VISUALIZING DATA: BEYOND ANSCOMBE'S NUMBER

A FOURTH YEAR PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF B.Sc. IN COMPUTATIONAL MATHEMATICS

BY

Bhupendra Kumar Sah (026609-19)
(shahbhupendra9211@gmail.com)

Nikhil Sharma (026611-19)
(nikhil12sharma10@gmail.com)

Diya Shrestha (026613-19)
(diyashrestha2001@gmail.com)

SCHOOL OF SCIENCE
KATHMANDU UNIVERSITY
DHULIKHEL, NEPAL

FEBRUARY 2024

CERTIFICATION

This project entitled **"Visualizing Data: Beyond Anscombe's Number"** is carried out under my supervision for the specified entire period satisfactorily, and is hereby certified as a work done by following students

1. Bhupendra Kumar Sah (026609-19)
2. Nikhil Sharma (026611-19)
3. Diya Shrestha (026613-19)

in partial fulfillment of the requirements for the degree of B.Sc.in Computational Mathematics, Department of Mathematics, Kathmandu University, Dhulikhel, Nepal.

Prof. Dr. Kanhaiya Jha
Department of Mathematics,
School of Science, Kathmandu University,
Dhulikhel, Kavre, Nepal
Date: February 1, 2024

APPROVED BY:
I hereby declare that the candidate qualifies to submit this report of the Mathematics Seminar Project (MATH-451) to the Department of Mathematics.

Head of the Department
Department of Mathematics,
School of Science, Kathmandu University,
Dhulikhel, Kavre, Nepal
Date: February 1, 2024

Table of Contents

List of Figures

Abstract

Anscombe's Quartet can be defined as a group of four data sets which are nearly identical in simple descriptive statistics, but there are some peculiarities in the dataset that fools the regression model if built. They have very different distributions and appear differently when plotted on scatter plots. In this report, we provide a general procedure to generate datasets with identical summary statistics (mean, standard deviation and correlation) but dissimilar graphics by using a genetic algorithm based approach.

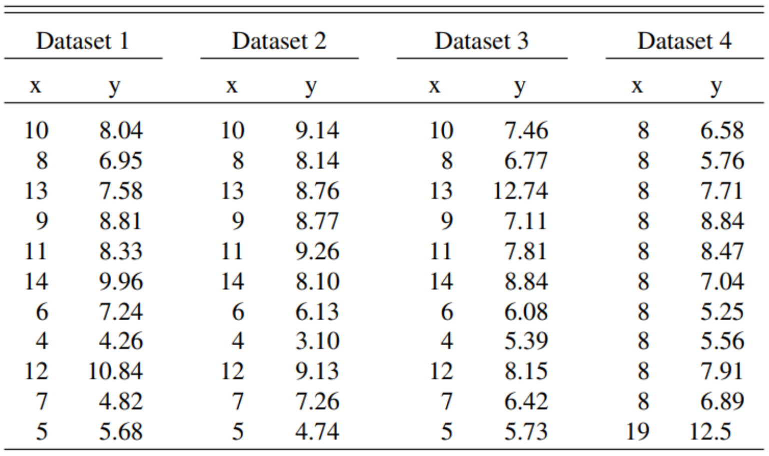
Keywords: Genetic algorithms, Ortho-normalization, Anscombe's Quartet

Chapter 1

Introduction

1.1 Background

Anscombe's quartet comprises a set of four datasets, having identical descriptive statistical properties in terms of means, variance, R-squared, correlations, and linear regression lines but having different representations when we scatter plots on a graph (3).



Dataset 1		Dataset 2		Dataset 3		Dataset 4	
x	y	x	y	x	y	x	y
10	8.04	10	9.14	10	7.46	8	6.58
8	6.95	8	8.14	8	6.77	8	5.76
13	7.58	13	8.76	13	12.74	8	7.71
9	8.81	9	8.77	9	7.11	8	8.84
11	8.33	11	9.26	11	7.81	8	8.47
14	9.96	14	8.10	14	8.84	8	7.04
6	7.24	6	6.13	6	6.08	8	5.25
4	4.26	4	3.10	4	5.39	8	5.56
12	10.84	12	9.13	12	8.15	8	7.91
7	4.82	7	7.26	7	6.42	8	6.89
5	5.68	5	4.74	5	5.73	19	12.5

Figure 1.1: The four datasets of Anscombe's quartet (4)

The four datasets that make up Anscombe's quartet each include 11 x-y pairs of data. When plotted, each dataset seems to have a unique connection between x and y, with unique variability patterns and distinctive correlation strengths. Despite these variations, each dataset has the same summary statistics, such as the same x and y mean and variance, x and y correlation coefficient, and linear regression line (3).

