Automatic

Control &

Systems

Engineering.

Generating distinct graphs while maintaining similar statistics using machine learning

Osinowo Michael

December 2019

Supervisor: Roderic Gross

A dissertation submitted in partial fulfilment of the requirements for the degree of BEng

ABSTRACT

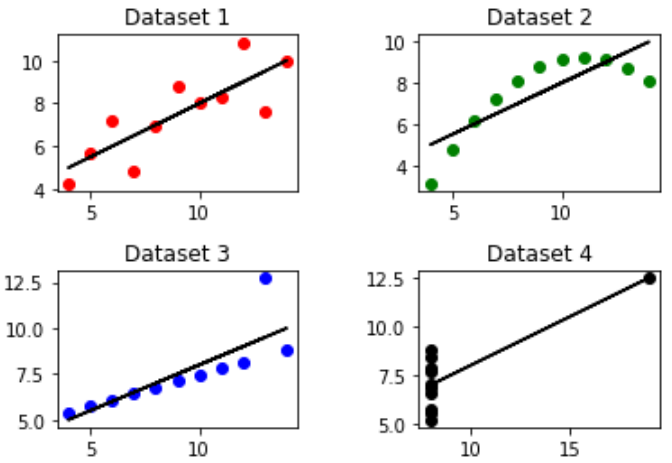
Significant amounts of effort have been put into statistics and data visualisations, this project takes a dive into a case of statistical and graphical anomaly that wasn't paid so much attention to in the past, until Anscombe discovered the occurrence of this anomaly and went ahead to describe it with four datasets (i.e. Anscombe's quartet). This project will aim to study if datasets other than Anscombe's quartet can be generated with an algorithm, with those datasets having the same summary statistics as Anscombe's quartet.

TABLE OF CONTENT

Chapter 1 - Introduction

1.1. Background and Motivation

Statistics typically is used to describe large datasets which are difficult to analyse by merely looking at the graphical representation but there are cases where numbers alone seem to be ineffective. Anscombe's quartet is a collection of four two-dimensional datasets which have similar statistic summary (i.e., mean, standard deviation and correlation) but have graphical representations that are distinct, this occurrence clearly shows the importance of data visualizations. Anscombe's quartet shows why summary statistics are not completely reliable. Figure 1 shows the scatter plot of the four datasets and their summary statistics, it is not well known how the original Anscombe's quartet was generated or what algorithms were used to produce these datasets as Anscombe did not address this issue in [1].




```
------DATASET 1-------
N:  44
X mean:  9.0
X SD:  3.198836997962679
Y mean:  7.50068181818182
Y SD:  1.9589251890784192
Pearson correlation:  0.8163662427614701
------DATASET 2-------
N:  44
X mean:  9.0
X SD:  3.198836997962679
Y mean:  7.50068181818182
Y SD:  1.9589251890784192
Pearson correlation:  0.8163662427614701
------DATASET 3-------
N:  44
X mean:  9.0
X SD:  3.198836997962679
Y mean:  7.50068181818182
Y SD:  1.9589251890784192
Pearson correlation:  0.8163662427614701
------DATASET 4-------
N:  44
X mean:  9.0
X SD:  3.198836997962679
Y mean:  7.50068181818182
Y SD:  1.9589251890784192
Pearson correlation:  0.8163662427614701
```

Figure 1: showing the four datasets and their scatter plots adapted from [1]

1.2. Aims and Objectives

The aim of this project is to employ the concept from Anscombe's quartet to design an algorithm which incorporates machine learning methods to generate more unique datasets from any reference dataset with similar statistic summary as the reference dataset. This algorithm, if designed properly will be able to generate counterfeit data which have the same summary statistics and the original. To fulfil the aims of this project, a set of objectives have to be achieved:

- Literature review on Anscombe's quartet, data visualisation and machine learning techniques relevant to the problem.
- Design an algorithm that generates a unique dataset with the nearly the same statistical summary as the Anscombe's quartet.
- Determine the accuracy of the generated datasets in comparison to the original dataset by comparing the statistical summary.
- Visualize generated datasets and original datasets.
- Analyse generated datasets.
- Analyse the algorithm designed.

