prevent re-identification from intruders. This condition used to be presumably satisfied in fully synthetic data that synthesises all original records, until full contingency tables are used to measure the utility.

Re-identification occurs when small counts appear on a multi-dimensional contingency table of a set of quasi-identifiers. Shlomo confirmed that re-identification risk is a function of both the dataset's statistical population and the sample [107]. Suppose $N_{(r,c)}$ is the population size in cell (r,c) in a bi-variate contingency table and $n_{(r,c)}$ is the sample size of the same cell. The disclosure risk is measured through:

$$\hat{r_1} = \sum_{(r,c)} I(n_{(r,c)} = 1)\hat{P}(N_{(r,c)} = 1 | n_{(r,c)} = 1)$$

$$\hat{r_2} = \sum_{(r,c)} I(n_{(r,c)} = 1)\hat{E}(\frac{1}{N_{c,r}} | n_{(r,c)} = 1)$$

For known $N_{(r,c)}$ from the population, the measurement of re-identification risk can simply be reduced to:

$$r = \sum_{(r,c)} \frac{I(f_{(r,c)} = 1)}{N_{(r,c)}} \tag{4.10}$$

Re-identification might also occur if there exist external datasets containing quasi-identifiers that some of them are the same as the variables presented in the released dataset that are also available to intruders [35]. Differential privacy is generally used to neutralise such risks from data linkage. It tells to what extent the data controllers can promise the indifference from an individual present or not present in a dataset. If outputs of a privacy model are almost equally likely to be observed on every neighbouring dataset, the model is $(\varepsilon, 0)$-differentially private [46]. By engaging full contingency tables in the utility measure, outputs from a GA synthesiser (M_{GA}) is $(\varepsilon, 0)$-differentially private. Suppose neighbouring datasets Y_1 and Y_2 that differ from only one record, it is impossible to distinguish outputs from $M_{GA}(Y_1)$ and $M_{GA}(Y_2)$ unless the original datasets are revealed. However, since GAs have different mechanisms from differential privacy, they can neither take ε as a model parameter nor give its value after the execution. I shall not discuss the issue any more in this thesis.

4.7.2.2 Attribute Disclosure Risk

Attribute disclosure risk assesses the sensitive association between variables. Taub et al. introduced a privacy model for measuring disclosure risk in synthetic data called *Differential Correct Attribution Probability* (DCAP) [47][119]. DCAP measures the probability if a specific value from a target variable can be learnt from the values of a set of quasi-identifiers by calculating empirical probability:

$$CAP_{o,j} = Pr(T_{o,j}|K_{o,j}) = \frac{\sum_{i=1}^{n}[T_{o,i} = T_{o,j}, K_{o,i} = K_{o,j}]}{\sum_{i=1}^{n}[K_{o,i} = K_{o,j}]}$$

where the square brackets are Iverson brackets, n is the number of cases, K_o and T_o as vectors for the quasi-identifiers and the target variable from the original data. Likewise, K_s and T_s are from the synthetic data. The CAP for record j based on a corresponding synthetic dataset is the same empirical, conditional probability but derived from the synthetic data,

$$CAP_{s,j} = Pr(T_{o,j}|K_{o,j})_s = \frac{\sum_{i=1}^{n}[T_{s,i} = T_{o,j}, K_{s,i} = K_{o,j}]}{\sum_{i=1}^{n}[K_{s,i} = K_{o,j}]} \tag{4.11}$$

DCAP is a feasible risk objective in GA synthesisers for the following reasons. First, it measures both re-identification and attribution risks if with proper modification. A scenario in DCAP is that in which rather than using the whole dataset, only the statistical uniques of the original dataset are deployed when calculating the CAP score (this method corresponds to a common focus for National Statistical Institutes). The non-matches (records on the original dataset which do not match any records in the synthetic dataset on the quasi-identifiers) in this instance of DCAP were scored as 0, that allows candidates with more non-matches to have lower scores but the one with matched uniques stands out. Second, it requires neither weights nor pre-determined parameters compared with information entropy measure (see Eq 2.5) or other privacy models in Sec 2.3.3. DCAP determines the level of risks contained in a synthetic dataset by comparing with a baseline DCAP score, which is derived by a dataset generated by uniformly sampling values from every variable in the original data[3]. Last but not least, DCAP can also be calculated through full contingency tables of both original and synthetic data (Eq 4.11), that reduces the processing time of GA synthesisers.

4.7.3 Evaluation Tests

It is important to validate the information utility of fully synthetic data as some of the statistical properties are not explicitly related to the full contingency table. The utility can

[3] Antal et al. also assumed in [6] that this kind of data contains no disclosure risk.

be evaluated from fundamental statistical analyses, including univariate histogram and bi-variate joint-distribution charts [89]. Some statistical models like logistic regression model and multiple correspondence analysis (MCA) can also be used to compare the closeness of the synthetic data to the original data.

Logistic regression model predicts the log-odds of the probability of the categories from the output variable. Its simplest form is binary logistic regression model that has only two categories in the predictor variable. Suppose that y is the predictor variable in original dataset Y that interests analysts. A logistic regression model of y can be expressed as:

$$\ln(\frac{p(y=1)}{1-p(y=1)} = \beta_0 + \beta_1 y_1 + ...\beta_w y_w,\ 1 \leq w \leq m$$

Suppose that X is the synthetic version of Y and x is the synthetic version of x, the corresponding logistic regression model in X is:

$$\ln(\frac{p(x=1)}{1-p(x=1)} = \beta'_0 + \beta'_1 x_1 + ...\beta'_w x_w,\ 1 \leq w \leq m$$

The original and synthetic data can then be compared through the correlation ρ between predicted logit probabilities. The closer ρ to 1, the similar the predictions from two logistic regression model are.

$$\rho = corr((\beta_0 + \beta_1 y_1 + ...\beta_w y_w), (\beta_0 + \beta_1 x_1 + ...\beta_w x_w)),\ 1 \leq w \leq m \tag{4.12}$$

MCA, on the other hand, carries out a pattern analysis on nominal variables and explores their underlying structures. It requires a pre-determined number of factors (which is usually small) and gives two sets of factor scores for all variables and cases. These scores are usually visualised on bi-plots [1]. The difference factor scores between the original and synthetic data can also be used to evaluate the latter's utility.

4.8 Chapter Summary

In this chapter, I have described different focuses of designing GA synthesisers. For some of those, I have been able to make design decisions on theoretical grounds or through the use of pilot empirical tests. For the others, more detailed empirical work is needed. The majority of the papers in Chapter Experiments and Results report on such experiments. In each experiment, there will be trials on selected datasets to evaluate the impact from particular parameters or operator variants to the speed of convergence of the generator. The capability of the GA synthesiser to generate a trade-off between utility and risk objectives will also be tested in the chapter.

Chapter 5

Experiments and Results

There are 6 papers in this chapter. They summarise important findings in this thesis. The publishing destinations and states of these papers can be checked in Table 1.1.

The first paper Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data shows the implementation of a simple GA on data synthesis on a small dataset. It demonstrates the potential of GAs in synthetic data production.

The following papers discuss different focuses of GA synthesiser design. The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods compared three crossover methods that are commonly used in GAs: variable parallelised crossover, matrix crossover and PUC. The comparison was based on their efficiency on restoring relationships between variables in synthetic data. The findings of this paper inspire the next paper: Matrix GA: building blocks in data synthesis, which discusses newly arising issues in identifying appropriate crossover methods for GA synthesisers when the efficiency of traditional crossover methods is not satisfactory. This paper involves Schema Theory in order to answer the question: how the 'blocks' are building when GAs synthesise data. The crossover method (CPC, see Fig 4.5) and its two variants (round CPC, see Fig 4.6 and whole CPC, see Fig 4.7) were first introduced and compared here, and whole CPC is demonstrated most suitable for GA synthesisers so far.

Impact of Full Contingency Table in Data Synthesis discusses the potential of using a unique utility objective for GA synthesisers and demonstrates that the full-contingency table is a reasonable choice for measuring the closeness of synthetic data to the corresponding original data. A further exploration of objective design is in Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator. Since the full-contingency table comes with high disclosure risks due to it being close to the original data by definition, we need to focus on finding the trade-off between information utility

and disclosure risks in the final output. This paper uses DCAP as the risk objective in the GA synthesiser to contest with the full contingency table as the only utility objective.

The Impact from Initial Population in GA Synthetic Data Generator investigates the impact of the diversity of initial populations on the efficiency of GA synthesisers. It confirms that in GA synthesisers, initial populations with more diversity have more exploratory power.

All papers above adopt fixed GAs, and their performance is acceptable. However, the advantage of fixed-rate operators is found not consistent in the process. Therefore, the paper: Exploring the Impact of Adaptive Parameters on a Genetic Algorithm Synthesiser explores whether adaptive parameters have a positive impact on the efficiency of GA synthesisers.

Paper Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator deploys a weighted-sum single objective GA to demonstrate the existence of a trade-off between full-contingency-table-measured utility and DCAP-measured risk. Implementing Pareto optimisation was the intuitive and obvious option while designing multi-objective GA synthesisers. However, the experiment showed that the performance of Pareto multi-objective GAs are no better (and sometimes worse) than the weighted-sum GA. As the purpose of the thesis was a proof of concept for the use of GAs as data synthesisers and this particular issue, whilst important, is a distraction from the primary goal. Therefore, the experiment results from Pareto multi-objective GAs are moved to Chapter 7 followed by a discussion of the underlying reasons for this unexpected lack of performance.

5.1 Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data

This paper is the implementation of an earlier position paper [20] (see Appendix 7.3) and explains how to use a new form of genetic algorithms (matrix GAs) to generate synthetic data and provides a proof of concept using a small individual-level microdata set. The new method is able to iteratively optimise synthetic data based on a set of utility parameters until its closeness from the original data achieves an acceptable level. The paper describes the advantages of this method and its potential in synthetic data production. It covers both theoretical and computerised model design and specifies further development of this study.

Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data

Yingrui Chen[*], Mark Elliot[**] and Joe Sakshaug[***]

[*] University of Manchester, yingrui.chen@manchester.ac.uk
[**] University of Manchester, mark.elliot@manchester.ac.uk
[***] University of Manchester, joe.sakshaug@manchester.ac.uk

Abstract: This paper is the implementation of an earlier position paper (Chen, Elliot & Sakshaug, 2016) and explains how to use a new form of genetic algorithms (matrix GAs) to generate synthetic data and provides a proof of concept using a small individual-level microdata set. The new method is able to iteratively optimise synthetic data based on a set of utility parameters until its difference from the original data achieves a desired level. The paper describes the advantages of this method and its potential in synthetic data production. It covers theoretical and computerised model design and specifies further development of this study.

1 Introduction

Algorithms are the core of machine learning (ML), which is designed for solving problems with non-trivial solution spaces and where prior knowledge cannot be completely specified. ML instantiates a series of instructions that transform input into output; modern machine learning algorithms are increasingly designed for unstructured global optimisation problems (Ethem, 2014). Notwithstanding this, algorithmic methodology presupposes that most problems can be reduced to optimisation problems with objectifiable constraints and objectives.

The properties of machine learning algorithms show significant potential for synthetic data production. Firstly, ML does not require an accurate pre-assumed model to cover all statistical properties that the synthetic data is supposed to have. Machine learning algorithms are interruptible so users can set any statistic as the objective any time during model processing. Secondly, it uses less human effort to find optimal solutions, which is more efficient with big datasets.

The idea of using machine learning algorithms in data protection has been mentioned by several researchers: Drechsler (2010) proposes modern optimisation tools as a generator of synthetic data. Navarro-Arribas and Torra (2015) describe data protection as an optimising problem with two opposite constraints: information loss and disclosure risks and proposed using Genetic Algorithms (GAs) to improve the protection of databases or to improve parameters in disclosure control models. More recently the current authors established a framework for a GA approach to producing synthetic data (Chen et al 2016).

1.1 Genetic Algorithms

Genetic algorithms (GAs) are one of the well-known machine learning algorithms. Conceptually, they mimic the process of natural selection. GAs use a parallel search to randomly select *individuals* from a *population* of candidates, apply crossover (exchange information between candidates) and mutate the candidates (perturb information) until the whole system reaches convergence (where all candidates are identical) or the system meets some user defined criterion (Goldberg, 1989). Every run of selection, crossover and mutation processes is called a generation. The number of generations can be used to measure the speed of convergence of a GA model. In GA, the set of candidates in the current generation is generally called "the

population", but in order to distinguish to the 'population' we generally use in statistics, the population of candidate datasets will be called "the GA population".

The most common representation of individuals in GA populations is in the form of binary strings (effectively one-dimensional arrays). There is a relative lack of work on what are called matrix-GAs (two-dimensional arrays). For the synthetic data production use case, there are strong reasons to use a matrix representation. Firstly, the matrix is the standard way to present microdata, which contains information directly collected from individual population units (Ciriani et al, 2007). Secondly, as Marchette and Solka (1996) observe, matrix GAs can most appropriately display data structures of their candidates so they can not only optimise records but also their relation to surrounding locations in a dataset.

2 The properties of GAs

2.1 Fitness Functions

As an optimisation problem, the objective of synthetic data production is to minimise its difference in statistical inference of the synthetic datasets compared to the original one (Abowd & Lane, 2004). In GAs, the measurement of how close an individual is to the objective is called fitness. Hence, the fitness of a candidate dataset within the GA population is measured by its analytical similarity to the original dataset – a property that is often called utility in the statistical disclosure control literature (see for example Duncan et al 2011). Practically, the similarity is evaluated by the divergence of a certain statistical properties between the two datasets: the smaller the divergence is the more similar they are and therefore the higher fitness of the synthetic dataset.

2.2 Selection Schemes

Candidates from the GA population in the current generation are selected into a pool before crossover. The purpose of selection is to eliminate worse candidates and give higher chance to better candidates to survive and pass their good properties to the next generation. Candidates are selected from the current GA population with replacement so that one each candidate can crossover with others more than once (Melanie, 1998). Standard selection schemes can be classified into two categories based on their mechanisms: Fitness-proportional selection schemes and Ranking-based selection schemes. Fitness-proportional schemes select candidates according to their fitness values. The fitter a candidate is the more likely it is to survive. Ranking based schemes select candidates based on the ranks of their fitness values; the candidate with highest rank is the most likely to survive. The probability from fitness-proportional selection depends on the value of fitness itself. In ranking-based selection, the candidate of the given rank always has fixed probability to be selected no matter how much it is better than others. One that is much better than the other candidates may only have slightly higher chance to be selected compared to others in the same generation. Fitness-proportional selection schemes avoid this issue and cause less elimination of fitter candidates.

Fitness-proportional selection	Ranking-based selection schemes
Roulette Wheel Selection Stochastic Universal Selection	Tournament selection Linear ranking selection Exponential ranking selection Truncation selection*

Table 1.1 Selection schemes

This paper only compares stochastic universal selection (SUS) and linear ranking selection that can be considered as representatives of the two categories. For more details about other selection schemes please check Blickle and Thiele's (1995) work. SUS selects candidates according to the probability that is proportional to one's fitness value. In linear ranking selection, Candidates in the GA population are ranked in ascending order of their fitness values from 1 to N, where N refers to the best and 1 refers to the worst. All candidates should be assigned into different ranks even though they may have same fitness values. The probability of a candidate to be selected is linearly proportional to its rank but not fitness (Chudasama et al, 2012). For each selection scheme we ran 10 simulations over population sizes 10, 50, and 100 (using a toy datasets described in section 3.0) and took the average number of generations when the process reaches convergence. It is expected to take longer when the GA population size increases. The process terminates after all candidates converge or there appears any candidate whose divergence value is less than 0.02 (calculation of divergence is described in 3.1). The average divergence values after convergence were also compared and the less the value is the higher the optimality the result has.

	Stochastic Universal Selection		Linear Ranking selection	
Population size	Average Number of generations before convergence	Average divergence value after convergence	Average Number of generations before convergence	Average divergence value after convergence
10	31	0.042	10	0.025
50	119	0.017	50	0.012
100	219	0.015	80	0.008

Table 1.2 *Comparison between SUS and linear ranking selection*

The table shows that SUS took longer to reach convergence compared to linear ranking selection and the optimality of its results is neither as satisfactory as the latter. It does not indicate that SUS is a weak selection scheme since it explored more possible solutions during the process.

2.3 Crossover

Crossover is a special operator of GAs that differs them from other algorithms. The crossover operator in this paper is to exchange a selected random block between two attributes in two candidates. Suppose there are N non-identical synthetic datasets in the population of GA that are denoted as $X^n, n \in \{1, \dots, N\}$. Each synthetic dataset consists of K attributes and M cases, $x_{ij}, i \in \{1, \dots, K\}, j \in [1, \dots, M]$. Crossover that occurred between a random pair of synthetic datasets, for example, X^1 and X^2 looks like:

$$\begin{pmatrix} x_{11}^1 & x_{12}^1 & \dots & x_{1k}^1 \\ x_{21}^1 & x_{22}^1 & \dots & x_{2k}^1 \\ x_{31}^1 & x_{32}^1 & \dots & x_{3k}^1 \\ x_{41}^1 & x_{42}^1 & \dots & x_{4k}^1 \\ x_{51}^1 & x_{52}^1 & \dots & x_{5k}^1 \\ x_{61}^1 & x_{62}^1 & \dots & x_{6k}^1 \\ \vdots & \vdots & \vdots & \vdots \\ x_{m1}^1 & x_{m2}^1 & \dots & x_{mk}^1 \end{pmatrix} \quad \begin{pmatrix} x_{11}^2 & x_{12}^2 & \dots & x_{1k}^2 \\ x_{21}^2 & x_{22}^2 & \dots & x_{2k}^2 \\ x_{31}^2 & x_{32}^2 & \dots & x_{3k}^2 \\ x_{41}^2 & x_{42}^2 & \dots & x_{4k}^2 \\ x_{51}^2 & x_{52}^2 & \dots & x_{5k}^2 \\ x_{61}^2 & x_{62}^2 & \dots & x_{6k}^2 \\ \vdots & \vdots & \vdots & \vdots \\ x_{m1}^2 & x_{m2}^2 & \dots & x_{mk}^2 \end{pmatrix}$$

Fig 1.1 Selected random block before crossover

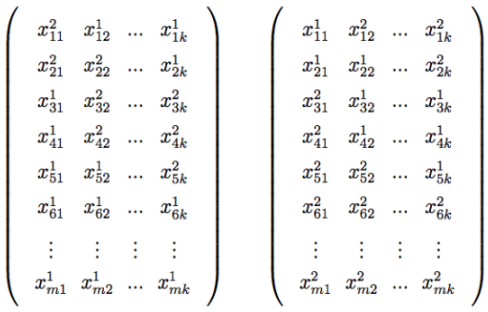


Fig 1.2 Selected random block after crossover

2.4 Mutation

Crossover was once considered as the main mechanism of variation to solution spaces in GAs, but Melanie (1998) argued that researchers should pay the same attention to mutation. Like crossover, there are at least two ways to perform mutation in matrix and this paper only studies mutation on single attributes. The process of mutation is: firstly, the model decides whether mutation happens on a single synthetic attribute based on a given probability. If the decision is "Yes", then the model replaces randomly selected positions by new values that are chosen from a user-determined distribution, which could be the univariate-distribution of the original attribute, uniform distribution or something else. The probability of mutation is usually fixed during the process.

3 Model and Simulations

3.0 Data

The toy dataset used here as a proof of concept a small dataset with 32 records and 4 attributes. There are three reasons to start small: 1) matrix GAs massively increase the workload compared to string GAs, so using a small dataset accelerates the process of programme development and testing; 2) the complexity of matrix GAs results in a higher likelihood of errors and using a small dataset allows to check results line by line; 3) matrix GAs are new in GAs so there are no previous cases to be referred to.

The process of optimisation starts from a set of datasets generated from the univariate distributions of original attributes. The initial GA population size can be any amount greater than 4. The larger the population is the slower the process is but the more variation that exists and the higher chance of discovering the global optima.

3.1 GA model Specification

The approach is designed to generate synthetic datasets for non-hierarchical categorical microdata. Users input a dataset and select the statistics that they wish to preserve. The system will firstly calculate those statistics for the dataset. Then it will generate the required number of synthetic versions using the univariate distributions. The number of synthetic versions is user-determined and will be set up as the size of the initial GA population. These synthetic datasets will then enter the GA process and become iteratively optimised until one or more of them reaches the desired level of closeness to the original data according to the selected statistics.

For the test case used for in this proof of concept, Cramer's V was employed as the single objective to measure the closeness between the synthetic and original data. Shlomo (2009) indicates that Chi-square and Cramer's V (c) are two key statistics to

measure relevance between pairs of categorical attributes. Cramer's V is derived from the Chi-square statistic and can evaluate the level of association between two categorical attributes. So the objective is to minimise the mean of differences of Cramer's V (ϕ_c) between each of the attributes. Suppose the 4 original attributes are X_1, X_2, X_3 and X_4 and its corresponding synthetic versions are X'_1, X'_2, X'_3 and X'_4, the objective can be expressed as a minimisation of the function F:

$$F = \binom{4}{2}^{-1} \sum_{i=1}^{3} \sum_{j=i+1}^{4} |\phi_c(x'_i, x'_j) - \phi_c(x_i, x_j)| \quad (1)$$

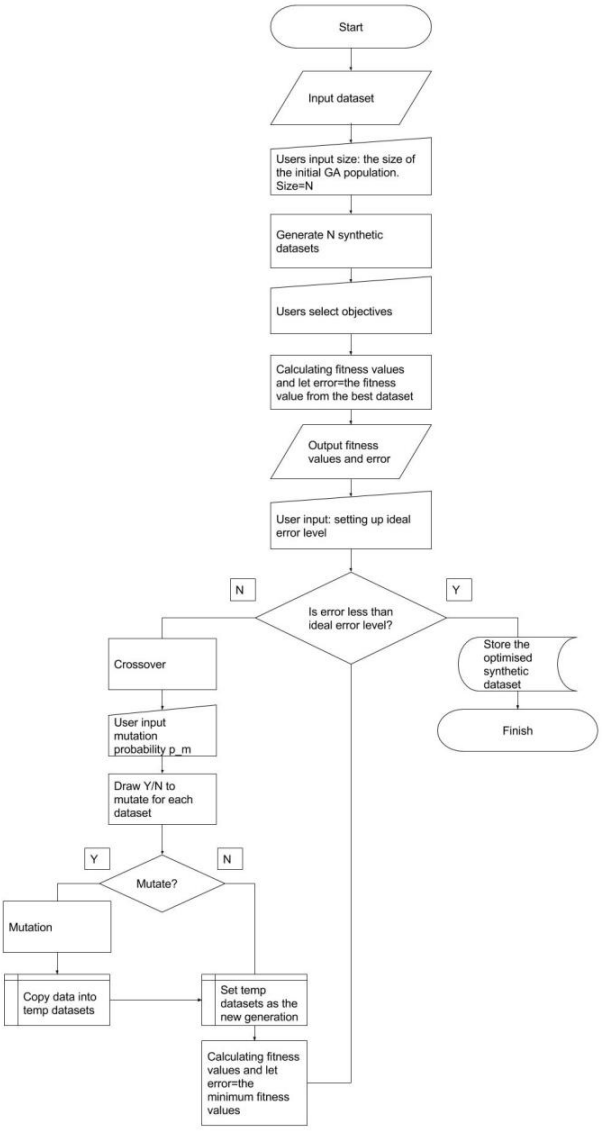


Fig 2.1 Flowchart of the GA model

3.2 Simulations from a general GA approach

The success of an algorithm depends on the details of its operators (Melanie, 1998). Although it might be better in some cases to use a GA model that is specifically designed based on prior knowledge of the particular problem, but general GA models can also work efficiently. The following graphs show how the divergence values (calculated using equation (1)) change over generations using a GA model with linear ranking selection. They show three situations: a GA model with crossover only, a GA model with mutation only and a GA model with both.

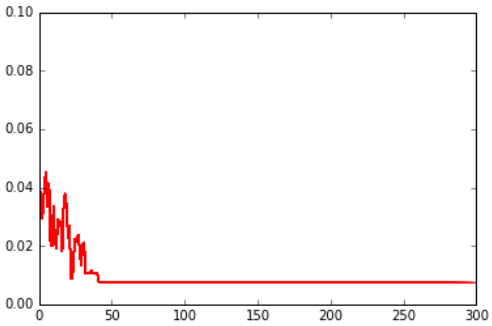


Fig 2.2 Mean fitness values by generation for Simulation with crossover only and toy dataset

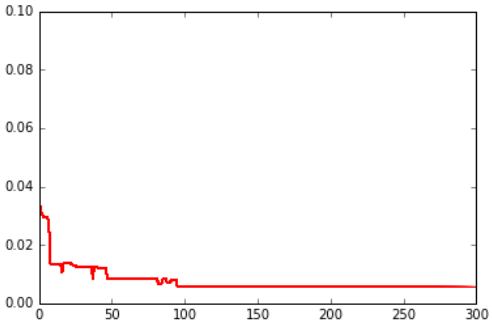


Fig 2.3 Mean divergence values by generation for simulation with mutation (mutation rate=0.1) only

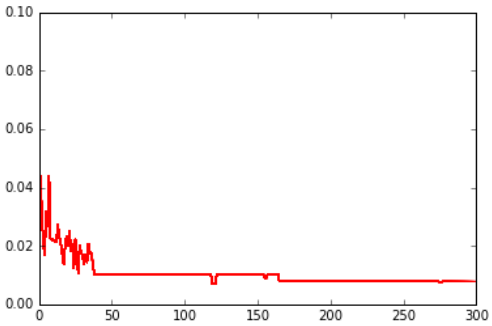


Fig 2.4 Mean divergence values by generation for simulation with crossover and mutation rate=0.1

Comparison between these graphs clearly justifies how crossover and mutation contributes variation to the searching space of optimal solutions and why a parallel search engine has higher chance to find a near-optimum solution. The simulation with crossover only (Fig 2.2) explored more solutions compared to the one with mutation only (Fig 2.3) but it converged quickly and stopped searching new solutions once the

convergence reached. The combination (Fig 2.4) apparently found more solutions before convergence and still searched for more after the convergence reached. Graphs 2.5 to 2.7 compare the three situations above over a dataset with the same four attributes as the toy dataset but with 1000 cases. They may not differ as much as the smaller dataset but still show that the combination of crossover and mutation explore more possible solutions and the convergence speed (in terms of number of generations) is still fast. However, the convergent solution is further away from the global optima.

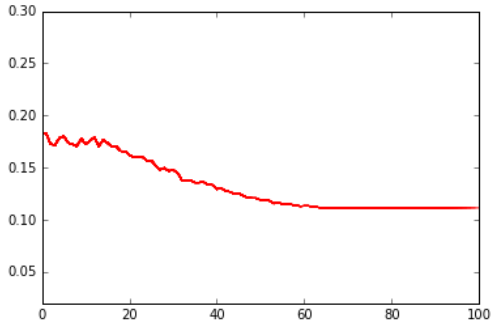


Fig 2.5 Mean divergence values by generation for simulation with crossover only for a larger dataset

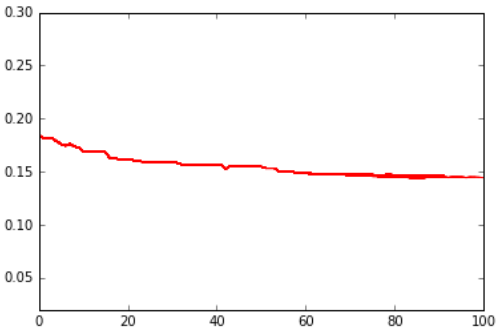


Fig 2.6 Mean divergence values by generation for simulation with mutation (mutation rate=0.1) only for a larger dataset

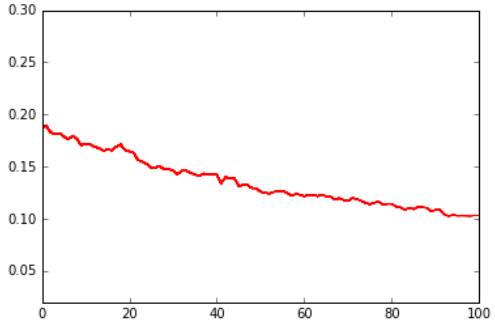


Fig 2.7 Mean divergence values by generation for simulation with crossover and mutation rate=0.1 for a larger dataset

4. Discussion

4.1 Selection

It is difficult to decide how "elitist" the selection schemes should be. A gentle selection operator like SUS assigns probabilities to all candidates to be selected but prolongs the time required for the whole GA population to mature. A restricted operator kills less fit candidates in one shot but results in faster convergence to the local optima. Identifying the appropriate selection operator for synthetic data generation is a topic for future work.

4.2 Adaptive Crossover

At the beginning the trial was implemented on a small GA population; thus, all selected candidates participated crossover to explore the solution space. For a larger GA population like 1000 candidates), an adaptive crossover with adaptive probability p_c is preferred to avoid uncontrollable growth from unordered crossover between candidates. Srinivas and Patnaik (1994) proposed an adaptive crossover rate that changes over generations. Suppose f_c is the divergence value of the best candidate in the generation. The existence of k_c constrains p_c in the range of $[0.0, 1.0]$.

$$p_c = \frac{k_c(f_c - f_{min})}{\bar{f} - f_{min}}, k_c < 1.0$$

There exist some drawbacks of using adaptive crossover. Firstly, there is no clear rule to tell when to change k_c, or whether the population size is large enough to perform adaptive crossover. Furthermore, the formula prevents the best candidate from crossing over with others, which limits evolution if the process is not sufficiently developed or the GA population size is small. In diagram 3.1 we compare the impact from different k_c to p_c over a group of 20 candidates:

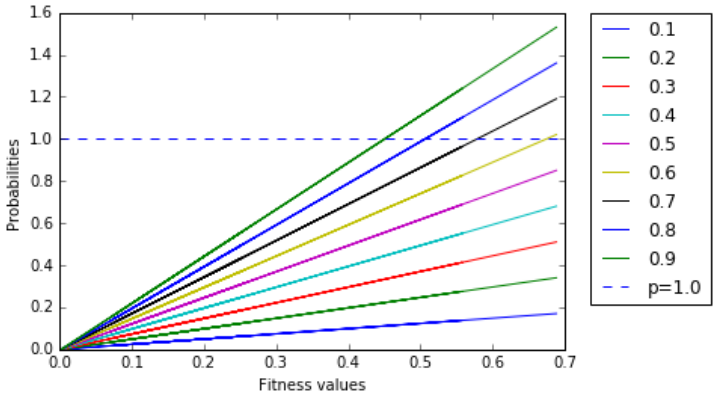


Fig 3.1 Comparison of the effect of difference k_c to p_c

It should be clarified that p_c is not a probability that sums to unity over all candidates but rather is specific to each candidate. The value of p_c returned by the formula can be greater than 1.0, and then needs to be adjusted to 1.0 heuristically. Future work will determine whether to use adaptive crossover by comparing fixed and adaptive crossover over the same GA population.

4.3 Adaptive Mutation

There are two common approaches to design mutation with adaptive probability p_m. One assumes that p_m is adaptive in favour of the difference between individual fitness f_i and the average fitness $\bar{f}$ of the current generation. If $f_i > \bar{f}$ (better than the average), then p_m should be low to keep properties of the good individual. If $f_i < \bar{f}$

(worse than the average), then p_m should be reasonably high to explore a more diverse range of new solutions (Libellin & Alba, 2000). An alternative is that p_m empirically decreases over the number of generations but is independent to individual fitness values. However, this reduces users' control on values of p_m (Thierens, 2002).

Previous research has shown that adaptive mutation is based on the fact that varying probabilities perform better than fixed probabilities in GAs. (see for example Thierens, 2002 for discussion) However, there are some downsides. First of all, it increases computational workload unnecessarily if fixed mutation probability in GAs can also give satisfactory outputs. Secondly, the involvement of adaptive mutation, especially when p_m depends on the individual fitness level, makes the whole process less predictable and impacts the consistency of the whole model (Thierens, 2002). It will require further research to decide whether - and which kind of - adaptive mutation should be used in the GA model for producing synthetic data.

5 Conclusion

We present here GAs as an alternative method for the production of synthetic data. GAs have the feature of being able to automatically optimise synthetic datasets based on given statistical properties and users can update these properties when the model is processing. The process of GAs is to select better candidates (synthetic datasets) and to produce new candidates through crossover and mutation. Since these candidates have higher chances to be selected and bear offspring, the process will eventually converge to an optimal solution. In further study we will make adjustments to different operators in the model and critically we will introduce disclosure control metrics to guarantee the generated dataset provides the statistical properties of the original dataset without duplicating it

References

Blickle, T. & Thiele, L. (1995). *A Comparison of Selection Schemes used in Genetic Algorithms.* Computer Engineering and Communication Networks Lab. Swiss Federal Institute of Technology.

Chen, Y., Elliot, M., Sakshaug, J. (2016). *A Genetic Algorithm Approach to Synthetic Data Production*. Proceedings of the 1st International Workshop on AI for Privacy and Security. Article No. 13.

Chudasama, C.; Shah, S.M. and Panchal, M. (2011) *Comparison of Parents Selection Methods of Genetic Algorithm for TSP*. International Conference on Computer Communication and Networks CSI- COMNET-2011.

Ciriani, V., di Vimercati, S.D.C., Foresti, S., Samarati, P., (2007). *Microdata Protection.* In: Yu T., Jajodia S. (eds.) Secure Data Management in Decentralized Systems, pp. 291–321, Springer, New York.

Cortez, P. (2014). *Modern Optimization with R*; Springer. p.v

Drechsler, J. (2010). *Using support vector machines for generating synthetic datasets; in* Privacy in Statistical Databases; Springer Berlin Heidelberg; pp. 148-161.

Ethem, A. (2014). *Introduction to Machine Learning*. Third Edition, MIT Press.

Liberllin, S., and Alba, P., (2000). *Adaptive mutation in genetic algorithms*, Soft computing. vol.4 pp.76-80.

Melanie, M. (1998). *An Introduction to Genetic Algorithms*. MIT press

Navarro-Arribas, G. and Torra, V. (2015). *Advanced Research in Data Privacy;* Studies in Computational Intelligence. Vol. 567. Springer Switzerland. pp.3-37.

Shlomo, N. (2009), *Releasing Microdata: Disclosure Risk Estimation.* D*ata Masking and Assessing Utility*. Working Paper M09/02. University of Southampton.

Srinivas, M. and Patnaik, L. M. (1994). *Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms*. IEEE Transactions on systems, Man and Cybernetics; 24(4). pp.656-668.

Thierens, D. (2002) *Adaptive mutation rate control schemes in genetic algorithms.* Evolutionary Computation, *2002. CEC '02.* Proceedings of the 2002 Congress on,vol.1, pp. 980 – 98

5.2 The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods

This paper compares three crossover methods that commonly used in GAs: parallelised crossover (namely variable parallelised crossover in the thesis), matrix crossover, and parametric uniform crossover. It aims to find if traditional crossover methods are applicable to data synthesis and if positional bias impacts the efficiency of a GA synthesiser. These methods are applied to three different datasets and compared on the basis of how well they reproduce the relationships between variables in the original datasets.

The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods

Yingrui Chen[1], Mark Elliot[2], and Duncan Smith[3]

[1] University of Manchester, Manchester, M13 9PL, UK
Yingrui.chen@postgrad.manchester.ac.uk
[2] University of Manchester, Manchester, M13 9PL, UK
Mark.Elliot@manchester.ac.uk
[3] University of Manchester, Manchester, M13 9PL, UK
Duncan.G.Smith@manchester.ac.uk

Abstract. Data synthesis is a data confidentiality method which is applied to microdata to prevent leakage of sensitive information about respondents. Instead of publishing real data, data synthesis produces an artificial dataset that does not contain the real records of respondents. This, in particular, offers significant protection against reidentification attacks. However, effective data synthesis requires retention of the key statistical properties of (and respecting the multiple utilities of) the original data. In previous work, we demonstrated the value of matrix genetic algorithms in data synthesis [4]. The current paper compares three crossover methods within a matrix GA: parallelised (two-point) crossover, matrix crossover, and parametric uniform crossover. The crossover methods are applied to three different datasets and are compared on the basis of how well they reproduce the relationships between variables in the original datasets.

Keywords: Genetic algorithms, Data synthesis, Data privacy

1 Introduction

Published data are provided in many formats, although the underlying data are often microdata collected from some population [5]. Confidentiality protection techniques for microdata attempt to camouflage sensitive information in the original data while retaining its statistical properties for analysts. Data synthesis is a protection technique that produces a synthetic dataset that is designed to

preserve the same statistical properties as the original data and provide sufficient variables to allow proper multivariate analyses [1].

The quality of synthetic data is strongly dependent on the design of the synthetic data generator [7]. Properties that are not explicitly included in the generator will not be present in the synthetic dataset (unless they are structurally or statistically related to properties that are, and therefore emerge from the synthesis process). Unforeseen analysis on fully synthetic data may therefore lead to different results from the same analysis on the original data [8].

In this paper we use Genetic Algorithms(GAs) to generate synthetic data. GAs are iterative optimising algorithms that simulate the process of natural evolution. They comprise of three main operators: selection, crossover and mutation. A group of candidate solutions are specified (the initial population). The fitnesses of these candidates are calculated and a selection operator selects a subset of the fitter candidates which are used to generate a new population. In crossover some pairs of these selected candidates are combined (using a variety of methods) to produce new candidate solutions. Some candidates are then subjected to mutation – random changes that will produce changes in fitness. After crossover and mutation we have the new population / generation. The process is repeated a number of times in order to (hopefully) generate fitter solutions than those in the initial population. Crossover and mutation rates can be varied from one iteration to the next, and tuning of these parameters can greatly influence performance.

GAs have been proposed as a potential method to protect respondents' from disclosures from published data. For example, Navarro-Arribas and Torra [9] mentioned that data protection could be treated as an optimisation problem with conflicting objectives and cite GAs as one approach to delivering this. Reasons for using GAs to produce synthetic data are: (i) they are designed to solve problems that have no observable solution space. The *a priori* knowledge required for setting up the initial population is minimal. (ii) GAs are interruptable so do not require complete *a priori* knowledge to set up objectives and, most crucially, (iii) GAs work well at optimising across competing constraints and therefore could, if well designed, have advantages over orthodox statistical model based synthesizers in: ameliorating overfitting, generating emergent properties and accommodating unforeseen analyses.

Matrix GAs are believed to capable of representing and solving more complex problem structures than the more orthodox bitstring GAs [12] [14] [15]. Although GAs have been used in various optimisation problems, the exploration of applications for matrix GAs has been limited. However, given that micro-

data are essentially matrices the production of synthetic microdata seems an obvious application. In previous work we have evaluated the potential for matrix GAs with promising initial results [3] [4]. The current paper explores the performance of three different crossover methods for matrix GAs in producing synthetic data. We consider three datasets, with different data structures, and sampled from different survey populations.