1.2 History

In 1973, Francis J. Anscombe published a paper titled, Graphs in Statistical Analysis. The idea of using graphical methods had been established relatively recently by John Tukey, but there was evidently still a lot of skepticism. Anscombe first listed some notions that textbooks were "indoctrinating" people with, like the idea that "numerical calculations are exact, but graphs are rough."

He then presented a table of numbers. It contained four distinct datasets (hence the name Anscombe's Quartet), each with statistical properties that are essentially identical: the mean of the x values is 9.0, mean of y values is 7.5, they all have nearly identical variances, correlations, and regression lines (to at least two decimal places) (7).

It is not known how Anscombe created his datasets. Since its publication, several methods to generate similar data sets with identical statistics and dissimilar graphics have been developed. One of these, the Datasaurus Dozen, consists of points tracing out the outline of a dinosaur, plus twelve other data sets that have the same summary statistics (8).

1.3 Statistical Values of Anscombe's Quartet

- **Mean;**
  x1(9), x2(9), x3(9), x4(9), y1(7.500909), y2(7.500909), y3(7.5), y4(7.500909)
- **Standard Deviation:**
  x1(3.316625), x2(3.316625), x3(3.316625), x4(3.316625), y1(2.031568), y2(2.031657), y3(2.030424), y4(2.030579)
- **Correlation:**
  x1y1(0.8164205), x2y2(0.8162365), x3y3(0.8162867), x4y4(0.8165214)

1.4 Representation of Anscombe's Quartet

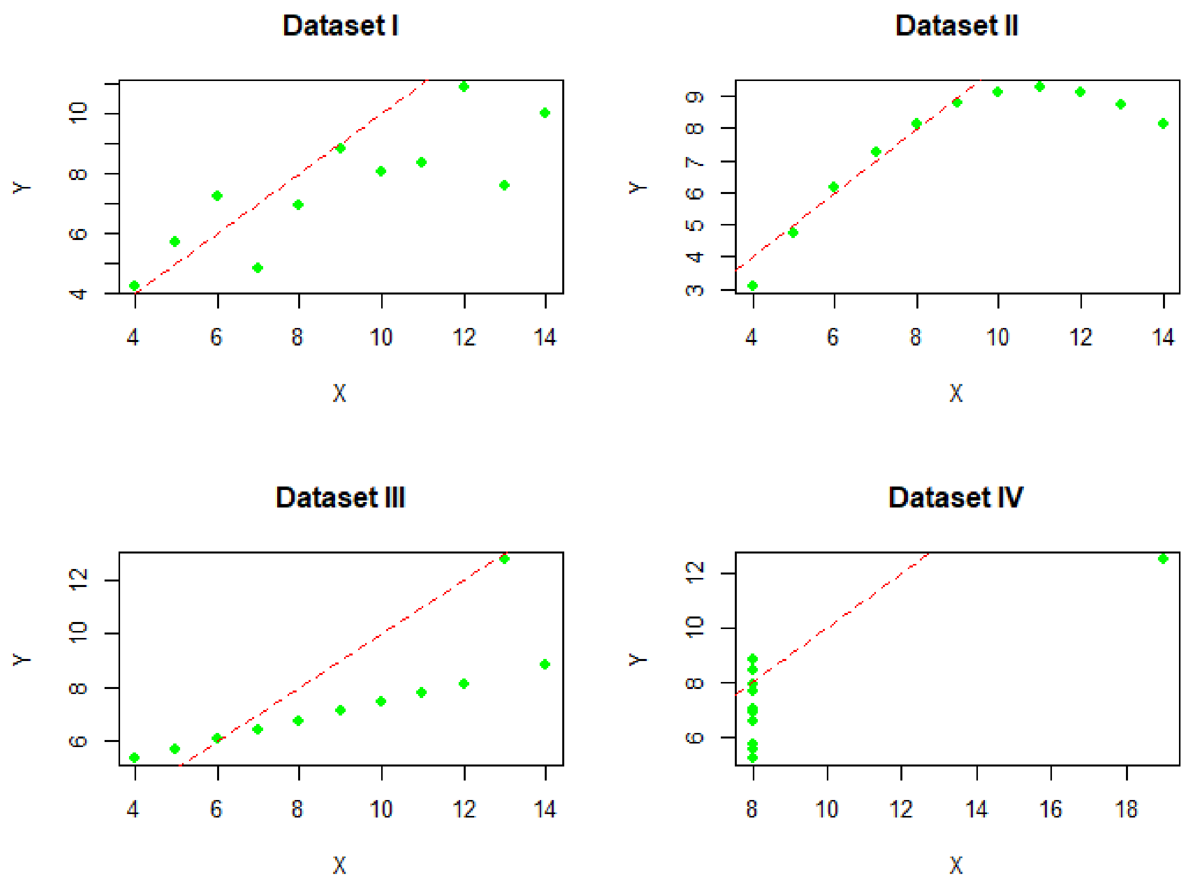


Figure 1.2: Scatter plot and linear regression line for each datasets

- **Dataset I:** consists of a set of (x,y) points that represent a linear relationship with some variance
- **Dataset II:** shows a curve shape but does not show a linear relationship
- **Dataset III:** looks like a tight linear relationship between x and y, except for one large outlier
- **Dataset IV:** looks like the value of x remains constant, except for one outlier as well (6)