The aim of the advanced set of objectives for this project will be to generalise the algorithm so it can perform the same task on larger and higher dimensional datasets. The efficiency and practicality of the algorithm will be analysed for the larger datasets. The following are the advanced objectives:

- Redesign the algorithm to work on datasets larger than 100 data points and multi-dimensional
- Analyse and visualise the generated datasets.
- Analyse the efficiency of the algorithm.

Chapter 2 - Literature Review

2.1. Origin of Anscombe's quartet

The anomaly of having datasets with similar summary statistics and distinct graphs was first introduced by Anscombe in 1973 with a paper titled "Graphs in Statistical Analysis" [1]. Anscombe begins the paper by stating three misconceptions about graphs at the time. The paper proves how most kinds of statistical analysis computations are based on assumptions of the datasets which can be misleading. Graphs help to expose these assumptions and gives the opportunity to observe broader features of the data.

Anscombe proved this by going into detail on a statistical analysis technique called regression analysis, focusing on a specific case where a scatter plot of a dataset is created and "*most of the points (x,y) lie close to a line or smooth curve, but a few are scattered a long way away"*. In Figure 1 the straight line in all four graphs show the general trend of those datasets generated by regression analysis, but the only dataset that seems to fit the general trend is dataset I, datasets II and IV have completely different trends to the first one. Dataset III shows that outliers can't be spotted using regression analysis.

Although Anscombe proves his point with four distinct datasets, he never went into detail on how these datasets were generated or how to create more like them.

2.2. Methods used to generate more unique datasets

There have been various attempts made to solving the problems of this nature. In this section two significant approaches will be discussed in detail as they both use different algorithms to arrive at the same solution, but one proves to be better than the other.

2.2.1. Genetic Algorithm Approach

There have been various methods used to generate unique datasets using the Anscombe's quartet. The earliest attempt was in 2007 by Chatterjee, S. and Firat, A. [2]. In this paper the method used was a genetic algorithm approach, where each gene in the genetic algorithm represents a randomly initialized dataset with same size as the reference dataset. A population of 1000 random genes were created and iteratively processed until the conversion criteria were met or there was no longer any improvement in the objective function. This algorithm ran for about 2500 generations, wherein the final generation, the genes with the highest fitness represented the solution to the problem. To prevent the algorithm from reproducing the original graph, the following statistical tests were used to measure the dissimilarity of the graphs and some of these tests were combined to produce more optimal results:

1. Ordered data values
2. Kolmogorov–Smirnov test, and
3. Quadratic coefficients of the regression fit,
4. Breusch-Pagan Lagrange multiplier,
5. Standardized skewness
6. Standardized kurtosis,
7. Maximum of the Cook's D statistic

Although the genetic algorithm approach managed to generate graphs with similar statistics, some of the graphs generated were not satisfyingly distinct. The technique used in this project aims to generate distinct graphs using machine learning methods.

2.2.2. Simulated Annealing Approach

A recent approach to generating unique datasets was by using the simulated annealing optimisation technique, which was done by a research team at Autodesk in 2017[3]. This paper shows how the simulated annealing optimization successfully generated specified datasets from a random dataset of size 182 with similar summary statistics with an accuracy of about 95%. Two significant test cases were carried out, the first test case was to coerce the random dataset to fit into target shapes(datasets whose scatter plots form defined shapes) while maintaining the summary statistics and the second test case was too coarse the initial

dataset which was defined to fit into the same target shapes. Both tests were successful and were completed in about 200, 000 iterations. Figure 2.2.1 shows experiment recreated for the two test cases.