Note that in this initial phase of this research, we are concerned only with optimising the utility of the synthesised data and not with the residual disclosure risk. The rationale for this is twofold: (i) optimising the utility of a synthetic dataset represents a difficult problem by itself and adding in the contrary constraint of disclosure control will introduce further complexity, and (ii) of the two elements the utility problem is the more significant for synthetic data; if this cannot be solved the efficiency of the risk optimisation will be irrelevant. Understanding the properties of the utility optimisation problem before introducing the complexity disclosure control as an objective is therefore the appropriate research strategy.

1.1 Microdata and Contingency Tables

A microdata set for n cases and m variables is usually represented as an n by m matrix indexed $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$. Here we use Y to denote and original dataset and its synthetic version is denoted as X. X shares the same structure as Y as illustrated in Fig. 1.

$$\begin{pmatrix} y_{11} & y_{12} & \ldots & y_{1m} \\ y_{21} & y_{22} & \ldots & y_{2m} \\ y_{31} & y_{32} & \ldots & y_{3m} \\ y_{41} & y_{42} & \ldots & y_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ y_{n1} & y_{n2} & \ldots & y_{nm} \end{pmatrix} \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1m} \\ x_{21} & x_{22} & \ldots & x_{2m} \\ x_{31} & x_{32} & \ldots & x_{3m} \\ x_{41} & x_{42} & \ldots & x_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nm} \end{pmatrix}$$

Fig. 1. Microdata Y and its synthetic version X

For categorical variables the same information can be encoded in a contingency table, which captures the between-variate structure of the candidate. Assume our variables take values in finite sets I_j so that $I = \underset{j \in [1..m]}{\times} I_j$ denotes the possible configurations of the variables. Then a contingency table is an m-dimensional table containing a count for each member of I. For example, if we

denote the jth column of a microdata set Y as $Y_{:,j}$, then the 2-dimensional contingency table constructed from distinct columns $Y_{:,j}$ and $Y_{:,k}$ is $CT(Y_{:,j}, Y_{:,k})$ with entries $n_{r,c}$ is,

$$n_{r,c} = \sum_{i=1}^{n} [Y_{i,j} = (I_j)_r \wedge Y_{i,k} = (I_k)_c] \tag{1}$$

where the square brackets are Iverson brackets and the levels of I_j and I_k are indexed $r \in [1..|I_j|]$ and $c \in [1..|I_k|]$ respectively.

1.2 Objectives

Respecting variable associations in the original data is an important aspect of producing high quality synthetic data. Thus, objective functions are designed based on the differences between synthetic (contingency) tables and original tables in low dimensions. A measure of the difference between a pair of contingency tables is the Jensen-Shannon distance D_{JS} between their normalised (to sum to 1) counterparts[4]. Suppose P and Q are two discrete probability distributions, then $D_{JS}(P||Q)$ is given by:

$$D_{JS}(P||Q) = (\frac{1}{2} D_{KL}(P||M) + \frac{1}{2} D_{KL}(Q||M))^{\frac{1}{2}} \tag{2}$$

where $M = \frac{1}{2}(P + Q)$ and D_{KL} is the well-known Kullback-Leibler divergence.

So our distance measure for a pair of 2-dimensional contingency tables is defined as:

$$\Delta(X, Y, \{j, k\}) = D_{JS}(\frac{1}{n} CT(X_{:,j}, X_{:,k}) || \frac{1}{n} CT(Y_{:,j}, Y_{:,k})) \tag{3}$$

Our first objective function is the mean of these distances over all pairs of variables:

$$F_1(X, Y) = \binom{m}{2}^{-1} \sum_{j=1}^{m-1} \sum_{k=j+1}^{m} \Delta(X, Y, \{j, k\}) \tag{4}$$

[4] Regarding the choice of divergence measure. The Kullback-Leibler divergence cannot be used directly because of the requirement for absolute continuity. Aside from that constraint there was no prior compelling reason for picking any specific measure, and there is no specific empirical work to guide us. The Jensen-Shannon distance was chosen mainly on the basis that it is a true metric, unlike e.g. the Jensen-Shannon divergence. The impact of using alternative measures is another area which future research could explore

Analogous measures are also considered for all 3-dimensional and all 4-dimensional contingency tables. So our other two objectives are defined as:[5]

$$F_2(X,Y) = \binom{m}{3}^{-1} \sum_{S \in P_3([1..m])} \Delta(X,Y,S) \tag{5}$$

$$F_3(X,Y) = \binom{m}{4}^{-1} \sum_{S \in P_4([1..m])} \Delta(X,Y,S) \tag{6}$$

where $P_k(Z)$ denotes the members of the powerset of Z of size K.

The fitness of each candidate is calculated by the Euclidean distance from the synthetic to the original data in the space delineated by the three objective functions. The fitness value is normalized to the range $[0,1]$ by dividing by $\sqrt{3}$.[6] So our overall objective function is:

$$F = \sqrt{3}^{-1} \sqrt{(F_1(X,Y)^2 + F_2(X,Y)^2 + F_3(X,Y)^2} \tag{7}$$

2 Crossover Methods

A crossover operator produces variation in a GA population. The operators considered here will change a pair of individuals by swapping randomly selected sub-matrices. In the case of uniform crossover these sub-matrices will necessarily have dimension 1×1 and we will essentially be swapping individual elements of the matrices.

The three crossover methods presented here have been used previously in various application areas, but not usually compared and certainly not in the context of synthetic data generation. Two of them use the mechanism of two-point crossover where not all sub-matrices (or elements) have an equal probability of being swapped (positional bias). The third operator, uniform crossover, does not suffer from positional bias and is included in order to examine the impact of positional bias on the effectiveness of matrix GA data synthesizers.

Parallelised Crossover The design of parallelised crossover is based on a two-point crossover method from linear GAs, which swaps the elements between two randomly selected crossover points a and b between a pair of bitstrings. Since

[5] Clearly this is not a complete set of possible objectives but these are probably necessary to produce reasonable synthetic categorical data and provide sufficient complexity for our crossover experiments

[6] So on this scale 0 is the best fitness possible and 1 is the worst

solutions that GAs generate are operationally independent (in that a change in one individual has no direct effect on another), crossover and mutation can be parallelised [2]. Parallelised crossover occurs on a single variable and therefore it is possible to have m sub-processors working separately on different variables in the generator. The generator works by randomly choosing a sub-matrix from within a single data column and swapping with the corresponding sub-matrix in the paired candidate. Fig. 2 illustrates parallelised crossover between a pair of candidates X^1 and X^2:

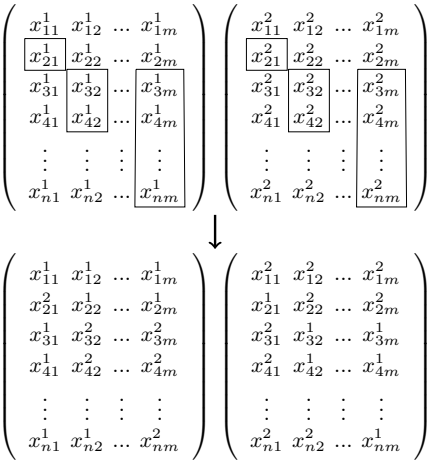


Fig. 2. X^1 and X^2 in parallelized crossover

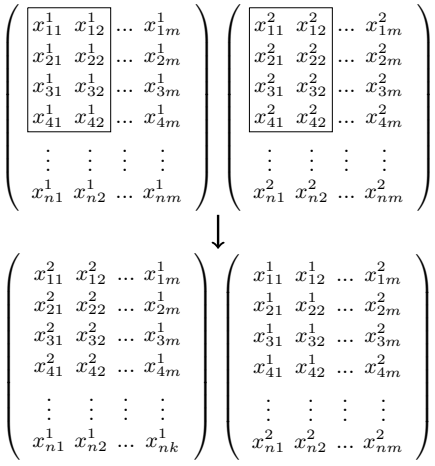


Fig. 3. X^1 and X^2 in Matrix crossover

Matrix Crossover Matrix crossover was first proposed by Wallet et al. [15]. Unlike parallelised crossover, matrix crossover generates crossover points for the rows as well as columns. Thus it swaps elements from a randomly generated sub-matrix (as opposed to the column vectors swapped in parallelised crossover). Fig. 3 illustrates matrix crossover.

Parametric Uniform Crossover (PUC) In PUC, the probability of crossover being applied to each element (1×1 sub-matrix) of the given candidate is determined by a user-specified parameter P_0. Fig. 4 illustrates PUC.

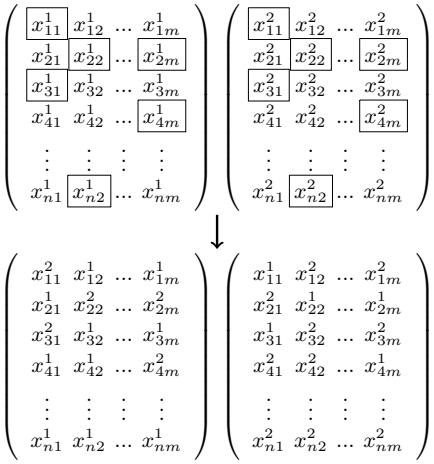


Fig. 4. X^1 and X^2 in PUC

2.1 Positional Bias

Both parallelised crossover and matrix crossover are based on the idea of two-point crossover. In parallelised crossover, the element with row index i will be swapped if, and only if, one of the selection points has index not greater than i while the other has index greater than i. Thus the swap probability is the hypergeometric probability:

$$P(\min(a,b) \leq i < \max(a,b)) = i(n-i+1)\binom{n+1}{2}^{-1} \tag{8}$$

where a and b are the indices of a pair of (distinct) randomly chosen crossover points.

It is trivial to show that this is a montone increasing function of i where $i < \frac{n}{2}$ and a monotone decreasing function of i where $i > \frac{n}{2}$.

For matrix crossover we also select a pair of crossover points for the columns and the probability of an element with index (i, j) being swapped is a product of hypergeometric probabilities. PUC, on the other hand contains no positional bias and it is therefore useful to provide us with an implicit evaluation of the effect of positional bias on the optimising ability of the matrix GA data synthesizer.

3 Empirical Study

3.1 Design

The three crossover methods were compared using three datasets that were each sampled from a different social survey. All three datasets contain 10 variables and 1000 cases. Dataset 1 was sampled from the Crime Survey for England and Wales, 2015-2016 [10] and has 10 binary variables. Dataset 2 was sampled from European Union Statistics on Income and Living Conditions, 2009 [11]. It has 6 binary variables, 1 three-category variable and 3 four-category variables. Dataset 3 was sampled from the Citizenship Survey, 2010-2011 [6]. It contains 1000 cases and 10 variables including 4 binary variables, 2 four-category variables, 2 six-category variables, 1 nine-category variable and 1 eleven-category variable.

For each dataset there was a fixed initial population of 100 candidates that was generated by independently sampling (with replacement) from the univariate distributions of the original data. Deterministic tournament selection[7] was used to select candidates with tournament size $t = 2$.

Synthetic data were generated using GAs with two distinct crossover rates. Matrix crossover used rates of 1.0 and 0.7. The corresponding crossover rates for parallelised crossover and PUC were chosen so that the probability of swapping individual elements was similar.

The synthetic data generator used a low mutation rate ($p_m = 0.01$) to reduce the noise in the final results.[8] Candidates chosen for mutation had a randomly selected sub-matrix swapped with data independently sampled from the original univariate distributions.

For each set of parameters we generated 10 synthetic populations. Each such trial was run for 100 generations.

[7] In generalised tournament selection, candidates are randomly selected into tournaments of size t (with or without replacement). The probability that a candidate wins the tournament and enters crossover is given by $p(1-p)^r$ where p is a parameter (such that $1/t < p \leq 1$) and r is the rank of the candidate's fitness within the tournament. In deterministic tournament selection p is set to 1.

[8] Mutation is another important operator in GA that helps find more promising candidates from the solution space and reduces the risk of becoming caught in local optima. However it can also reduce the fitness of a candidate. Here our focus is on comparing crossover operators so we selected a low mutation rate to reduce the noise in the final results. In future work will examine the relationship between the two operators.

Table 1. Fitness values of the initial population of each of the three test dataset

Fitness of Fixed Population (size=100) for each Data			
	Best fitness	Mean	s.d.
Data 1	0.0734	0.0800	0.0034
Data 2	0.2176	0.2259	0.0031
Data 3	0.2544	0.2610	0.0027

3.2 Experimental Results

Table 2 shows the means and standard deviations of the fittest solutions in the final (100th) populations. The rightmost column shows the fitness of the best individual that was generated over the 10 trials.

The generator used the objective function in Equation 7. The number of individual contingency tables compared depends on the number of variables and would increase substantially if we extended the measure to, say, 5-dimensional tables. Table 2 shows that the fitness of candidates for Dataset 1 is always closer to the original data compared with the other two no matter which crossover operator is used, followed by Dataset 2 and Dataset 3. This is monotonic with the complexity of the data structures of the three datasets. This issue will need further exploration to establish how the degree of complexity affects the viability of GA generated synthesis.

The experimental results also indicate that positional bias does impact the effectiveness of matrix GA generator. All the best means and individuals after 100 generations for the three datasets are generated by the synthesizer with the PUC operator that has $p_0 = 0.3818$. The second best mean and individuals are generated by the same synthesizer with $p_0 = 0.1911$.

Moreover, there was a significant improvement on the initial population no matter which crossover method was used. The increase of fitness (decrease in distance from the original data) indicates that matrix GA is efficient in generating synthetic data with real-coded data or even more complex data structures. Table 3 shows the mean improvement of the fitness of the population from the beginning to the 100th generation over all trials.

4 Conclusions

Our experimental results indicate that PUC performs better than matrix and parallelised crossover in producing synthetic data for all three datasets. This is likely to be due to the lack of positional bias. Results also indicate that the

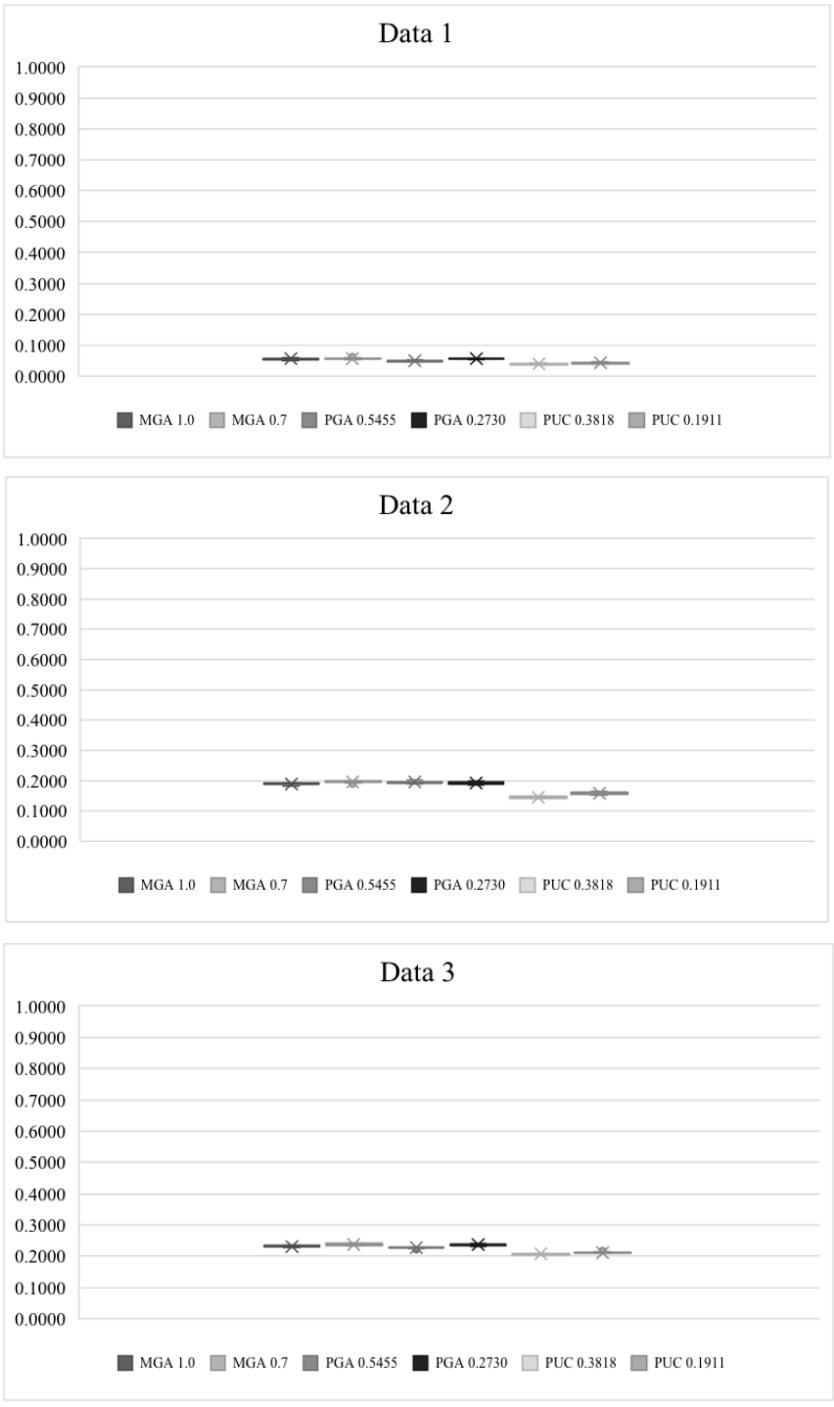


Fig. 5. Box plots of final fitness values of the best individual for Dataset 1, 2 and 3 from ten trials of matrix GA synthetic data generator using different crossover operators

Table 2. Summary statistics from ten trials of matrix GA data synthesizers equipped with three different operators: matrix crossover (MGA), parallelized crossover (PGA) and PUC.

Crossover Type	Crossover Rate	Data	Best Fitness Value		Best Individual
			Mean	s.d.	
MGA	1	Data 1	0.0564	0.0026	0.0517
		Data 2	0.1887	0.0030	0.1818
		Data 3	0.2319	0.0020	0.2281
	0.7	Data 1	0.0579	0.0022	0.0541
		Data 2	0.1958	0.0034	0.1889
		Data 3	0.2375	0.0032	0.2340
PGA	0.5455	Data 1	0.0497	0.0022	0.0472
		Data 2	0.1936	0.0038	0.1885
		Data 3	0.2259	0.0019	0.2213
	0.273	Data 1	0.0561	0.0015	0.0533
		Data 2	0.1929	0.0041	0.1867
		Data 3	0.2363	0.0025	0.2315
PUC	0.3818	Data 1	0.0393	0.0021	0.0362
		Data 2	0.1450	0.0025	0.1408
		Data 3	0.2060	0.0009	0.2048
	0.1911	Data 1	0.0429	0.0022	0.0397
		Data 2	0.1576	0.0038	0.1521
		Data 3	0.2112	0.0020	0.2092

performance of matrix GA on synthetic data generation strongly depends on the structure of data and the number of cases. For example, the optimisation of Dataset 1 is the most effective because it has the simplest data structure (containing only binary variables) compared to Dataset 2 and Dataset 3.

Beyond the issue of positional bias, the overall performance for all three crossover operators in producing synthetic data is reasonable. All approaches significantly improved the fitness of the 100 candidates from the initial population over 100 generations.

Our future research will focus on testing the effectiveness and practicality of the matrix GA generator by introducing adaptive crossover rates, more objectives and larger datasets. A key element missing from these initial experiments has been the assessment of disclosure risk. As outlined in the introduction, this was a rational approach to isolate the difficult problem of optinmising utility. However, a full GA data synthesiser should incorporate risk. Therefore, in future work we will bring measures of disclosure risk into the GA framework. In

Table 3. Mean fitness improvement over ten trials on the best fitness value of initial population

On mean of the best fitness values over all trials	
Data 1	0.0296
Data 2	0.0470
Data 3	0.0362

many ways this is when the GA approach will come into its own. The risk utility trade-off is usually dealt with as a two step-process and optimisng both within a single framework is likely to be more efficient.

Overall, these initial experiments using matrix GA generators to generate synthetic data show that matrix GA is of interest for the problem of data synthesis and for solving problems with higher-dimensional and complex structures in general.

References

1. J. M. Abowd, and J. Lane, New approaches to confidentiality protection: Synthetic da-ta, remote access and research data centers; In Privacy in statistical databases, Springer Berlin Heidelberg, 282-289. (2004)
2. E. Cantu-Paz and D. Goldberg, Efficient Parallel Genetic Algorithms: Theory and Prac-tice, Computer Methods in Applied Mechanics and Engineering, vol.186, 221-238. (2000)
3. Y. Chen, M. Elliot and J. Sakshaug, A Genetic Algorithm Approach to Synthetic Data Production, in Proceedings of the 1st International Workshop on AI for Privacy and Se-curity. Article No. 13. (2016)
4. Y. Chen, M. Elliot and J. Sakshaug, 2017. Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data, UNECE Work Session on Statistical Data Con-fidentiality. `https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.46/2017/2_Genetic_algorithms.pdf`. Last access 20/12/2017. (2017)
5. V. Ciriani, S. D. C. di Vimercati, S. Foresti and P. Samarati, Microdata protection, in Se-cure Data Management in Decentralized Systems, T Yu, and S. Jajodia (edt.) Springer, New York, 291321. (2007)
6. Department for Communities and Local Government, Ipsos MORI. Citizenship Survey, 2010-2011. [data collection]. UK Data Service. SN: 7111, `http://doi.org/10.5255/UKDA-SN-7111-1`, Last access: 20/12/2017. (2012)
7. J. Drechsler, Synthetic data, where do we come from? Where do we want to go?, in Synthetic Data Workshop; Office of National. (2014)

8. O. Maimon and L. Rokach, Data Mining and Knowledge Discovery Handbook, Springer Science and Business Media; p. 704. (2010)
9. G. Navarro-Arribas and V. Torra, Advanced Research on Data Privacy in the ARES Pro-ject; Advanced Research in Data Privacy; Studies in Computational Intelligence. Vol. 567. Springer Switzerland; 3-14. (2015)
10. Office for National Statistics. Crime Survey for England and Wales, 2015-2016. [data collection]. UK Data Service. SN: 8140, http://doi.org/10.5255/UKDA-SN-8140-1. Last access 11/01/2018. (2017)
11. Office for National Statistics. Social Survey Division, Northern Ireland Statistics and Research Agency, Eurostat. (2011). European Union Statistics on Income and Living Conditions, 2009. [data collection]. UK Data Service. SN: 6767, http://doi.org/10.5255/UKDA-SN-6767-1. Last access 11/01/2018. (2009)
12. 3. P. Pongcharoen, A. Khadwilard, and A. Klakankhai, Multi-matrix Real-coded Genetic Algorithm for Minimising Total Costs, in Logistics Chain Network, World Academy of Science, Engineering and Technology International Journal of Economics and Man-agement Engineering, Vol:1, No.11, 574-597. (2007)
13. M. Srinivas and L. M. Patnaik, Genetic algorithms: A Survey, Computer, vol. 27, no. 6, 17-26. (1994)
14. L. Sun, Y. Zhang, and C. Jiang, A matrix real-coded genetic algorithm to the unit commitment problem, in Electric Power Systems Research; 76; pp.716-728. (2006)
15. B. C. Wallet, D. J. Marchette and J. L. Solka, A matrix Representation for Genetic Algorithms, in Proceedings of Automatic Object Recognition IV of SPIE Aerosense, Na-val Surface Warfare Center Dahlgren; Virginia. (1996)

5.3 Matrix GA: building blocks in data synthesis

The efficiency of GA synthesisers was not satisfying using orthodox crossover methods in The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods. In order to find or design more suitable crossover methods, this paper addresses a key question: how 'blocks' are building when GAs synthesise data. Building blocks are a hypothesis within the GA field that asserts that optimum results are from efficiently combining fit blocks of candidates and dismantling unfit ones. A new crossover method and its two variants are introduced and compared in the paper, and the one that is most suitable for data synthesis so far is identified.

Matrix GA: Building Blocks in Data Synthesis

YINGRUI CHEN* and MARK ELLIOT*, Cathie Marsh Institute, University of Manchester

This paper discusses newly arisen issues of finding appropriate crossover methods in genetic algorithm (GA) data synthesisers. Synthetic data is a microdata protection technique. It aims to protect personal confidential information from record-level and prevent possible privacy leakage. The mechanism of synthetic data is to produce new dataset with similar statistical properties but hides individual data subjects confidential records. The potential of using GA in data synthesis has been evaluated previously, but the efficiency is currently unsatisfactory using orthodox crossover parameters. In order to find or design more suitable crossover parameters, this paper addresses a key question: what the building "blocks" of a good solution are when synthesising data using GAs. Within the GA field the"Building blocks" hypothesis asserts that optimum results are most efficiently of combining and fit "blocks" of data and dismantling unfit ones. A new crossover method and two variants are introduced and compared, and the one that is most suitable for data synthesis is identified. As a general conclusion, this example illustrates that the optimum size and shape for the elements (the smallest units of crossover) may depend on the characteristics of the domain.

CCS Concepts: • **Computing methodologies** → **Genetic algorithms**; • **Security and privacy** → *Data anonymization and sanitization.*

Additional Key Words and Phrases: data synthesis, data privacy, genetic algorithms, evolutionary computing

ACM Reference Format:
Yingrui Chen and Mark Elliot. 0. Matrix GA: Building Blocks in Data Synthesis. *ACM Trans. Evol. Learn.* 0, 0, Article 0 (0), 14 pages. https://doi.org/0

1 INTRODUCTION

Microdata often underlies other data formats. It contains information that is collected straightforward from or assembled for individual population units [3]. The recent update of GDPR requires data holders to apply appropriate technical processes on microdata before processing and publishing it (in any format) to avoid individual's confidential information being revealed. The orthodox technology for protecting privacy from microdata is called *statistical disclosure control (SDC)* and data synthesis is one such approach. Instead of removing, aggregating, distorting or modifying records from the original data, data synthesis produces a synthetic dataset that is designed to preserve the statistical properties of the original data and provide sufficient variables to allow proper multivariate analyses. Thus, the quality of synthetic data is strongly dependent on the design of the synthetic data generator. Properties that are not explicitly included in the generator will not be present in the synthetic dataset (unless they are structurally or statistically related to properties that are, and therefore emerge from the synthesis process). Unforeseen analyses on fully synthetic data may therefore lead to different results from the same analysis on the original data.

*Both authors contributed equally to this research.

Authors' address: Yingrui Chen, yingrui.chen@postgrad.manchester.ac.uk; Mark Elliot, mark.elliot@manchester.ac.uk, Cathie Marsh Institute, University of Manchester, Oxford Road, Manchester, M13 9PL.

2688-3007/0/0-ART0 $15.00
https://doi.org/0

Presented mathematically, a microdata set for n cases and m variables is usually represented as an n by m matrix indexed $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$. Here we use Y to denote and original dataset and its synthetic version is denoted as X. X shares the same structure as Y as illustrated in Fig. 1.

$$\begin{pmatrix} y_{11} & y_{12} & \ldots & y_{1m} \\ y_{21} & y_{22} & \ldots & y_{2m} \\ y_{31} & y_{32} & \ldots & y_{3m} \\ y_{41} & y_{42} & \ldots & y_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ y_{n1} & y_{n2} & \ldots & y_{nm} \end{pmatrix} \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1m} \\ x_{21} & x_{22} & \ldots & x_{2m} \\ x_{31} & x_{32} & \ldots & x_{3m} \\ x_{41} & x_{42} & \ldots & x_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nm} \end{pmatrix}$$

Fig. 1. Microdata Y and its synthetic version X

The use of GAs to construct synthetic datasets has been examined in our previous work (see for example [1] and [2]) and shows promise. Here we consider the optimum design of crossover operators for a GA data synthesiser.

2 BACKGROUND

Investigating the potential of implementing a machine learning algorithm for data synthesis is not completely new (see for example [14] and [5]) but these implement different mechanism to GAs. GAs are iterative optimising algorithms that simulate the process of natural evolution. They comprise of three main operators: selection, crossover and mutation. A group of candidate solutions are specified (the initial population) while set up GAs. The fitness of these candidates is then calculated and a selection operator selects a subset of the fitter candidates as "parents". In crossover, these parents are paired by a pre-determined method to produce new candidates. Some of the new candidates are then subjected to mutation – random changes allows them to carry different information from their parents. After crossover and mutation we have the new population / generation. The process is iterated until there is at least one candidate reaches the desirable fitness level. Compared to existing machine learning methods in data synthesis, GAs simulate more complicated aspects from real world problems through interaction between learning agent and environments by given reward functions. Involving these features in data synthesis can allow using both utility and risk objectives to determine whether individuals can survive and reproduce. Moreover, it achieves dynamic monitoring on the trade-off between utility and risks in candidates during the whole process.

2.1 Matrix GAs

Although GAs were initially designed to operate on binary-coded strings, In this case I chose to used matrix as the structure for candidates. Matrices can more appropriately display microdata and can be well applied to multivariate control. It has greater exploration power than linear GAs in complex problems like data synthesis. Matrix GAs were tested to be applicable on higher-dimensional, real-coded problems like [17][15][13]. We demonstrated that Genetic Algorithms (GAs) can be used for generating synthetic data in previous papers (for example [2]).

2.2 Crossover Operators in GAs

Operator design is a vital step for implementing GAs for a particular problem. Although it always possible to use certain primitive schema for GA operators, their efficiency may not be ideal for the

any given problem and sometimes may prevent the whole process from optimising. Our previous work [2] compared three primitive crossover methods for GA with matrix-format candidates: (i) Matrix crossover, which is applied to a whole data matrix; (ii) Parallelised crossover, applied at the level of individual variables within the data and (iii) Parametric uniform crossover (PUC), which is applied to individual elements of the data. In the remainder of this paper we will refer to these methods as the common crossover methods set. The first two on the set (Matrix crossover and Parallelised crossover) were developed from two-point crossover, a traditional method in linear GA. This is defined in definition 2.1.

Definition 2.1 (Two-point Crossover). Suppose any two strings have length l, an integer position k along the string is chosen randomly between $[1, l-1]$. Then two new strings are created by swapping all characters between k 1 and l from the original strings [7].

A critical point for this domain is that two-point crossover carries positional bias, i.e. probabilities of elements in a candidate to be swapped is biased and depends on its position in the candidate, one that is closer to the centre has higher chance to be swapped with another individual. [2] indicated that positional bias does affect the efficiency of any GA synthesiser and concluded that PUC performed the best of the three methods over three test datasets. However, this was not to conclude that PUC is the best crossover operator for generating synthetic data as there are many different possible crossover methods that can used beyond those in the common crossover methods set [16]. In order to find, or design the most suitable crossover operator for GA synthesiser, a fundamental question must be answered: what should be the “building blocks” of a GA synthesiser?

2.3 Building Blocks

When Holland proposed the linear GAs, he described the work of GAs as discovering, emphasising and recombining good “building blocks” of individuals in the GA population. "Blocks" are determinants of the fitness of candidates that have changeable shapes during the crossover procedure which was proposed have three stages in [6] . In stage 1, the algorithm searches for small blocks that confer good fitness. In stage 2, these small blocks are combined to form larger blocks. These larger blocks will eventually take over the population and lead to convergence at stage 3. The initial framework of "building blocks" known as "Schema Theory" was only applicable to simple GA, meaning, a GA with linear, binary and fixed-length candidates, using proportional selection, single-point crossover and single-point mutation. In the linear GA context, a block, or schema, is a template consisting of ones, zeros and asterisks, where asterisks represent wildcards. For example, consider the schema $H = 1 * *0*$: any binary string that has 1 and 0 at the first and fourth position is called an instance of H. The number of fixed bits is referred to as the schema’s order, $o(H)$, and the distance between the first and last fixed bits is referred to as the schema’s distance, $\delta(H)$. In this example, $o(H) = 2$ and $\delta(H) = 4 - 1 = 3$. With schema, the evaluation of fitness values of a given generation in the GA process can be considered to implicitly be the evaluation of the average fitness values of any instances of a schema. In Schema Theory, Holland confirmed that the number of instances from schemas with low order and short length but whose average fitness is greater than the mean fitness value increases exponentially over the run time of the GA [9].

Schema Theory represents an advance in thinking about GAs but is limited in usage; in addition to it only being valid in certain types of GAs, some (for example [7]) also questioned the practicality of the formula as its application was based on an infinite population thus it might not always work on finite populations due to sampling error.

Inspired from Schema Theory, Holland developed another hypothesis that GA works well when instances of low-order, short schema with high fitness would be recombined to instances of high-order, long schema that confer even higher fitness, which is the so called "Building Block

Hypothesis" [8]. “Building blocks” were observed frequently in linear and traditional GAs (see [6] and [8]) but whether the same process happens in the matrix GA (and more specifically within a GA data synthesiser) is unknown.

The reason for considering new crossover methods is that since [2] concluded that parametric uniform crossover (PUC) - where the crossover is applied to a single element (1×1 matrix) in the dataset - outperforms the other two crossover methods, subsequent experiments have shown that when the population is well-evolved, PUC will actually start to reduce the fitness. Fig 2 demonstrates this showing the impact of PUC over 200 generations on the minimum divergence from original data (which for current purposes we equate to the fitness of the best possible candidate) of a starting population of fit candidates and for a population with less fit candidates.

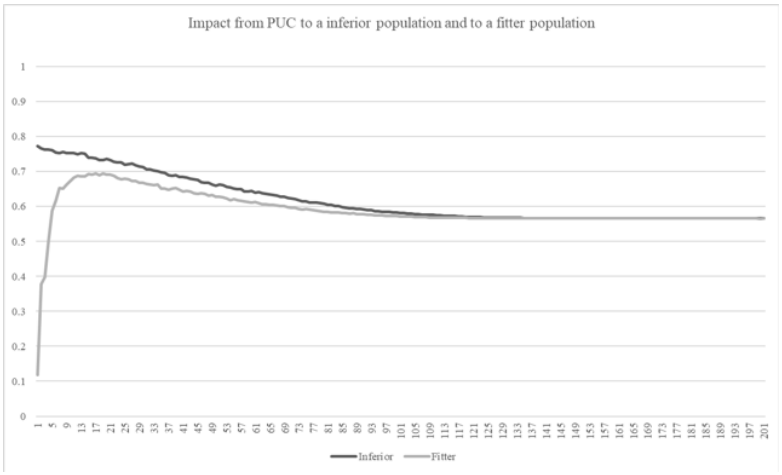


Fig. 2. The impact of PUC on a population of inferior candidates and a population of fit candidates.

The reason for this is that PUC, which can affect every single element in the candidate, is more likely to break up good data structures (compared to the other crossover methods). In the terminology of schema theory, PUC is good at stage 1 where it is selecting small blocks but does not perform well at stage 2 as it continues to select small blocks and breaks up large ones. Therefore, for using GAs for data synthesis it is necessary to find a new crossover method that is capable to do both jobs (and also is free from positional bias).

The goal of designing these was to increase the likelihood of building good blocks. Given that data structures in microdata are based on the relationships within cases it is reasonable to assume it is the relationships between variables (or fields) within cases (or records) that we wish to optimise. Therefore, a good crossover operator will respect within case structure and this, in turn, implies that case orientated crossover methods are worth considering. There are three possibilities.

- *Case-Oriented Parallelised Crossover (CPC)*: This is similar to the variable based parallelised crossover but all exchanged units will be contained within a single case.
- *Whole-CPC*: in this method whole cases only would be swapped.
- *Round-CPC*: This variant allows selected units to wrap around the ends of the case.

In this paper, we report on experiments using each of these three methods.

3 EXPERIMENTS

We report on two experiments in this paper. First, we compared CPC with the common crossover methods set. The experiment tries to address if using case as entity in crossover has better impact to

the efficiency of GA synthesiser than others. Once it was confirmed, we run the second experiments by comparing Whole-CPC, Round-CPC and PUC to assess if any of these method outperform PUC.

All crossover methods were compared using the same three datasets used in [2]. Each sampled from a different social surveys. The three datasets contain ten variables and a thousand cases. Dataset1 was sampled from the [12] and has ten binary variables. Dataset2 was sampled from [11]. It has six binary variables, one three-category variable and three four-category variables. Dataset3 was sampled from [4]. It contains four binary variables, two four-category variables, two six- category variables, one nine-category variable and one eleven-category variable.

3.1 Positional Bias and Crossover Rates

The parameters in the GA generator have the same design as those used in the experiments in [2]. We note that any crossover method that developed from two-point crossover carries positional bias naturally so the probability of a single element to be swapped depends on its position in the candidate and follows hypergeometric distribution:

$$P(a \leq j < b) = j(m - j \quad 1)\binom{m \quad 1}{2}^{-1}$$

, where $a, b, j, m | a \leq b, j = 1, ..., m$ are two random crossover points, index of position and the total number of elements in each case. As in the previous paper, two crossover rates were used for each crossover operator. crossover rate of matrix crossover was set as standard rates of 1.0 and 0.7 (where 1.0 means that every candidate will be selected for crossover, since matrix crossover works on the whole dataset, it is reasonable to put every candidate into competition at the beginning.) For the other operators crossover rates were set at a level which means that the same expected number of elements will be swapped. Therefore, the two crossover rates for CPC and its two variants was 0.5005 and 0.3505.

3.2 Objectives

The synthesiser used the same objectives as in [2]. The between-variates structure in a categorical data can be captured in contingency tables. Assume $I = \times_{j \in [1..m]} I_j$ denotes the possible configurations of the variables that take values from finite sets, a full contingency table is an m-dimensional table containing a marginal for each member of I. Jensen-Shannon distance D_{JS} is a method to give the level of divergence between two multi-dimensional probability distributions, which in effect measures the divergence between a pair of contingency tables. Suppose P and Q are two discrete probability distribution, $D_{JS}(P \parallel Q)$ is defined by:

$$D_{JS}(P \parallel Q) = (\frac{1}{2}D_{KL}(P \parallel M) \quad \frac{1}{2}D_{KL}(Q \parallel M))^{\frac{1}{2}} \tag{1}$$

, where$M = \frac{1}{2}(P \quad Q)$ and D_{KL} is *Kullback-Leibler divergence.*

In this paper, the set of objectives of GA is to minimise mean of D_{JS} between every equivalent pair of 2, 3 and 4-dimensional contingency tables drawn from the synthetic and original data.

$$F_1(X,Y) = \binom{m}{2}^{-1} \sum_{j=1}^{m-1} \sum_{k=j\ 1}^{m} \Delta(X,Y,\{j,k\}) \tag{2}$$

$$F_2(X,Y) = \binom{m}{3}^{-1} \sum_{S \in P_3([1..m])} \Delta(X,Y,S) \tag{3}$$

$$F_3(X,Y) = \binom{m}{4}^{-1} \sum_{S \in P_4([1..m])} \Delta(X,Y,S) \tag{4}$$

, where $P_k(Z)$ denotes the members of powerset of Z of size K.