1.5 Objectives

- Anscombe's Quartet emphasizes the importance of visualizing data, as graphs can reveal patterns and outliers that summary statistics alone may overlook (5).
- By showing that four datasets with different patterns can have the same summary statistics, it challenges the notion that summary statistics alone provide a complete understanding of data.
- The quartet emphasizes that even when summary statistics suggest a linear relationship, the actual data may exhibit non-linear patterns or outliers that significantly affect interpretations.
- It underscores the importance of robustness testing in statistical analysis, urging researchers to explore alternative methods and visualizations to ensure the validity and reliability of their findings.
- It encourages analysts to question assumptions and delve deeper into their data, rather than relying solely on superficial summary measures.

1.6 Limitations

- Anscombe's quartet consists of only four artificially constructed datasets, limiting its ability to represent the diversity of real-world data scenarios.
- The datasets are intentionally crafted to have identical summary statistics and specific patterns, which may not fully capture the randomness and complexity of natural data.
- It primarily focuses on bivariate relationships and may not address the challenges associated with multivariate datasets.
- It does not account for the contextual nuances and domain-specific knowledge that can influence data analysis and interpretation.
- While visualization is crucial, Anscombe's quartet may overemphasize its importance compared to other analytical techniques, such as statistical analysis and domain expertise.

Chapter 2

Mathematical Results

2.1 Problem Description

Consider a given data matrix X^* consisting of two data vectors of size n: the independent variable x^*, and the dependent variable y^*. Let $\overline{x^*}$, $\overline{y^*}$ be the mean value, and s_x*, s_y* be the standard deviation of vectors, and r^* be the correlation coefficient between vectors x^* and y^*. Let X be another data matrix containing two data vectors of size n : x, y. The problem is to find at least one X that has identical summary statistics as X^*. At the same time, scatter plots of x, y should be dissimilar to those of x^*, y^* according to a function g(X, X^*), which quantifies the graphical difference between the scatter plots of x, y and x^*, y^*. This problem can be formulated as a mathematical program as follows:

$$\begin{gathered}\text{maximize g(X, } X^*\text{)}\\ \text{s.t.}\\ |\overline{x^*} \text{ - } \overline{x}| + |\overline{y^*} \text{ - } \overline{y}| + |\ s_x* \text{ - } s_x| + |\ s_y* \text{ - } s_y| + |\text{r}^* \text{ - r}| = 0\end{gathered}$$

In the above formulation, the objective function to be maximized is the graphical dissimilarity between X and X^*, and the constraint ensures that the summary statistics will be identical. Different tests are used to check the differences between these statistical parameters (4). They are listed below:

- Ordered data differentials:

$$g_{ord} = \sum |x_i - x_i^*| + |y_i - y_i^*|$$

- Kolmogorov-Smirnov Test differentials:

$$g_{ks} = max\left(|F(a) - F^{*}(a)|\right)$$

- Quadratic regression coefficient differentials:

$$g_{reg} = |b_2 - b_2^{*}|$$

- Breusch-Pagan heteroscedasticity test differentials:

$$g_{LM} = |LM - LM^{*}|$$

- Skewness differentials:

$$g_{skewness} = \left| \frac{\sum (y_i - \overline{y})^3}{(s_y)^3} - \frac{\sum \left(\overset{*}{y_i} - \overline{y^{*}}\right)^3}{(s_y^{*})^3} \right|$$

- Kurtosis differentials:

$$g_{kurtosis} = \left| \frac{\sum (y_i - \overline{y})^4}{(s_y)^4} - \frac{\sum \left(\overset{*}{y_i} - \overline{y^{*}}\right)^4}{(s_y^{*})^4} \right|$$

- Cook's d statistic differentials:

$$g_{cookd} = |max(d_i) - max(d_i^{*})|$$

2.2 Methodology

We follow a genetic algorithm based solution for our problem to create datasets similar to Anscombe's Quartet. Genetic Algorithm (GA) is a search-based optimization technique based on the principles of Genetics and Natural Selection. It is frequently used to find optimal or near-optimal solutions to difficult problems which otherwise would take a lifetime to solve. It is frequently used to solve optimization problems, in research, and in machine learning (9). There are five phases of genetic algorithm which needs to be considered - initial population, fitness function, selection, crossover and mutation.