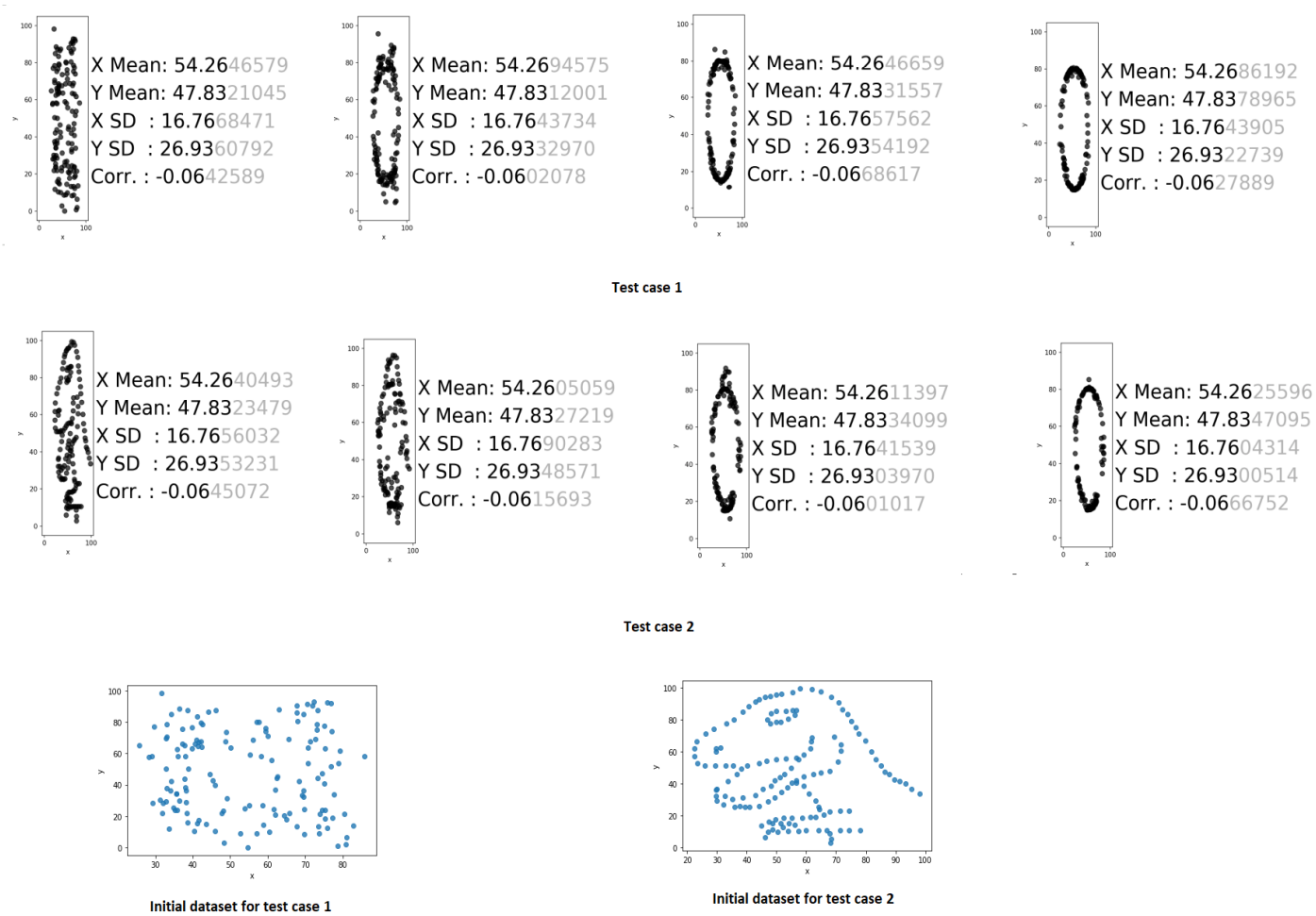


Figure 2.2.1 shows the recreated experiment from [3].