3.3 Experiment 1

In this experiment case-oriented parallelised crossover (CPC) is introduced and compared with the common crossover method set, especially with the variable-oriented parallelised crossover that was introduced in the previous paper (Chen et al, 2018). CPC has parallel operation on every case in the dataset as illustrated in Fig 3.

$$\begin{pmatrix} \boxed{x^1_{11}\ x^1_{12}} \dots\ x^1_{1m} \\ x^1_{21}\ x^1_{22} \dots\ x^1_{2m} \\ x^1_{31}\ \boxed{x^1_{32} \dots\ x^1_{3m}} \\ x^1_{41}\ x^1_{42} \dots\ x^1_{4m} \\ \vdots\ \vdots\ \vdots\ \vdots \\ x^1_{n1}\ x^1_{n2} \dots\ x^1_{nm} \end{pmatrix} \begin{pmatrix} \boxed{x^2_{11}\ x^2_{12}} \dots\ x^2_{1m} \\ x^2_{21}\ x^2_{22} \dots\ x^2_{2m} \\ x^2_{31}\ \boxed{x^2_{32} \dots\ x^2_{3m}} \\ x^2_{41}\ x^2_{42} \dots\ x^2_{4m} \\ \vdots\ \vdots\ \vdots\ \vdots \\ x^2_{n1}\ x^2_{n2} \dots\ x^2_{nm} \end{pmatrix}$$

$$\downarrow$$

$$\begin{pmatrix} x^2_{11}\ x^2_{12} \dots\ x^1_{1m} \\ x^1_{21}\ x^1_{22} \dots\ x^1_{2m} \\ x^1_{31}\ x^2_{32} \dots\ x^2_{3m} \\ x^1_{41}\ x^1_{42} \dots\ x^1_{4m} \\ \vdots\ \vdots\ \vdots\ \vdots \\ x^1_{n1}\ x^1_{n2} \dots\ x^1_{nm} \end{pmatrix} \begin{pmatrix} x^1_{11}\ x^1_{12} \dots\ x^2_{1m} \\ x^2_{21}\ x^2_{22} \dots\ x^2_{2m} \\ x^2_{31}\ x^1_{32} \dots\ x^1_{3m} \\ x^2_{41}\ x^2_{42} \dots\ x^2_{4m} \\ \vdots\ \vdots\ \vdots\ \vdots \\ x^2_{n1}\ x^2_{n2} \dots\ x^2_{nm} \end{pmatrix}$$

Fig. 3. Two datasets before and after CPC

Fig 4, 5, 6 show box-plots capturing the mean (over ten trials) of the best individuals after 100 generations. MGA, PGA, PUC and CP represent Matrix crossover, Parallelised crossover, Parametric uniform crossover and CPC respectively. They indicate that CPC, was outperformed by PUC. On the basis of previous work this is likely to be due to its positional bias. However, CPC outperformed the other two crossover methods. This demonstrates that positional bias may not be the only factor that influences the efficiency of GA synthesiser. It is reasonable to say that an unknown factor also plays critical role, we might hypothesise that this is related to how the operator is affecting good quality "blocks".

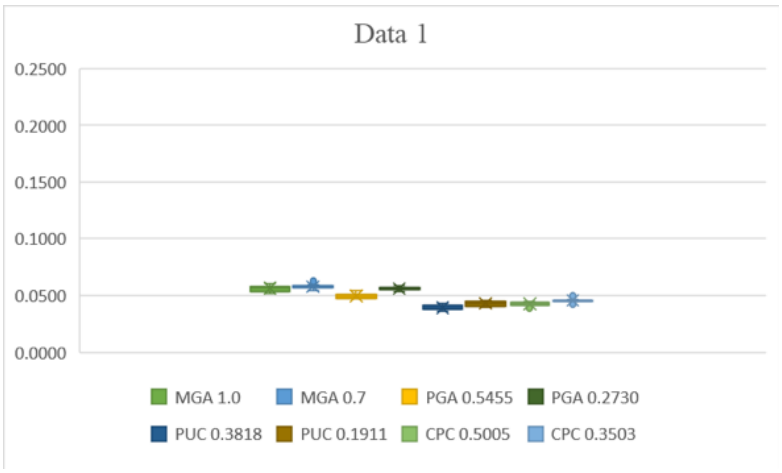


Fig. 4. Box plots of the average final fitness values of the best individual for Dataset1 from ten trials of matrix GA synthetic data generator using different crossover operators. NB (Fitness scales are measures of divergence so 0 is optimum)

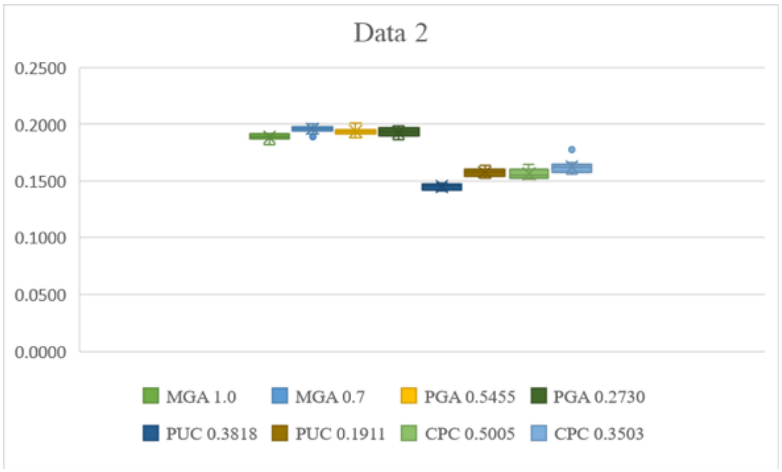


Fig. 5. Box plots of the average final fitness values of the best individual for Dataset2 from ten trials of matrix GA synthetic data generator using different crossover operators. NB (Fitness scales are measures of divergence so 0 is optimum)

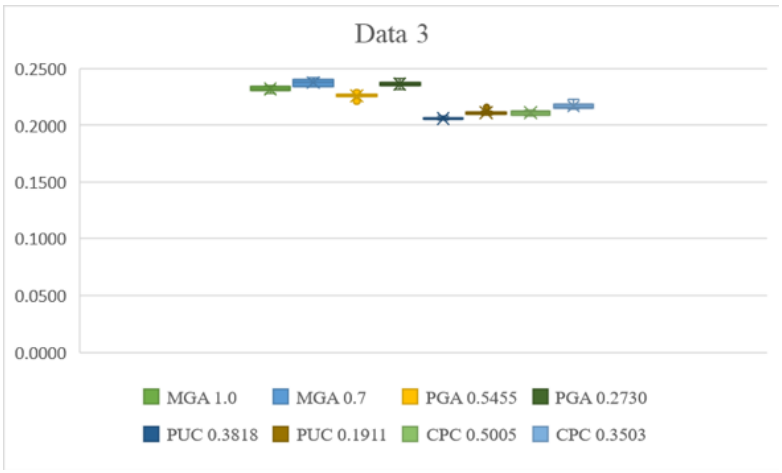


Fig. 6. Box plots of the average final fitness values of the best individual for Dataset3 from ten trials of matrix GA synthetic data generator using different crossover operators. NB (Fitness scales are measures of divergence so 0 is optimum)

3.4 Experiment 2

We designed two new crossover methods based on CPC. They involve the same mechanism as CPC but are free from positional bias. Round-CPC (Fig 7) uses the same method as CPC to select the two endpoints from a case firstly. Then it decides to swap the elements between or out of the two endpoints with equal probability; this gives an equal chance of selection to all elements in a row (case) regardless of position.

$$\left(\begin{array}{cccc} \boxed{x^1_{11}} & \boxed{x^1_{12}} & \dots & x^1_{1m} \\ x^1_{21} & x^1_{22} & \dots & x^1_{2m} \\ \boxed{x^1_{31}} & \boxed{x^1_{32}} & \dots & \boxed{x^1_{3m}} \\ x^1_{41} & x^1_{42} & \dots & x^1_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ x^1_{n1} & x^1_{n2} & \dots & x^1_{nm} \end{array}\right)\left(\begin{array}{cccc} \boxed{x^2_{11}} & \boxed{x^2_{12}} & \dots & x^2_{1m} \\ x^2_{21} & x^2_{22} & \dots & x^2_{2m} \\ \boxed{x^2_{31}} & \boxed{x^2_{32}} & \dots & \boxed{x^2_{3m}} \\ x^2_{41} & x^2_{42} & \dots & x^2_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ x^2_{n1} & x^2_{n2} & \dots & x^2_{nm} \end{array}\right)$$

$$\downarrow$$

$$\left(\begin{array}{cccc} x^2_{11} & x^2_{12} & \dots & x^1_{1m} \\ x^1_{21} & x^1_{22} & \dots & x^1_{2m} \\ x^2_{31} & x^2_{32} & \dots & x^2_{3m} \\ x^1_{41} & x^1_{42} & \dots & x^1_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ x^1_{n1} & x^1_{n2} & \dots & x^1_{nm} \end{array}\right)\left(\begin{array}{cccc} x^1_{11} & x^1_{12} & \dots & x^2_{1m} \\ x^2_{21} & x^2_{22} & \dots & x^2_{2m} \\ x^1_{31} & x^1_{32} & \dots & x^1_{3m} \\ x^2_{41} & x^2_{42} & \dots & x^2_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ x^2_{n1} & x^2_{n2} & \dots & x^2_{nm} \end{array}\right)$$

Fig. 7. Two datasets before and after round-CPC

Instead of swapping elements between or out of two random end points, whole-CPC (Fig 8) exchanges the entire case based on pre-determined probability. Since each case in the dataset is selected independently, this crossover method also has no positional bias.

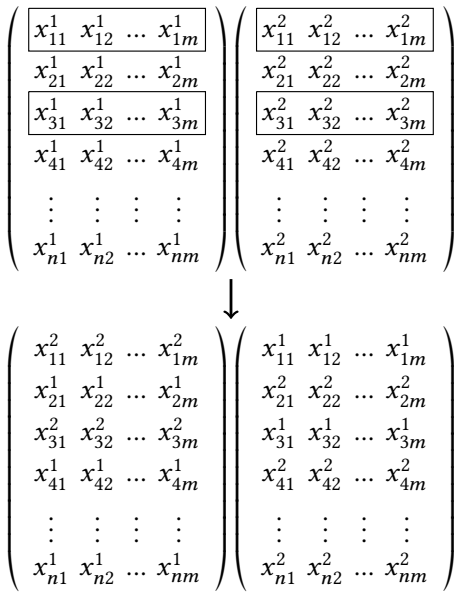


Fig. 8. Two datasets before and after whole-CPC

In this experiment, we compared the two new case-based crossover methods with PUC and CPC. Both Round-CPC and whole-CPC outperform CPC over all the three datasets, their impact on the efficiency of the GA generator is similar to PUC, (which still produces the best results among all crossover operators in Dataset2). We have found that, compared with matrix-based and variable-based crossover methods, case-based crossover show greater efficiency in optimising the population. One conjecture is that the fitness functions assess the divergence from original data by the number of cases in multi-dimensional contingency table, where cases have more straightforward influence than variables.

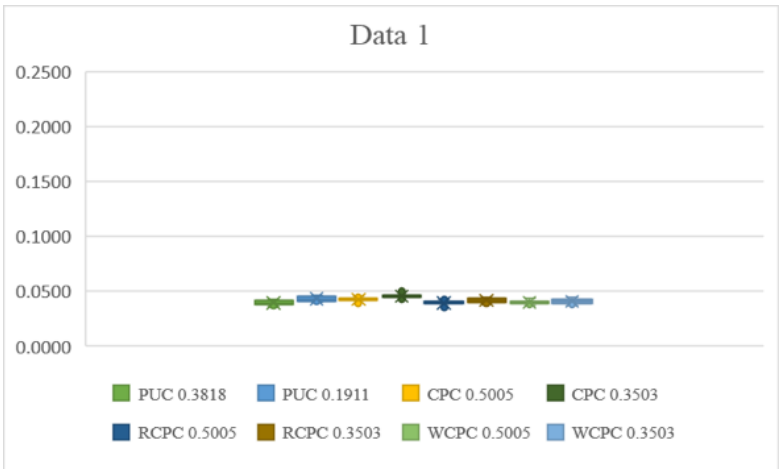


Fig. 9. Box plots with different scales of final fitness values of the best individual for Dataset1 from ten trials of matrix GA synthetic data generator using different crossover operators. NB (Fitness scales are measures of divergence so 0 is optimum)

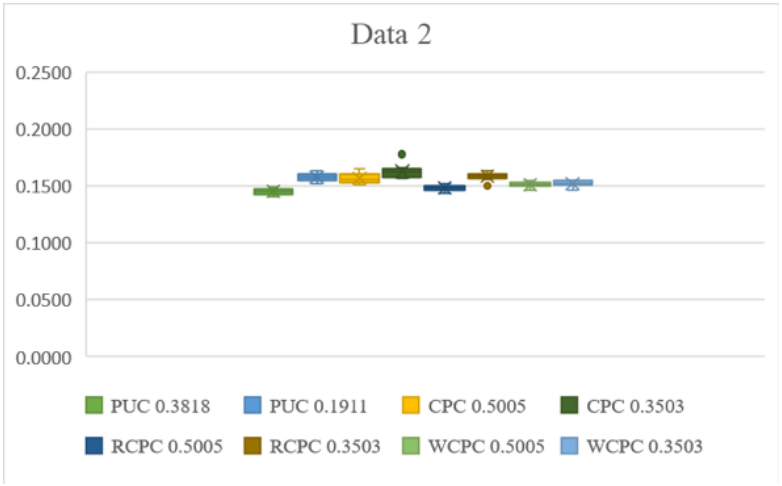


Fig. 10. Box plots with different scales of final fitness values of the best individual for Dataset2 from ten trials of matrix GA synthetic data generator using different crossover operators. NB (Fitness scales are measures of divergence so 0 is optimum)

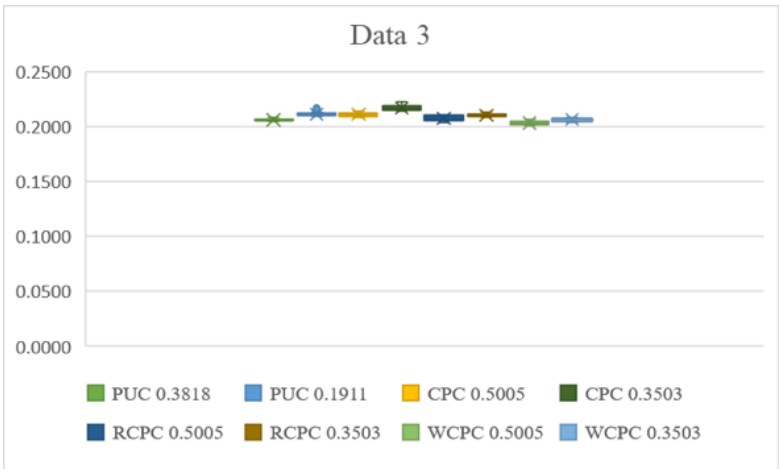
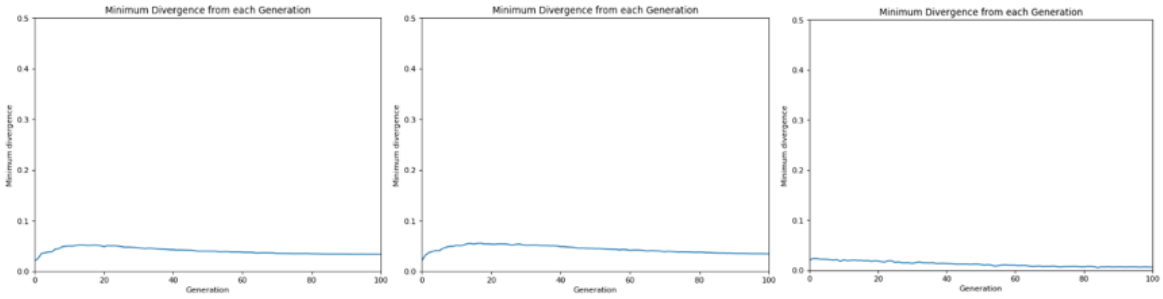


Fig. 11. Box plots with different scales of final fitness values of the best individual for Dataset3 from ten trials of matrix GA synthetic data generator using different crossover operators. NB (Fitness scales are measures of divergence so 0 is optimum)

Since we used the same setting as the experiment carried out by [2], we can only conclude that there is no significant improvement of the efficiency in the GA generator when using round-CPC or whole-CPC compared to PUC. One conjecture is that the experiment was terminated after the 100th generation, when the population was not well developed. In other words, the generator was in the stage of searching for good "blocks" that had not yet joined into a larger one that confer higher fitness. In order to deal with this issue we carried out a third experiment on a mature population.

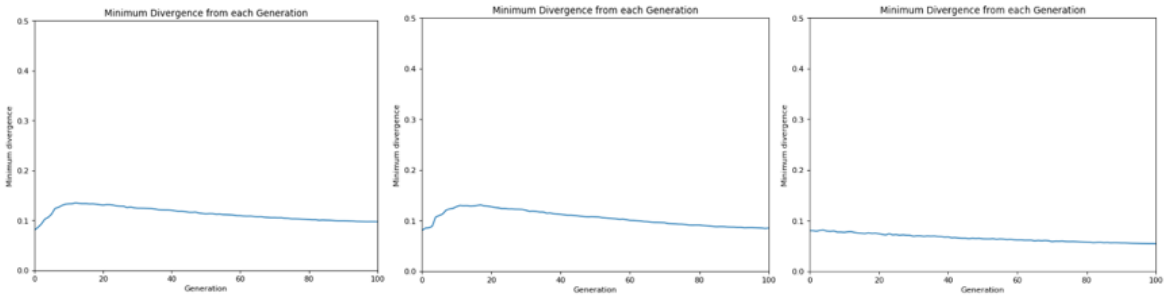
3.5 Additional experiment: Comparison of Sustainability

A further experiment is carried out to test whether the impact of the three crossover methods is sustainable when the population is closer to optimal. This ensures that the advantage of a certain crossover method persistent. We used the same three datasets in this section, but changed the initial population of each dataset compared to experiment 2. Each initial population now consists of candidates generated using a CART synthesiser[10]. These candidates are closer to the original data and are therefore believed to carry large blocks that confer good fitness. We compared the performance of all three crossover methods optimising these populations in the 100th generation (See Figs 12, 13 and 14).



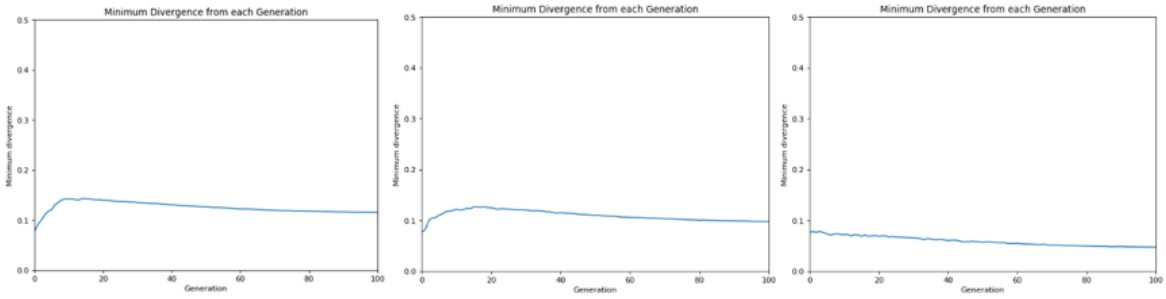
(a) PUC (b) Round-CPC (c) Whole-CPC

Fig. 12. Performance of PUC, Round-CPC and Whole-CPC on a sufficiently optimal population for Dataset1. NB (Fitness scales are measures of divergence so 0 is optimum)



(a) PUC (b) Round-CPC (c) Whole-CPC

Fig. 13. Performance of PUC, Round-CPC and Whole-CPC on a sufficiently optimal population for Dataset2. NB (Fitness scales are measures of divergence so 0 is optimum)



(a) PUC (b) Round-CPC (c) Whole-CPC

Fig. 14. Performance of PUC, Round-CPC and Whole-CPC on a sufficiently optimal population for Dataset3. NB (Fitness scales are measures of divergence so 0 is optimum)

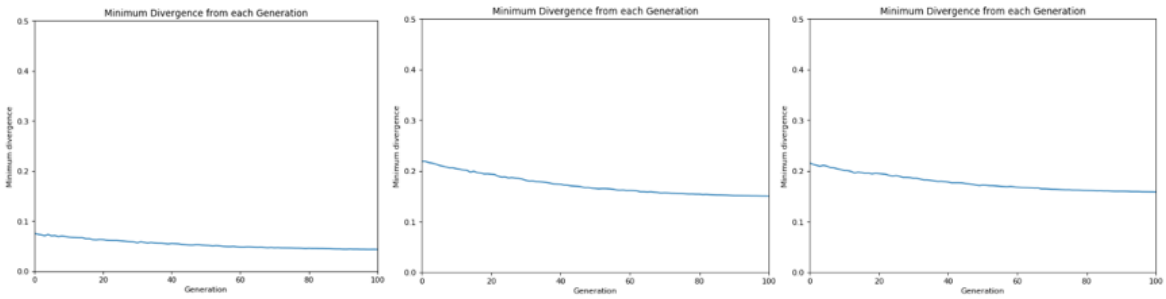
The results indicate that, except for whole-CPC that continues to reduce the divergence fo the fitter population from the original dataset, the other two present undesirable trends of reducing the best fitness in the population for the first 10-20 generations. It seems as if, although the three

crossover methods have similar impact to the efficiency of GA generator in the first 100 generations, not all of them can sustain it when the population is well developed. In these experiments, PUC and round-CPC show less capacity to retain good structures from candidates, which causes them to be less likely to reach the optima than whole-CPC. the most likely explanation for this is that, in a fitter population, candidates are formed by large "blocks" and a crossover operator that swaps blocks of any size smaller than those large blocks will break up the good data structure and diminish the candidate's fitness.

3.6 Illustration of Building Block Process in GA synthesiser

Combined with the experimental results from [2], it is reasonable to conjecture that building blocks in GA synthesiser are produced in one of two stages.

- In the first stage, the process of building blocks happens with each element (1×1 matrix) simultaneously at first and here the process is similar to linear GA as summarised by [6]. This explains why PUC has the best performance among all crossover operators in the first 100 generations (given an initial population where the multivariate relationships in the original data are not respected). Blocks that confer higher fitness are combined with others and gradually at the later stage they merge to larger blocks that may become the whole row (a case) or even have higher dimensions (two dimensions in our case), and at this point the effect from PUC becomes negative.
- The second way to simply set the minimum shape of blocks as a row (a case) and reduces the dimension in candidates to one in order to follow the process in linear GA. This also explains why the whole-CPC also appears to work well in the first 100 generations without mutation (Fig 15).



(a) Dataset1 (b) Dataset2 (c) Dataset3

Fig. 15. Performance of Whole-CPC on a non-optimal population for Dataset1, 2 and 3 without mutation operator.

4 CONCLUSION

In this paper, we reran the experiments from [2] using three new, case-level, crossover operators and illustrated the process of building blocks in GA synthesiser in microdata. We concluded that, case-level crossover methods are suitable in building good blocks and their performances are similar to PUC and better than matrix crossover or variable-oriented parallelised crossover. The fitness levels from PUC, CPC and round-CPC are not sustainable. They are therefore unsuitable for use on developed populations. However, whole case crossover continues to produce increased performance with mature populations.

Combining the experimental results from our previous and this paper, we gave two hypothesises about how "building blocks" happen in GA synthesiser and concluded that whole-CPC is the most appropriate method for this problem.

REFERENCES

[1] Yingrui Chen, Mark Elliot, and Joseph Sakshaug. 2017. Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data. Retrieved February 28, 2018 from https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.46/2017/2_Genetic_algorithms.pdf.Lastaccess20/12/2017

[2] Yingrui Chen, Mark Elliot, and Duncan Smith. 2018. The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods. In *Proceedings of Privacy in Statistical Database 2018*. Springer.

[3] Valentina Ciriani, S. Vimercati, S. Foresti, and P. Samarati. 2007. *Microdata Protection*. Vol. 33. 291–321. https://doi.org/10.1007/978-0-387-27696-0_9

[4] Communities and Local Government. 2012. Citizenship Survey. Retrieved December 12, 2017 from http://doi.org/10.5255/UKDA-SN-7111-1

[5] Jörg Drechsler. 2010. Using Support Vector Machines for Generating Synthetic Datasets. In *Proceedings of the 2010 International Conference on Privacy in Statistical Databases (PSD'10)*. Springer-Verlag, Berlin, Heidelberg, 148–161. http://dl.acm.org/citation.cfm?id=1888848.1888865

[6] Stephanie Forrest and Melanie Mitchell. 1993. Relative Building-Block Fitness and the Building-Block Hypothesis. In *Foundations of Genetic Algorithms*, L. DARRELL WHITLEY (Ed.). Foundations of Genetic Algorithms, Vol. 2. Elsevier, 109 – 126. https://doi.org/10.1016/B978-0-08-094832-4.50013-1

[7] David E. Goldberg and Jon Richardson. 1987. Genetic Algorithms with Sharing for Multimodal Function Optimization. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 41–49. http://dl.acm.org/citation.cfm?id=42512.42519

[8] John H. Holland. 1992. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA.

[9] Melanie Mitchell. 1998. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA. 124–131 pages.

[10] Beata Nowok, Gillian Raab, and Chris Dibben. 2016. synthpop: Bespoke Creation of Synthetic Data in R. *Journal of Statistical Software, Articles* 74, 11 (2016), 1–26. https://doi.org/10.18637/jss.v074.i11

[11] ONS. 2009. European Union Statistics on Income and Living Conditions. Retrieved November 11, 2018 from http://doi.org/10.5255/UKDA-SN-6767-1

[12] ONS. 2017. Crime Survey for England and Wales. Retrieved November 11, 2018 from http://doi.org/10.5255/UKDA-SN-8140-1

[13] Pupong Pongcharoen, Aphirak Khadwilard, and Anothai Klakankhai. 2007. Multi-matrix Real-coded Genetic Algorithm for Minimising Total Costs in Logistics Chain Network. *International Journal of Economics and Management Engineering* 1, 11 (2007). https://publications.waset.org/15279/pdf

[14] Jerome P. Reiter. 2005. Using CART to Generate Partially Synthetic, Public Use Microdata. *Journal of Official Statistics* 21 (01 2005).

[15] Liyong Sun, Yan Zhang, and Chuanwen Jiang. 2006. A matrix real-coded genetic algorithm to the unit commitment problem. *Electric Power Systems Research* 76, 9 (2006), 716 – 728. https://doi.org/10.1016/j.epsr.2005.10.005

[16] Anantkumar Umbarkar and Pranali Sheth. 2015. CROSSOVER OPERATORS IN GENETIC ALGORITHMS: A REVIEW. *ICTACT Journal on Soft Computing (Volume: 6 , Issue: 1)* 6 (10 2015). https://doi.org/10.21917/ijsc.2015.0150

[17] Bradley C. Wallet, David J. Marchette, and Jeffery L. Solka. 1996. Matrix representation for genetic algorithms. In *Automatic Object Recognition VI*, F. A. Sadjadi (Ed.), Vol. 2756. 206–214. https://doi.org/10.1117/12.241153

5.4 Impact of Full Contingency Table in Data Synthesis

This paper discusses the potential of using only full contingency table as the utility objective in GA synthesisers to optimising data utility. The final output is compared with synthetic data generated by CART. The finding shows that GA-synthetic data has better utility than CART synthetic-data with taking full contingency table as the only utility objective. Moreover, since this property measures divergence between two datasets, the potential of use it as risk measure is also discussed in the last section.

The Impact of using the Full Contingency Table as an Objective for Data Synthesis

Yingrui Chen
Mark Elliot
University of Manchester

Abstract: Data synthesis allows the publication of an artificial copy of a dataset that maitains the statistical properties to the original but contains no sensitive records. It has been understood that synthetic data only carries statistical properties specified by the generating model or properties derived from ones that accounted in that generating model. This issue creates challenges in designing synthesisers (Criani et al, 2007). This paper discusses the potential of using only the full contingency table as the unique objective of a data synthesiser in order to optimise data utility. Our final output is compared with synthetic data generated by CART synthesiser in the R package *synthpop* (Nowok et al, 2006). The finding shows that GA-synthetic data has better utility than CART synthetic-data with taking full contingency table as the only utility objective. Moreover, since this property measures divergence between two datasets, the potential of use it as risk measure will be discussed in the last section.

1 Introduction

Data synthesis provides significant protection against reidentification attacks and therefore is a potentially valuable tool in data protection. An effective data synthesiser requires retention of as many key statistical properties of (and respecting the multiple utilities of) the original data as possible. Orthodox methods of data synthesis require the building a model of the original data and then draw from that model to build the synthetic data. This implies that only structure that is specified by the generation model is captured in the synthetic data.

Genetic algorithms (GAs) are a branch of natural computing that simulate evolutionary processes therefore they deploy three operators: selection, crossover and mutation. Starting with group of candidate solutions (the initial population). The fitness of these candidates is firstly evaluated. Fitter candidates are randomly paired using a crossover operator to generate new candidate solutions. These candidates are then passed to mutation operator and given a probability to mutate. The offspring form the new population and the process is repeated until a desired optimality achieved. The standard process of GAs is illustrated in Figure 1.1.

The general efficiency in generating synthetic data has been evaluated in previous works (Chen et al, 2017, 2019). Objective design is a vital part of building GAs and this is particularly true in the application to data synthesis because it specifies the extent to which the final output is similar to the original dataset. In any data synthesis process, statistical properties that are not explicitly included in the synthesising model will not be present in the synthetic dataset, unless they are structurally or statistically related to properties that are and therefore emerge from the synthesis process (see Drechsler, 2014, Chen et al, 2018).

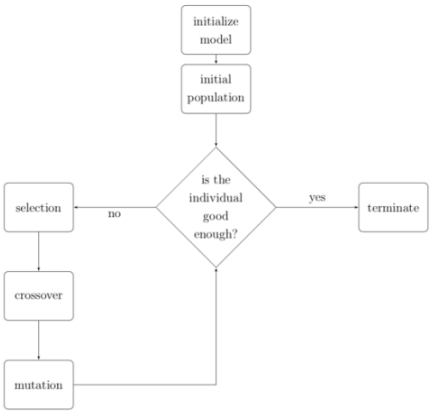


Fig 1.1 The flowchart of GA process

The contingency table is a table of counts (or proportions) of categories in a set of categorical variables, which is an essential measure of the utility of the synthetic data (Barak et al, 2007). In this paper, we aims to use a full contingency table contingency table to capture the variation of all variables in a categorical dataset and compare its divergence between original and synthetic data . We will also examine if full contingency table might be used as the only objective to measure the utility of synthetic data in GA synthesisers as we assume that if two datasets have similar full contingency tables then they also have similar statistical properties. The utility of synthetic datasets will then be evaluated by several tests, and be compared with CART-generated synthetic data. CART is a strong, non-parametric prediction tool generally used to synthesise data (Reiter, 2005) and it is capable to produce high-utility synthetic data.

2 Utility Objective and Evaluation Test

2.1 Utility Objective

The between-variates structure in a categorical data can be captured by a contingency table. Assume $\mathrm{I} = \underset{\mathrm{j}\in[1..\mathrm{m}]}{\times} \mathrm{I_j}$ denotes the possible configurations of the variables that take values from finite sets, a full contingency table is an m-dimensional table containing a marginal for each member of I. Jensen-Shannon distance $\mathrm{D_{JS}}$ is a method to give the level of divergence between two multi-dimensional probability distribution, which is capable measure the divergence between a pair of contingency tables. Suppose P and Q are two discrete probability distribution, $\mathrm{D_{JS}}(P||Q)$ is defined by:

$$\mathrm{D_{JS}}(P||Q) = \left(\frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M)\right)^{\frac{1}{2}}$$

, where $M = \frac{1}{2}(P + Q)$ and D_{KL} is Kullback-Leibler divergence.

In this paper, the single objective of GA is to minimise $\mathrm{D_{JS}}$ between full contingency tables CT_{FULL} from the synthetic and original data.

$$F(X,Y) = D_{JS}\left(\frac{CT_{FULL}(X)}{N}||\frac{CT_{FULL}(Y)}{N}\right)$$

2.2 Evaluation Test

Visualisation comparison to original data is commonly used in evaluating synthetic data. In order to demonstrate our hypothesis that full contingency table can be used as the single utility objective in data synthesis modelling, utility of synthetic data will also be evaluated by

several statistical properties that are not explicitly related to full contingency table (there is no theoretical proof). Logistic regression is a form of regression modelling that has a categorical variable as the response. The simplest form of logistic regression model is the binary logistic regression model, which has a two-category output variable. Logistic regression models predict the log-odds of the probability of the categories from output variable. Suppose Y_{binary} is a binary variable in original dataset $\boldsymbol{Y}$ that interests analysts. A logistic regression model that has some of the other variables in $\boldsymbol{Y}$ as the explanatory variables of Y_{binary} can be expressed as:

$$\ln\left(\frac{p(Y_{binary}=1)}{1-p(Y_{binary}=1)}\right)=\beta_0+\beta_1 Y_1+\cdots+\beta_w Y_w\,,1\le w\le m$$

Suppose $\boldsymbol{X}$ is the synthetic version of $\boldsymbol{Y}$ and X_{binary} is the synthetic version of Y_{binary}, the corresponding logistic regression model in $\boldsymbol{X}$ is:

$$\ln\left(\frac{p(X_{binary}=1)}{1-p(X_{binary}=1)}\right)=\beta_0{}'+X_1+\beta_2{}'X_2+\cdots+\beta_w{}'X_w\,,1\le w\le m$$

the logistic regression models can be compared by the correlation U_1 between predicted logit probability from the same training dataset. In this case, the training dataset can be simply set as the all cases from explanatory variables in the original dataset. The closer U_1 to 1, the similar the predictions from two logistic regression model are.

$$U_1=Corr\left((\beta_0+\beta_1 Y_1+\cdots+\beta_w Y_w,),\left(\beta_0^{'}+\beta_1^{'}Y_1+\cdots+\beta_w^{'}Y_w\right)\right),\qquad 1\le w\le m$$

Multiple correspondence analysis (MCA) is another post hoc test which carries out a pattern analysis on categorical variables. It explores underlying structures of a set of nominal variables. By applying correspondence analysis (CA) on the data matrix X that has only nominal variables, MCA will give two sets of factor scores on variables and cases. These factor scores scaled their variance to equal to their eigenvalues. Since MCA can also work on dimension reduction, it is a convention to set the number of factors in MCA small. This paper will compare factor scores from two factors of original data and its synthetic versions. These scores will be visualised on projection maps for comparison (Abdi & Valentin, 2007).

3 Experiments

The experiment was carried in a trial dataset that has 4 variables shown as the table below. The output synthetic data will be generated by GA synthesiser with full contingency table as the only objective (GA-synthetic data). It will be compared with the synthetic version generated by the CART (CART-synthetic data).

'coalmine'	Categorical: "0", "1", "2"
'nucar'	Categorical: "0", "1", "2"
'agegrp'	Categorical: "0", "1", "2", "3", "4", "5"
'ltillr'	Categorical: "0", "1",

Table 3.1 variables and their corresponding levels in the trial data

3.1 Utility Test

The GA synthesiser destined when its population has evolved into an acceptable level. The output data has 0.08 in D_{JS} in the full-contingency table objective $F(X, Y)$. The histogram confirms that the synthetic data has very similar univariate distribution in each variable compared with the original data (Fig 3.2 and Fig 3.3).

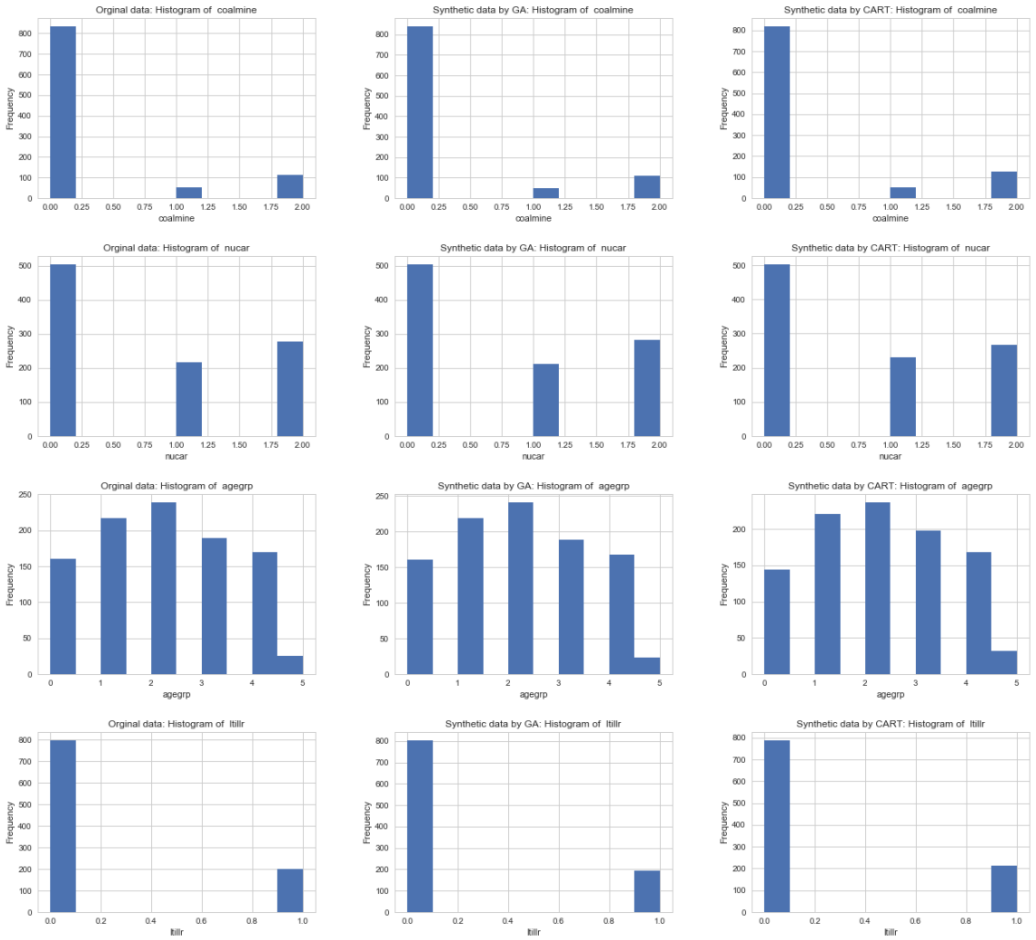
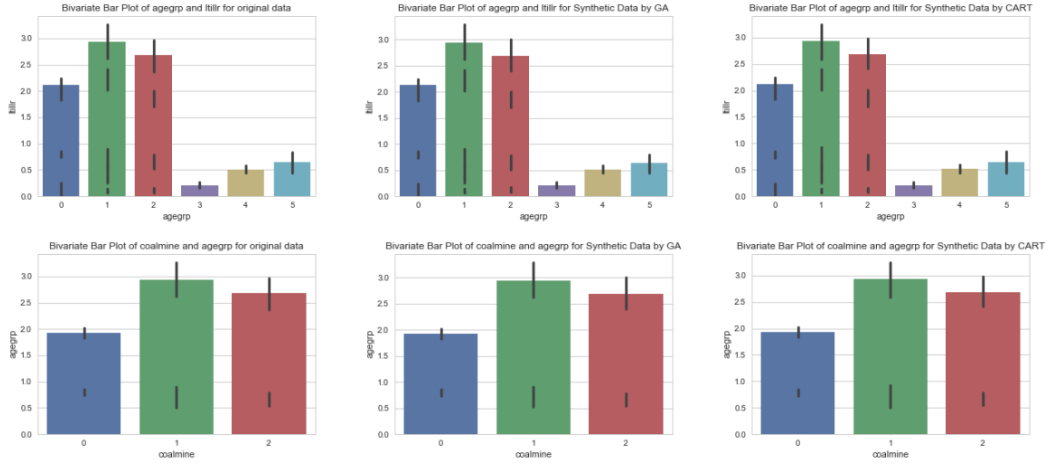


Fig 3.2 comparison of histogram between original, GA-synthetic and CART-synthetic dataset.