The steps for the completion of genetic algorithm are listed below:

- Individual in population compete for resources and mate
- Those individuals who are successful (fittest) then mate to create more offspring than others
- Genes from "fittest" parent propagate throughout the generation, that is sometimes parents create offspring which is better than either parent
- Thus each successive generation is more suited for their environment

2.2.1 Representation

We conceptualize a gene as a matrix of size n $\times$ 2 having real values.

2.2.2 Initial Population Creation

Given an original data matrix X^* of size n $\times$ 2, we accomplish this through orthonormalization and a transformation as outlined in the following for a single gene.

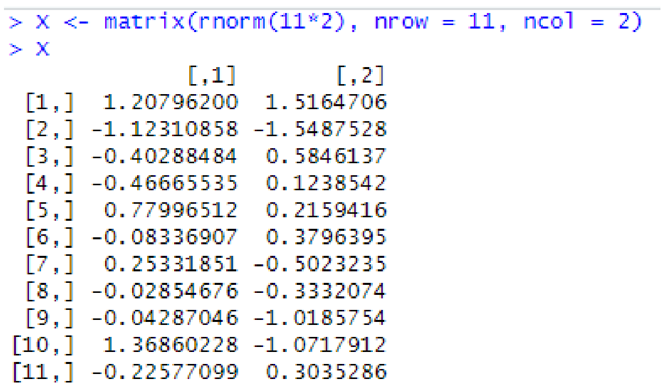
Step 1

Generate a matrix X of size n$\times$2 with randomly generated data from a standard normal distribution. Distributions other than the standard normal can also be used in this step.

R Script to generate a random dataset with 11 rows and 2 columns :

```
X = matrix(rnorm(11*2), nrow = 11, ncol = 2)
X
```

Output:



```
> X <- matrix(rnorm(11*2), nrow = 11, ncol = 2)
> X
             [,1]       [,2]
 [1,]  1.20796200  1.5164706
 [2,] -1.12310858 -1.5487528
 [3,] -0.40288484  0.5846137
 [4,] -0.46665535  0.1238542
 [5,]  0.77996512  0.2159416
 [6,] -0.08336907  0.3796395
 [7,]  0.25331851 -0.5023235
 [8,] -0.02854676 -0.3332074
 [9,] -0.04287046 -1.0185754
[10,]  1.36860228 -1.0717912
[11,] -0.22577099  0.3035286
```

Figure 2.1: Output showing the generated random dataset X with 11 rows and 2 columns

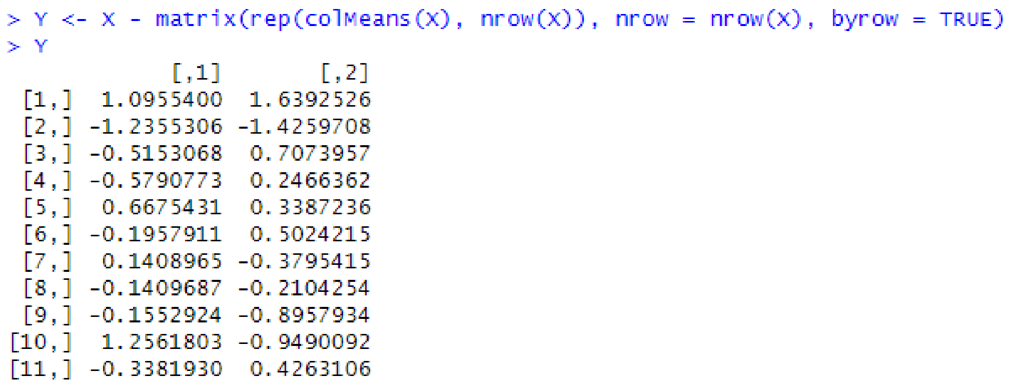
Step 2

Set the mean values of X's columns to zero using X = X-$e_{n\times1}$ $*\overline{X}$, where $e_{n\times1}$ is an n-element column vector of ones. This step is needed to make sure that after ortho-normalization the standard deviation of the columns will be equal to the unit vector norm.