The simulated annealing optimization arrives at an optimal solution to difficult tasks based on the exploration and exploitation technique, where you have two parameters, the start and end temperature and every iteration represents a cooling step. For higher temperatures, the more likely the program takes the risk to accept less optimal solutions and otherwise for lower temperatures. In the case of coercing the dataset to fit a specific shape, the program starts with a random two-dimensional dataset and perturbs each data point by a random small amount and repeats the process iteratively while maintaining similar statistic summary. At the end of every iteration, the fitness of the new dataset is calculated as the average distance between all data points on the current perturbed dataset and the target shape. At the early stages of the iterations, the

program can accept less optimal solutions more frequently than later stages of the iteration where the algorithm mostly exploits the optimal solution.

The benefits of the optimisation technique used was that the solution never got stuck in a local optimal solution where the datasets looks quite similar to the initial dataset. Based on the working principle of the simulated annealing technique it is always bound to find the globally optimal solution.

2.3. Research on Turing Learning

The term “Turing Learning” is derived from the famous Turing test developed by Alan Turing in 1950. The aim of the test was for a human to interact with a machine and another human without the human spotting the difference. Turing learning according to Wei Li, Melvin Gauci and Roderic Gross in [4] is a metric-free systems identification method. This method was aimed at solving problems (in this case inferring the behaviour of swarms) where typical metric-based system identification methods could not perform, whereas Turing learning achieved more accuracy in inferring the behaviour of swarms. Turing Learning also has the property of generality meaning it can be applied to other problems with similar setup i.e. (problems where you have to mimic the behaviour of a system).

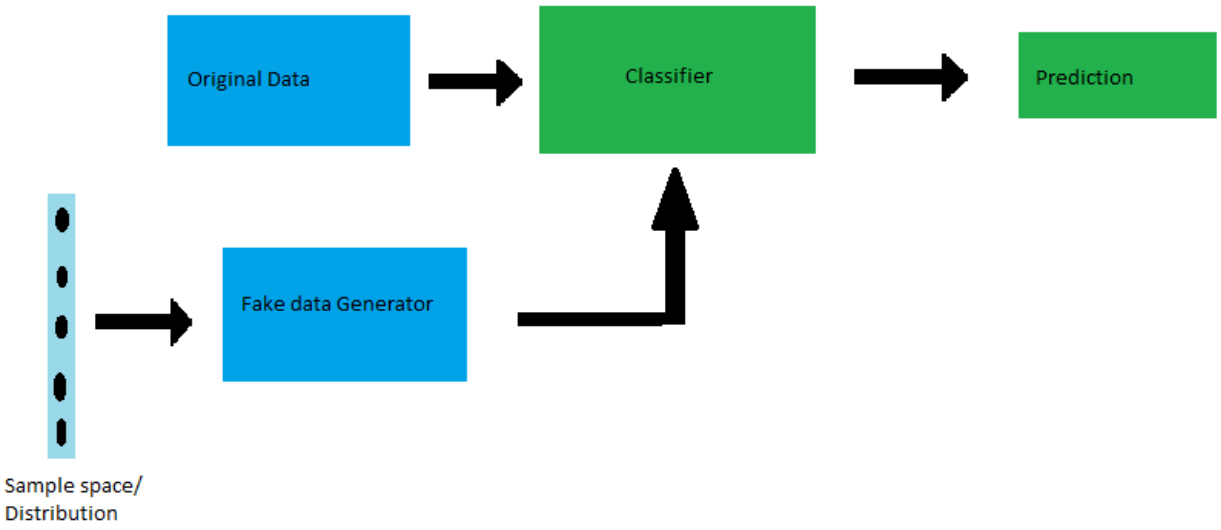


Figure 2-3: simple representation of a GANs architecture for image generation. Image copied from [A]