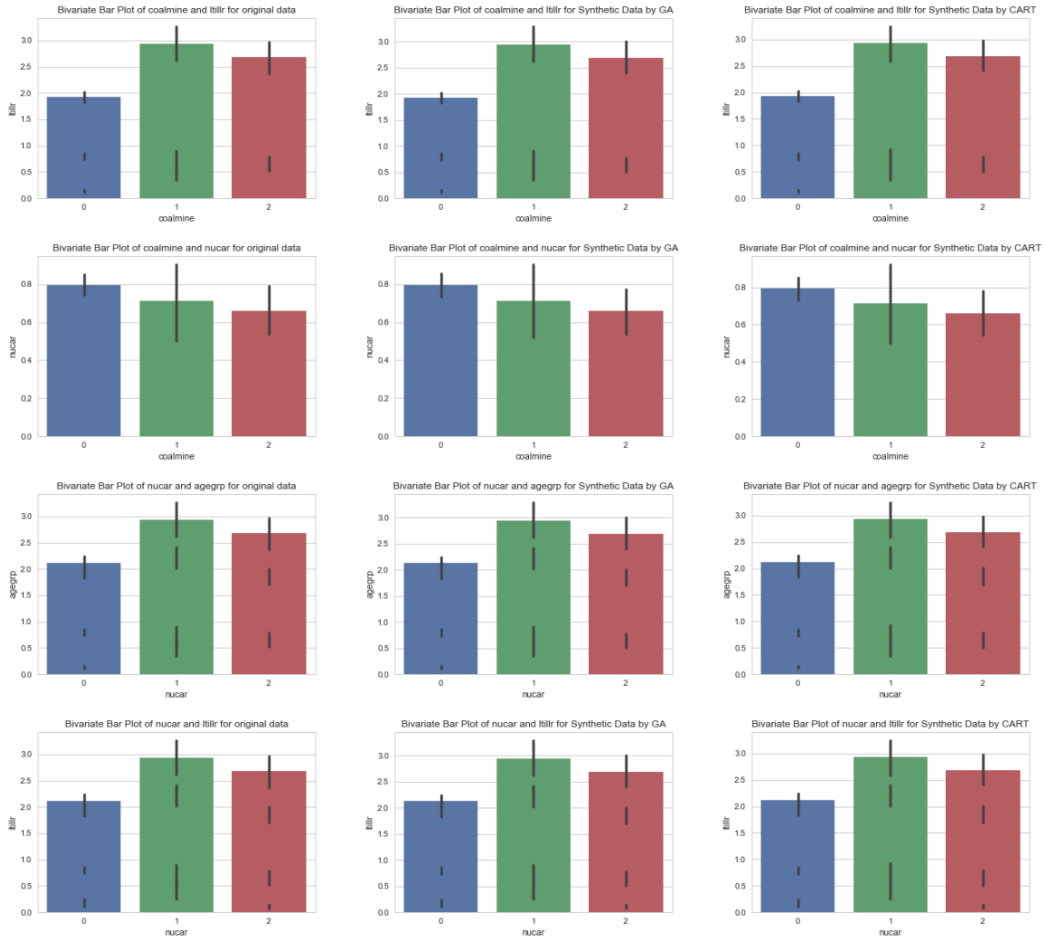


Fig 3.3 comparison of bivariate bar charts between original, GA-synthetic dataset and CART-synthetic dataset.

In logistic regression evaluation test, 'ltillr' was selected as the dependent variable and the other variables are its explanatory variables. The following table records coefficients and intercepts of explanatory variables and corresponding dummy variables from logistic regression models generated from the three datasets: original dataset, GA-synthetic dataset and CART-synthetic dataset. It shows that GA-synthetic dataset has closer values to the original dataset than CART-synthetic data in almost all coefficients except "agegrp_5".

Data	Model Summary
Original Data	Estimate Std. Error z value Pr(>|z|) (Intercept) -3.4674 0.4064 -8.532 < 2e-16 *** coalmine1 1.3807 0.3271 4.221 2.44e-05 *** coalmine2 2.0674 0.2584 8.001 1.23e-15 *** agegrp1 1.0254 0.4546 2.256 0.02410 * agegrp2 1.2922 0.4388 2.944 0.00323 ** agegrp3 1.8140 0.4312 4.207 2.59e-05 *** agegrp4 2.9712 0.4222 7.038 1.95e-12 *** agegrp5 3.1594 0.6050 5.222 1.77e-07 *** nucar1 0.2182 0.2245 0.972 0.33122 nucar2 -0.4997 0.2499 -2.000 0.04552 *
GA-synthetic	Estimate Std. Error z value Pr(>|z|)

Data	(Intercept) -3.1524 0.3679 -8.568 < 2e-16 *** coalmine1 0.9371 0.3578 2.619 0.008818 ** coalmine2 1.9871 0.2593 7.662 1.83e-14 *** agegrp1 0.7508 0.4187 1.793 0.072934 . agegrp2 0.9479 0.4050 2.341 0.019249 * agegrp3 1.4885 0.3961 3.757 0.000172 *** agegrp4 2.6561 0.3890 6.827 8.65e-12 *** agegrp5 1.7580 0.6920 2.540 0.011073 * nucar1 0.2013 0.2263 0.890 0.373589 nucar2 -0.4582 0.2483 -1.846 0.064956 .
CART-synthetic Data	Estimate Std. Error z value Pr(>|z|) (Intercept) -3.6022 0.4356 -8.270 < 2e-16 *** coalmine1 1.4167 0.3260 4.345 1.39e-05 *** coalmine2 2.0300 0.2525 8.040 9.00e-16 *** agegrp1 1.4599 0.4696 3.109 0.00188 ** agegrp2 1.4598 0.4718 3.094 0.00198 ** agegrp3 2.1500 0.4533 4.743 2.11e-06 *** agegrp4 2.9362 0.4516 6.502 7.94e-11 *** agegrp5 3.2711 0.5881 5.562 2.67e-08 *** nucar1 0.2673 0.2112 1.266 0.20562 nucar2 -0.8620 0.2730 -3.158 0.00159 **

Table. X comparison of logistic regression model coefficients and intercepts generated by GA-synthetic dataset and CART-synthetic dataset

Moreover, the predicted logit probability correlation indicates that the logistic model generated by GA-synthetic data also outperforms the model from CART-synthetic data. It has 1000 same prediction outputs as the original model for the 1000 cases compared with the 984 same outputs from the latter.

GA_synthetic data	Ltillr="0"	0.9999
	Ltillr="1"	0.9999
CART_synthetic data	Ltillr="0"	0.9860
	Ltillr="1"	0.9823

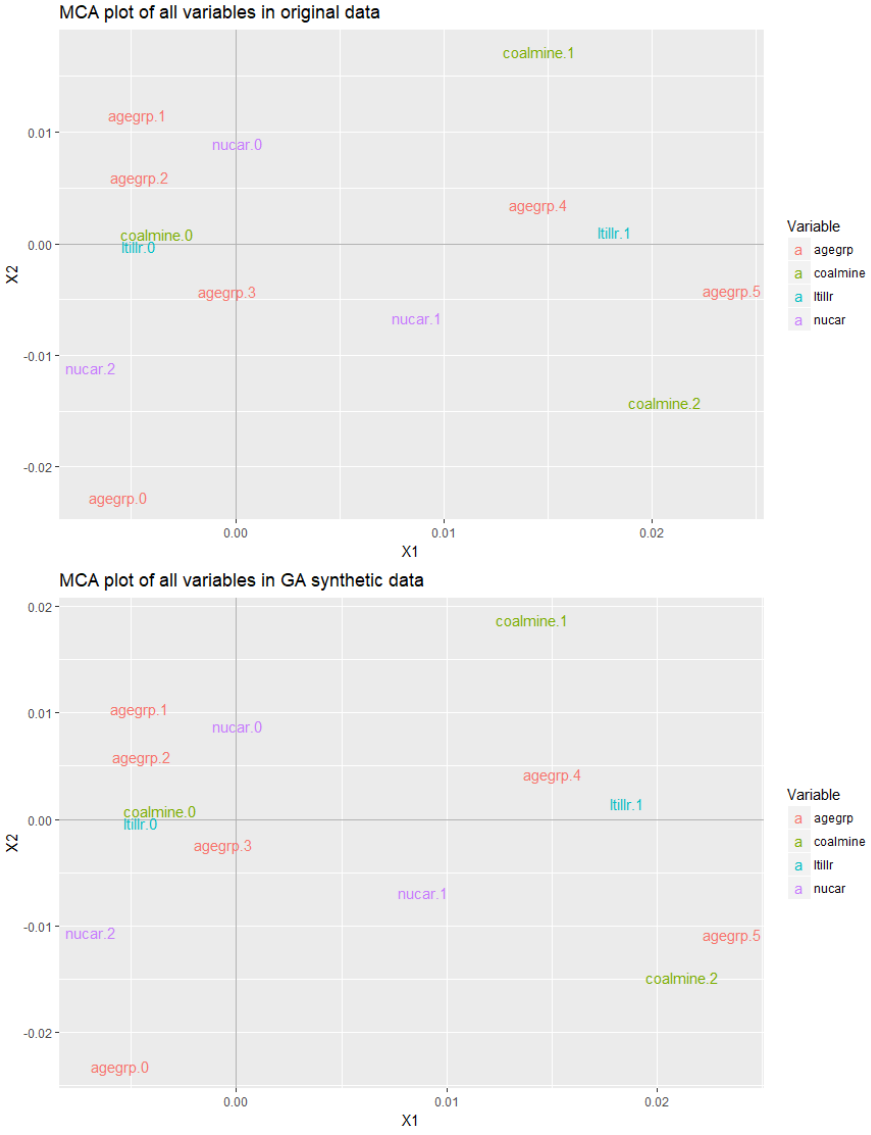
Table 3.4 comparison of predicted logit probability correlation to the original model generated by GA-synthetic dataset and CART-synthetic dataset

In MCA evaluation test, we set the number of factors as 2. The table below shows factor scores of different variables and their dummy variables, where the highlighted cells indicate that the score of this variable in GA synthetic dataset is more similar to its value in the original dataset than CART-synthetic dataset. By looking at the score-plots in Fig. X, we can conclude that although the two synthetic datasets have similar performance in MCA, GA-synthetic dataset is more similar to the original dataset in some of the variables.

	Original Data		GA-Synthetic Data		CART-Synthetic Data	
Factors	1	2	1	2	1	2
coalmine.0	-3.73E-03	8.64E-04	-3.62E-03	8.22E-04	-3.50E-03	7.13E-04
coalmine.1	1.46E-02	1.72E-02	1.41E-02	1.87E-02	1.58E-02	1.86E-02
coalmine.2	2.06E-02	-1.42E-02	2.12E-02	-1.48E-02	2.09E-02	-1.51E-02
nucar.0	9.21E-05	8.98E-03	9.34E-05	8.83E-03	-1.25E-04	9.36E-03
nucar.1	8.73E-03	-6.64E-03	8.91E-03	-6.86E-03	9.39E-03	-7.40E-03
nucar.2	-6.93E-03	-1.11E-02	-6.89E-03	-1.06E-02	-7.61E-03	-1.27E-02
agegrp.0	-5.61E-03	-2.27E-02	-5.48E-03	-2.32E-02	-6.15E-03	-2.01E-02

agegrp.1	-4.67E-03	1.15E-02	-4.56E-03	1.04E-02	-5.04E-03	1.10E-02
agegrp.2	-4.57E-03	6.00E-03	-4.46E-03	5.88E-03	-2.91E-03	6.24E-03
agegrp.3	-3.54E-04	-4.21E-03	-6.14E-04	-2.31E-03	-1.19E-03	-4.51E-03
agegrp.4	1.45E-02	3.55E-03	1.51E-02	4.31E-03	1.37E-02	-2.90E-04
agegrp.5	2.38E-02	-4.17E-03	2.36E-02	-1.08E-02	2.16E-02	8.55E-03
ltillr.0	-4.61E-03	-2.62E-04	-4.53E-03	-3.60E-04	-4.74E-03	-7.14E-05
ltillr.1	1.82E-02	1.04E-03	1.86E-02	1.48E-03	1.85E-02	2.78E-04

Table 3.5 Factor score of MCA (number of factors=2) from different datasets



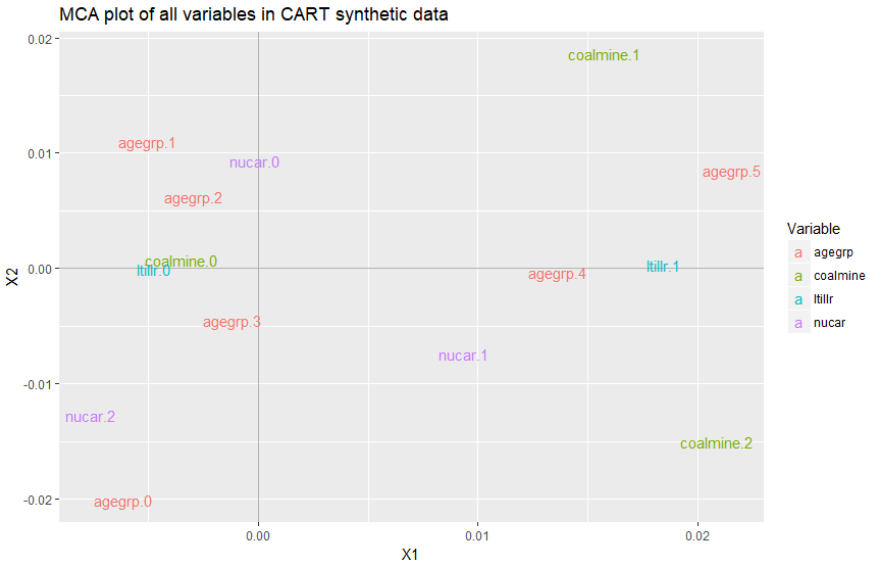


Fig 3.6 Factor scores plots of original dataset, GA synthetic dataset and CART-synthetic dataset

It is reasonable to say that 0.08 in D_{JS} in the full-contingency table preserves a quite good level of information from the original data and it is better than CART-synthetic data. Since GA is a dynamic generator, the final output is more controllable than CART. It is important because when utility is increasing, the dataset carries more risk in disclosure sensitive information from the original dataset.

4 Disclosure Risk Assessment

Using full contingency table as objective can measure both the utility and disclosure risk of the synthetic data. Compared with other utility measurements, it gives a more straightforward view of the divergence between original data and its synthetic version in marginals. Such information can help to measure other utility or disclosure objectives for example, differential privacy, which tests how two datasets differ by removing or adding particular cases (Work & Roth, 2014) (Barak et al, 2007), and differential correct attribute probability (DCAP), which measures attribute disclosure risks for synthetic data (Taub et al, 2018). Moreover, the experimental results show a strong proof that full contingency table can maintain other statistical properties without including them into the generator, which breaks the rule that synthetic data can only holds statistical properties that contains in its generator (Drechsler, 2014).

It is confirmed that utility and disclosure risk are conflict objectives in data synthesis. Duncan et al (2001) illustrated the utility-risk maps for some of known synthesiser and concluded that the higher the utility is the riskier the output data or vice versa. The risk of GA synthesiser with only utility objective is also higher because using full contingency table as the only objective will eventually end up to the original data. The following sections illustrates some scenarios happened in evaluation disclosure risks for a real-world dataset.

4.1 Re-identification Risks

Since our data is a small sample and there is no clear perspective to the population, we only use risk assessment as a post hoc test to the synthetic data. Theoretically fully synthetic data does not require re-identification risk assessment because it masks all real-records from

original data. However, this needs to be re-considered when using full contingency table. During the process of evolution in GA generator, the synthetic data is likely to have sensitive individuals from the original data. Shlomo (2010) confirmed that re-identification risk is a function of both population and sample. Suppose $N_{(r,c)}$ is the population size in cell (r, c) in the contingency table (r, c are indices of row and column) and $N_{(r,c)}$ is the sample size of the same cell. The disclosure risk can be measured by

$$\tau = \sum_{r,c} I\left(n_{(r,c)} = 1\right)/N_{(r,c)}$$

Due to the absence of original population, it is impossible to derive the value of τ for the synthetic data. If we only compare the two datasets, what we can tell is that synthetic data releases 13 of 16 unique records from original data. The other unique records have $\tau' = 0.5$, $\tau' = 0.25$ and an unidentified one due to no corresponding record in the original data. It is not evident to say that the synthetic data is dangerous because the dataset is a small sample with unknown population, but it would be dangerous in realistic case if 13 of 16 unique individuals were exposed in a published dataset.

4.2 Attribute Disclosure Risks

DCAP works on the assumption that the intruder knows the values of a set of key variables for a given unit and is seeking to learn the specific value of a target variable. The Correct Attribution Probability (CAP) for the record indexed *j* is the empirical probability of its target variables given its key variables

$$CAP_{o,j} = \Pr\left(T_{o,j} \middle| K_{o,j}\right) = \frac{\sum I\left(T_{o,j} = T_{o,j} \middle| K_{o,j} = K_{o,j}\right)}{I(K_{o,j} = K_{o,j})}$$

d_o is the original data and K_o and T_o as vectors for the key and target information respectively. Likewise, d_s is the synthetic dataset, with the vectors K_s and T_s. The CAP for record j based on a corresponding synthetic dataset d_s is the same empirical, conditional probability but derived from d_s,

$$CAP_{s,j} = \Pr\left(T_{s,j} \middle| K_{s,j}\right) = \frac{\sum I\left(T_{s,j} = T_{o,j} \middle| K_{s,j} = K_{o,j}\right)}{I(K_{s,j} = K_{o,j})}$$

Suppose 'Ltillr' is the target variable and the others are key variables in DCAP test, the CAP score of the GA-synthetic dataset is only 0.0045 less than the original dataset, which seems to put the data in a vulnerable place in attribute disclosure.

4.3 Differential Privacy

Differential privacy intuitively guarantees that a privacy model, which involves randomness in its process, behaves similarly on similar input dataset. Although it is a pre-setting of model rather than a post hoc assessment, the principle can be analogised in synthetic data if full contingency table is used.

Suppose the model used to generate synthetic data is M, The model M is (ε, δ)-differential privacy if for all $S \subseteq range(M)$ and for all x, y such as $\left||x - y|\right|_1 \leq 1$:

$$\Pr[M(x) \in S] \leq e^{\varepsilon} \Pr[M(y) \in S] + \delta$$

When a privacy model M is $(\varepsilon, 0)$-differential privacy ($\delta = 0$), the output of a record is equally likely to be output in different runs of the model. In contrast, if $\delta \neq 0$, then the output of a record is very unlikely to be output in different runs of the model. Meanwhile, given an

output $\xi \sim M(x)$, probably we could find a dataset y such that ξ looks more like an output from $M(y)$ than from $M(x)$. By setting full contingency table as the objective, outputs from GA synthetic data generator is naturally $(\varepsilon, 0)$-differential privacy before a certain level of optimality. For example, suppose X_1 and X_2 are neighbouring outputs from GA generator M_{GA} that differs from only one record, it is very unlikely to distinguish $M_{GA}(X_1)$ and $M_{GA}(X_2)$ before they eventually turn to original datasets. In another word, the output of GA generator is almost equally likely to be observed on every neighbouring datasets.

5 Conclusion

The paper shows that using full contingency table as the only objective in GA synthetic data generator is capable to produce an acceptable result. As a fundamental property for categorical data, it sufficiently retains other statistical properties at the same time. However, this objective comes with high disclosure risks due to the closeness to the original data. Fortunately, most of the known disclosure risk assessments like re-identification assessment, differential privacy and DCAP measures the data from case-level, which can be derived from full contingency table directly. In order to achieve dynamic control to the synthetic data in GA synthesiser, or any other potential learning algorithms, we should focus on finding the trade-off between data utility and disclosure risks.

Reference

[1] Drechsler, J., 2014; Synthetic data, where do we come from? Where do we want to go?, in Synthetic Data Workshop; Office of National Statistics., https://www.ons.gov.uk/ons/guide- method/method-quality/general-methodology/statistical-disclosure-control/ons-workshops-on-synthetic- data/synthetic-data--where-do-we-come-from--where-do-we- want-to-go-.pdf+&cd=1&hl=en&ct=clnk&gl=uk

[2] Duncan, G.T., Keller-McNulty, S. A., and Strokes, S. L., 2001, Disclosure Risk vs. Data Utility: The R-U Confidentiality Map, National institute of Statistical Science, NISS.

[3] Dwork, C., and Roth, A., 2014, The Algorithmic Foundations of Differential Privacy, *Foundations and Trends in Theoretical Computer Science*, vol.9 3-4.

[4] Maimon, O., and Rokach, L., 2010, Data Mining and Knowledge Discovery Handbook, Springer Science and Business Media; p. 704

[5] Taub, J., Elliot, M., Pampaka, M., and Smith, D., 2018, Differential Correct Attribution Probability for Synthetic Data: An Exploration, Privacy in Statistical Database, 2018

[6] Abdi, H. and Valentin, D., 2007, Multiple Correspondence Analysis, Encyclopedia of Measurement and Statistics, Neil Salkind (Ed.), Thousand Oaks (CA): Sage.

[7] Shlomo, N., (2010), Releasing Microdata: Disclosure Risk Estimation, Data Masking and Assessing Utility' *Journal of Privacy and Confidentiality*, vol. 2, no. 1, pp. 73-91

[8] Ciriani, V., di Vimercati, S.D.C., Foresti, S., Samarati, P., (2007) Microdata protection. In: Yu T., Jajodia S. (eds.) Secure Data Management in Decentralized Systems, 291–321. Springer, New York

[9] Barak, B., Chaudhurim, K., et al (2007) Privacy, Accuracy, and Consistency Too: A Holistic Solution to Contingency Table Release, PODS' 07, Beijing, China

[10] Chen, Y., Elliot, M., and Sakshaug, J., (2016), A Genetic Algorithm Approach to Synthetic Data Production, in Proceedings of the 1st International Workshop on AI for Privacy and Se-curity. Article No. 13.

[11] Chen, Y., Elliot, M., and Sakshaug, J., (2017), Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data, UNECE Work Session on Statistical Data Confidentiality, https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.46/2017/2_Genetic_algorithms.pdf. Last access 20/12/2017.

[12] Chen, Y., Elliot, M., and Smith, D., (2018), The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods, Privacy in Statistical Database 2018, Springer.

[13] Reiter, J.P., (2005) Using CART to generate partially synthetic, public use microdata, Journal of Official Statistics 21, 441–462

[14] Nowok, B., Raab., M. G., and Dibben, C., (2016), synthpop: Bespoke Creation of Synthetic Data in R, Journal of Statistical Software, 74(11), DOI: 10.18637/jss.v074.i11

5.5 The Impact from Initial Population in GA Synthetic Data Generator

The initial population is considered as an important parameter in GAs, so does in a GA synthesiser. As information utility is first concerned when I designed the GA synthesiser, Most experiments used a good initial population (very similar to the original data) for quicker and better results. This contradicts with traditional fashions of GAs, where diversity is considered as the most important characteristic in an initial population since it expands the solution space and prevents converging to local optima. This paper investigates the performance of GA synthesisers with initial populations that carry different levels of diversity and, based on the findings.

The Impact from Initial Population in GA Synthetic Data Generator

Yingrui Chen*, Mark Elliot*

* Cathie Marsh Institute, The University of Manchester, Manchester, UK,
{yingrui.chen, mark.elliot}@manchester.ac.uk

Abstract. The initial population is considered as an important parameter in the design of genetic algorithms (GAs). It is widely accepted that a the initial population should be specifically designed for the problem it is designed to solve. As information utility is central to functional data synthesis, in most of our experiments we use an initial population concentrated on good utility candidates for quicker and better results [6]. This contradicts with orthodox view of GAs, where diversity is considered as the most important characteristic in setting up a population since it expands the solution space prevents converging to local optima [2]. This paper investigates the performance of GA synthetic data generator (aka GA synthesiser) with initial populations with its overall fitness and different levels of diversity and fitness.

1 Introduction

Data synthesis is a statistical disclosure control technique for preventing leakage of confidential information from data. Its main feature is to preserve analytical properties of the original data whilst protecting the true information of data subjects. A microdata set of n cases and m variables is usually represented as an n by m matrix indexed $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, m\}$. Here we use Y to denote and original dataset and its synthetic version is denoted as X. X shares the same structure as Y as illustrated in Fig 1.

$$\begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1m} \\ y_{21} & y_{22} & \cdots & y_{2m} \\ y_{31} & y_{32} & \cdots & y_{3m} \\ y_{41} & y_{42} & \cdots & y_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nm} \end{pmatrix} \quad \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & x_{2m} \\ x_{31} & x_{32} & \cdots & x_{3m} \\ x_{41} & x_{42} & \cdots & x_{4m} \\ \vdots & \vdots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nm} \end{pmatrix}$$

Figure 1: Microdata Y and its synthetic version X

The potential of synthetic data is tremendous because, in principle, it allows the publication of data without concerns about privacy leakage. Hence many more

sensitive datasets with significant potential economic and social value can be opened to public. However, the utility of synthetic data strongly relies on the design of synthetic data generator [10]. Properties can only be present in the synthetic dataset if they are structurally or statistically related to properties that included in the generator. Unforeseen analysis on fully synthetic data may therefore lead to different results from the same analysis on the original data [12].

GAs are learning optimisation algorithms that simulate natural evolution. They were first proposed to solve linear problems. Subsequent research showed its suitability for solving higher-dimensional and more complex problems [13, 14, 15]. Starting with an initial population consisting of a group of *candidates*, the process of GAs is presented below (Fig 2).

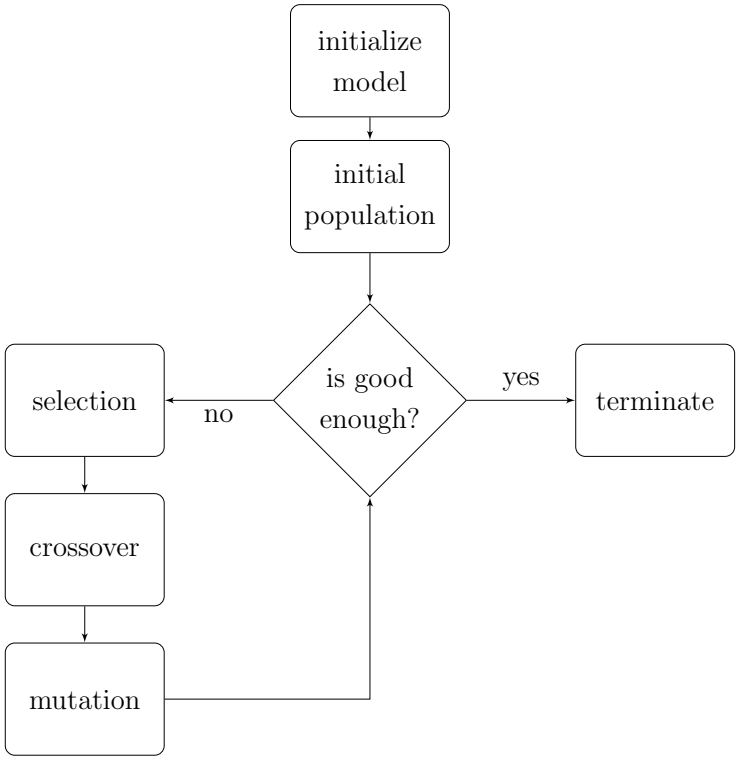


Figure 2: Flowchart of Genetic Algorithms

We have investigated GA data synthesisers in the past of our research [7, 3, 4, 5, 6]. We have demonstrated the efficiency and effectiveness of this new method in synthesising data and finding the trade-off between information utility and disclosure risks. We demonstrated that the efficiency of a GA synthesiser can be increased if the parameters are aligned to the specific data.

1.1 Initial Population

Diversity and fitness are considered as two influential features in GA populations. A population with higher diversity is more likely to jump out of local optima [2] [1]. However, there is no general standard for measuring the diversity of datasets, and in common with other features of GAs, diversity needs to be tied to the specific problem.

In the data synthesis domain, We can hypothesise that population diversity will depend on the amount of structure implied in the generating model. On the other hand, the fitness of initial population may also impact on the efficiency of the whole GA process. The initial population can be generated from some observed descriptive properties of the original data such as means, frequencies, equivalence class structures, covariance or Chi-squared statistics [9]. Generating initial populations that have those properties will reduce the running time of the whole process compared with pure random data because properties embedded in the initial population need not to be re-gained from the GA process. Meanwhile, the initial population can still carry a certain level variation between candidates because simple properties are not sufficient to describe the original data.

2 Experiment Design

This paper considers three models that can be used to build the initial population of the GA synthesiser. They are the uniform distribution model, the univariate distribution model and a synthesiser model. Individuals in uniform model are created by independently sampling (with replacement) from the uniform distributions of each variable within the original data. The univariate distribution model creates individuals by independently sampling (with replacement) from the univariate distributions of each variable in the original data. As for the synthesiser model, we use a well known synthesising method CART. [2]. The reason for comparing these three generation models for initial population is to check:

1. The impact of fitness level from initial population on the efficiency of a GA synthesiser: Populations generated by the three models vary arbitrary distributions to feasible solutions and therefore they contain different levels of utility (U) and risk (R): with the expecation that $U(\text{uniform}) < U(\text{univariate}) < U(\text{CART})$ and $R(\text{uniform}) < R(\text{univariate}) < R(\text{CART})$. In the first set of

[1] Brabazon, et al (2015) prefer the word "dispersion" as being more descriptive of the feature in question, but we use the more common "diversity" here as dispersion has a particular meaning in statistics.

[2] Specifically we use the software `synthpop.R` created [20]

experiments the paper aims to find which initial population produces the most fit solutions.

2. The impact of diversity from the initial population on the efficiency and effectiveness of GA synthesiser: we would expect that a freer model, like the uniform model will produce Initial populations that carry higher diversity. Candidates in this population are more like a random noise has which in the expectation maximises distance between any two generated candidates. However, a more restricted model, like the CART model, is likely to produce similar candidates to the initial population and to each other.

In this paper, we will use the same GA parameter values that we used in [6]: the fixed initial population size: 100, tournament selection with tournament size $t = 2$ [3], (whole-case) parallelised crossover with crossover rate 0.5 [4], and uniform mutation with mutation rate $p_m = 0.01$ [5]. With the same settings to all parameters, we are now able to investigate how different features of the initial population impact the efficiency and effectiveness of the GA synthesiser. The experiment will be implemented on three datasets with different levels of complexity in their structures [17, 18, 19]: Data 1 has 4 binary variables, Data 2 has 1 binary variables, 1 six-category variable, 1 nine-category variables and 1 eleven-category variables and Data 3 contains 2 binary variables, 1 nine-category variables and 1 ten-category variable. All of the experiments will be carried on the following objective settings for the synthesiser.

2.1 Objectives

The synthesiser used a set of two opposite objectives: Utility and Risk. The utility measure is carried by Jensen-Shannon distance D_{JS} in full contingency tables from synthetic data $CT_{FULL}(X)$ to the original data $CT_{FULL}(Y)$. Since the overall structure in a categorical data can be captured by a full contingency table and many statistical analysis methods are based on this statistic, it is undoubted that a synthetic data that is close enough to the original data retains most - if not all - of its statistical properties.

Jensen-Shannon distance D_{JS} is a method to measure the level of divergence between two probability distributions, and specifically can measure the divergence

[3] In tournament selection, candidates are randomly selected into tournaments of size t (with replacement). only the winning candidate in the tournament can enters crossover.

[4] Whole-case parallelised crossover occurs on every case in the candidate, the case was chosen by determined crossover rate p_r in this paper and it is then switched with the corresponding case in paired candidate.

[5] uniform mutation gives every single element in the candidate a chance p_m to mutate.

between a pair of high-dimensional contingency tables while their values are normalised. Suppose P and Q are two discrete probability distribution, $D_{JS}(P||Q)$ is defined by:

$$D_{JS}(P||Q) = (\frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M))^{\frac{1}{2}} \tag{1}$$

, where $M = \frac{1}{2}(P + Q)$ and D_{KL} is Kullback-Leibler divergence, which cannot be used directly because of the requirement for absolute continuity. The objective is to minimise $U(X, Y)$, which is the divergence of full contingency tables CT_{FULL} from the synthetic and original data.

$$U(X, Y) = D_{JS}(CT_{FULL}(X)||CT_{FULL}(Y)) \tag{2}$$

As for risk measure, [11] and [16] introduced a measure for disclosure risk of synthetic data called the *Differential Correct Attribution Probability* (DCAP), which consists of a Correct Attribution Probability (CAP) score. DCAP was originally used as a post-hoc test to assess attribution risk. DCAP works on the assumption that the intruder knows the values of a set of key variables for a given unit and is seeking to learn the specific value of a target variable. The Correct Attribution Probability (CAP) for the record indexed j is the empirical probability of its target variables given its key variables,

$$CAP_{o,j} = Pr(T_{o,j}|K_{o,j}) = \frac{\sum_{i=1}^{n}[T_{o,i} = T_{o,j}, K_{o,i} = K_{o,j}]}{\sum_{i=1}^{n}[K_{o,i} = K_{o,j}]} \tag{3}$$

where the square brackets are Iverson brackets, n is the number of records, and d_o is the original data and K_o and T_o as vectors for the key and target information. Likewise, d_s is the synthetic dataset, with the vectors K_s and T_s. The CAP for record j based on a corresponding synthetic dataset d_s is the same empirical, conditional probability but derived from d_s,

$$CAP_{s,j} = Pr(T_{o,j}|K_{o,j})_s = \frac{\sum_{i=1}^{n}[T_{s,i} = T_{o,j}, K_{s,i} = K_{o,j}]}{\sum_{i=1}^{n}[K_{s,i} = K_{o,j}]} \tag{4}$$

[6] introduce a scenario in DCAP in which rather than using the whole dataset, only the statistical uniques of the original dataset are used in calculating the CAP score (this method corresponds to a common focus for National Statistical Institutes). The non-matches (records on the original dataset which do not match any records in synthetic dataset on the key) in this instance of DCAP were scored as 0, this allowed for candidates with more non-matches to have lower scores on the risk measure, which is intuitive.

As [6], the paper will take the normalised Euclidean distance of the two objectives as a single fitness function:

$$F(X, Y) = \sqrt{2}^{-1}\sqrt{U(X, Y)^2 + R(X, Y)^2} \tag{5}$$

3 Experimental Results

Firstly the hypothesis that population generated from freer models has more diversity needs to be verified. The diversity of a pair of candidates can also be evaluated by Jensen-Shannon divergences D_{JS} as the method is capable of measuring the degree of similarity between two high-dimensional probability distributions. In this paper we evaluate diversity in a population D_P by the standard deviation of D_{JS} between full contingency tables of each pair of candidates.

$$D_P(X_i, X_j \in P, i \neq j) = \sigma(D_{JS}(CT_{FULL}(X_i)||CT_{FULL}(X_j))) \tag{6}$$

The standard deviation verified our assumption that the population generated from uniform model has greater diversity than the one from univariate model and CART model (Table 1).

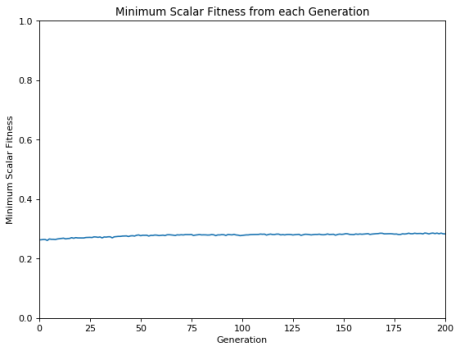
	Population Diversity D_P		
	uniform	univariate	CART
data1	0.0766	0.0772	0.0744
data2	0.2653	0.1589	0.0067
data3	0.1540	0.1314	0.0951

Table 1: Population diversity of initial populations generated from different models for the three datasets

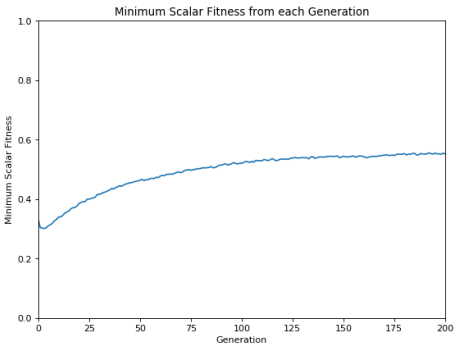
3.1 The Impact of Fitness Level of the Initial Population to the Efficiency of GA Synthesisers

The following table (Table 2 recorded performances of population generated by the three models during 200 generations. It compares the best candidate in the initial population and the best candidate in the last generation. The differences between them indicated how much the population has been improved.

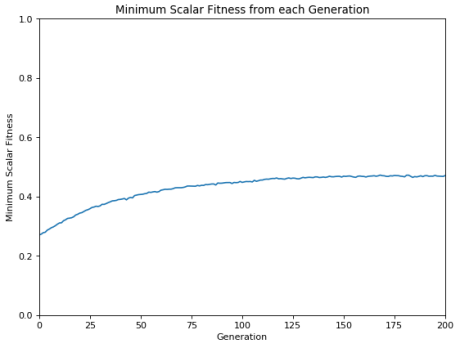
The result indicated that a more optimal population (generated by CART) cannot lead to better results (Fig 3). The synthesiser indeed produced satisfactory results for the inital population but its fitness deteriorated after 200 generations. A reasonable conjecture is that the crossover rate was too high (0.5) to stop mature candidates swapped cases with its paired individual, therefore its structures were broken during the process. Initial populations generated from uniform model had the best improvement except in Data 2 and the population generated by univariate model had the best final outputs.