R Script to set the mean values of generated random dataset X to zero:

```r
Y = X - matrix(rep(colMeans(X), nrow(X)), nrow = nrow(X), byrow = TRUE)
Y
```

Output:



```
> Y <- X - matrix(rep(colMeans(X), nrow(X)), nrow = nrow(X), byrow = TRUE)
> Y
            [,1]       [,2]
 [1,]  1.0955400  1.6392526
 [2,] -1.2355306 -1.4259708
 [3,] -0.5153068  0.7073957
 [4,] -0.5790773  0.2466362
 [5,]  0.6675431  0.3387236
 [6,] -0.1957911  0.5024215
 [7,]  0.1408965 -0.3795415
 [8,] -0.1409687 -0.2104254
 [9,] -0.1552924 -0.8957934
[10,]  1.2561803 -0.9490092
[11,] -0.3381930  0.4263106
```

Figure 2.2: Output showing the updated dataset of random dataset X

Step 3

Ortho-normalize the columns of X. For this, we use the Gram-Schmidt process, by taking a nonorthogonal set of linearly independent vectors x and y constructing an orthogonal basis e_1 and e_2 as follows (in R^2):

$$u_1 = x, u_2 = y - proj_{u_1} y,$$

where

$$proj_u v = \frac{\langle v, u \rangle}{\langle u, u \rangle} u,$$

and $\langle v_1, v_2 \rangle$ represents the inner product. Then

$$e_1 = \frac{u_1}{||u_1||}, and e_2 = \frac{u_2}{||u_2||},$$

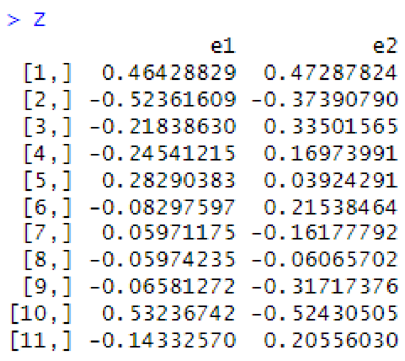
and

$$X_{ortho-normalized} = [e_1, e_2]$$

R Script to ortho-normalize the columns of updated dataset Y:

```r
grams = function(data) {
  if (ncol(data) == 1) {
    return(data)}
  u1 = data[, 1]
  u2 = data[, 2] - (t(u1) %*% data[, 2]) / (t(u1) %*% u1) * u1
  e1 = u1 / sqrt(t(u1) %*% u1)
  e2 = u2 / sqrt(t(u2) %*% u2)
  return(cbind(e1, e2))}
Z = grams(Y)
Z
```

Output:



```
> Z
               e1          e2
 [1,]  0.46428829  0.47287824
 [2,] -0.52361609 -0.37390790
 [3,] -0.21838630  0.33501565
 [4,] -0.24541215  0.16973991
 [5,]  0.28290383  0.03924291
 [6,] -0.08297597  0.21538464
 [7,]  0.05971175 -0.16177792
 [8,] -0.05974235 -0.06065702
 [9,] -0.06581272 -0.31717376
[10,]  0.53236742 -0.52430505
[11,] -0.14332570  0.20556030
```

Figure 2.3: Output showing the random dataset Y after ortho-normalization

Step 4

Transform X with the following equation:

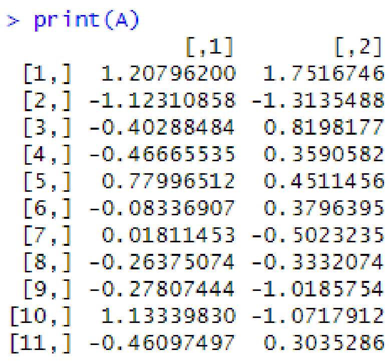
$$X = \sqrt{n-1} * X_{ortho-normalized} * cov(X^*)^{1/2} + e_{n\times 1} * \overline{X^*},$$

where $cov(X)^*$ is the covariance matrix of X^*, $\overline{X^*} = \left[\overline{x_1^*}, \overline{x_2^*}\right]$.
$\sqrt{n-1}$ is needed since we are using the sample standard deviation in covariance calculations.

R Script to transform the ortho-normalized form of updated random dataset Z:

```
cov_matrix = cov(X)
A = sqrt(10) * Z %*% chol(cov_matrix) + matrix(rep(colMeans(X), each =
    nrow(X)), nrow = nrow(X), ncol = ncol(X), byrow = TRUE)
print(A)
```

Output:



```
> print(A)
             [,1]       [,2]
 [1,]  1.20796200  1.7516746
 [2,] -1.12310858 -1.3135488
 [3,] -0.40288484  0.8198177
 [4,] -0.46665535  0.3590582
 [5,]  0.77996512  0.4511456
 [6,] -0.08336907  0.3796395
 [7,]  0.01811453 -0.5023235
 [8,] -0.26375074 -0.3332074
 [9,] -0.27807444 -1.0185754
[10,]  1.13339830 -1.0717912
[11,] -0.46097497  0.3035286
```

Figure 2.4: Output showing the transformed and ortho-normalized random dataset A

The dataset obtained after performing all these steps will have similar statistical properties as that of the initially generated random dataset.