Turing learning works by optimizing a model and a classifier simultaneously which is similar to General Adversarial Networks (GAN) proposed by Ian Goodfellow in [5], Turing learning is the underlying concept used by GANs. It is called an *adversarial* network because the generator is pitted against the classifier. Figure 2-3 shows a simplified representation of a GAN or Turing learning model. The purpose of the model (Generator Network) is to produce data/ behaviour that imitates the original data/ system behaviour. The purpose of the classifier (Discriminative Network) is to discriminate/differentiate between the behaviour of the original system and the behaviour generated by the model. The property of generality in GANs is owed to the use of neural networks, the neural network this property because it is able to learn patterns in datasets which can then be used to produce or classify data with similar patterns from the original dataset. Regardless of the kind of dataset fed into a neural network, it will always learn patterns given patterns exist in the dataset, hence the generality. Neural networks come in different forms and the type of neural network used in a GAN varies depending on the application and system under observation. In inferring the behaviour of swarm robots, an Elman Neural Network was used as the classifier which also known as a Recurrent Neural Network (RNN). The RNN was used in this case as the classifier because it was dealing with time-series data as the swarm changes states as time progresses and RNNs are built specifically for this type of data, more details on Elman neural networks can be found in [6]. Another case where a different neural network

can be used is in the generation of images. In using GANs to generate fake images from a random distribution/ data sample, a Convolutional Neural Networks (CNNs) is used in this case to encode the images into smaller dimensions and extract important features and patterns in the image. The encoded vectors are used to classify and generate other images.

2.3.1. Training and optimization of the models and classifiers

This is the most important part of the Turing learning concept because the generative model and classifier have to be trained and optimized simultaneously. If either the generative model or classifier of them optimizes faster or slower than the other, then the entire system will collapse and settle in a locally optimal solution and neither of them will have the correct error feedback/weight updates to improve.

The training process for the networks is done via an algorithm called back-propagation. The algorithm simply updates the weights of the connection between the network layers by penalising connections that contribute to false prediction and strengthening connection that leads to accurate predictions, the weights of the layers are represented by matrices(2-dimensional)/tensors(3-dimensional and higher) depending on the hyper-parameters of your network. The weights of the neural networks are updated iteratively after every item in a dataset or after a batch of items in a dataset depending on the application, size and nature of the dataset. More details on the training of neural networks can be found in [7]

An optimization algorithm is used to compute the best weights in a neural network for better classification and prediction. There are various optimization techniques that can be used to train a neural network depending on the preference of the designer or the system under observation e.g. (Simulated Annealing optimization was used to train the network in [3], Stochastic Gradient Descent was used in [5] and in [4] pg. 8, it is stated that any population-based optimization algorithm can be used).

2.4. Work done

To date an algorithm has been designed based on the simulated annealing approach. Figure 1 shows the flowchart of the implementation of the simulated annealing approach. There are two main functions in this program i.e. The error function and perturb function. The criteria on which the error is measured determines what solutions the program accepts. The error function used in the program was the Mean Squared Error (MSE) of the summary statistics between Anscombe's quartet dataset and the generated dataset. The perturb function updates a random row at every iteration by adding a random number multiplied by a shift factor to the initial value, a shift factor of 0.1 is used to avoid large increment on the function.

The algorithm at this stage is inconsistent, the program was unable to function properly for iterations greater than 400 and did not perform well for random datasets with relatively larger error values i.e. (error values greater than 25). Figure 2.4.1 shows the performance trend of the algorithm on a dataset with relatively lower initial error values with the following initial and final statistic summary in Table 2.1. The Table below shows that the algorithm can successfully perturb a randomly generated dataset to achieve similar summary statistics as the Anscombe's quartet and graphically distinct as shown in Figure 2.4.1.