(a) Data1



(b) Data2



(c) Data3

Figure 3: Performances of populations generated from the CART model by original data of Data1, Data2 and Data3 over 200 generations

	Risk and utility values before and after 200 generations								
	uniform			univariate			CART		
	Before	After	Difference	Before	After	Difference	Before	After	Difference
Data1	(0.6135 ,0.3903)	(0.2398 ,0.0000)	(0.2232, 0.3903)	(0.2958 ,0.0000)	(0.1455 ,0.0000)	(0.1503, 0.0000)	(0.3724 ,0.0000)	(0.4002 ,0.0000)	(-0.0278, 0.0000)
Data2	(0.9231 ,0.0762)	(0.7526 ,0.0545)	(0.1705, 0.0217)	(0.7633 ,0.0287)	(0.5658 ,0.0188)	(0.3895 ,0.1818)	(0.3895 ,0.1818)	(0.7796 ,0.0416)	(-0.3901, 0.1402)
Data3	(0.7383 ,0.0417)	(0.5014 ,0.0000)	(0.2369, 0.0417)	(0.6269 ,0.0000)	(0.4640 ,0.0000)	(0.1629, 0.0000)	(0.3863 ,0.0000)	(0.6658 ,0.0000)	(-0.2795, 0.0000)
	Single objective fitness values before and after 200 generations (normalised)								
	uniform			univariate			cart		
	Before	After	Difference	Before	After	Difference	Before	After	Difference
Data1	0.5142	0.1696	0.3446	0.2092	0.1029	0.1063	0.2633	0.283	-0.0197
Data2	0.655	0.5336	0.1214	0.5401	0.4003	0.1398	0.3039	0.5520	-0.2481
Data3	0.5229	0.3545	0.1684	0.4433	0.328	0.1153	0.2731	0.4708	-0.1977

Table 2: The difference of best candidates in the first and 200-th generations in GA synthesiser, whose initial populations generated from uniform, univariate and CART models, in Data1, Data2 and Data3.

3.2 The Impact of Diversity of the Initial Population to the Efficiency of GA Synthesisers

In order to eliminate the impact from fitness from the initial population. In this set of experiments we introduced a reference data for each dataset. A reference data is sampled from uniform distribution from each data and we generate initial populations by the three models from the reference data. Compared with populations generated from the original data, populations generated by reference data presumably has more closeness in their overall fitness values. However, this cannot eliminate the difference in overall fitness between the population generated by the uniform model and populations generated by the other two models, as the former carried less utility naturally. The diversity of populations calculated by Eq 6 (see Table 3).

	Population Diversity D_P		
	uniform	univariate	CART
data1	0.0782	0.0767	0.0727
data2	0.2659	0.1598	0.0787
data3	0.2648	0.1590	0.0771

Table 3: Population diversity of initial populations generated from different models by reference datasets of the three datasets

Table 4 recorded the performance of different populations after 200 generations. Result shows that populations generated from univariate model have the best outcome among the three models in 200 generations and populations generated from uniform model have the most significant improvement (Table 4).

	Risk and utility values before and after 200 generations								
	Uniform			Univariate			CART		
	Before	After	Difference	Before	After	Difference	Before	After	Difference
Data1	(0.6240, 0.3121)	(0.2430, 0.0000)	(0.3810, 0.3121)	(0.2899, 0.0000)	(0.1540, 0.0000)	(0.1359, 0.0000)	(0.3160, 0.0000)	(0.1582, 0.0000)	(0.1578, 0.0000)
Data2	(0.9253, 0.0749)	(0.7372, 0.0412)	(0.1881, 0.0337)	(0.7662, 0.0359)	(0.5521, 0.0212)	(0.2147, 0.0147)	(0.7720, 0.1023)	(0.5580, 0.0253)	(0.2140, 0.0770)
Data3	(0.7380, 0.0417)	(0.5045, 0.0000)	(0.2335, 0.0417)	(0.6415, 0.0000)	(0.4578, 0.0000)	(0.1837, 0.0000)	(0.6535, 0.0000)	(0.4631, 0.0000)	(0.1904, 0.0000)
	Fitness values before and after (normalised)								
	Uniform			Univariate			CART		
	Before	After	Difference	Before	After	Difference	Before	After	Difference
Data1	0.4933	0.1719	0.3214	0.2049	0.1089	0.0960	0.2234	0.1119	0.1115
Data2	0.6564	0.5221	0.1343	0.5424	0.3907	0.1517	0.5507	0.3950	0.1557
Data3	0.5227	0.3567	0.1660	0.4536	0.3237	0.1299	0.4621	0.3275	0.1346

Table 4: The difference of best candidates in the first and 200-th generations in GA synthesiser, whose initial populations generated from uniform, univariate and CART models by reference data, in Data1, Data2 and Data3.

4 Discussion

The two sets of experiments demonstrate that: (1) it is not always true that GAs with fitter initial populations are more efficient in searching for optima. (2) in GAs, populations with more diversity have more exploratory powers. They validate that diversity is truly the vital feature in the initial population in GAs but a population with high diversity but low fitness level may not outperform a population with moderate diversity and moderate fitness in the same number of generations (see the performance of populations from uniform and univariate models).

An interesting phenomenon in the first set of experiments is the performance of the population generated by CART using the original data, which developed towards an unwanted direction. It is possibly caused by high crossover rates which may destroy data structures in well-developed candidates. Another likely reason is that population with less diversity is unlikely to escape from local optima.

References

[1] Abdi, H., and Valentin, D., (2007) Multiple Correspondence Analysis, Neil Salkind (Ed.), *Encyclopedia of Measurement and Statistics*, The University of Texas at Dallas, Richardson, TX 75083–0688, USA

[2] Brabazon, A., O'Neill, Michae.l, and McGarraghy, Seán., (2015). *Natural computing algorithms*, Springer.

[3] Chen, Y., Elliot, M., and Sakshaug, J., (2016), A Genetic Algorithm Approach to Synthetic Data Production, in Proceedings of the 1st International Workshop on AI for Privacy and Se-curity. Article No. 13.

[4] Chen, Y., Elliot, M., and Sakshaug, J., (2017), Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data, UNECE Work Session on Statistical Data Confidentiality, `https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.46/2017/2_Genetic_algorithms.pdf`, Last access 20/12/2017.

[5] Chen, Y., Elliot, M., and Smith, D., (2018), The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods, *Privacy in Statistical Database*, 2018, Springer.

[6] Chen, Y., Taub, J., and Elliot, M (2019) Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator, UNECE 2019.

[7] Chen, Y. and Elliot, M, (in preparation), Matrix GA: building blocks in data synthesis.

[8] Department for Communities and Local Government, Ipsos MORI. (2012). Citizenship Survey, 2010-2011, UK Data Service. SN: 7111, http://doi.org/10.5255/UKDA-SN-7111-1

[9] Drechsler, J. (2010). Using support vector machines for generating synthetic datasets, *Privacy in Statistical Databases*, Springer Berlin Heidelberg, 148-161.

[10] Drechsler, J. (2014), Synthetic data, where do we come from? Where do we want to go?, Synthetic Data Workshop, Office of National Statistics.

[11] Elliot, M. (2014). Final Report on the Disclosure Risk Associated with the Synthetic Data Produced by the SYLLS Team. [online] CMIST. Available at: https://tinyurl.com/syllsDR

[12] Maimon, O. and Rokach, L. (2010) *Data Mining and Knowledge Discovery Handbook*, Springer Science and Business Media, p. 704

[13] Pongcharoen, P., Khadwilard, A., and Klakankhai, A., (2007) Multi-matrix Real-coded Genetic Algorithm for Minimising Total Costs in Logistics Chain Network, World Academy of Science, *Engineering and Technology International Journal of Economics and Management Engineering*, 1(11), 574-597

[14] Wallet, B. C., Marchette, D. J. and Solka, J. L., (1996), A matrix Representation for Genetic Algorithms, Proceedings of Automatic Object Recognition IV of SPIE Aerosense, Naval Surface Warfare Center Dahlgren, Virginia.

[15] Sun, L. Zhang, Y and Jiang, C (2006) A matrix real-coded genetic algorithm to the unit commitment problem, *Electric Power Systems Research*, 76, 716-728

[16] Taub, J., Elliot, M., Pampaka, M. and Smith, D. (2018). Differential Correct Attribution Probability for Synthetic Data: An Exploration. J. Domingo-Ferrer and F. Montes (Eds.): PSD 2018, LNCS 11126. pp. 122-137

[17] Department for Communities and Local Government, 2012, Ipsos MORI. Citizenship Survey, 2010-2011. [data collection]. UK Data Service. SN: 7111, http://doi.org/10.5255/UKDA-SN-7111-1, Last access: 20/12/2017.

[18] NatCen Social Research. (2017). British Social Attitudes Survey, 2016. [data collection]. UK Data Service. SN: 8252, http://doi.org/10.5255/UKDA-SN-8252-1

[19] Office for National Statistics. Crime Survey for England and Wales, 2015-2016. [data collection]. UK Data Service. SN: 8140, http://doi.org/10.5255/UKDA-SN-8140-1. Last access 11/01/2018. (2017)

[20] Nowok, B., Raab., M. G., and Dibben, C., (2016), synthpop: Bespoke Creation of Synthetic Data in R, Journal of Statistical Software, 74(11), DOI: 10.18637/jss.v074.i11

5.6 Exploring the Impact of Adaptive Parameters on a Genetic Algorithm Synthesiser

The exploration power of GA synthesiser is not consistent with fixed parameters, like fixed crossover rate and fixed mutation rate, during the process. This paper explores the impact of adaptive parameters in a GA synthesiser and describes the possible influence of the power of the parameters on the efficiency and effectiveness of the synthesiser.

Exploring the Impact of Adaptive Parameters on a Genetic Algorithm Synthesiser

Abstract: Data synthesis is a statistical disclosure control method used to protect data subjects' privacy by replacing real records with artificial ones while maintain statistical features from the original data. We designed Genetic Algorithm data synthesiser (GA synthesiser) and demonstrated that GAs can an efficient approach used in data synthesis. The exploration power of genetic algorithms (GAs) increased when operators that specifically designed for this problem are equipped. Unfortunately, the advantage of these operators is not consistent with fixed parameters (crossover rate and mutation rate) during the process. This paper explores the impact from adaptive and dynamic parameters in a GA synthesiser and describes the possible influence of the power of the parameters on the efficiency and effectiveness of the synthesiser.

1 Introduction

Data synthesis offers significant protection against reidentification attacks. However, an effective data synthesiser requires retention of as many key statistical properties of (and respecting the multiple utilities of) the original data as possible. The potential of GAs in producing synthetic data is clearly stated (Chen et al, 2016)(Chen et al, 2017). GAs are iterative optimising algorithms that simulate the process of natural evolution. They comprise of three main operators: selection, crossover and mutation. Begin with a group of candidates (the initial population), fitness of these candidates is calculated and a selection operator selects a subset of the fitter candidates which are used to generate a new population. In crossover some pairs of these selected candidates are combined (using a variety of methods) to produce new candidate solutions. Some candidates are then subjected to mutation – random changes that will produce changes in fitness. After crossover and mutation, we have the new population / generation. The process is repeated a number of times in order to (hopefully) generate fitter solutions than those in the initial population. Crossover and mutation rates can be varied from one iteration to the next and tuning of these parameters may greatly influence performance.

The model of GAs in the application of data synthesis has been well constructed in the past. we have demonstrated that the model of GAs in the application to data synthesis is feasible (Chen et al, 2018)(Chen et al 2019). This work shows the effectiveness of GAs in searching solutions, however the efficiency is not satisfying yet. Our current state of the art model shows the lack of ability topass the best candidates to following generations, which results fluctuation in the development of the population's fittest members. **Figure 1** is an example of the change of population's minimum divergence to the original data during 1000 generations.

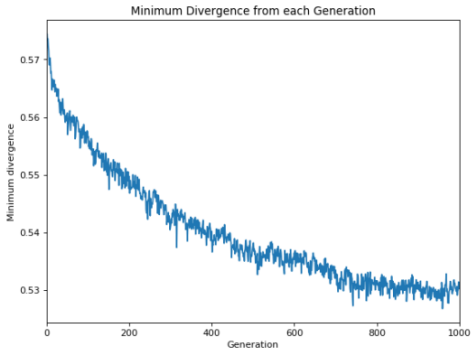


Figure 1.1 An example of change of population's fittest value vs number of generations in a state of the art using fixed-parameter model.

1.1 Adaptive and Dynamic Parameters

Adaptive parameters are suggested in order to improve efficiency of GAs in the later stage of the process. The consideration of using dynamic parameters is based on that non-constant parameters perform better than fixed parameters in evolutionary studies. However, there are drawbacks if the parameters are used inappropriately. Firstly, it probably increases computational workload unnecessarily; sometimes using fixed parameters in GAs is sufficient to give satisfactory outputs. Secondly, the involvement of dynamic parameters makes the whole process more unpredictable (Thierens, 2002). Thierens also classified parameters in adaptive GA into dynamic parameter control and adaptive parameter control. The former tunes parameters dependent upon the number of generations or the convergence level of current population whereas the later adapts parameters according to individual fitness. Either type can be applied to any of three operators; this paper mainly discusses the potential of using adaptive parameters in crossover and mutation.

Adaptive or dynamic parameters normally applied on crossover and mutation operators. The Crossover rate p_c is a parameter to moderates the power of the crossover operator, it can stop all candidates are peremptorily swapped with others and retain the goodness to compete in the next generation. Meanwhile, adaptive mutation rate p_m, assumes that p_m of a particular candidate is determined by the optimality of the candidate. Suppose the fitness value of the candidate is f_i and the average fitness value in its generation is $\bar{f}$. p_m is determined as the following principles (Libellin & Alba, 2000): If f_i is better than $\bar{f}$, then p_m should be low to keep the schema from the good individual. If f_i is worse than $\bar{f}$, then p_m should be reasonably high to generate new species into the pool.

2 Model Design

In this paper we will use the same settings that used in (Chen et al, 2019): the fixed initial population size 100, tournament selection[1] with tournament size $t = 2$, (whole-case) parallelised crossover [2]and uniform mutation[3].

Minimising Jensen-Shannon divergence between full contingency tables of candidate and the original data is the only objective of this synthesiser. The Full contingency table is fully capable to be the only objective in the GA generator. It captures the between-variate structure among categorical variables by counting the number of cases by possible configurations of related variables. We used Jenson-Shannon divergence D_{JS} to transform the divergence between two datasets in full contingency table scalar. Suppose P and Q are two full contingency tables for data Y and its synthetic version X in the format of probability distribution, i.e.

$$P = \frac{CT_{FULL}(X)}{N}$$
$$Q = \frac{CT_{FULL}(Y)}{N}$$

The D_{JS} between P and Q is defined by:

[1] In tournament selection, candidates are randomly selected into tournaments of size t (with replacement). only the winning candidate in the tournament can enters crossover.

[2] Whole-case parallelised crossover occurs on every case in the candidate, the case was chosen by determined p_r and it is then switched with the corresponding case in paired candidate.

[3] uniform mutation gives every single element in the candidate a chance p_m to mutate.

$$\mathrm{D}_{\mathrm{JS}}(P||Q) = \left(\frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M)\right)^{\frac{1}{2}}$$

, where $M = \frac{1}{2}(P + Q)$ and D_{KL} is Kullback-Leibler divergence.
Thus, the fitness function is $f(X) = \mathrm{D}_{\mathrm{JS}}(P||Q)$ and the objective is to minimise $f(X)$ for X in the population.

2.1 Algorithm Design for Adaptive Parameters

Suppose $F_c(f(X))$ and $F_m(f(X))$ in the range of [0,1] are formula to determine adaptive rates of crossover and mutation. The algorithm can generally be written as:

```
{
Initialise population;
Evaluate fitness f(X) of each individual X in the population;
Evaluate p_c = F_c(f(X)) and p_m = F_m(f(X)) for each individual X;
While stopping condition:
        {
        Do selection;
        Do crossover and mutation;
        Evaluate fitness f(X) of each individual X in the population;
        Evaluate p_c = F_c(f(X)) and p_m = F_m(f(X)) for each individual X;
        }
}
```

Algorithm 2.1 Algorithm of adaptive GA

Previous work has discussed many designs of $F_c(f(X))$ and $F_m(f(X))$ in traditional GAs [2, 6, 7] and they are all based on the principle that the rate should stay low for fitter candidates to pass their goodness into the new generation but be reasonably high for worse candidates to create more variation for the population. We decided not to use any the pre-designed formula because 1) they only proved to be applicable in traditional GAs, which have different problem structures and candidates to ours. 2) they are overspecified for the problem that we worked on and may create problems in exploring the search space effectively.

As our problem is a minimisation program and the objective function is normalised, the value of fitness of a candidate can naturally be proportional to its rates.

$$F_c\big(f(X)\big) = k_c f(X)$$
$$F_m\big(f(X)\big) = k_m f(X)$$

, where k_c and k_m are arbitrary non-negative constants and $k_c f(X),\ k_m f(X) \subseteq [0, 1]$.

2.2 Algorithm Design for Dynamic Parameters

Binguil (2007) argued that the relationship between population fitness and parameters were complex so that it is better to be described linguistically [8]. Since there is no universal rule of parameter control in GAs and the problem is more complex in either candidate structures or reproduction process. Therefore, in this paper we decided to tune the parameters by piecewise tuning functions (conditions) empirically rather than applying hyper-parameters on the generator. Based on the above algorithm of adaptive parameters, the new design that involves in dynamic parameters is shown below.

```
{
Initialise population;
Evaluate fitness f(X) of each individual X in the population;
```

Evaluate $p_c = F_c(f(X))$ and $p_m = F_m(f(X))$ for each individual X;
While *stopping condition:*
    While *tuning condition 1:*
    {
    Do selection;
    Do crossover and mutation;
    Evaluate fitness $f(X)$ of each individual X in the population;
    Evaluate $p_c = F_c(f(X))$ and $p_m = F_m(f(X))$ for each individual X;
    }
    While *tuning condition 2:*
    {
    Do selection;
    Do crossover and mutation;
    Evaluate fitness $f(X)$ of each individual X in the population;
    Evaluate $p_c = F_c(f(X))$ and $p_m = F_m(f(X))$ for each individual X;
    }

    …
}

Algorithm 2.2 Algorithm of dynamic GA with tuning conditions

3 Experiment Design

3.1 Experiment Design for Exploring Adaptive Parameters

This exploration of the impact of adaptive parameters uses survey data from Citizenship survey 2010-2011 (Department for Communities and Local Government, 2012). This paper run trials on GA generators equipped with different k_c, k_m values for both crossover and mutation operators. The initial population is randomly and independently sampled from univariate distributions from original data. We empirically picked a set of values for k_c and k_m in this experiment and recorded the minimum fitness value of the 200th generations from corresponding generator.

3.2 Experiment Design for Exploring Dynamic Parameters

This paper will neither compare the methods nor give deterministic tuning conditions. Its goal is to demonstrate the benefits of using dynamic parameters as a part of GA generators. The design of tuning conditions can be flexible and based on the user's preference.

We summarise three feasible methods to adjust adaptive parameters during the process of GA synthetic data generator: 1) Adjusting parameters by the level of dominance of the best candidate in current generation. Parameters are adjusted by monitoring the proportion of current population that are exact copies of the best candidate. Since the whole population will eventually be taken over by some fitter candidates in GAs, this method helps prevent premature convergence of the generator. 2) Adjusting parameters by the level of monopoly of the best candidate in a set of consecutive generations, for example, the most recent 10. Once a good candidate rises, it spends several generations to take over the whole population unless a fitter candidate appears during the process. This method customises parameters by checking the number of generations that a candidate outperforms other competitors to stop it consuming the whole population. And 3) adjusting parameters by the optimality level of current generation. Determining dynamic parameters for GA synthetic data generator totally

depends on users' knowledge of the dataset and it is not limited to choosing only one method. We give an example in the next section about how to adjust parameters using method 3).

4 Experiment Results

We picked a set of values $k_c = \{0.0, 0.25, 0.5, 0.75, 1.0, 1.25\}$ and $k_m = \{0, 0.001, 0.0025, 0.0050, 0.0075, 0.01, 0.0025, 0.05\}$ for Algorithm 2.1 and compared fitness of the best candidate in 200th generation.

Clearly, generators with lower mutation rates have smoother and gentle trends in 200th generations (see Appendix 1 and 2). A sharp increase in fitness (decrease in divergence) in early stage is not always helpful to the generator in exploring the solution space in following generations. No matter how small the value of k_m is, mutation performed a vital role in pushing the generator to reach places in the search space that cannot be reached by using crossover operator only. However, a larger k_m can also disrupt the goodness of developed candidates, which was shown by fluctuations in the graphs. As for k_c, a larger value will result quick improvement in fitness from the beginning but also a quicker and premature convergence without involving mutation. The good news is that the adequate values of k_c and k_m is not in a restricted range. Although $k_c = 0.75, k_m = 0.005$ best output in Fig 4.l, the surrounding values also give acceptable results.

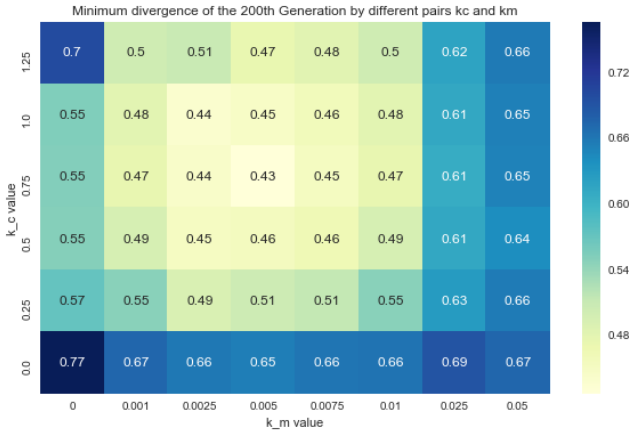


Fig 4.1 Heatmap of the capability of adaptive GA synthetic data generator with different values of k_c and k_m

Although we may deduct the acceptable range k_c and k_m with this particular dataset, it is hard to say this conclusion can be applied universally to all other datasets. However, since our objective limit of the fitness value is [0,1], there might be only slight deviation in using the similar pairs of values to optimise synthetic data for other datasets. We tested the same sets of k_c and k_m on other four datasets (ONS 2011) (ONS 2017) (National Records of Scotland, 1901), that have different structures and the result indicates that, the impact of k_c and k_m depend on the level of optimality over the initial population. An initial population that already consists of better candidates needs higher k_c and k_m values to boost the performance (see the 1st and 2nd datasets in Fig 4.2). As for a weaker initial population, its candidates need a mild pair of k_c and k_m to discover better solutions.

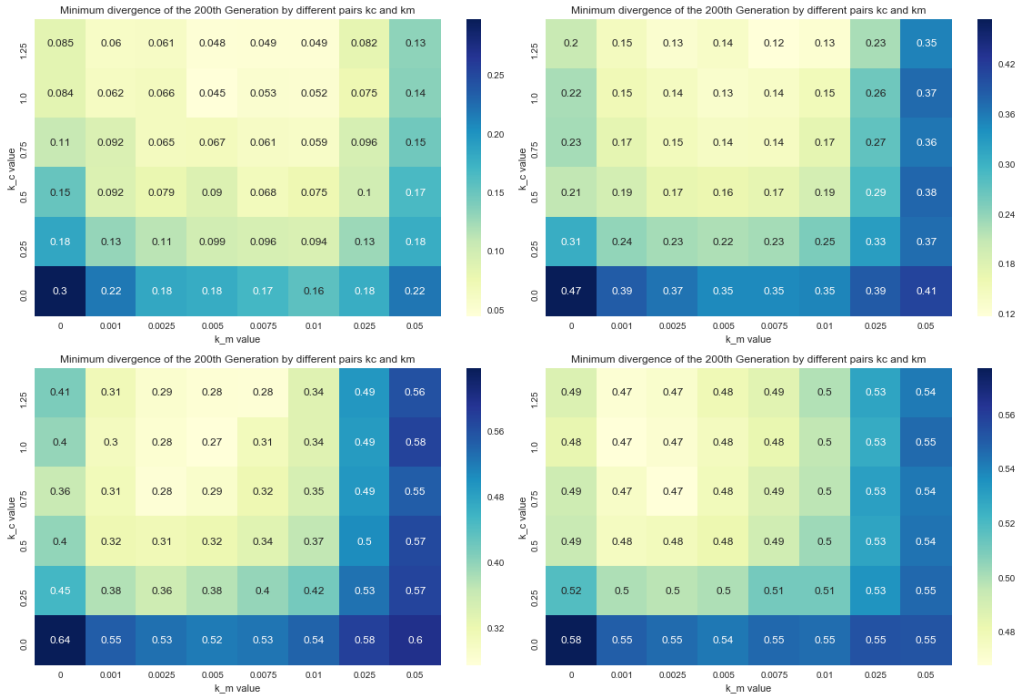


Fig 4.2 Heatmap of the capability of adaptive GA synthetic data generator with different values of k_c and k_m for other four datasets

The conclusion is only applicable in early stage of the GA process. According to our experience, a high mutation rate has no does not help a developed population. Fig 4.4 draws the change of minimum divergence in 1500th to 2500th generations from the same generator that gives the optimal output for the trial dataset. The fluctuation indicates that the generator loses its capability to explore the solution space when the population evolved into a certain level. Parameters must be adjusted to escape from this situation. We designed a set of tuning conditions for dynamic GA synthesiser for the citizenship dataset (Algorithm. 4.3).

{
Initialise population;
Evaluate fitness $f(X)$ of each individual X in the population;
Evaluate $p_c = F_c(f(X))$ and $p_m = F_m(f(X))$ for each individual X;
While *stopping condition:*
    if $f(X) \leq 0.25$*:*
    {
    $k_c = 0.75$ and $k_m = 0.001$ for each individual X;
    }
    elIf $f(X) \leq 0.20$*:*
    {
    $k_c = 0.5$ and $k_m = 0.0005$ for each individual X;
    }
    else*:*
    {
    $k_c = 0.75$ and $k_m = 0.005$ for each individual X;
    }
    Do selection;

Do crossover and mutation;
Evaluate fitness $f(X)$ of each individual X in the population;
Evaluate $p_c = F_c(f(X))$ and $p_m = F_m(f(X))$ for each individual X;
}
}

Algorithm 4.3 Algorithm of dynamic GA with tuning conditions for trial dataset

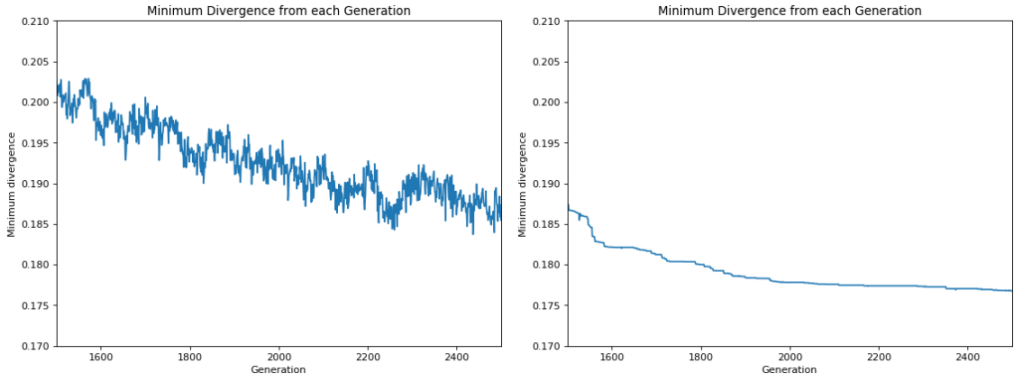


Fig 4.4 Comparing performance from adaptive GA generator (left) and dynamic GA generator (right) between 1500th and 2500th generations

The comparison of the performance between two generators is evidence of the positive effect from using dynamic parameters in GA. Despite the lack of theoretical proof , the above conclusion illustrates the potential of using the level of optimality as a tuning condition in a GA synthetic data generator.

5 Conclusion

This paper explores the impact from using adaptive parameters and tuning conditions to improve the performance of GA synthetic data generator. Results show distinct influence on efficiency of generators by different settings of adaptive functions. Dynamic parameter will be helpful to retain good candidates in the latter process of GA. It is not necessary but can definitely accelerate the speed in finding optima. However, this paper cannot give a universal theory about the impact from adaptive or dynamic parameters to GA synthesiser. These two cannot be determined unless users have good understanding of the dataset and the relation between its structure and optimality in the generation. The complex relation between parameters and dataset itself creates questions like: Does it depend on the design of objective? How does the landscape of the solution space influence the tuning conditions? What is the mathematical relation between the convergence level of population and the adaptive function?

Reference

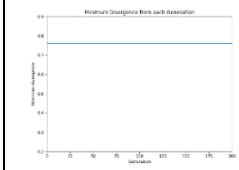
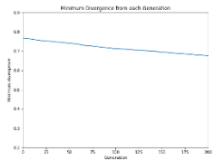
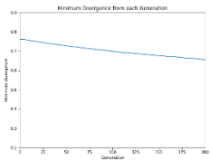
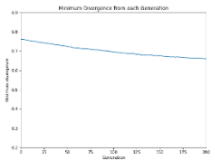
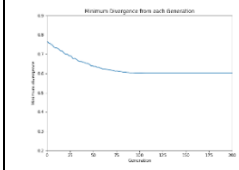
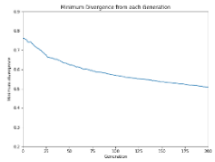
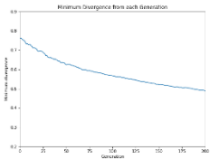
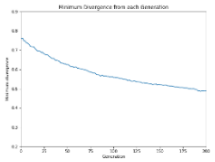
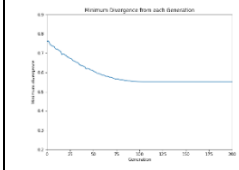
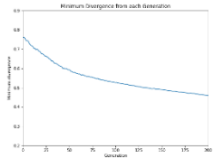
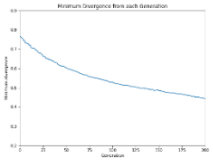
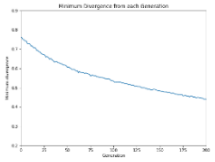
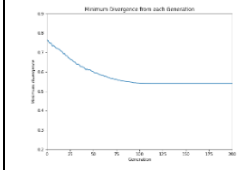
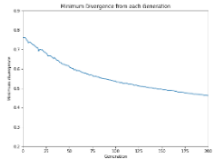
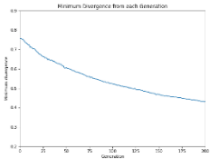
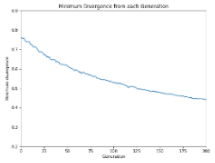
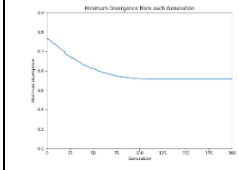
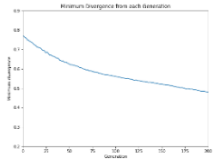
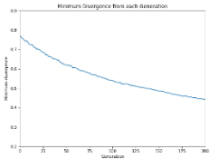
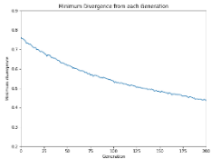
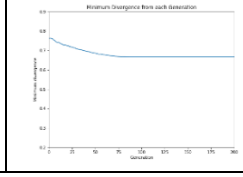
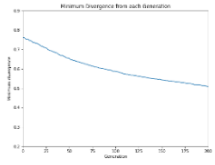
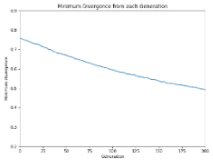
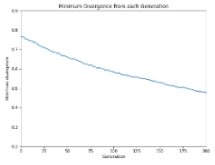
[1] Bingul, Z., (2007) Adaptive Genetic Algorithms Applied to Dynamic Multi-objective Problems, Applied Soft Computing, 7, 891-799

[2] Chen, Y., Elliot, M., and Sakshaug, J., (2016), A Genetic Algorithm Approach to Synthetic Data Production, in Proceedings of the 1st International Workshop on AI for Privacy and Se-curity. Article No. 13.

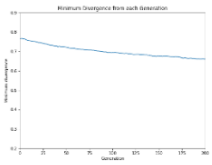
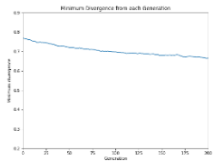
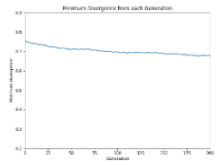
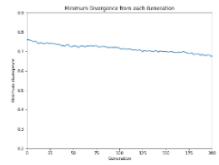
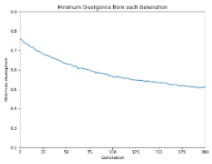
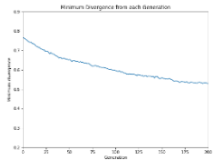
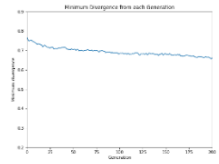
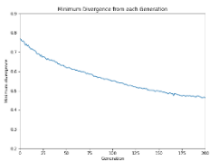
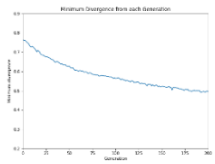
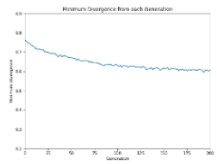
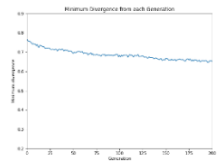
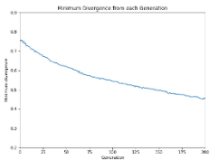
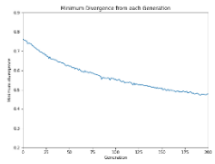
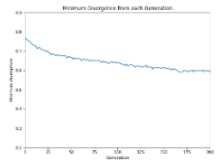
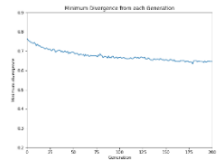
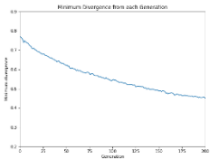
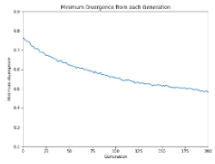
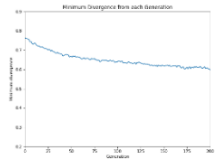
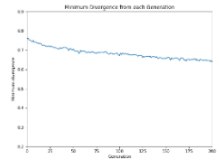
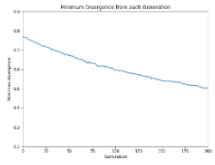
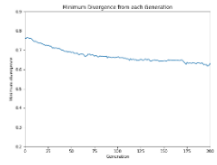
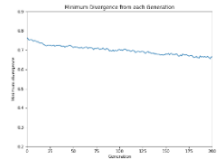
[3] Chen, Y., Elliot, M., and Sakshaug, J., (2017), Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data, UNECE Work Session on Statistical Data Confidentiality,

https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.46/2017/2_Genetic_algorithms.pdf. Last access 20/12/2017.
[4] Chen, Y., Elliot, M., and Smith, D., (2018), The Application of Genetic Algorithms to Data Syn- thesis: A Comparison of Three Crossover Methods. In *Proceedings of Privacy in Statistical Database 2018*. Springer.
[5] Chen, Y., (in preparation). Genetic Algorithms and their Application to Synthetic Data Generation. PhD Thesis to be submitted to the University of Manchester. Expected submission date December 2019.
[6] Department for Communities and Local Government, Ipsos MORI. (2012). *Citizenship Survey, 2010-2011*. [data collection]. UK Data Service. SN: 7111, http://doi.org/10.5255/UKDA-SN-7111-1
[7] Liberllin, S., and Alba, P., (2000) Adaptive mutation in genetic algorithms, *Soft computing*, vol.4 pp.76-80.
[8] Srinivas, M. and Patnaik, L. M. (1994a) Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms; IEEE Transactions on systems, Man and Cybernetics; 24(4); pp.656-668
[9] Thierens, D. (2002) Adaptive mutation rate control schemes in genetic algorithms, *Evolutionary Computation, 2002. CEC '02. P roceedings of the 2002 Congress on*, vol.1, pp. 980 – 985
[10] Office for National Statistics, (2017), Crime Survey for England and Wales, 2015-2016. [data collection]. UK Data Service. SN: 8140, http://doi.org/10.5255/ UKDA-SN-8140-1. Last access 11/01/2018.
[11] Office for National Statistics. Social Survey Division, Northern Ireland Statistics and Research Agency, Eurostat. (2011). European Union Statistics on Income and Living Conditions, 2009. [data collection]. UK Data Service. SN: 6767, http:// doi.org/10.5255/UKDA-SN-6767-1. Last access 11/01/2018. (2009)
[12] National Records of Scotland, (1901), 1901 Scottish Census.

Appendices

k_c \ k_m	0.0000	0.0010	0.0025	0.0050
0.00				
0.25				
0.50				
0.75				
1.00				
1.25				

Appendix 1 Divergence change of the best candidate in 200 generations from the generator equipped by different values of k_c and k_m (1)

k_c \ k_m	0.0075	0.0100	0.0250	0.0500
1.25				
1.00				
0.75				
0.50				
0.25				
0.00				

Appendix 2 Divergence change of the best candidate in 200 generations from the generator equipped by different values of k_c and k_m (2)

5.7 Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator

Impact of Full Contingency Table in Data Synthesis has shown that GAs are capable of producing good synthetic data using the full contingency table as the only objective, however, this implies higher disclosure risks (which are by definition one such statistical property). This paper uses DCAP as the risk objective in GA synthesiser to contest with the full contingency table and looks for the trade-off between the two objectives.

Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator

Yingrui Chen*, Jennifer Taub*, Mark Elliot*

* Cathie Marsh Institute, The University of Manchester, Manchester, UK,
{yingrui.chen, jennifer.taub, mark.elliot}@manchester.ac.uk

Abstract. Data synthesis is a data confidentiality method which is applied to microdata to prevent leakage of sensitive information about respondents. Instead of publishing real data, data synthesis produces an artificial dataset that does not contain the original records of respondents. This, in particular, offers significant protection against re-identification attacks. Previous work has shown that Genetic algorithms (GAs) are capable of producing good synthetic data using the full contingency table as the only objective, because it completely specifies the frequencies of equivalence class structure; using this objective could in theory lead to the original data as the solution; although the search space for any non trivial dataset appears be fractal and therefore convergence to the original data is improbable in polynomial time [3].

As a fundamental property for categorical data, the full table necessarily retains all statistical properties present in the original data. However, this implies higher disclosure risks (which are by definition one such statistical property).