R Script to compare the two datasets X and A:

```
plot(X,
     main="Scatterplot␣of␣Random␣Candidate␣Dataset",
     xlab="X", ylab="Y", pch=1)
points(A,
       main="Scatterplot␣of␣Random␣Candidate␣Dataset",
       xlab="X", ylab="Y", pch=16)
```

Output:

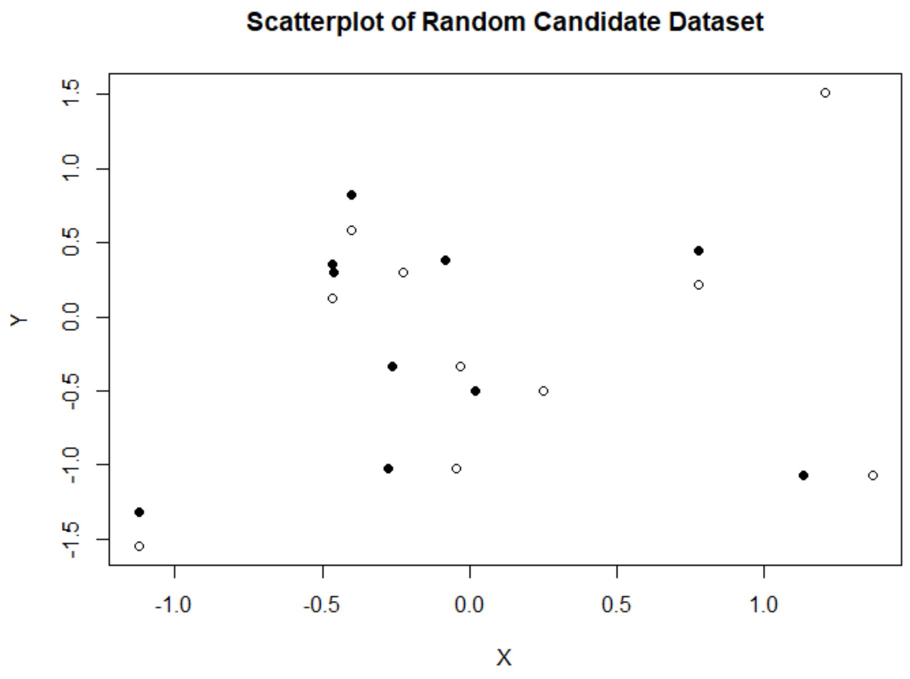


Figure 2.5: Output showing the visual representation of the datasets X and A

Table 2.1: Table with comparison of mean and standard deviation of the generated datasets

Parameter	Random dataset X	Random dataset A
Mean	-0.005180004	-0.005180004
Standard Deviation	0.8045735	0.8148566

2.2.3 Fitness Values

At each generation, a fitness value needs to be calculated for each population member. For our problem a gene's fitness is proportional to its graphical difference from the given data matrix X^*. We used the graphical difference functions mentioned in the problem description section in different runs of experiments.

2.2.4 Creating The Next Generation

Selection

Once the fitness values are calculated, parents are selected for the next generation based on their fitness. We use the “stochastic uniform” selection procedure. This selection algorithm first lays out a line in which each parent corresponds to a section of the line of length proportional to its scaled fitness value. Then the algorithm moves along the line in steps of equal size. At each step, the algorithm allocates a parent from the section it lands on.

New Children

Three types of children are created for the next generation:

- **Elite Children**
  Individuals in the current generation with the top two best fitness values are called elite children, and automatically survive in the next generation.

- **Crossover Children**
  These are children obtained by combining two parent genes. A child is obtained by splitting two selected parent genes at a random point, and combining the head of one with the tail of the other and vice versa. Eighty percent of the individuals in the new generation are created this way.

- **Mutation Children**
  Mutation children make up the remaining members of the new generation. A parent gene is modified by adding a random number, or mutation, chosen from Gaussian distribution(with mean 0, and variance 0.5) to each entry of the parent vector. Mutation amount decreases at each generation according to the following formula:

  $$var_k = var_{k-1}\left(1 - 0.75 * \frac{k}{Generations}\right)$$

  where var_k is the variance of the Gaussian distribution at the current generk, and Generations is the number of total generations (4).

Ortho-normalization and Transformation

After the children for the new generation are created, they are ortho-normalized and transformed so that they also satisfy the constraint like the initial population.

2.2.5 Final Generation

The algorithm runs for 2,500 generations or until there is no improvement in the objective function during an interval of 20 seconds. Within the final generation, genes with large fitness values are obvious solutions to our problem.