Summary Statistics	Final dataset (400 iterations)	Final dataset (200 iterations)	Initial Dataset	Anscombe's quartet
X mean	8.998173761931694	8.89591607601954	6.41584566247530 5	9.00000000000000
X SD	3.163364913029186	3.105074769885613 6	3.75945142403717 8	3.16227766016837 9
Y mean	7.501491187837787	7.5161411915701	5.62097269923834	7.50068181818181 8
Y SD	1.936245219088570 6	1.940736495058810 9	2.92045422763370 7	1.93653662393127 6
Pearson correlation	0.811514680632933 5	0.442171840259476 3	- 0.11893625218166 3	8.16366299680795 9

Table 2.1

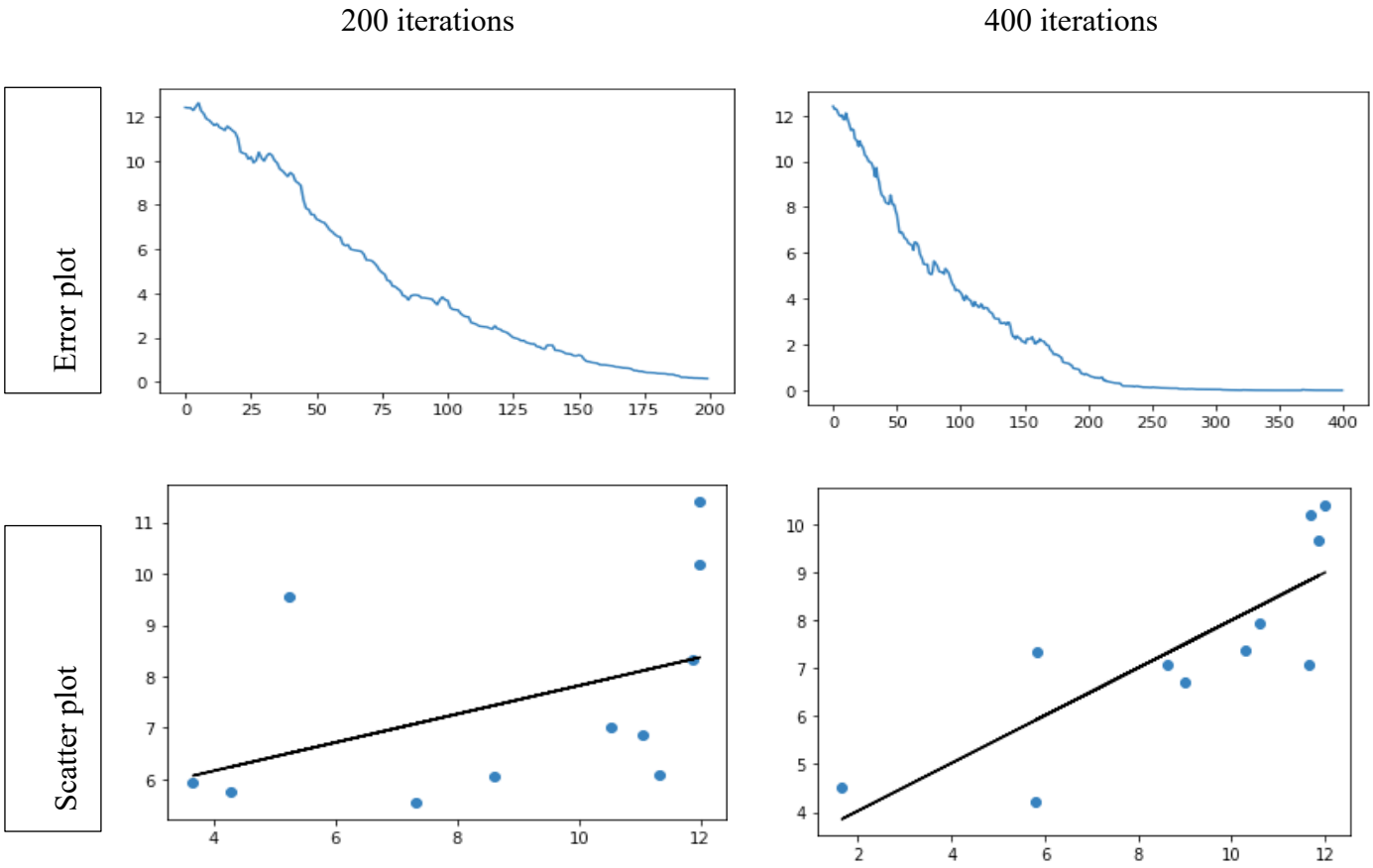


Figure 2.2.1. scatter plot of generated data set and error plot of the generated dataset