This technique therefore presents precisely the same issue of managing the trade-off between disclosure risks and data utility as orthodox SDC. However, GAs have a big advantage. A GA is an iterative learning algorithm that used to solve complicated problems and critically supports multiple contrary objectives by allowing the dynamically monitoring of the relationship between them.

This paper uses the differential correct attribute probability (DCAP), which measures attribute disclosure risks for synthetic data, as the risk objective in GA synthesiser to contest with the full contingency table as the only utility objective. .

1 Introduction

Disclosure control for microdata attempts to protect sensitive information in an original dataset whilst retaining that datasets statistical properties for analysts.

Data synthesis can be regarded a statistical disclosure control (SDC) technique that produces a synthetic dataset that is designed to preserve the statistical properties of the original data and provide sufficient variables to allow proper multivariate analyses (Abowd and Lane, 2004). Therefore, data synthesis can be treated as an optimisation problem with two opposite constraints: data utility and disclosure risks.

In this paper, we use a genetic algorithm (GA) to generate synthetic data. GA's form a branch of evolutionary computing which aims to solve optimisation problems. There are three main operators within GAs: *selection*, *crossover* and *mutation*. Starting with group of candidate solutions (the initial population). The fitness of these candidates are evaluated. Fitter candidates are randomly paired using a crossover operator to generate new candidate solutions. These candidates are then passed to mutation operator and given a probability to mutate. The offspring form the new population and the process is repeated until a desired optimality achieved.

2 Objectives

Objectives within GAs are user defined values for a set of (usually latent) attributes which will hold for any optimal solution and so define a standard against which the fitness of a candidate should be measured. When designing a GA for a real-world problem, there is invariably more than one objective to be considered. These objectives are often opposed to one another, and this situation holds for data utility and disclosure risk in the synthetic data problem [7]. in general, as utility decreases, disclosure risk rises. When dealing with multi-objective problems, we can adopt one of two solutions; the first is to simultaneously optimise all objectives using a method such as *Pareto optimisation*. However, Chen's work suggests that it is very difficult to arrive at a stable optimal solution using this method[3]. This leads to the second alternative which is to simply compose the objectives into a single function with determined weights (referred as SOGA in this paper). The objectives must be normalised to avoid an objective with a larger numerical variance dominating the less varied one, otherwise the selection of weights is easy to get wrong and even a small difference in weighting can lead to different solutions.

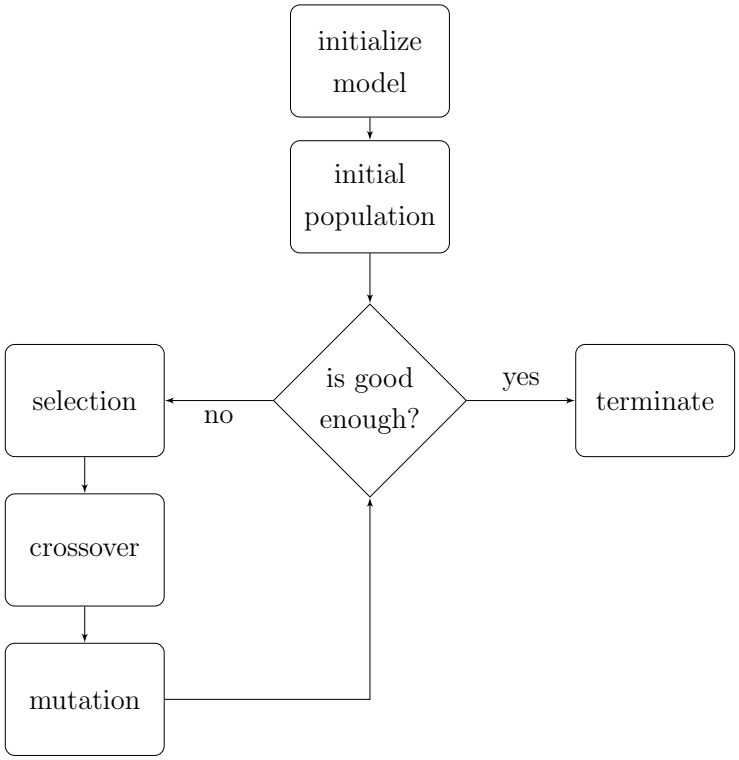


Figure 1: Flowchart of Genetic Algorithms

2.1 The Utility Measure

The between-variate structure in a categorical data can be captured by contingency table. Assume $I = \times_{j\in[1,,m]} I$ denotes the possible configurations of the variables that take values from finite sets, a contingency table is an m-dimensional table containing a count for each member of I. Since many statistical analysis methods are based on contingency tables, it is undoubted that a synthetic data that is close enough to the original data retains most - if not all - of its statistical properties.

Jensen-Shannon distance D_{JS} is a method to measure the level of divergence between two probability distributions, and specifically can measure the divergence between a pair of contingency tables. Suppose P and Q are two discrete probability distribution, $D_{JS}(P||Q)$ is defined by:

$$D_{JS}(P||Q) = (\frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M))^{\frac{1}{2}} \tag{1}$$

, where $M = \frac{1}{2}(P + Q)$ and D_{KL} is Kullback-Leibler divergence, which cannot be used directly because of the requirement for absolute continuity.

The objective is to minimise $U(X, Y)$, which is the divergence of full contingency tables CT_{FULL} from the synthetic and original data.

$$U(X, Y) = D_{JS}(CT_{FULL}(X)||CT_{FULL}(Y)) \tag{2}$$

Compared with other utility measurements, the full contingency table gives a more straightforward view of the divergence between original data and its synthetic version in a case-level. The disadvantage is that the synthesiser may overfit to the original data.This may not be immediately obvious, but there maybe structure within the data that is not useful. That is although the fitter output carries more sufficient information, it has highly chance of exposing sensitive information.

2.2 The disclosure risk measure

Most of the functions that are used to evaluate disclosure risks from released datasets are based on post-hoc measurements. Although there exists an inverse relationship between utility and disclosure risks in data privacy [4], involving risk in fitness evaluation the early stage of process may eliminate candidates with good properties. Therefore, we replace the starting point of the generator from a set of inferior candidates by a set of near optimal candidates (on the utility function) generated by

modest mutation if the original data, which makes the bi-objectives model more reasonable. We note in passing here that this makes the process more like a traditional SDC approach than an traditional data synthesis, but as we will discuss later within an optimisation framework, we consider the distinction between SDC and synthesis to be entirely arbitrary.

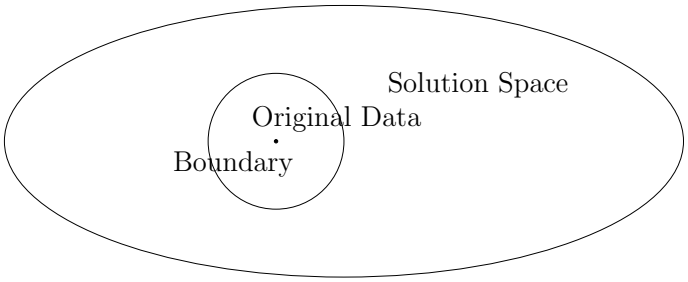


Figure 2: Illustration of Solution Space in Data Synthesis

Elliot (2014) and Taub et al (2018) introduced a measure for disclosure risk of synthetic data called the *Differential Correct Attribution Probability* (DCAP), which consists of a Correct Attribution Probability (CAP) score. DCAP was originally used as a post-hoc test to assess attribution risk [5, 14].

It is considered differential in that the score is to be compared to that of both the original dataset and that of a baseline (based on the univariate distribution of a given sensitive variable). DCAP works on the assumption that the intruder knows the values of a set of key variables for a given unit and is seeking to learn the specific value of a target variable. The Correct Attribution Probability (CAP) for the record indexed j is the empirical probability of its target variables given its key variables,

$$CAP_{o,j} = Pr(T_{o,j}|K_{o,j}) = \frac{\sum_{i=1}^{n}[T_{o,i} = T_{o,j}, K_{o,i} = K_{o,j}]}{\sum_{i=1}^{n}[K_{o,i} = K_{o,j}]} \tag{3}$$

where the square brackets are Iverson brackets, n is the number of records, and d_o is the original data and K_o and T_o as vectors for the key and target information. Likewise, d_s is the synthetic dataset, with the vectors K_s and T_s.

The CAP for record j based on a corresponding synthetic dataset d_s is the same empirical, conditional probability but derived from d_s,

$$CAP_{s,j} = Pr(T_{o,j}|K_{o,j})_s = \frac{\sum_{i=1}^{n}[T_{s,i} = T_{o,j}, K_{s,i} = K_{o,j}]}{\sum_{i=1}^{n}[K_{s,i} = K_{o,j}]} \quad (4)$$

However, a test run on 9,000 datasets mutated from the original dataset in with different levels mutation showed an unexpected near perfect correlation between DCAP and $U(X, Y)$ (-0.9920). A linear model with multiple $R^2 = 0.9839$ confirmed the linear relationship between the two objectives. In another words, using this measure there will not be an ideal solution with high utility but low disclosure risk. In effect this mean that the pure DCAP measure is a measure of utility rather than risk[1]. This actually makes sense since this full DCAP measure captures the capability of user to make inferences about based on the conditional distribution of the target variable. On the other hand drawing from the posterior distribution for any arbitrary record regardless of it really makes non sense for an intruder.

As one method of circumnavigating this, Taub et al (2018) introduce a scenario in which rather than using the whole dataset, only the statistical uniques of the original dataset are used in calculating the CAP score [14] (this method corresponds to a common focus for National Statistical Institutes). Correlation between DCAP-Statistical Uniques and $U(X, Y)$ dropped to -0.58. The non-matches (records on the original dataset which do not match any records in synthetic dataset on the key) in this instance of DCAP were scored as 0, this allowed for candidates with more non-matches to have lower scores on the risk measure, which is intuitive.

3 Experiments

3.1 Model Design in GA Synthesiser

The GA synthesiser in our previous work [2] was equipped with Deterministic tournament selection operator with tournament size $t = 2$, Candidates are randomly selected into tournaments of size t (with replacement). The probability that a candidate wins the tournament and enters crossover is given by $p(1 - p)r$ where p is a parameter (such that $1/t < p1$) and r is the rank of the candidates fitness within the tournament. In deterministic tournament selection p is set to 1. And the same mechanism applies to this problem by combining the two objectives using the Euclidean distance from the origin $(0, 0)$:

[1]We are thankful to Gillian Raab [10]for drawing attention to this general problem which led to us running the above experiment

$$
\begin{aligned}
&\text{Minimise } F(X,Y) \\
&\qquad F(X,Y) = \frac{1}{\sqrt{2}}\sqrt{U(X,Y)^2 + R(X,Y)^2} \\
&\text{s.j.t } U(X,Y) \in [0,1] \\
&\qquad R(X,Y) \in [0,1]
\end{aligned}
$$

Crossover and mutation are the main operators used in GA to provide variation within the next generation. We used whole-case crossover and uniform mutation, which has been shown to be efficient in generating synthetic data. The following figures illustrate how the crossover and mutation operators work.

$$
\begin{pmatrix}
\boxed{x^1_{11}\ x^1_{12}\ \ldots\ x^1_{1m}} \\
x^1_{21}\ x^1_{22}\ \ldots\ x^1_{2m} \\
\boxed{x^1_{31}\ x^1_{32}\ \ldots\ x^1_{3m}} \\
x^1_{41}\ x^1_{42}\ \ldots\ x^1_{4m} \\
\boxed{x^1_{51}\ x^1_{52}\ \ldots\ x^1_{5m}} \\
x^1_{61}\ x^1_{62}\ \ldots\ x^1_{6m} \\
\vdots\quad \vdots\quad \vdots\quad \vdots \\
x^1_{n1}\ x^1_{n2}\ \ldots\ x^1_{nm}
\end{pmatrix}
\begin{pmatrix}
\boxed{x^2_{11}\ x^2_{12}\ \ldots\ x^2_{1m}} \\
x^2_{21}\ x^2_{22}\ \ldots\ x^2_{2m} \\
\boxed{x^2_{31}\ x^2_{32}\ \ldots\ x^2_{3m}} \\
x^2_{41}\ x^2_{42}\ \ldots\ x^2_{4m} \\
\boxed{x^2_{51}\ x^2_{52}\ \ldots\ x^2_{5m}} \\
x^2_{61}\ x^2_{62}\ \ldots\ x^2_{6m} \\
\vdots\quad \vdots\quad \vdots\quad \vdots \\
x^2_{n1}\ x^2_{n2}\ \ldots\ x^2_{nm}
\end{pmatrix}
$$

Figure 3: X^1 and X^2 before whole-case crossover

$$
\begin{pmatrix}
x^2_{11}\ x^2_{12}\ \ldots\ x^2_{1m} \\
x^1_{21}\ x^1_{22}\ \ldots\ x^1_{2m} \\
x^2_{31}\ x^2_{32}\ \ldots\ x^2_{3m} \\
x^1_{41}\ x^1_{42}\ \ldots\ x^1_{4m} \\
x^2_{51}\ x^2_{52}\ \ldots\ x^2_{5m} \\
x^1_{61}\ x^1_{62}\ \ldots\ x^1_{6m} \\
\vdots\quad \vdots\quad \vdots\quad \vdots \\
x^1_{n1}\ x^1_{n2}\ \ldots\ x^1_{nm}
\end{pmatrix}
\begin{pmatrix}
x^1_{11}\ x^1_{12}\ \ldots\ x^1_{1m} \\
x^2_{21}\ x^2_{22}\ \ldots\ x^2_{2m} \\
x^1_{31}\ x^1_{32}\ \ldots\ x^1_{3m} \\
x^2_{41}\ x^2_{42}\ \ldots\ x^2_{4m} \\
x^1_{51}\ x^1_{52}\ \ldots\ x^1_{5m} \\
x^2_{61}\ x^2_{62}\ \ldots\ x^2_{6m} \\
\vdots\quad \vdots\quad \vdots\quad \vdots \\
x^2_{n1}\ x^2_{n2}\ \ldots\ x^2_{nm}
\end{pmatrix}
$$

Figure 4: X^1 and X^2 after whole-case crossover

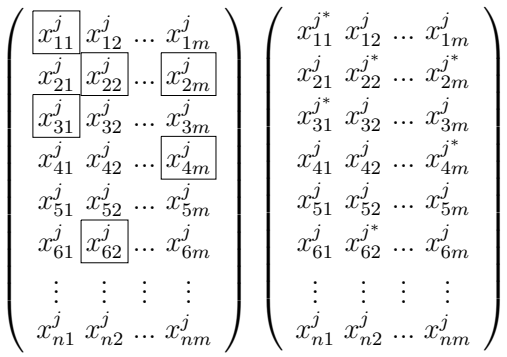


Figure 5: X^j before and after uniform mutation

3.2 Experiment design

The dataset is from the 1901 Scottish Census[9] and consists of 82,851 records. It was subsetted to of 5 variables; parish, sex, marital status, age(recoded into agegroup), and employment status. The employment variable was used as the target variable, with the other variables serving as as the key. We are comparing solutions from a GA with different level of elitism[2], followed by testing if it is possible to find a synthetic data with both of the opposite objectives acceptable. The initial population was formed by 100 datasets that are mutated from the original data, thus they are sufficiently high in utility at the beginning. The generator takes 0.1 and 0.001 as crossover and mutation rates respectively.

3.3 Experiment Results and Discussion

As an output we synthesised a single synthetic dataset which has CAP-U= 0.3964 and $U = 0.1257$. The process took 57 generations to converge to a solution.

[2]Elitism is a concept which indicates the degree to which an GA focuses on the best candidates - it is equivalent to the term "greedy" used in other algorithmic contexts

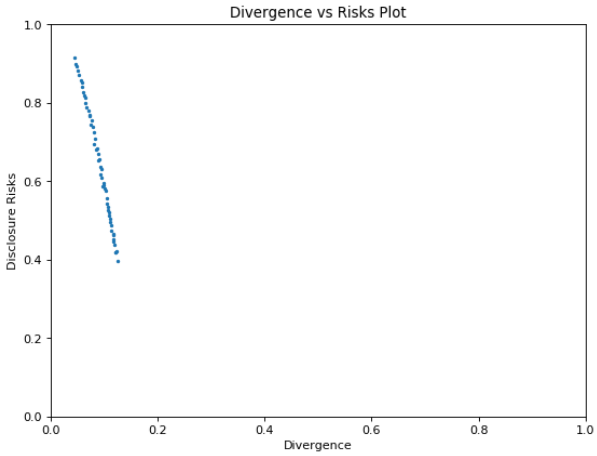


Figure 6: Changing of risk and utility from the best candidates in every generation during optimising process: 2D

3.4 Comparisons to Other Synthesis Methods

As well as the GA synthesised dataset, we also generated a CART and parametric synthetic datasets[3]. Figure 7 shows histograms comparing the counts for the five variables the the original and the synthetic datasets.

The CART synthetic dataset has DCAP= 0.4186 and the parametric synthetic dataset has DCAP= 0.3278. Given that the baseline DCAP score for the univariate is 0.415[4], the GA and parametric dataset would be considered no risk since they are below the baseline and the CART synthetic dataset would have minimal risk since it is very close to the baseline. While visually Figure 7 shows that all three synthetic datasets closely follow the univariate distributions of the original we tested this with a series of chi square tables where:

$$\begin{aligned} \mathrm{H}_0&: \text{original=synthetic} \\ \mathrm{H}_A&: \text{original}\neq \text{synthetic} \end{aligned}$$

The results shown in Table 1 show that for all variables for all synthetic datasets that the null hypothesis is not rejected, implying that the synthetic dataset have the same univariate distributions for their variables.

[3]We used *synthpop* using the generic precepts with synthesising order *parish, sex, marstat, agegroup, employ.*

[4]Wherein the baseline CAP for record j is the marginal probability of its target variables estimated from the original dataset,

$$CAP_{b,j} = Pr(T_{o,j}) = \frac{1}{n}\sum_{i=1}^{n}[T_{o,i} = T_{o,j}] \tag{5}$$

Synthetic Dataset	Variable	Marital Status	Sex	Age Group	Parish	Employment
GA	Chi^2	0.2251	0.0604	1.083	1.6138	0.3425
	P-Value	0.9411	0.8059	0.9992	1.00	0.8426
CART	Chi^2	2.0766	1.4065	5.2701	14.621	2.343
	P-Value	0.7217	0.2356	0.8102	0.9822	0.3099
Parametric	Chi^2	2.4702	0.2878	7.8011	8.7484	1.0358
	P-Value	0.65	0.5916	0.5543	0.9998	0.5958
	DF	4	1	9	28	2

Table 1: Chi Square Comparison Between Original and Synthetic Datasets

We also ran two sets of alternative utility tests. Table 2 shows the propensity mean square error (pMSE) score (Woo et al, 2009) for the three synthetic datasets[5]. Table 2 also includes the standard pMSE and the pMSE ratio. Both introduced by Snoke et al (2018)[12]. The closer the pMSE is to 0 the better the data performs. In this instance all synthetic datasets have quite low pMSE scores, however the GA performs slightly better than the CART and parametric datasets. The GA does not perform as well on the standardised pMSE and pMSE ratio. According to Snoke et al, the standardised pMSE has an expectation of 0 and the pMSE ratio of 1. Hence the CART and the parametric synthetic datasets do perform better in this respect.

[5]To calculate the propensity scores we used a logistic regression consisting of $k = 45$ parameters with no interaction variables.

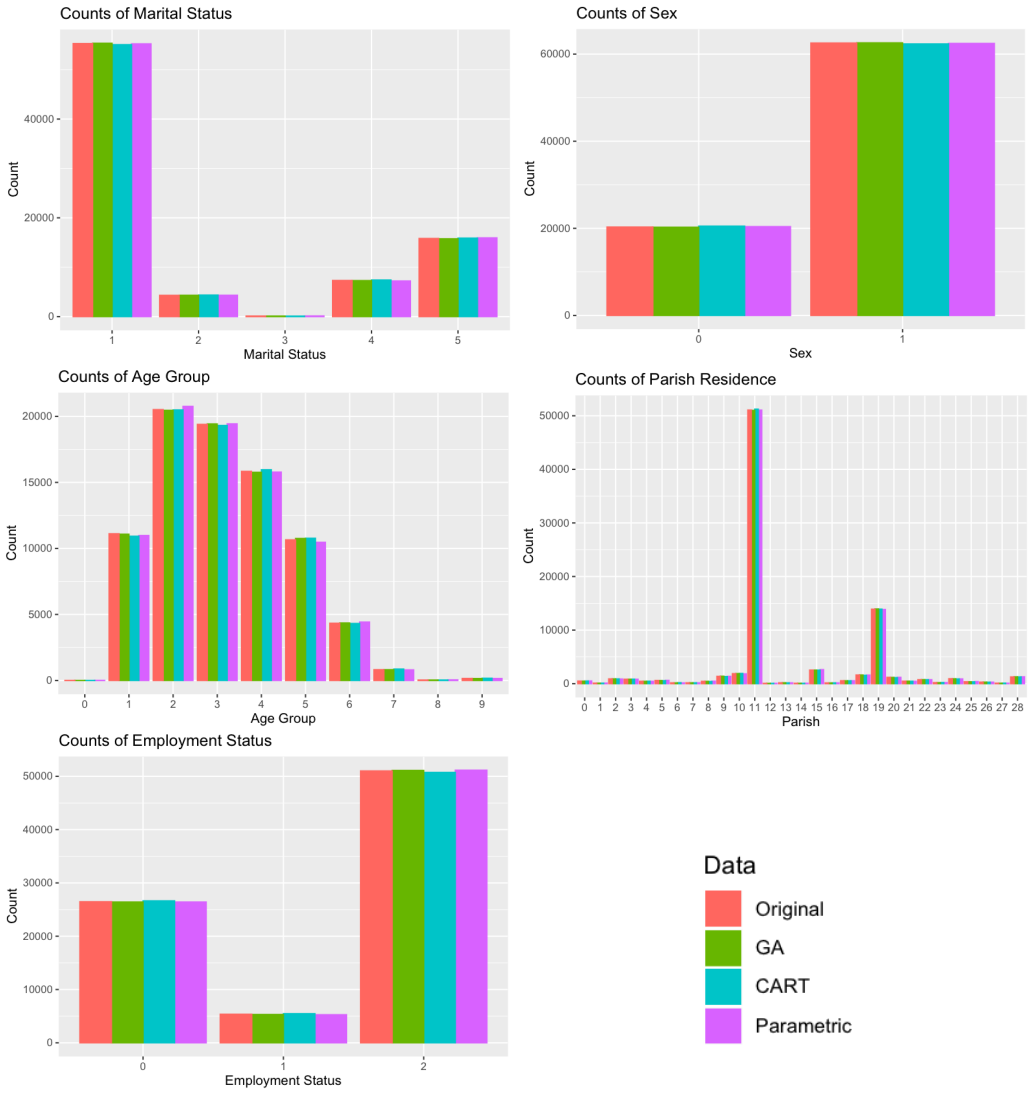


Figure 7: Histogram Comparing Original Data to GA, CART, and Parametric Synthetic Data

Synthetic Dataset	pMSE	Standardized pMSE	pMSE ratio
GA	5.44E-06	-3.9221	0.1638
CART	3.397e-05	0.1106	1.0236
Parametric	3.17e-05	7.077e-06	0.9553

Table 2: Propensity Scores for Synthetic Datasets

While the pMSE is a good broad measure of data utility, we also wanted to test the synthetic datasets in terms of narrow measures. To that end, we also ran a ratio

of estimates (ROE) (see Taub et al, 2016) over a series of bi-variate cross-tabulations, shown in Table 3. Table 3 shows that all three synthetic datasets average high ROE scores, wherein a score of 1 would be a replica of the original datasets. The GA average is slightly less than that of the CART and parametric synthetic datasets.

Variable 1	Variable 2	GA	CART	Parametric
Marital Status[6]	Sex	0.7891	0.9486	0.9454
Marital Status	Age Group	0.8582	0.9218	0.8742
Marital Status	Parish	0.9051	0.8864	0.8487
Marital Status	Employment	0.8858	0.9426	0.9469
Sex	Age Group	0.8988	0.9661	0.9031
Sex	Parish	0.9529	0.9215	0.9462
Sex	Employment	0.8931	0.9886	0.9818
Age Group	Parish	0.9262	0.8548	0.8798
Age Group	Employment	0.8665	0.8693	0.9324
Parish	Employment	0.9485	0.8985	0.9106
	Average	0.8924	0.9198	0.9169

Table 3: ROE Scores for Two Variables Cross-Tabulations

4 Conclusion

In this paper, we have reported on the use of GAs to produce synthetic data and in particular of embodying the risk utility trade-off within a single algorithm.

Our experiments indicate that GAs are viable alternative to standard synthesising methods. The GA produced synthetic data was below the baseline CAP score indicating that it had low disclosure risk. The GA synthetic data performed similarly to the CART and parametric synthetic datasets (two established forms of data synthesis) on the utility tests.

Further experiments testing different parameter settings and then implementations with larger (more realistic) datasets are needed. GA unlike previous methods for data synthesis could prove a very useful tool in that it's disclosure risk level can be pre-set, instead of being left as a post-hoc question.

References

[1] Abowd, J. M., and Lane, J. (2004). New approaches to confidentiality protection: Synthetic data, remote access and research data centres, In *Privacy in statistical databases*, Springer, Berlin Heidelberg, 282-289

[2] Chen,Y., Elliot M., and Sakshaug, J. (2017). Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data, UN-ECE Work Session on Statistical Data Confidentiality. Ljubljana, October 2018. `https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.46/2017/2_Genetic_algorithms.pdf.Lastaccess20/12/2017`.

[3] Chen, Y.(in preparation). Genetic Algorithms and their Application to Synthetic Data Generation. PhD Thesis to be submitted to the University of Manchester. Expected submission date December 2019.

[4] Duncan, G. T., Keller-McNulty, S. A. and Stokes, S. L. (2004). Database security and confidentiality: Examining disclosure risk vs. data utility through the R-U confidentiality map, Technical report. Downloaded from https://tinyurl.com/Duncanetal04 [accessed 12/09/2019]

[5] Elliot, M. (2014). Final Report on the Disclosure Risk Associated with the Synthetic Data Produced by the SYLLS Team. [online] CMIST. Available at: https://tinyurl.com/syllsDR

[6] Konak, A., Coit, D. W., and Smith, A. E., (2006). Multi-objective optimization using genetic algorithms: A tutorial, *Reliability Engineering and System Safety*, 91(9), 992-1007.

[7] Lu, H., and Yen, G., (2002), Rank-Density-Based Multiobjective Genetic Algorithm, *Proceedings of the 2002 Congress on Evolutionary Computation 2002*, 944-949

[8] Navarro-Arribas, G. and Torra, V. (2015). Data Privacy: A Survey of Results, Advanced Research in Data Privacy, *Studies in Computational Intelligence*. Vol. 567. Springer Switzerland, 27-37

[9] National Records of Scotland, (1901), 1901 Scottish Census.

[10] Raab, G. (2018). Personal correspondence.

[11] Shlomo, N., (2010). Releasing Microdata: Disclosure Risk Estimation, Data Masking and Assessing Utility', *Journal of Privacy and Confidentiality*, vol. 2, no. 1, 73-91.

[12] Snoke, J., Raab, G., Nowok, B., Dibben, C. and Slavkovic, A. (2018). General and specific utility measures for synthetic data. Journal of the Royal Statistical Society Series A (Statistics in Society).

[13] Taub, J., Elliot, M., and Saukshaug, J. (2017) A Study of the Impact of Synthetic Data Generation Techniques on Data Utility using the 1991 UK Samples of Anonymised Records. In proceedings of UNECE Statistical Data Confidentiality Work Session.

[14] Taub, J., Elliot, M., Pampaka, M. and Smith, D. (2018). Differential Correct Attribution Probability for Synthetic Data: An Exploration. J. Domingo-Ferrer and F. Montes (Eds.): PSD 2018, LNCS 11126. pp. 122-137

Chapter 6

Summary and Model Integration

The thesis demonstrates that imputation is not the only applicable method in data synthesis. With reasonable settings of utility and disclosure objectives, GAs are also capable of producing satisfying outputs. The GA synthesiser designed in the thesis has shown the ability to produce promising results for different datasets (see chapter 5)

Data synthesis, as an SDC technique, aims to generate a new, full or partial synthetic data that have a measurable closeness to the original data but hide individual respondent's confidential information. The background of SDC was summarised in the chapter Statistical Disclosure Control, including several common techniques for SDC like perturbative and non-perturbative masking. Data synthesis, one of the SDC approaches, was specifically reviewed in the remainder of the chapter. Starting from the initial conceptualisation by Rubin [103], the chapter introduces most of common data synthesisers now in use, such as multiple imputation [103][98], CART [99][17] and DPMPM [65][66], and common measurements of information utility and disclosure risk. The whole chapter outlines the theoretical basis for designing GA synthesisers.

Chapter: Machine Learning, Natural Computation and Genetic Algorithms introduced the fundamental theories of GAs, including different methods for operators and settings for parameters in traditional GAs, common frames for multi-objective GAs and Matrix Real Coded GAs (MRCGAs), which provided the practical basis for the design of GA synthesisers. It also raised issues to be considered in later chapters, such as which method for each operator the GA synthesiser should adopt, how to implement multi-objective GA synthesisers and whether to use Pareto GAs here.

The idea of implementing GAs in data synthesis was expanded and developed in the next chapter: Model Design. It explored the potential of GAs in data synthesis by describing almost all possible methods for the different operators in the GA synthesiser.

Since parameter design is a key to improve the efficiency of a GA for a particular problem, this chapter assists in making decisions on what and how operator methods would be compared. Moreover, it also discussed how to adapt information utility and disclosure risks measures to objectives in a GA synthesiser.

Chapter Experiments and Results consists of seven papers that have been or are likely to be published. The experiments as a group aim to identify suitable operators/parameters settings for a GA synthesiser. The paper The Impact from Initial Population in GA Synthetic Data Generator tested two characteristics, fitness and diversity, in the initial population and summarised their impacts on the efficiency and effectiveness in GA synthesisers. The paper defined the diversity in the population of a GA synthesiser as the standard deviation of Jensen-Shannon divergences between all pairs of candidates. It concludes that it is not always true that a fitter initial population performs better in searching and diversity also matters. Papers The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods [22] and *Matrix GA: Building Blocks in Data Synthesis* concentrated on crossover operator design. Besides finding that whole-case parallelised crossover (illustration of this method is in Fig 4.7) is the most suitable for GA synthesisers compared with all six methods in section 4.3, it also conjectures how synthetic data are optimised in a GA synthesiser by illustrating the process of building blocks. The paper Exploring the Impact of Adaptive Parameters on a Genetic Algorithm Synthesiser shows that a plain and fixed operator may not deliver the best candidates to next generation at all points in the process and claims that endurance of power from suitable operators can last by using adaptive parameters or dynamic control to the whole process. Nevertheless, there are no definite rules about how to set up self-adaptive parameters in a GA synthesiser due to the lack of any theory with which to evaluate the impact of adaptive or dynamic parameters on a GA synthesiser. So far, adaptive functions and dynamic control points can only be defined with a good understanding of the input data. The paper, Impact of Full Contingency Table in Data Synthesis, explored the potential of using only the full contingency table when measuring utility and compared synthetic datasets generated by GA and CART. The paper demonstrates that, while using full contingency table as the only objective in data synthesiser will eventually terminate on the original data, GA synthesiser still outperform CART. Paper Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator involves DCAP as a risk objective in GA synthesisers to contest with full contingency tables and shows that GA is capable of finding the trade-off between conflicting objectives and therefore produces satisfying synthetic data.

6.1 Model Integration and Flowchart

As most of the previous contents studied only a component (either an operator or parameters) in GA synthesiser, in this section, I shall integrate them and deliver a full view of the whole model. The synthesiser equips all operator methods that were tested in previous chapters. It is agile on operator and parameter settings.

GA synthesisers require initialisation (Fig 6.1). i.e. to select particular operator methods and parameters for the model before running. The first two inputs are the original dataset and the population size. For most experiments in this thesis, the population size is set to 100. The initial population can be generated from a selected generator. Options in the synthesiser include uniform model, univariate model and mutation model. Uniform and univariate models generate candidates by independently sampling (with replacement) from the uniform and univariate distributions of each variable from the original data. Mutation model generates candidates by mutating each cell in the dataset with a pre-determined mutation rate. Alternatively, users can choose to load the initial populations generated by other synthesisers (like CART) from a `.csv` file. In the next step, there are two boolean questions to be answered: 'Is the synthesiser adaptive? and 'Is the synthesiser dynamic?. If the answer of the first question is 'Yes then the model includes adaptive functions for crossover and mutation rates $F(p_c), F(p_m)$ (the synthesiser uses linear functions with pre-determined coefficients so far, see paper: Exploring the Impact of Adaptive Parameters on a Genetic Algorithm Synthesiser), otherwise users are required to inputs fixed crossover and mutation rates p_c, p_m. As for the second question, if the answer is 'Yes then a pre-determined turning condition will be active (also see paper: Exploring the Impact of Adaptive Parameters on a Genetic Algorithm Synthesiser).

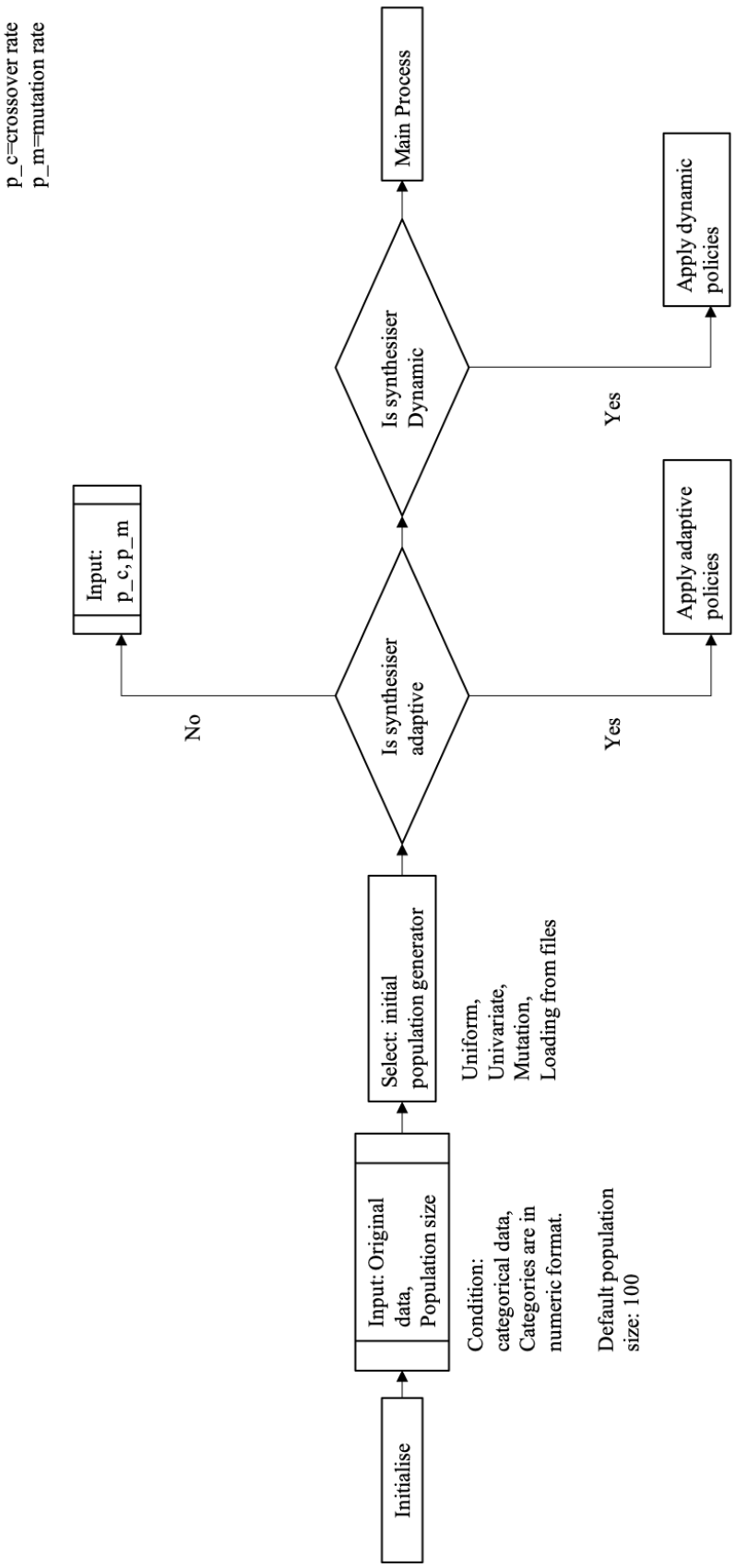


Figure 6.1: GA synthesiser flowchart part I: initialising synthesiser

In the next step, users make a decision on which methods are used in the selection operator, the crossover operator and the mutation operator in the synthesiser. Methods included in the thesis are listed in Table 6.1. Tournament selection ($t = 2$) and Whole-CPC are demonstrated to be the most suitable methods for this synthesiser. Other methods can also be used depending on users' preference or for any interesting research in the future.

Selection methods	Crossover methods	Mutation methods
Linear ranking	Case Parallelised	Uniform*
Tournament*	Round-CPC	
Roulette wheel	Whole-CPC*	
	Variable Parallelised	
	Matrix	
	PUC	

Table 6.1: List of methods of selection, crossover and mutation, * indicates the method is proved to be the most suitable one for the synthesiser so far.

In the main process, the synthesiser (Fig 6.2) starts from evaluating fitness values of candidates the initial population. The model uses a full contingency table and DCAP in measuring utility and risk objectives. The whole process iteratively optimises the population until terminating conditions satisfied. There are two terminating conditions (1) at least one candidate in current generation reaches the trade-off of the risk and utility, and (2) the whole population converges to one candidate for a specific number (the default value is 20) of generations.