2.3 Statistical Tests

To perform the tests, the steps of genetic algorithm mentioned should be performed. The detailed description is explained in the R Script below:

R Script to setup environment to perform statistical tests:

Load necessary libraries.

```
library(ggplot2)
library(MASS)
library(e1071)
```

Choose a dataset from Anscombe's Quartet

```
data = anscombe[, c("x1", "y1")]
data
```

Fit a linear regression model

```
model = lm(y1 ~ x1, data = data)
```

Perform Gram-Schmidt orthogonalization

```
U = matrix(0, nrow = nrow(data), ncol = ncol(data))
U[, 1] = data[, 1] / sqrt(sum(data[, 1]^2))
for (i in 2:ncol(data)) {
  U[, i] = data[, i]
  for (j in 1:(i - 1)) {
    U[, i] = U[, i] - (sum(U[, j] * U[, i]) * U[, j])
  }
  U[, i] = U[, i] / sqrt(sum(U[, i]^2))
}
```

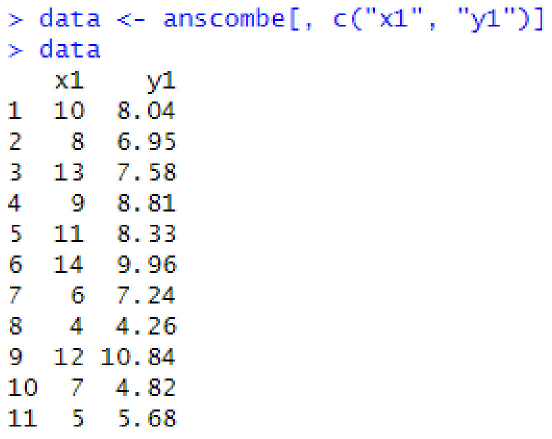
Display the corresponding orthogonal matrix

```
cat("Corresponding␣orthogonal␣matrix␣is\n")
print(U)
```

Perform the transformation

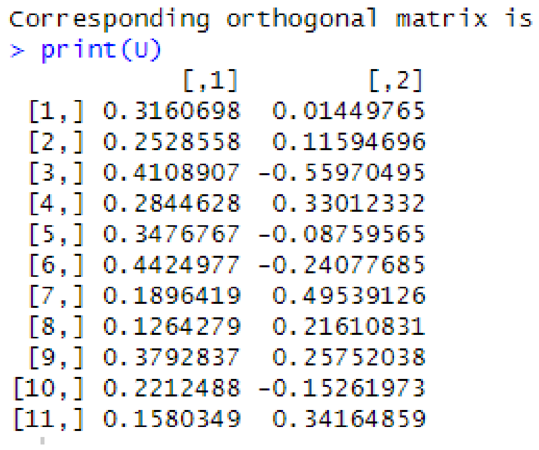
```
cat("After␣transformation,␣the␣matrix␣is\n")
cov_matrix = cov(data)
Y = sqrt(10) * U %*% chol(cov_matrix) + matrix(rep(colMeans(data), each =
    nrow(data)), nrow = nrow(data), ncol = ncol(data), byrow = TRUE)
print(Y)
```

Outputs:



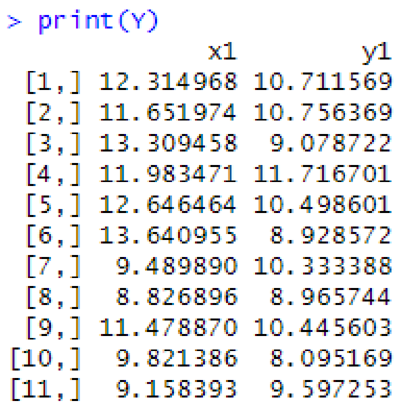
```
> data <- anscombe[, c("x1", "y1")]
> data
   x1    y1
1  10  8.04
2   8  6.95
3  13  7.58
4   9  8.81
5  11  8.33
6  14  9.96
7   6  7.24
8   4  4.26
9  12 10.84
10  7  4.82
11  5  5.68
```

Figure 2.6: Choosing and displaying a dataset from Anscombe's Quartet



```
Corresponding orthogonal matrix is
> print(U)
           [,1]        [,2]
 [1,] 0.3160698  0.01449765
 [2,] 0.2528558  0.11594696
 [3,] 0.4108907 -0.55970495
 [4,] 0.2844628  0.33012332
 [5,] 0.3476767 -0.08759565
 [6,] 0.4424977 -0.24077685
 [7,] 0.1896419  0.49539126
 [8,] 0.1264279  0.21610831
 [9,] 0.3792837  0.25752038
[10,] 0.2212488 -0.15261973
[11,] 0.1580349  0.34164859
```