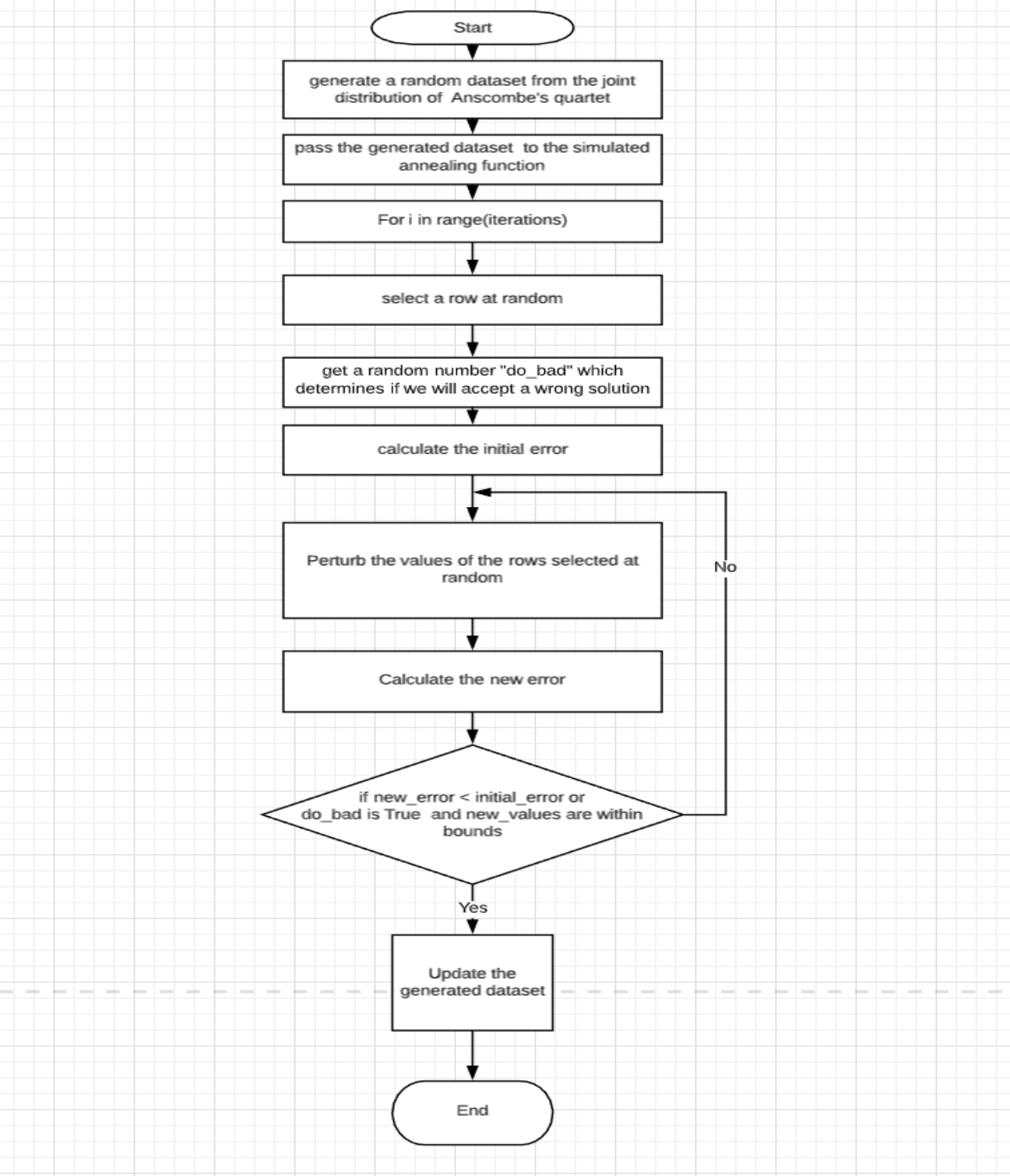


Figure 2-4-2 flowchart of the simulated annealing approach.