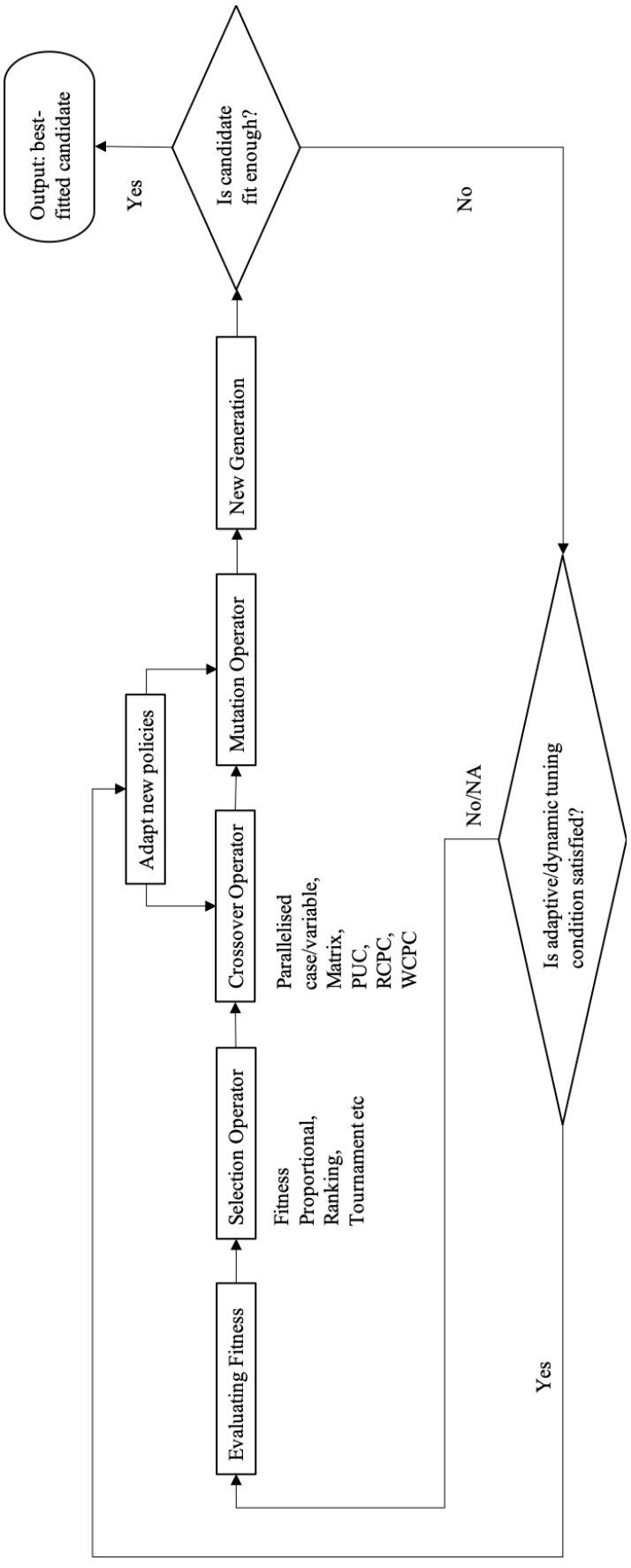


Figure 6.2: GA synthesiser flowchart part II: main process

Chapter 7

Impacts and Critical Analysis of the Model

7.1 Impacts

This thesis investigates the potential of implementing a genetic algorithm to data synthesis. Although machine learning is not new to synthetic data production (see [99][38]), the thesis represents the first attempt to tackle this problem using GAs. GAs have different properties to other machine-learning data synthesisers to date. GAs simulate more complicated aspects of real-world problems through the interaction between learning agents and environments by given reward functions. Candidates gain rewards (chance to survive) from interacting with the environment (objectives). Building these features into the design of a data synthesiser enables the use of both utility and risk objectives to determine whether individuals can survive and reproduce. Specifically, it allows us dynamic monitoring on the change of utility and risks in candidates during the whole process.

7.1.1 Impacts on the SDC field

The GA synthesiser successfully produced synthetic data by using the full contingency table as the unique utility objective. It confirms that synthetic data that has better utility also has less divergence from the full contingency table of the original data and demonstrates the possibility of producing sufficiently similar synthetic data with a limited set of statistical properties. Full contingency tables are capable of measuring both the utility and some disclosure risks of the synthetic data and, compared with other utility measurements, they give a more straightforward view of the divergence to the original data. Therefore, using only this measurement can help to capture the change of risk and utility in synthetic data

during the process of GA synthesisers.

Paper Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator demonstrates the existence of a trade-off between disclosure risk and information utility in data synthesis even though they were proved to conflict with each other [42][43]. The success of GA synthesiser in this paper also indicated that utility and risk measures, which used to be post-hoc tests, now can be parts of synthesising models.

Using a full contingency table as the only objective in GA synthesisers will eventually end up to the original data. Therefore, on the other hand, it proves that there is no actual gap between data synthesis and data masking in categorical data. First, crossover in GAs has a similar mechanism to data swapping. Instead of swapping records with the equivalent data within another set of records chosen based on matching variables, crossover swaps records in the same position from two synthetic datasets. Depend on which crossover method is adopted, the number of records swapped can vary from a single cell (PUC) to a sub-matrix (matrix crossover). Secondly, it unlikely to have another dataset that has completely different records as the original data has the same statistical utility. The better the utility the synthetic data is, the closer it to the original data. GA synthesisers illustrate the solution space of data synthesis by recording how risk and utility change during its optimising process, and it will be discussed later in the chapter.

7.1.2 Impacts on the GA field

The thesis connects the two fields: GAs and data science and confirms that the algorithm works well in protecting data confidentiality in a multi-objective environment.

The thesis systematically studied how MRCGA is applied to data synthesis. It investigated whether operator methods in binary GAs like two-point crossover and PUC can be adapted to MRCGAs. After finding that the common crossover methods are not efficient, two new crossover methods (round-CPC and whole-CPC) were designed. Round-CPC and whole-CPC retain the relationships between variables better than some of the common crossover methods and have no positional bias. They might have future applications of GAs in other data science problems. The thesis also explained how building blocks and self-adaption from binary GAs changed in MRCGAs by investigating how building blocks and adaptive parameters work in GA synthesisers: Paper Matrix GA: building blocks in data synthesis illustrates two processes that a GA synthesiser combines lower-order schema from inferior candidates to higher-order schema to improve the population's overall fitness. Meanwhile, paper Exploring the Impact of Adaptive Parameters on a Genetic Algorithm Synthesiser discusses whether adaptive parameters can be applied to matrix GAs or GA synthesisers, and how they impact on the efficiency and effectiveness of

the model.

7.2 Critical Analysis of the Thesis

7.2.1 Full Synthesis and Partial Synthesis

Full synthesis produces synthetic versions for all of the records within the original data. It presumably contains lower disclosure risks than partially synthetic data. It was declared at the beginning of the thesis that the GA synthesiser is designed for synthesising full, categorical, single-level data because (1) there is no need to decide target variables and (2) I only considered the utility of synthetic data at that time and the utility function ought to be designed over the whole dataset. However, in paper Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator we used utility function over the whole dataset but risk function over one target variable. It is then unable to tell, in this circumstance, if the data produced is fully synthetic data or partially synthetic data. Moreover, using full contingency tables in the utility objective loses the advantages in full synthesis (because it will eventually stop at the original data). Therefore risk should also be concerned while using GA synthesisers.

7.2.2 Model Stability

Even though GA synthesisers produced good-quality synthetic data in many of the cases, the stability of the model is not measured yet, i.e. whether the model outputs are homogeneous. Stability of a data synthesiser can be evaluated through the variance of the estimators of interesting properties from their outputs [95] [40]. Ideally, the smaller the variance is the stable the synthesiser is and the more homogeneous result it produces. This is also observed in GA synthesisers from where experiment results in papers The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods and Matrix GA: building blocks in data synthesis, the standard deviation of a set of outputs from one dataset is sufficiently small. However, there is no mathematical proof yet for this hypothesis, and it might be an interesting matter for future research.

7.2.3 The Application of GA Synthesisers in Continuous and Mixed Datasets

The whole thesis focused on synthesising categorical data because they are commonly used in social survey and census data. Working on only categorical data also obeys the convention of data confidentiality protection, which asks one model to be applied to one type of variables [27]. Nevertheless, since GAs use a different mechanism from other synthesisers, the synthesiser has potentials to work on continuous data and data that forms with both continuous and categorical variables (mixed data). Theoretically, as long as the format of microdata stays the same, all operator methods in the categorical GA synthesiser apply to continuous/mixed datasets. The main obstacles of adapting current GA synthesisers in continuous or mixed data is the design of fitness function, which has two possible solutions: (1) to re-design fitness functions for continuous/mixed datasets and (2) to discrete-lise continuous variables in the datasets.

7.2.3.1 Fitness Function Design for Continuous Datasets

Mateo-Sanz et al. used to suggest that the utility of continuous synthetic data should retain at least mean and covariance matrix [81]. These were used as the only two properties in their synthesising model [80]. However, mean and covariance only capture bi-variate structures, which is not sufficient in multivariate data. Partial correlation coefficients are commonly used in determining relations between multiple continuous variables. Given a set of controlling variables $\mathbf{x} = \mathbf{x_1}, \mathbf{x_2}, ...$, the partial correlation between variable $\rho_{x_1 x_2, \mathbf{x}}$ is the correlation between residuals e_{x_1} and e_{x_2} from linear regression of x_1 and of x_2 with bfx. Alternatively, the relation in multiple continuous variables can be assessed by coefficients from their analysis models, like multiple correlation R^2 from their linear regression models.

7.2.3.2 Fitness Function Design for Mixed Datasets

Fitness function design is more complicated in a mixed dataset. Olkin and Tate ever suggested a multivariate correlation model for mixed data [91]. Suppose $\mathbf{x} = (x_1, x_2, ...x_m)$ and y be a set of independent random categorical and a continuous variable. Also suppose $\mathbf{x}$ is multinomially distributed and every variable in $(x_1, x_2, ...x_m)$ has binomial distribu-

tion.

$$f(x_1, x_2, ...x_m) = \prod_{k=1}^{m} p_k^{x_k},\ x_k = 0, 1;$$
$$\sum_{k=0}^{m} x_k = 1,$$
$$0 < p_k^{x_k} < 1,\ \sum_{k=0}^{m} p_k^{x_k} = 1$$

For given $x_k = 1$, the conditional distribution of $\mathbf{y}$ is multivariate normal $\sim N(\mu^{(k)}, \Sigma)$ where $\mu^{(k)} = (\mu_{1k}, ..., \mu_{pk})$, where $k = 0, ..., m$ and Σ is a positive definite covariance matrix. Results from multivariate correlation models can then be assembled to the full contingency table by adding extra dimension. The following table illustrates situation for $\mathbf{x} = (x_1, x_2)$ and y:

	$x_2 = 0$	$x_2 = 1$
$x_1 = 0$	$n_{(0,0)}$	$n_{(0,1)}$
$x_1 = 1$	$n_{(1,0)}$	$n_{(1,1)}$

Table 7.1: A contingency table for $\mathbf{x} = (x_1, x_2)$

	$x_2 = 0, y$	$x_2 = 1, y$
$x_1 = 0, y$	$y\|(x_1 = 0, x_2 = 0)$	$y\|(x_1 = 0, x_2 = 1)$
$x_1 = 0, y$	$y\|(x_1 = 1, x_2 = 0)$	$y\|(x_1 = 1, x_2 = 1)$

Table 7.2: Adding extra dimension for the conditional distribution of y to table 7.1

The model assumes every categorical variable in the dataset to be binary, otherwise it requires extra dummy variables, which massively increases the workload to the synthesiser. A more efficient model is required for evaluating the utility of mixed datasets, and it will be an interesting topic for future research. Without such a feasible measure to the dataset's fitness, an alternative approach is to discrete-lise continuous variables in mixed datasets, so they are usable in the current (GA synthesiser) model.

7.2.3.3 Discrete-lising Continuous Variable in Mixed Datasets: an Example

This section gives an example of how to discrete-lise a continuous variable step by step. One possible way to discrete-lise a continuous variable is to convert its records according to its histogram bins. For example, LIMIT_BAL is a continuous variable recording a bank's monthly credits given to its clients. The value varies from £10000 to £800000.

CLIENT_ID	LIMIT_BAL
1	210000
2	330000
3	50000
4	180000
⋮	⋮
22500	750000

Table 7.3: LIMIT_BAL

Its default histogram generated by `numpy.histogram` in Python 3.7 contains 8 bins, and each can be assigned a position index, which is integer in $[0, 7]$.

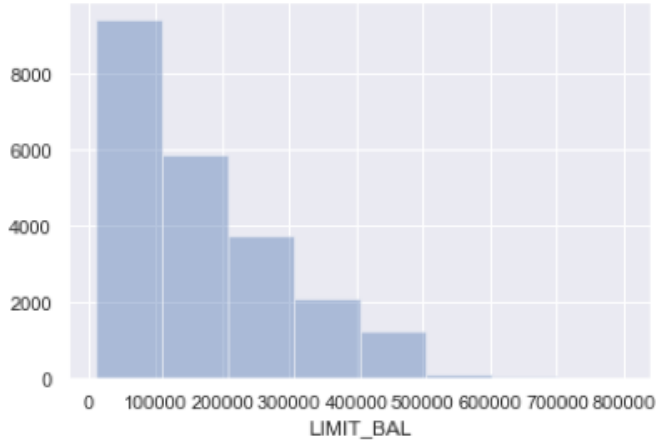


Figure 7.1: The histogram of LIMIT_BAL

Bin	[10000, 108750)	[108750, 207500)	[207500, 306250)	[306250, 405000)	[405000, 503750)	[503750, 602500)	[602500, 701250)	[701250, 800000)
Index	0	1	2	3	4	5	6	7
Counts	9414	5865	3760	2099	1220	84	42	16

Table 7.4: Explanation of the histogram of LIMIT_BAL, including boundaries of bins, position indices and counts of client

Using information from Table 7.4, LIMIT_BAL now can be discrete-lised to an 8-categories variable by replacing its true records with their corresponding position index.

CLIENT_ID	LIMIT_BAL
1	2
2	3
3	0
4	1
⋮	⋮
22500	7

Table 7.5: Discrete-lised LIMIT_BAL

It should be aware that discrete-lising continuous variables always sacrifices some information utility because the variable will not be perfectly restored. Therefore, selecting an appropriate approach to restore the records in discrete-lised variables is important in controlling information loss under a certain level. These records can be restored by randomly sampling values within the range of their corresponding bins using a chosen distribution. Uniform distribution $U(a,b)$ was used in this case, of those parameters (a,b) are determined by the boundaries of bins, for example, if the record in discrete-lised LIMIT_BAL is 2, then it will be restored by sampling from $U(207500,306250)$, where $[207500,306250]$ is the range of the corresponding bin with position index 2. The restored version of LIMIT_BAL is:

CLIENT_ID	LIMIT_BAL
1	225925
2	332914
3	93020
4	175166
⋮	⋮
22500	752882

Table 7.6: Restored LIMIT_BAL by randomly sampling values from their corresponding bins using its uniform distribution

Pearsons correlation between the original and restored LIMIT_BAL is 0.9984, which confirms that the two variables have very similar trends. Their histograms also prove the similarity between these two variables and verify that uniformly sampling values from its corresponding bin is a feasible approach to restore the discrete-lised version of LIMIT_BAL thus, discrete-lising by its histogram works for this variable.

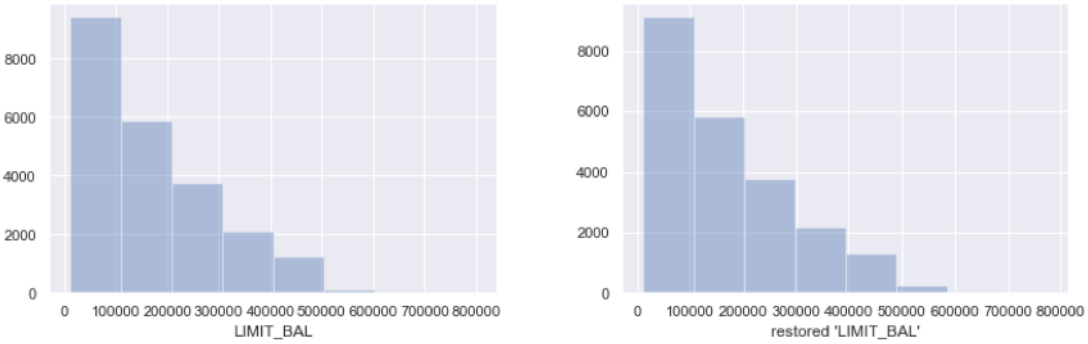


Figure 7.2: Histograms of original LIMIT_BAL and restored LIMIT_BAL

Discrete-lising is so far the most feasible approach to synthesise mixed datasets in GA synthesisers. However, as the method always sacrifices information utility and sometimes the modified variables may not be restored. It should be used with cautions. Finding proper fitness functions for mixed datasets is still desirable and will be an interesting research question.

7.2.4 The Comparison between Single-objective and Pareto-Optimal GA Synthesisers

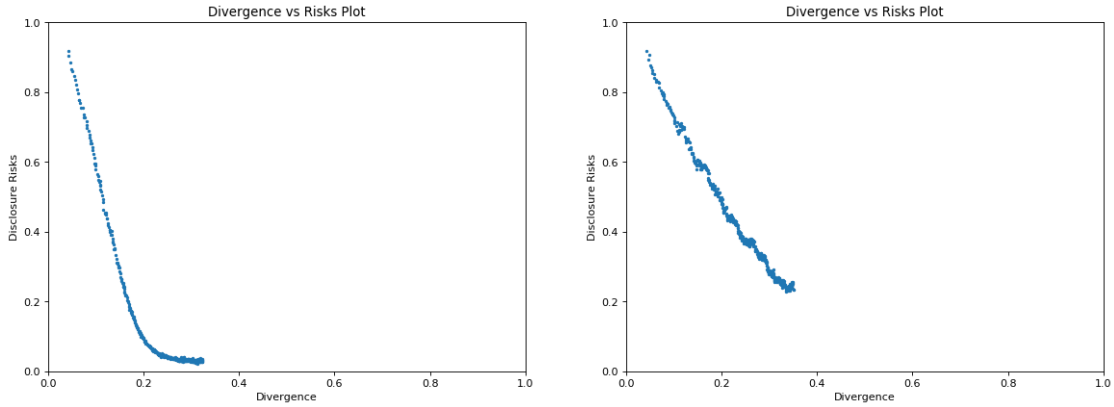
When designing a GA for a real-world problem, there is invariably more than one objective to be considered. These objectives are sometimes opposed to one another, and this applies to information loss and disclosure risk in synthetic data. One general approach to deal with these objectives is to compose them into a single function with determined weights (single-objective GAs). We used this method in paper Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator by given equal weights to utility and risk objectives. However, the selection of weights is easy to get wrong and even a small difference in weighting can lead to different solutions. An alternative method is to compute a Pareto optimal solution set in where no candidate is dominated by one another (see Def 3.9.1. Common frameworks of Pareto-optimal GAs were introduced and compared previously (section 3.9). In this section, I shall discuss the potential of using Pareto-optimal GA in data synthesis by using Niched-Pareto GA (NPGA).

Horn et al. proposed NPGA in searching for Pareto-optimal solutions. Its selection operator consists of two parts: Pareto domination tournaments and non-dominant tournament. The first part helps to find dominantly optimal candidates, and the second one keeps diversity in the population during the process [64]. Details of NPGA was given in section 3.9.1.1.

7.2.4.1 Experimental Results for an NPGA and a SOGA models

NPGA is run with the same settings as in paper: Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator. Recall the data and experiment design in this paper: the dataset consists of 82,851 records. It was subsetted to consist of 5 variables; *parish*, *sex*, *maristal status*, *agegroup*, and *employ*. The *employ* variable was used as the target variable, with the other variables serving as the key. Initial populations for both GAs contain 100 datasets that slightly mutated from the original data. They are sufficiently high in utility at the beginning so low rates for crossover and mutation, 0.1 and 0.001 respectively, are used. Selection method is the only difference between the two models: SOGA was equipped with tournament selection with tournament size $t = 2$ and NPGA used Pareto tournament with $m_i = 5$ and $d_{i,j} = 0.05$.

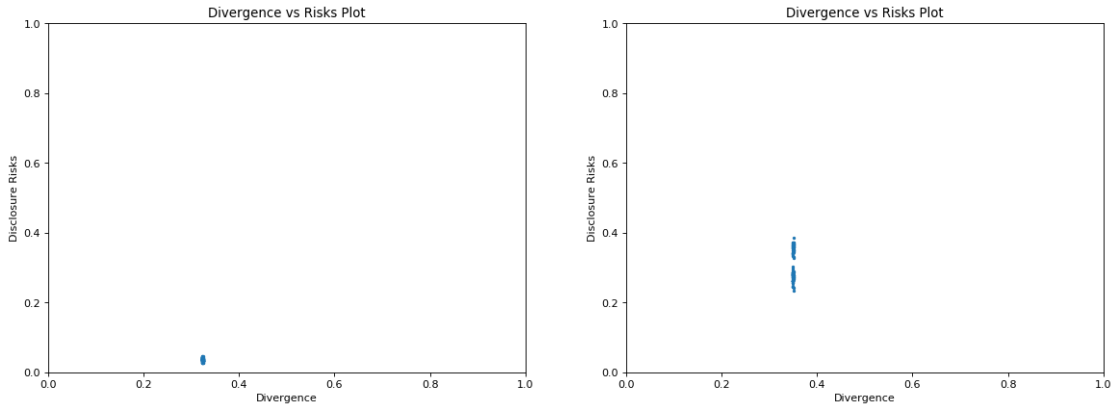
The two GAs were ran for 500 generations and the **best candidate** in every generation was recorded. As shown in Fig 7.3, SOGA showed a sharper trend then NPGA; it instantly reduced the risk at the beginning then the whole population converging after approximately 200 generations. Meanwhile, SOGA did not explore much in the solution space. NPGA, on the other hand, intended to find more solutions during its process. Both models went to a very low-risk level after 500 generations.



(a) SOGA (b) NPGA

Figure 7.3: Changing of the risk and utility from the best candidate in SOGA and NPGA in 500 generations

Initial populations for both models were formed by datasets that slightly mutate from the original data so they had high utility (low divergence to the original data) and all candidates were similar at the beginning. However, after 500 generations, populations in SOGA and NPGA showed different distributions (Fig 7.4).



(a) SOGA (b) NPGA

Figure 7.4: Divergence-Risk map for the last population in SOGA and NPGA after 500 generations

The experiment confirms the capability of either SOGA or NPGA in optimising synthetic data with contrast objectives. SOGA is more efficient in finding optimal (trade-off) solutions. However, more diversity appeared in solutions found during the process of NPGA as well as its last population, which considerably takes cares of different preference from users.

7.2.5 Jensen-Shannon Divergence in Full Contingency Table: Advantages and Disadvantages

Jensen Shannon divergence D_{JS} in full contingency table is an important statistic the thesis: it was used as the only measure of information utility in most experiments. Paper Impact of Full Contingency Table in Data Synthesis demonstrated that synthetic data with smaller D_{JS} in the full contingency table to the original data would maintain more statistical properties. Full contingency tables were also used to determine disclosure risk (DCAP) in paper Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator. Barak et al. confirmed that contingency table could evaluate not only the accuracy of a synthetic dataset by comparing synthetic and original records but also privacy and consistency of the synthetic dataset by checking the presence or absence of any single record that may or may not substantially contain disclosure risks. Thus, besides of DCAP, full contingency table enables any independent and post-hoc privacy model to serve as a risk objective for GA synthesisers, for example, differential privacy. Given the definition of differential privacy (Def 2.3.5), the relationship between ε, δ and contingency table is defined as [8]:

Definition 7.2.1. Let C be a set of marginals of the contingency table for at most j bi-

nary variables. Marginals from C', a positive, integral contingency table, preserves ε-differential privacy, such that with probability $1-\delta$ for any marginal $c \in C$:

$$||c - c'||_1 \leq 2^{j+3}|C| \log \frac{|C|}{\delta}/\varepsilon + |C|$$

Moreover, paper Impact of Full Contingency Table in Data Synthesis indicates that GA synthesiser is $(\varepsilon, 0)$-differentially private, where outputs from this model are equally likely to be observed on every neighbouring dataset of the original dataset (neighbouring datasets are defined as datasets that only has one record different from each other [46]). Meanwhile, in the solution space of GA synthesiser, ε is equivalent to the boundary of candidates whose risk exceed the maximum tolerable value (see Fig 7.5).

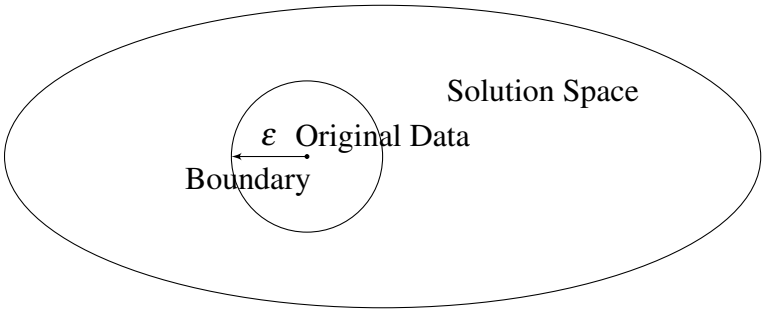


Figure 7.5: ε in the solution space of a GA synthesiser

Although D_{JS} in full contingency tables enabled the GA synthesiser to successfully produced many synthetic datasets in this thesis. I noticed that the calculation of the statistic is less efficient when the number of variables increases as the function creates more sparsity in the contingency table. Sparsity describes a matrix or dataset in which most of the elements are zero. It costs more storage space and operational time and undesirably diminishes the difference between two datasets or matrices when the volume of zeros overwhelms non-zeros. There are several solutions. For example, Connor et al. proposed an algebraic deduction to reduce the similarity in sparse data [28]. Recall the definition of D_{JS}: D_{JS} is designed from Kullback-Leibler divergence D_{KL}, which measures the divergence between two probability distributions. For two discrete probability distributions P and Q:

$$D_{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$$

By given $M = \frac{1}{2}(P+Q)$,

$$D_{JS}(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M)$$

Connor et al. introduced a kernel function $\mathscr{F}(x,y)$:

$$\mathscr{F}(x,y) = h(x) + h(y) - h(x+y)$$

, where $h(x) = -x\log_2(x)$, to transfer the formulae of D_{JS} to:

$$D'_{JS}(P||Q) = 1 - \frac{1}{2}\sum_i \mathscr{F}(P(i), Q(i)) \tag{7.1}$$

Now the divergence only takes values in $P(i)$ *and* $Q(i)$ that are non-zeros. $\mathscr{F}$ can be regarded as a similarity 'accumulator' with threshold value r, where $r = 2$ means perfect similarity:

$$\sqrt{1 - \frac{1}{2}\sum_i \mathscr{F}(P(i), Q(i))} < r$$

Other solutions that can be adopted by GA synthesisers to solve sparsity in D_{JS} include [113] and [77]. Solving sparsity in high-dimensional statistics like full contingency table is an interesting topic, and I shall leave it for future research.

Besides of sparsity, another disadvantage of D_{JS} is that the divergence is not a metric. A metric d is a distance measure between every pair of points x, y in a metric space M. It is used in measuring the diversity of populations in GAs [70] or similarity between datasets [126]. M must satisfy three properties:

1. $d(x,y) \geq 0$ and $d(x,y) = 0$ iff $x = y$.
2. (*symmetry*) $d(x,y) = d(y,x)$
3. (*triangle inequality*) $d(x,y) \leq d(x,z) + d(z,x)$

Mathematically, these properties express intuitive notions about the concept of **distance** but D_{JS} indeed does not obey the two of them (triangle inequality and symmetry). It prevents us from thinking about the solution space using common senses that we usually used in Euclidean geometry. Therefore, although we attempted to illustrate solution space and divergence-risk map for data synthesis in the thesis like Fig 7.5 and Fig 7.4, they did not accurately present the real information of the solution space. So does in paper The Impact from Initial Population in GA Synthetic Data Generator, where D_JS was used to measure population diversity, but the diversity is not in a metric space. Moreover, using a non-metric distance measure may be the primary cause of failure in the NPGA model, which requires to calculate the distance between candidates in niches. It is foreseeable that other Pareto-GAs may fail due to this reason and research on how to metricise the divergence between two datasets should be conducted.

This section suggests two topics for future research: an efficient way to ameliorate the effect of sparsity in Jenson-Shannon divergence and a metricised distance measure between synthetic datasets. The former can enable GA synthesisers to work on larger datasets and improve its computational efficiency. The latter provides not only the real representation of the solution space of GA synthesisers or other privacy models but also might enable multi-objective GAs to find Pareto-optimal solutions.

7.3 Chapter Summary and Closing Remarks

In this chapter, I gave some critical analysis to the thesis and proposed potential research topics for future study of the GA synthesiser. Although the implementation of GAs in data synthesis was demonstrated successful, there are still many under-investigated potentials.

GAs showed agility during the process of synthesising and efficiency in producing acceptable synthetic data by monitoring both utility and risk objectives. The success of generating synthetic data using GAs give the idea that imputation is not the only way to synthesise microdata and it can be achieved by reinforcement learning algorithms as long as the definitions and functions of objectives are clear and universal.

Appendix

A Genetic Algorithm Approach to Synthetic Data Production

The paper explained the initial framework to synthetic data generation using genetic algorithms. This framework was believed applicable because its evolutionary and competitive process allows to comparing different synthetic versions of the original data and inheriting their advantages into the optimal solution.

A Genetic Algorithm Approach to Synthetic Data Production

Yingrui Chen
University of Manchester
Oxford Road
Manchester M13 9PL
+44 (0)161-275-4257
yingrui.chen@manchester.ac.uk

Mark Elliot
University of Manchester
Oxford Road
Manchester M13
+44 (0)161-275-4257
mark.elliot@manchester.ac.uk

Joseph Sakshaug
University of Manchester
Oxford Road
Manchester M13 9PL
+44 (0)161-275-0271
joe.sakshaug@manchester.ac.uk

ABSTRACT

This paper explains a potential approach to synthetic data generation using genetic algorithms. It based on the principle that optimisation can be strong, accurate and efficient if there is sufficient prior knowledge of solution space. This approach is applicable because its evolutionary and competitive process compares different synthetic versions of the original data and combines their fitness in the finalised dataset.

Key words: Synthetic Data; Data Privacy; Genetic Algorithms; Modern Optimisation

CCS Concepts

Security and privacy→ Database and storage security→ Data anonymization and sanitization

Keywords

Synthetic Data; Data Privacy; Genetic Algorithms; Modern Optimisation;

1. INTRODUCTION

A key question in data privacy is whether anonymised versions of data can be created so that sharing or dissemination for re-use is possible. Orthodox approaches to anonymisation are largely based on statistical disclosure control. Residual risk is always present in disclosure controlled datasets as through linkage processes, statistical disclosure can still occur as units are re-identified.

Synthetic data generation is a strong method for controlling the risk of statistical disclosure. The goal is to preserve the analytical properties of the original data so that the utility of the original data is retained but privacy is protected. The methods in generating synthetic data are classified into full synthesis and partial synthesis. With partially synthetic dataset residual re-identification risk will still be present in the non-synthesised variables although this can be minimised through careful selection of the variables to be synthesised. In a fully synthetic dataset re-identification risk is effectively negligible even though all variables are retained. Privacy risks are therefore minimised and because of this, the potential of usable synthetic data is huge. For example, it would be possible to publish open data versions of sensitive datasets without concern for a confidential breach and this in turn would mean that much of the economic and social value of data currently hidden behind firewalls could be unlocked.

PrAISe '16, August 29 - 30, 2016, The Hague, Netherlands
ACM 978-1-4503-4304-6/16/08...$15.00
DOI: http://dx.doi.org/10.1145/2970030.2970034

The key problem is retaining sufficient analytical value in the synthetic version of the data. With orthodox methods of synthetic data production this entirely depends on the comprehensiveness of the data generation model used [12]. Properties that are not explicitly captured in the model used to generate synthetic data will not usually be retained in the dataset. Thus, unforeseen analyses might lead to results rather different to the same analysis on the original data [16]; this is the problem of analytical invalidity which does also beset orthodox disclosure controlled data [20]. So in essence, synthetic data has not – at present – fulfilled its undoubted potential as safe mechanism for releasing useful data because analysts do not trust its utility.

A key issue is that because most synthetic data methods are model driven; optimisation of solutions is essentially post hoc; an alternative approach is to make optimisation the core of the process and indeed Drechsler (2010) has proved that modern optimisation tools have some potential in synthetic data generation [11]. Although Mateo-Sanz et al (2004) commented that iterated data generation processes will prolong running time and increase method complexity [17], iteration can also optimise an object globally and its operation is – in principle - flexible and controllable.

Genetic algorithms are a form of optimization that use random processes to search a solution space. The unique combination of reproduction and crossover procedures means that their overall efficiency exceeds that of random search (Cortez, 2014) [6].

This paper will discuss the potential for genetic algorithms (GA) in synthetic data production. In short, we believe that GA has the potential to solve the synthetic data utility problem. With a well specified set of analytical properties extracted from the original data acting as a fitness function the GA would search the space of possible datasets for a good solution. *A priori* we know that there exists at least one solution - the original data - which will perfectly meet the criteria. There will also be a very large but finite set of possible datasets which are a close approximation to the original dataset. Our initial task is to specify a methodology for finding one of those datasets. Once we have achieved that then we can also add into our fitness function the additional constraint of protecting privacy (preventing re-identification and/or attribute disclosure). The impact of the utility constraints will be to attract the search towards the original dataset; the impact of the privacy constraint will be to repel the search away from the original data.

In the remainder of paper we talk trough the elements of the standard GA approach and demonstrate how they apply to the synthetic data generation use case.

2. A GENETIC ALGORITHM APPROACH

2.1 Preliminaries

Genetic Algorithms (GA) are bio-inspired optimisation methods. They start with an initial population[1] with N > 2 candidates. These candidates generate new generations by exchanging information with each other and/or mutating. Under a fitness test a "good" candidate will be more likely to bear offspring that bring its properties whereas a "bad" one will die off. The fitness tests (which are essentially a set of optimization criteria) mean that the pool of candidates will tend to converge on the optima with each generation. The success of an algorithm greatly depends on the quality of its fitness function. A general GA contains initialization and recombination in its process. It is usually made up of 5 steps: initial population selection, reproduction, crossover, mutation and replacement [18].

2.2 Initial Selection of Candidates

The initial selection of candidates does not need to preserve any properties from the original data and can be any same sized dataset filled by any data usually with the constraint that the metadata is respected. Although research suggests that diversity of the initial population is vital for efficiently reaching global optima, we will add to the computational workload by introducing randomly generated datasets in this problem. Based on Mateo-Sanz, Martínez-Ballesté and Domingo-Ferrer (2004) work on utility parameters for synthetic data, the initial population of candidate datasets could be composed of N datasets that preserve variable mean and the covariance matrix of original data [17]. However, this would increase the difficulty in setting up the candidates in the initial population (which would itself become an optimization problem) and so we will only preserve one or other of the conditions in any given candidate.

2.3 Fitness Functions

A good synthetic dataset will preserve inferences drawn from the original observed data. So, the fitness of an individual dataset should be tested against the inference parameters. The fitness function is formed by multiple constraints, which are referred to as *objectives* in GA. Multi-objective problems normally prevent simultaneous optimization. Generally we combine these objectives into a single function with empirically or theoretically derived weights. For example, we take variable means and covariance as two parameters to evaluate data utility. As proved by Drechsler (2010) [10], they sufficiently cover basic features of a dataset. The objectives of the process are expressed as:

(1) Minimise the sum of squared error of variable means.

(2) Minimise the sum of squared error of variable covariance.

The fitness values of candidates can be calculated as the scalar product of the two sums of squared error. However, selection of weights can be critical and small differences can lead to different solutions. Moreover, a single function will return a single solution thus the trade-off between objectives cannot be examined. The second way to deal with this problem is to compute a Pareto optimal solution set. This is a set of solutions that are non-dominated with respect of each other and there is always a certain amount of sacrifice in some objectives to achieve a desired level of other objectives while moving from one Pareto solution to another [14].

When using similarity between two datasets as the optimal strategy, the optimal solution will tend to be the original dataset; this implies that disclosure risks (which are also a de facto analytical property) will also re-emerge. Therefore, new objectives will be employed to minimise disclosure risk once the average fitness of the population reaches an acceptable level.

2.4 Reproduction

Reproduction, or selection, increases the likelihood of datasets passing on good properties to the next generation. Selection schemes are described based on the fitness distribution of the population before and after selection, thus only two factors are involved in this step: fitness values and candidates [4]. The fitness of each candidate dataset is evaluated by user-determined fitness functions, as described in the previous section. Higher-scoring datasets will (be more likely to) mate with others and generate offspring datasets, whereas lower-scoring ones are eliminated. Parents are selected with replacement so that one candidate can mate more than once [18]. Selection methods should be designed for particular problems but they all follow the principle that a better-fitted individual can be assigned a higher reproduction probability. The selection method can also kill less fit candidates to shrink population size and improve average performance of this generation and guarantees that no "bad" datasets pass to the next generation [3, 19]. Following Blickle and Thiele (1995), a selection method is a function to change the way that candidates allocated to each fitness value, also known as fitness distribution. The change between two fitness distributions over the population can be measured by the difference between average fitness in every generation, also known as selection pressure. High selection pressure is ideal. However, it is difficult so say which selection method provide the greatest pressure best because their selection pressure partly depends on how the methods themselves are operationalised [4]. For example, the selection pressure in tournament selection depends on the sample size drawn from the population. In truncation selection it depends on the proportion of best candidates in the population T. In the very beginning of the process we can use truncation selection because we have a dataset population generated from different conditions and the method gives the fraction T best candidates the same selection probability.

The reproduction rate denotes the ratio of the number of individuals with a certain fitness value before and after selection. It helps control the size of gene pool. A selection method should preserve datasets with the best fitness by assigning them a higher reproduction rate. According to Blickle and Thiele (1995), the reproduction rates should be greater than 1 for good candidates but less than 1 for bad candidates [4]. However, there is no clear boundary between good and bad individuals, so an adaptive reproduction rate will be preferable for this problem. For example, reproduction rates in one generation might be exponential or proportional to the ranks of fitness of individuals in the generation. Similarly to selection pressure, the reproduction rate also depends on the embedded features of selection methods so we cannot say if there is a best method. However, since the

[1] The term "population" has a meaning within the dataset milieu that refers to the set of entities that the data represents (or is drawn from). So that a reader fram that community is clear, here we mean the set of all datasets currently in the pool of candidates to reproduce.

process is dynamic, we could switch to different selection methods anytime to control the balance between utility and disclosure parameters.