2.5 Summary

An overview of the origin of Anscombe's quartet was covered in detail. This occurrence shows the importance of combining both graphs and statistics in interpreting and understanding data.

Methods used to recreate generate more distinct datasets/graphs with similar statistics were reviewed by looking two main approaches i.e. 1) Using genetic algorithms and 2) Using machine learning with simulated annealing optimization. The genetic algorithm is still a valid solution when generating random distinct graphs while maintaining summary statistics without the intention of gearing it towards a target shape. The Simulated Annealing technique is a really good approach when altering the original dataset to form the graph plot of a target shape. The two approaches cannot be compared directly as they do not solve the same problem. The two approaches generally fulfil the target of generating distinct graphs while maintaining similar statistics.

Finally, having gone through previous work related to this project, concepts and solutions from previous works an algorithm has been developed which achieves the aims of the project and requires further development.

Chapter 3

3.1 Project management

The basic objectives for this project have been partially fulfilled, the algorithm needs to be improved on to produce more consistent results before moving ahead to advanced objectives. The genetic algorithm approach will also be developed to compare their performance and accuracy. To completely fulfil the basic objectives of the project the following deliverables should be achieved.

1. Implement a more stable algorithm for the simulated annealing approach.
    - Make the algorithm run for iterations greater than 400.
    - Ensure the algorithm works on data sets with higher error values i.e. (error values greater than 25).
2. Implement another algorithm solving the same problem using the genetic algorithm approach.
3. Analyse and visualise results generated from both algorithms.

An updated version of the project Gantt chart is a shown in Appendix A.

3.2 Self review

So far, I have been able to achieve most of the basic objectives for this project. I fell short in some areas concerning my conduct towards the project which I discussed with my supervisor and I will work towards making improvements in these areas. I have poorly managed my time on the project due to the demand of the other course works I am currently taking. To tackle this problem, I will begin to keep track of the time I spent on the project and the other course work, my goal is to spend at least 2hours 30minuites every day on the project and documenting my progress daily. I also noticed I need to improve on my technical writing skills and grammar in order to express my progress and ideas properly and comprehensively. To achieve this, I will need to spend more time proofreading my writing and schedule more one-on-one meetings with my supervisor to review my project.

Apart from the stated short comings I believe I will be able to achieve the basic objectives faster than expected and will have time to attempt the advanced objectives. This will give me more time to focus on my short comings.

Referencing

1. Anscombe, F.J. (1973). Graphs in Statistical Analysis. The American Statistician.
2. Chatterjee, S. and Firat, A. (2007). Generating Data with Identical Statistics but Dissimilar Graphics. The American Statistician.
3. Justin Matejka and George Fitzmaurice (2017). Same Stats, Different Graphs: Generating Datasets with Varied Appearance and Identical Statistics through Simulated Annealing.
4. Wei Li, Melvin Gauci and Roderich Groß (2016). Turing learning: a metric-free approach to inferring behaviour and its application to swarms.
5. Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio (2014). Generative Adversarial Nets.
6. David Samek (2008). Elman Neural Networks in model predictive control.
7. Joseph Rocca (2019). Understanding Generative Adversarial Networks. https://towardsdatascience.com/understanding-generative-adversarial-networks-gans-cd6e4651a29.
8. B. Mehlig (2019). Artificial Neural Networks.
9. The code for this project https://github.com/micaris/project/blob/master/SimulatedAnnealing.py

Appendix

Updated Gantt chart for final year project.