2.5 Crossover

Selected datasets mate on randomly chosen positions with crossover probability p_c, normally drawn from a parameterised range [21]. Although since Goldberg (1989), GA has normally been processed using encoded strings and there are examples using real-coded matrices [19, 22] and this is the appropriate formulation for the synthetic data problem. Matrix GA can display more appropriate phenomena structures and has proved useful for multivariate adaptive control. It not only optimises the fitness of individual values but their relationship with surrounding locations. The fitness of candidate is evaluated by comparing all chromosome matrices to a reference matrix (also known as key matrix) obtaining fitness values from the degree of convergence [23]. Suppose a set of synthetic datasets Y_i is displayed in matrix format with entries $y_{i_{jk}}$. In the crossover step, a "rectangle" is randomly positioned in any two selected Y_i and their records would be exchanged:

$$\begin{pmatrix} y_{11} & y_{12} & y_{13} \\ y_{21} & y_{22} & y_{23} \\ y_{31} & y_{32} & y_{33} \end{pmatrix} \quad \begin{pmatrix} y_{211} & y_{212} & y_{213} \\ y_{221} & y_{222} & y_{023} \\ y_{231} & y_{232} & y_{233} \end{pmatrix}$$

Figure 1. Matrices and selected "rectangle" before crossover

$$\begin{pmatrix} y_{211} & y_{212} & y_{113} \\ y_{221} & y_{222} & y_{123} \\ y_{131} & y_{132} & y_{133} \end{pmatrix} \quad \begin{pmatrix} y_{111} & y_{112} & y_{213} \\ y_{121} & y_{122} & y_{023} \\ y_{231} & y_{232} & y_{233} \end{pmatrix}$$

Figure 2. Matrices and selected "rectangle" after crossover

A high crossover probability can result in population explosion in the pool whereas a low probability may lose the chance to pass good properties to the new generation. Considerations of computational capacity, suggest using small initial population to avoid the growth of population outpacing the increase of average fitness. Srinivas and Patnaik (1994) indicated that crossover probability p_c could be adaptive. In fact, the value of p_c should be negatively related to the level of convergence. The convergence of the current population can be observed from the difference between the maximum fitness value $f_{\max}$ and the average fitness value of current population $\bar{f}$. Therefore,

$$p_c = k_c/(f_{\max} - \bar{f})$$

where the existence of $k_c \leq 1.0$ constrains the value of p_c to the range of [0.0, 1.0] [22].

2.6 Mutation

Mutation ensures that the database is not developed from a uniform population. However, it does not always advance the search optimally. Alba and Marsili Libelli (2000) suggest that the mutation rate should be adaptive and inversely proportional to the individual's fitness value [2]. Srinivas and Patnaik (1994) claimed that mutation probability should be inversely proportional to the level of convergence of current population just like the crossover case.

$$p_m = \frac{k_m}{f_{\max} - \bar{f}},\ k_m \leq 1.0$$

The use of mutation reduces the likelihood of the pool of candidate solutions becoming stuck in local optima, especially when we intend to use a small population at the beginning [22]. One approach that we will explore is to implement mutation for only part of the GA process.

2.7 Replacement

Traditionally, new datasets after crossover will replace old datasets and from the new generation. In order to preserve the good performance, Wei (2003) proposed preserving the whole of previous generations in the gene pool. Allowing parents to compete does reduce the risk of cycling around within local optima but also increases running time [24]. Therefore, we will explore a variety of mechanisms for including the most fit parents from the previous generations.

3. SUMMARY

In this paper we have outlined a new approach to synthetic data generation based on genetic algorithms. Implementing GA for synthetic data generation involves random searching in a solution space of possible datasets based on optimisation criteria (which are properties of the original dataset) expressed as a fitness function. This randomness breaks with the traditional approach to fully synthetic data based on predictive distributions, and especially those using the multiple imputation approach. Generating synthetic data through GA has potential value because its evolutionary and competitive process compares different synthetic versions and combines their fitness in the finalised dataset. This means that we can explore different fitness function permutations to establish which are sufficient for a full set of analytical properties to emerge. This will move us away from the core problem of the model based approach that it arbitrarily constrains the solution.

In order to have a fast-converging GA approach, datasets in the initial population are designed to share at least one property with the original dataset. These properties could include (but are not limited to): mean vectors, covariance matrices, univariate distributions and equivalence class structures. A selection of these properties will form the initial suite of fitness indicators that will be combined into a single fitness measure. Assessing the appropriate mix of indicators and their weights will be a significant element of the research programme.

Once we have established whether it is possible to produce high utility synthetic datasets using this approach we will then consider the issue of residual disclosure risk. Although, as we have asserted, the risk of re-identification is negligible their remains the possibility that an unconstrained synthetic data generator may reproduce the original dataset or something very close to it and therefore we will need to consider the disclosure risk as a competing constraint.

One advantage in GA is real-time parameter control. Thus our process will be flexible. We will add in parameters to the fitness function in later stages of the process to balance information utility and disclosure risks. We set up crossover probability and mutation probability changes by the state of convergence of the current population to avoid pool explosion or pre-converging to local optima.

4. ACKNOWLEDGMENTS

Our thanks to ACM SIGCHI for allowing us to modify templates they had developed.

5. REFERENCES

[1] Abowd, J. M., and Lane, J. 2004. New approaches to confidentiality protection: Synthetic data, remote access and research data centres. In J. Doming-Ferrer, and V. Torra (eds), *Privacy in statistical databases*, Berlin Heidelberg: Springer. 282-289.

[2] Alba, P., and Marsili Libelli, S. 2000. Adaptive Mutation in Genetic algorithms. *Soft computing*, 4, 2, 76-78.

[3] Alabsi, F., and Naoum, R. 2012. Comparison of Selection Methods and Crossover Operations using Steady State Genetic Based Intrusion Detection System. *Journal of Emerging Trends in Computing and Information Sciences*, 3, 7, 1053-1058.

[4] Blickle, T., and Thiele, L. 1995. A Comparison of Selection Schemes used in Genetic Algorithms, TIK Report 11, 2 (December, 1995). DOI=ftp://129.132.2.212/pub/publications/TIK-Report11.pdf

[5] Bolboaca, S. D., Jantschi, L., Sestras, A. F., Sestras, R. E., and Pamfil, D. C. 2011. Pearson-Fisher Chi-Square Statistic Revisited. *Information*, 2, 3. 528-545.

[6] Chang, W. A., and Ramakrishna, R. S. 2003. Elitism-based compact genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 7, 4. 367-386.

[7] Chudasama, C., Shah, S.M., and Panchal, M. 2011 Comparison of Parents Selection Methods of Genetic Algorithm for TSP. In *International Conference on Computer Communication and Networks CSI-COMNET-2011*. 85-87.

[8] Cortez, P. 2014. *Modern Optimization with R.* New York: Springer. v.

[9] Dillard, B. L., and Gibson, H. R. 2016. *Elementary Statistics; 4th Edition.* Dubuque, IA: Kendall Hunt Publishing Company.

[10] Domingo-Ferrer, J., and Rebollo-Monedero, D. 2009. Measuring risk and utility of anonymized data using information theory. In *Proceedings of the 2009 EDBT/ICDT Workshops*, ACM. 126-130.

[11] Drechsler, J. 2010. Using support vector machines for generating synthetic datasets. *Privacy in Statistical Databases*, Berlin Heidelberg: Springer: 148-161.

[12] Drechsler, J. 2014. Synthetic data, where do we come from? Where do we want to go? Presentation at Synthetic Data Workshop, Office of National Statistics (Titchfield, UK, December, 2014).

[13] Duncan, G., Elliot, M. and Salazar-Gonzalez, J. 2011. *Statistical Confidentiality: Principles and Practice*. New York: Springer.

[14] Konaka, A., Coitb, D., and Smith, A. 2006. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering and System Safety*, 91, 9. 992-1007.

[15] Little, R. 1993. Statistical Analysis of Masked Data; *Journal of Official Statistics*, 9, 2. 407-426.

[16] Torra, V. 2010. Privacy in Data Mining. In O. Maimon and L. Rokach (eds). *Data Mining and Knowledge Discovery Handbook*. New York: Springer Science & Business Media, 687-716.

[17] Mateo-Sanz, J. M., Martınez-Balleste, A., and Domingo-Ferrer, J. 2004. Fast generation of accurate synthetic microdata. In J. Domingo-Ferrer and V. Torra (eds.) *Privacy in Statistical Databases*. Berlin Heidelberg: Springer. 298-306.

[18] Mitchell, M. 1998. *An Introduction to Genetic Algorithms.* Cambridge MA: MIT Press.

[19] Miller, B., and Goldberg, D. 1995. Genetic Algorithms, Tournament Selection, and the Effects of Noise, *Complex Systems*, 9, 3. 193-212.

[20] Purdam, K., and Elliot, M. 2007. A case study of the impact of statistical disclosure control on data quality in the individual UK Samples of Anonymised Records. *Environment and Planning A*, 39, 5. 1101-1118.

[21] Sun, L., Zhang, Y., and Jiang, C. A. 2006. Matrix real-coded genetic algorithm to the unit commitment problem. *Electric Power Systems Research*, 76, 9. 716-728.

[22] Srinivas, M., and Patnaik, L. M. 1994. Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms. *IEEE Transactions on systems, Management and Cybernetics*, 24, 4. 656-668.

[23] Wallet, B. C., Marchette, D. J., and Solka, J. L. 1996. A matrix Representation for Genetic Algorithms. In *Proceedings of Automatic Object Recognition IV of SPIE Aerosense*; Naval Surface Warfare Center Dahlgren (Virginia. May, 1996). 206-214.

[24] Wei, G. 2003. An Improved Fast-convergent Genetic Algorithm. In *Proceedings of 2003 IEEE International Conference on, Robotics, Intelligent Systems and Signal Processing*, 2: 1197-1202.

Glossary

- **Adaptive GAs** are GAs those parameters (crossover rates, mutation rates and population sizes) depend on the current stage of the process.
- **Attribution or attribute disclosure** refers to disclosure of target variable values from one or more equivalence classes.
- **Background knowledge attack** occurs when intruders build up a connection between published quasi-identifiers and auxiliary information that may come from their knowledge or external data.
- **Building block hypothesis** GAs optimise a population through the juxtaposition of short, low-order, high-performance building blocks (schema).
- **Candidates** refer to an individual in the searching space of GAs.
- **CPC** stands for 'case-oriented parallelised crossover' that has parallel crossover operations on every case (row) between paired candidates.
- **Crossover** allows candidates in the pool have a randomly chosen part of its information swapped with its paired candidate under a pre-determined crossover rate p_c.
- **Equivalence class** (in a dataset) is a set of records that have same values for a given set of variables (normally the quasi-identifiers).
- **Fitness distribution** is the function $s : \mathbb{R} -> \mathbb{Z}_0^+$ that maps a fitness value $f \in \mathbb{R}$ to the number of candidates in the population that has the same fitness value.
- **Full synthesis/Fully synthetic data** is artificial data that contains no actual information of respondents.

- **Homogeneity attacks** occurs when an equivalence class in a dataset does not have enough diversity therefore intruders can find values of sensitive variables in the class.
- **Loss of diversity** is the proportion of candidates in the population that is not selected by a selection operator.
- **MOOPs** stand for multi-objective optimisation problems.
- **Microdata** is a dataset that can represent in a matrix format in which columns are variables and rows are cases.
- **MRCGAs** are matrix real-coded GAs.
- **Mutation** enables a newborn candidate to change a piece of its information under a certain mutation rate.
- **Partial synthesis/Partially synthetic data** synthesises only selected records, for example quasi-identifiers, in the original data.
- **Pool** is where selected candidates stored and where the later process (crossover and mutation) occurs during the process of GAs.
- **Positional bias** occur when the disruption and recombination of a schemata depend on its position in the candidate.
- **Population** (in GAs) refer to all candidates in the current generation.
- **Population unique** is a population unit that has unique values on a set of quasi-identifiers within that population.
- **PUC** stands for 'parametric uniform crossover', namely the crossover occurs on each cell of the dataset in a pre-determined probability p_0.
- **Quasi-identifiers** are not themselves uniquely identify respondents but may create a unique identifier when combined with other quasi-identifiers.
- **Ranking selection** is a selection method in GAs in where all candidates are first rearranged in ascending order according to their fitness values and their chance to be selected is either *linearly* or *exponentially* proportional to the ranks.
- **Replacement** occurs after crossover and mutation, where the old population is replaced by new candidates.

- **Reproduction rate** calculates the ratio of the number of candidates with a certain fitness value before and after selection.
- **Re-identification or identification disclosure** occurs when attackers can identify a natural person within released data.
- **Roulette wheel selection** is a selection method in GAs that spreads the chance of selection to every candidate according to its fitness value. It simulates a roulette wheel with one pointer and plate that the area of each candidate is given by dividing its fitness by the total fitness.
- **Round-CPC** uses the same method as CPC to select the two endpoints from a case, then it decides to swap the elements between or out of the two endpoints by equal probability, which gives equal chance to the elements at the margin or centre of each row (case).
- **Schema** are the formal notions of building blocks that are discovered, emphasised and recombined during the process of GAs.
- **SDC** is the orthodox technology for controlling the balance between data utility and disclosure risks.
- **Selection** is an operator in GAs that selects candidates to enter crossover and mutation operators.
- **Selection pressure** measures to what extent that a better candidate is selected in a selection operator.
- **Selection variance** determines the change of variance in the population after selection, which is usually used for comparing methods together with selection pressure.
- **Stochastic universal selection (SUS)** is a selection method in GAs that modifies the roulette wheel selection by spinning the roulette wheel once with N evenly distributed pointers.
- **Targets or target variables** contain sensitive information of respondents and thus are the *raison detre* for privacy protection.
- **Tournament selection** is a selection method in where candidates are randomly selected with replacement and compete in a tournament with other t candidates and only the winner will be selected.
- **Whole-CPC** is a variant of CPC that exchanges the entire case between paired candidates.

Reference

[1] Abdi, H., and Valentin, D., (2007) Multiple Correspondence Analysis, Neil Salkind (Ed.), *Encyclopedia of Measurement and Statistics*, The University of Texas at Dallas, Richardson, TX 750830688, USA

[2] Abowd, J. M., and Lane, J. (2004). New approaches to confidentiality protection: Synthetic data, remote access and research data centres, In *Privacy in statistical databases*, Springer, Berlin Heidelberg, 282-289

[3] Alabsi, F. and Naoum, R. (2012) Comparison of Selection Methods and Crossover Operations using Steady State Genetic Based Intrusion Detection System, *Journal of Emerging Trends in Computing and Information Sciences*, 3(7), 1053-1058.

[4] Alpaydin, E. (2010) *Introduction to Machine Learning*, 2nd edition, The MIT Press, 1-9

[5] Andersen, E. B. (1990) *The Statistical Analysis of Categorical Data*, Springer-Verlag, Berlin

[6] Antal, L., Shlomo, N., and Elliot, M. (2014). Measuring disclosure risk with entropy in population based frequency tables. In International Conference on Privacy in Statistical Databases (pp. 62-78). Springer, Cham.

[7] Amjady, N., and Shirzadi, A., (2009), Unit commitment using a new integer coded genetic algorithm. Euro. Trans. Electr. Power, 19: 1161-1176. doi:10.1002/etep.297

[8] Barak, B., Chaudhurim, K., et al. (2007) Privacy, Accuracy, and Consistency Too: A Holistic Solution to Contingency Table Release, *PODS*'07, Beijing, China

[9] Barto, G., Sutton, S., Watkins, H. (1990), Learning and sequential decision making, *Learning and Computational Neroscience*, Gabriel, M., Moore, W., (Eds.), MIT press, Cambridge, MA

[10] Baum, E. B., Boneh, D., and Garrett, C., (2001), Where Genetic Algorithms Excel, *Evolutionary Computation*, vol:9(1), 93-124

[11] Benschop, T., Machingauta, C., and Welch, M., (2018), Statistical Disclosure Control: A Practice Guide, Avaliable at https://buildmedia.readthedocs.org/media/pdf/sdcpractice/latest/sdcpractice.pdf, [10/05/2019]

[12] Bingul, Z., (2007) Adaptive Genetic Algorithms Applied to Dynamic Multi-objective Problems, *Applied Soft Computing*, 7, 891-799

[13] Blickle, T. and Thiele, L. (1995) A Comparison of Selection Schemes used in Genetic Algorithms, *Computer Engineering and Communication Networks Lab*, Swiss Federal Institute of Technology.

[14] Brabazon, A., O'Neill, Michae.l, and McGarraghy, Sean., (2015). *Natural computing algorithms*, Springer

[15] Branke, J., (2012), Evolutionary optimisation in dynamic environments, Vol. 3. Springer Science & Business Media

[16] Burgees, M. (2019), What is GDPR? The summary guide to GDPR compliance in the UK, WIRED magazine, January, 21, 2019, Available at https://www.wired.co.uk/article/what-is-gdpr-uk-eu-legislation-compliance-summary-fines-2018 [09/04/2019]

[17] Caiola, G., and Reiter, J. P., (2010). Random Forests for Generating Partially Synthetic, Categorical Data. *Trans. Data Privacy* 3, 1 (April 2010), 27-42.

[18] Cano, I., Ladra, S., and Torra, V., (2010), Evaluation of Information Loss for Privacy Preserving Data Mining through comparison of Fuzzy Partitions, *Proceedings of FUZZ-IEEE 2010/WCCI.*

[19] Cantù-Paz, E., and Goldberg, D., (2000), Efficient Parallel Genetic Algorithms: Theory and Practice, *Computer Methods in Applied Mechanics and Engineering*, vol.186, 221-238

[20] Chen, Y., Elliot, M., and Sakshaug, J., (2016), A Genetic Algorithm Approach to Synthetic Data Production, in Proceedings of the 1st International Workshop on AI for Privacy and Se-curity. Article No. 13.

[21] Chen, Y., Elliot, M., and Sakshaug, J., (2017), Genetic Algorithms in Matrix Representation and Its Application in Synthetic Data, *UNECE Work Session on Statistical Data Confidentiality*, 2017, https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.46/2017/2_Genetic_algorithms.pdf, [20/12/2017].

[22] Chen, Y., Elliot, M., and Smith, D., (2018), The Application of Genetic Algorithms to Data Synthesis: A Comparison of Three Crossover Methods, *Privacy in Statistical Database*, 2018, Springer.

[23] Chen, Y., Taub, J., and Elliot, M, (2019) Trade-off between Information Utility and Disclosure Risk in GA Synthetic Data Generator, *UNECE Work Session on Statistical Data Confidentiality* 2019, https://statswiki.unece.org/display/confid/Work+Session+on+Statistical+Data+Confidentiality+2019, [01/12/2019].

[24] Chertov, O., and Pilipyuk, A., (2009), Statistical Disclosure Control Methods for Microdata, *2009 International Symposium on Computing, Communication, and Control*, Proceeding of CSIT, vol.1, (2011), IACSIT Press, Singapore

[25] Chiong, R., Weise, T., and Michalewicz, Z., (Editors), (2012), *Variants of Evolutionary Algorithms for Real-World Applications*, Springer, 2012, ISBN 3642234232

[26] Chudasama, C., Shah, S.M. and Panchal, M. (2011) Comparison of Parents Selection Methods of Genetic Algorithm for TSP, International Conference on Computer Communication and Networks CSI- COMNET-2011

[27] Ciriani, V., di Vimercati, S.D.C., Foresti, S., Samarati, P., (2007) Microdata protection. In: Yu T., Jajodia S. (eds.) Secure Data Management in Decentralized Systems, Springer, New York, 291321

[28] Connor, R., Cardillo, F., A., Moss, R., and Fausto, R., (2013) Evaluation of Jensen-Shannon distance over sparse data. In: Similarity Search and Applications. Lecture Notes in Computer Science, 8199. Springer, Berlin, pp. 163-168. ISBN 9783642410611, http://dx.doi.org/10.1007/978-3-642-41062-8_16, [01/03/2017]

[29] Cortez, P. (2014) *Modern optimisation with R*, Springer, v

[30] Dandekar, R. A., Cohen, M., and Kirkendall, N., (2001), Applicability of Latin Hypercube Sampling to create multi variate synthetic micro data, ETK-NTTS 2001 Pre-proceedings of the Conference, 839–847.

[31] Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T., (2002), A fast and elitist multiobjective genetic algorithm: NSGA-II, I*EEE Transactions on Evolutionary Computation*, vol. 6, no. 2, 182-197

[32] Deb. K., (2014) Multi-objective optimisation, *Search Methodologies: Introductory Tutorials 403 in optimisation and Decision Support Techniques*, Burke, E. K., and Kendall, G., (eds.), Springer New York, Boston, MA, 2014, 403-449

[33] Domingo-Ferrer, J. and Rebollo-Monedero, D. (2009). Measuring risk and utility of anonymized data using information theory. *Proceedings of the 2009 EDBT/ICDT Workshops*, ACM, 126-130

[34] Domingo-Ferrer, J. Mateo-Sanz, J. M. and Torra. V. (2001): Comparing SDC methods for microdata on the basis of information loss and disclosure risk, *Pre-proceedings of ETK-NTTS2001* (vol. 2):, 807826, Luxemburg, Eurostat.

[35] Domingo-Ferrer, J., and Torra, V, (2001) Disclosure control methods and information loss for microdata, Confidentiality, disclosure, and data access, *Theory and practical applications for statistical agencies*, 91-110

[36] Domingo-Ferrer, J., (2014), Data Anonymisation: a tutorial, Universitat Rovira i Virgili, Tarragona, Catalonia.

[37] Drechsler, J., Bender, S., and Rassler, S. (2008) Comparing fully and partially synthetic datasets for statistical disclosure control in the German IAB Establishment Panel, *Transactions in Data Privacy*, 1, 105-130.

[38] Drechsler, J. (2010). Using support vector machines for generating synthetic datasets, *Privacy in Statistical Databases*, Springer Berlin Heidelberg, 148-161.

[39] Drechsler, J. (2014), Synthetic data, where do we come from? Where do we want to go?, Synthetic Data Workshop, Office of National Statistics

[40] Drechsler, J. (2018) Some Clarifications Regarding Fully Synthetic Data, *Privacy in Statistical Databases*, Springer, Spain, 2018.

[41] Dumitrescu, D., Lazzerini, B., Jain, L. C., and Dumitrescu, A., (2000), *Evolutionary Computation*, CRC Press, N. W. corporate Blvd, Boca Raton, Florida, 33431, 218-225

[42] Duncan, G. T., Keller-McNulty, S, A., and Stokes, S. L., (2001), Disclosure Risk vs. Data Utility: The R-U Confidentiality Map, National Institute of Statistical Sciences, Report Number 121.

[43] Duncan, G. T., and Stokes, S. L., (2004), Disclosure Risk vs. Data Utility: The R-U Confidentiality Map as Applied to Topcoding, *CHANCE* 17:3, 16-20

[44] Duncan, G. T., Elliot, M. and Salazar-Gonzalez, J. (2011) *Statistical Confidentiality: Principles and Practice*, Springer New York, 23

[45] Duncan, G. T., Elliot, M. and Salazar-Gonzalez, J. (2011) *Statistical Confidentiality: Principles and Practice*, Springer New York, 42

[46] Dwork, C., and Roth, A., 2014, The Algorithmic Foundations of Differential Privacy, *Foundations and Trends in Theoretical Computer Science*, vol.9 3-4.

[47] Elliot, M. (2014). Final Report on the Disclosure Risk Associated with the Synthetic Data Produced by the SYLLS Team. [online] CMIST. Available at: https://tinyurl.com/syllsDR, [12/9/2016]

[48] Elliot, M., and Domingo-Ferrer, J., (2018), The future of statistical disclosure control, Paper published as part of The National Statistician's Quality Review. London, December 2018, Available at: https://arxiv.org/pdf/1812.09204.pdf, [30/11/2019]

[49] Ethem, A., (2014), *Introduction to Machine Learning*, The MIT Press

[50] Evfimievski, A., Gehrke, J., and Srikant, R., (2003) Limiting privacy breaches in privacy preserving data mining, *PODS*, 2003.

[51] Fellegi, I., Sunter, A. (1969).A Theory for Record Linkage.*Journal of the American Statistica Association*, 64(328):.11831210.

[52] Forrest, S. and Mitchell, M. (1993), Relative building-block fitness and the building- block hypothesis. In D. Whitley (ed.), *Foundations of Genetic Algorithms* 2, 109- 126. San Mateo, CA: Morgan Kaufmann.

[53] Galaviz J., and Xuri, A., (1996) A self-adaptive genetic algorithm for function optimisation, *Proceedings Mexico-USA Collaboration in Intelligent Systems Technologies*, Cancun, Mexico, 1996, 156-161.

[54] Garfinkel, S. L., Abowd, J.M., and Powazek, S., (2018), Issues Encountered Deploying Differential Privacy, *arXiv e-prints*, arXiv:1809.02201

[55] General Data Protection Regulation 2016/679 (GDPR), The EU, GDPR Key Changes, Avaliable at https://eugdpr.org/the-regulation/, [06/05/2019]

[56] Goldberg, D., Jon, R., (1987). Genetic algorithms with sharing for multimodal function optimisation. 2nd *Int'l Conf. on Genetic Algorithms and their applications*, 41-49

[57] Goldberg, D. E. (1989) *Genetic Algorithms in search, optimisation, and Machine Learning*, The University of Alabama.

[58] Gotshall, S. and Rylander, B., (2002), Optimal Population Size and the Genetic Algorithm, *Proceedings of the 2002 WSEAS International Conferences*, Mexico.

[59] Gouweleeuw, J., Kooiman, P., Willenborg, L., and De Wolf. P., (1998) , The Post Randomisation Method for Protecting Microdata, *QUETIIO*, 22(1), .145-156

[60] Herzog, T. N., Scheuren, F. J., Winkler, W. E., (2007), *Data Quality and Record Linkage Techniques*, Springer, Science+Business Media, LLC, New York.

[61] Henry, K. M., (2012). *Penetration Testing: Protecting Networks and Systems*. IT Governance Ltd. ISBN 978-1-849-28371-7.

[62] Holland, J. (1992). Genetic Algorithms, *Scientific American*, 267(1), 66-73. www.jstor.org/stable/24939139, [23/04/2020]

[63] Holland, J. H., (1992). *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA.

[64] Horn, J., Nafpliotis, N., and Goldberg, E., (1994), A Niched Pareto Genetic Algorithm for Multiobjective optimisation, *Proceedings of the First IEEE Conference on Evolutionary Computation*, 1994.

[65] Hu, J., Reiter, J. P. and Wang, Q. (2014) Disclosure Risk Evaluation for Fully Synthetic Categorical Data, Proceedings of Privacy in Statistical Database 2016, Domingo-Ferrer, J., and Montes, F., (Eds.), 185-199.

[66] Hu, J., and Hoshino, N. (2018). The Quasi-Multinomial Synthesizer for Categorical Data. Proceedings of Privacy in Statistical Database 2018, Domingo-Ferrer, J., and Montes, F., (Eds.), 7591

[67] Isaac, M., and Frenkel, S. (2018) Facebooks Woes Rise as Hackers Expose Data of 50 Million Users, The New York Times, Sept. 29, 2018. Available at https://www.nytimes.com/2018/09/28/technology/facebook-hack-data-breach.html [09/04/2019].

[68] Konak, A., Coit, D. W., and Smith, A. E., (2006), Multi-objective optimisation using genetic algorithms: A tutorial, *Reliability Engineering & System Safety*, 91(9), 992-1007, https://doi.org/10.1016/j.ress.2005.11.018., [01/07/2018]

[69] Korejo, I., Yang, S., and Li, C., (2009) A Comparative Study of Adaptive Mutation Operators for Genetic Algorithms, MIC 2009: The VIII Metaheuristics International Conference, Hamburg, Germany

[70] Leung, Y., Gao, Y., and Xu, Z., (1997), Degree of population diversitya perspective on premature convergence in genetic algorithms and its Markov chain analysis, *IEEE Transactions on Neural Networks*, vol. 8, no. 5, 11651176

[71] Li, N., Li, T., and Venkatasubramanian, S., (2007), t-closeness: Privacy beyond k-anonymity and l-diversity, in *ICDE*

[72] Libelli, S., and Alba, P., (2000) Adaptive mutation in genetic algorithms, *Soft computing*, vol.4 76-80.

[73] Little, R., (1993) Statistical Analysis of Masked Data, *Journal of Official Statistics*, 9, 407-426.

[74] Liu, F. and Little, R. (2003) SMILKe. Vs Data Swapping and PRAM for Statistical Disclosure Control in Microdata: a Simulated Study, 2003 Joint Statistical Meeting, San Francisco, California.

[75] Lu, H., and Yen, G., (2002), Rank-Density-Based Multiobjective Genetic Algorithm, Proceedings of the 2002 Congress on Evolutionary Computation 2002, 944-949

[76] Machanavajjhala, A., et al., (2007) l-diversity: Privacy beyond k-anonymity, *ACM Transactions on Knowledge Discovery from Data*, 1(1), 2007.

[77] Matthews, A., Hensman, J., et al., (2015), On Sparse Variational Methods and the Kullback-Leibler Divergence between Stochastic Processes, https://arxiv.org/abs/1504.07027, [01/06/2017]

[78] Maimon, O. and Rokach, L. (2010) *Data Mining and Knowledge Discovery Handbook*, Springer Science and Business Media, p. 704

[79] Matwin, S., Nin, J., Sehatkar, M. and Szapiro, T. (2015) A Review of Attribute Disclosure Control, *Advanced Research in Data Privacy, Studies in Computational Intelligence*. Vol. 567. Springer Switzerland, 41-62

[80] Mateo-Sanz, J. M., Martnez-Ballest, A., and Domingo-Ferrer, J. (2004), Fast generation of accurate synthetic microdata, *In Privacy in Statistical Databases*, Springer Berlin Heidelberg, 298-306

[81] Mateo-Sanz, J. M., Domingo-Ferrer, J., and Sebé, F., (2005) Probabilistic Information Loss Measures, *Confidentiality Protection of Continuous Microdata, Data Mining and Knowledge Discovery*, Vol.11, 181-193

[82] Mitchell, M., Holland, J. H., and Forrest, S., 1993. When will a genetic algorithm outperform hill climbing?. *In Proceedings of the 6th International Conference on Neural Information Processing Systems (NIPS'93)*, J. D. Cowan, G. Tesauro, and J. Alspector (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 51-58.

[83] Mitchell, M. (1998a) *An Introduction to Genetic Algorithms*, Massachusetts Institute of Technology, 7-23

[84] Mitchell, M. (1998b) *An Introduction to Genetic Algorithms*, Massachusetts Institute of Technology, 124-131

[85] Moriarty, D., Schultz, A., and Grefenstette, J., (1999), Evolutionary Algorithms for Reinforcement Learning, *Journal of Artificial Intelligence Research*, vol.11, 241-276

[86] Murata, T., Ishibuchi, H., (1995), MOGA: multi-objective genetic algorithms. Proceedings of the 1995 IEEE international conference on evolutionary computation 1995, Perth, WA, Australia: IEEE; 1995.

[87] Navarro-Arribas, G. and Torra, V. (2015a) Advanced Research on Data Privacy in the ARES Project, *Advanced Research in Data Privacy*. Vol. 567. Springer Switzerland, 3-14

[88] Navarro-Arribas, G. and Torra, V. (2015b), Data Privacy: A Survey of Results, Advanced Research in Data Privacy, *Studies in Computational Intelligence*. Vol. 567. Springer Switzerland, 27-37

[89] Nowok, B., Raab., M. G., and Dibben, C., (2016), synthpop: Bespoke Creation of Synthetic Data in R, Journal of Statistical Software, 74(11), DOI: 10.18637/jss.v074.i11

[90] Oganian, A., (2014) V-Dispersed Synthetic Data Based on a Mixture Model with Constraints, *Privacy in Statistical Databases*, Springer International Publishing, 200-212

[91] Olkin, B., I., and Tate, R., F., (1960), Multivariate Correlation Models with Mixed Discrete and Continuous Variables, The Annuals of Mathematical Statistics.

[92] Pavez-Lazo, B., and Soto-Cartes, Jessica., (2010), A deterministic annular crossover genetic algorithm optimisation for the unit commitment problem, *Expert Systems with Applications*, Volume 38, Issue 6, 2011, 6523-6529, https://doi.org/10.1016/j.eswa.2010.11.089., [23/04/2020]

[93] Pistner, M., Slavkovic, A. B., and Vilhuber, L. (2018). Synthetic data via quantile regression for heavy-tailed and heteroskedastic data. In F. Montes, and J. Domingo-Ferrer (Eds.), Privacy in Statistical Databases 2018, Proceedings, Springer, Verlag, 92-108, https://doi.org/10.1007/978-3-319-99771-1_7, [22/12/2018]

[94] Pongcharoen, P., Khadwilard, A., and Klakankhai, A., (2007) Multi-matrix Real-coded Genetic Algorithm for Minimising Total Costs in Logistics Chain Network, World Academy of Science, *Engineering and Technology International Journal of Economics and Management Engineering*, 1(11), 574-597

[95] Raghunathan, T.E., Reiter, J.P., and Rubin, D.B. (2003), Multiple Imputation for Statistical Disclosure Limitation, *Journal of Official Statistics*, 19

[96] Raab, G, M., Nowok, B., and Dibben, c., (2016), Practical data synthesis for large samples, *Journal of Privacy and Confidentiality (2016-2017)*, 7, no.3, 6797

[97] Reeves, C.R. (2010). *Genetic Algorithms, Handbook of metaheuristics*, Vol. 2, editors: Gendreau, M. and Potvin, J.Y., New York: Springer. 124-125

[98] Reiter, J. P., (2003), Inference for partially synthetic, public use microdata sets, *Survey Methodology* 29, 181189.

[99] Reiter, J.P., (2005) Using CART to generate partially synthetic, public use microdata, *Journal of Official Statistics*, 21, 441462

[100] Reiter, J. P. and Kinney, S. K. (2012) Inferentially Valid, Partially Synthetic Data: Generating from Posterior Predictive Distributions Not Necessary, *Journal of Official Statistics*, 28(4), 583-590

[101] Roque, L.A.C., Fontes, D.B.M.M. and Fontes, F.A.C.C. (2014), A hybrid biased random key genetic algorithm approach for the unit commitment problem. J Comb Optim 28, 140166, https://doi.org/10.1007/s10878-014-9710-8

[102] Rozenberg, G. (2012) *Handbook of Natural Computing*, Bäck,Thomas. Kok, Joost N. editor, SpringerLink.

[103] Rubin, B. D., (1993), Discussion Statistical Disclosure Limitation, *Journal of Official Statistics*, vol.9, no.2, 461-468

[104] Salimans, T., Ho, J., Chen, X., and Sutskever, I., (2017), Evolution Strategies as a Scalable Alternative to Reinforcement Learning, arXiv preprint, arXiv:1703.03864.

[105] Sarathy, R., and Muralidhar, K., (2011) Evaluating Laplace Noise Addition to Satisfy Differential Privacy for Numeric Data, *Transactions on Data Privacy*, 4(1), 1-17.

[106] Schaffer, J., D., (1985), Multiple Objective Optimisation with Vector Evaluated Genetic Algorithms, *Proceedings of the 1st International Conference on GAs*, 1985, 93-100

[107] Shlomo, N., (2010), Releasing Microdata: Disclosure Risk Estimation, Data Masking and Assessing Utility, *Journal of Privacy and Confidentiality*, 2(1), 73-91.

[108] Smith, D., and Elliot, M. (2008). A Measure of Disclosure Risk for Tables of Counts. *Trans. Data Privacy, 1*(1), 34-52.

[109] Snoke, J. , Raab, G. M., Nowok, B. , Dibben, C. and Slavkovic, A. (2018), General and specific utility measures for synthetic data. J. R. Stat. Soc. A, 181: 663-688. doi:10.1111/rssa.12358

[110] Si, Y. and Reiter, J. p. (2013) Nonparametric Bayesian Multiple Imputation for Incomplete Categorical Attributes in Large-Scale Assessment Surveys, *Journal of Educational and Behavioral Statistics*, 38(5), 499-521

[111] Spear, W. M., and De Jong, K. A., (1991), On the Virtues of Parameterized Uniform Crossover, *Proceedings of the Fourth International Conference on Genetic Algorithms*, 230-236

[112] Sun, L. Zhang, Y and Jiang, C (2006) A matrix real-coded genetic algorithm to the unit commitment problem, *Electric Power Systems Research*, 76, 716-728

[113] Surya, K., and Mahara, T., (2016), A New Similarity Measure Based on Mean Measure of Divergence for Collaborative Filtering in Sparse Environment, *12th International Multi-Conference on Information Processing-2016, Procedia Computer Science*, 89, 450 456

[114] Sutton, R., and Barto, A., (2012), *Reinforcement Learning: An Introduction*, 2nd edition, the MIT press

[115] Srinivas, M. and Patnaik, L. M. (1994a) Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms, *IEEE Transactions on systems, Man and Cybernetics*, 24(4), 656-668

[116] Srinivas, M., and Patnaik, L. M., (1994b), Genetic algorithms: A Survey, *Computer*, 27(6), 17-26,.

[117] Srinivas, M., Deb, K., (1994) Multiobjective optimisation using nondominated sorting in genetic algorithms. *J Evol Comput 1994*, 2(3), 22148.

[118] Sweeney, L., (2002) k-anonymity: a model for protecting privacy, *International Journal on Uncertainty, Fuzziness and Knowledge-based Systems*, 10(5), 557570

[119] Taub, J., Elliot, M., Pampaka, M., and Smith, D., (2018), Differential Correct Attribution Probability for Synthetic Data: An Exploration, *Privacy in Statistical Database*, 2018

[120] Taub, J., Elliot, M., and Sakshaug, J. (2017). The Impact of Synthetic Data Generation on Data Utility with Application to the 1991 UK Samples of Anonymised Records. In *TRANSACTIONS ON DATA PRIVACY*, 12, 123

[121] Thierens, D. (2002) Adaptive mutation rate control schemes in genetic algorithms, Evolutionary Computation, 2002. CEC '02. *Proceedings of the 2002 Congress on*, 1, 980 985

[122] Anonymisation, UK data service, viewed 21/03/2019, https://www.ukdataservice.ac.uk/manage-data/legal-ethical/anonymisation, [03/12/2016]

[123] Villalobos-Arias, Mario and Coello, Carlos A. Coello and Hernández-Lerma, (2005), Asymptotic Convergence of Some Metaheuristics Used for Multiobjective optimisation, *Proceedings of the 8th International Conference on Foundations of Genetic Algorithms*, 95-111

[124] Wallet, B. C., Marchette, D. J. and Solka, J. L., (1996), A matrix Representation for Genetic Algorithms, Proceedings of Automatic Object Recognition IV of SPIE Aerosense, Naval Surface Warfare Center Dahlgren, Virginia.

[125] Wei, G. (2003) An Improved Fast-convergent Genetic Algorithm, Intemational Conference on Robotics, *Intelligent Systems and Signal Processing Changsha*, China

[126] Zezula, P., Amato, G., Dohnal, V., and Batko, M., (2006), Similarity Search. The Metric Space Approach. USA: Springer, 2006.

[127] Zhang, Q., and Li, H., (2008). MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *Evolutionary Computation, IEEE Transactions*, 11. 712 - 731

[128] Zhu, X., Gao,Z., Du, Y., Cheng, S., and Xu, F., (2018) A decomposition-based multi-objective optimisation approach considering multiple preferences with robust performance, *Applied Soft Computing*, Volume 73, 2018, 263-282, DOI:https://doi.org/10.1016/j.asoc.2018.08.029.

[129] Ziztler, E., and Künzil, S., (2004), Indicator-Based Selection in Multiobjective Search, *Proceedings of the 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, September 2004, Birmingham, UK, Available at https://www.simonkuenzli.ch/docs/ZK04.pdf [31/03/2020]