Figure 2.7: Displaying the orthogonal matrix resulting from performing Gram-Schmidt Orthogonalization



```
> print(Y)
             x1        y1
 [1,] 12.314968 10.711569
 [2,] 11.651974 10.756369
 [3,] 13.309458  9.078722
 [4,] 11.983471 11.716701
 [5,] 12.646464 10.498601
 [6,] 13.640955  8.928572
 [7,]  9.489890 10.333388
 [8,]  8.826896  8.965744
 [9,] 11.478870 10.445603
[10,]  9.821386  8.095169
[11,]  9.158393  9.597253
```

Figure 2.8: Displaying the tranformed orthogonalized matrix

2.3.1 Kolmogorov-Smirnov differential test

The Kolmogorov Smirnov test (KS test or K-S test) is used to compare two distributions to determine if they are pulling from the same underlying distribution. The goal of the KS test is to determine if two distributions A and B are different. The KS test statistic gives a numeric value related to that difference. The KS test statistic is defined as the maximum value of the difference between A and B's cumulative distribution functions (CDF). In machine learning settings, CDFs are typically derived empirically from samples of the datasets and would be called eCDFs (11).

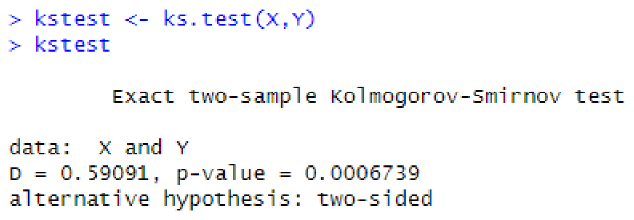
The formula to calculate Kolmogorov-Smirnov differential coefficient is:

$$g_{ks} = max\left(|F(a) - F^{*}(a)|\right)$$

R Script to run Kolmogorov-Smirnov differential test:

```
kstest <- ks.test(X,Y)
kstest
```

Output:



```
> kstest <- ks.test(X,Y)
> kstest

        Exact two-sample Kolmogorov-Smirnov test

data:  X and Y
D = 0.59091, p-value = 0.0006739
alternative hypothesis: two-sided
```

Figure 2.9: Displaying the calculated Kolmogorov-Smirnov differential coefficient

R Script of scatter plot with Kolmogorov-Smirnov differential:

```
ggplot(data.frame(value = c(X,Y), sample = rep(c("X", "Y"), each = length(
   X)))) +
  geom_point(aes(x = value, y = 0, color = sample)) +
  labs(x = "Variable", y = "", title = "Scatter␣Plot␣with␣Kolmogorov-
     Smirnov␣Differential") +
  theme(axis.text.y = element_blank(), axis.ticks.y = element_blank())
```

Output:

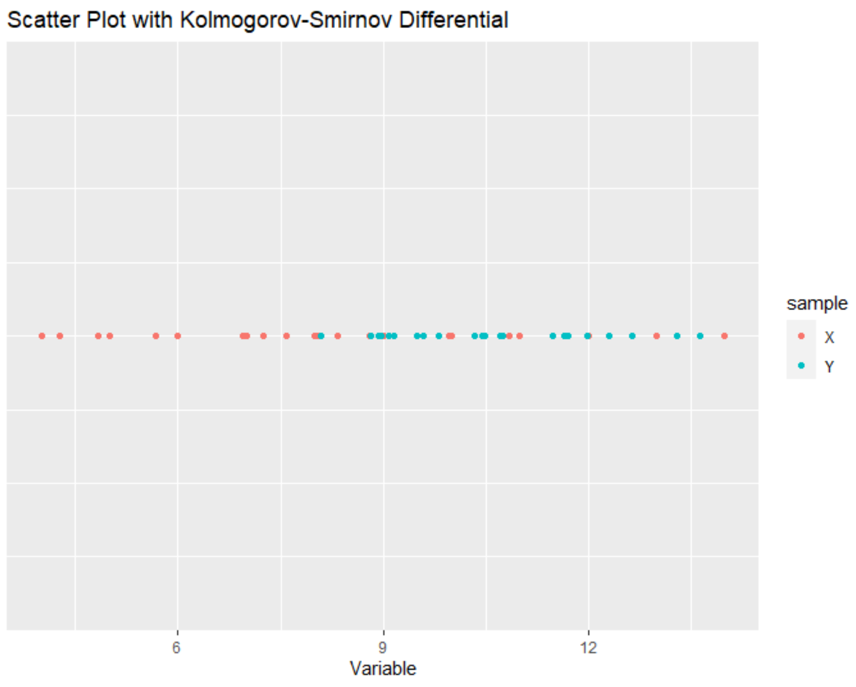


Figure 2.10: Scatter plot with Kolmogorov-Smirnov differential

Advantages

- The KS test doesn't require assumptions about the distribution of data, making it versatile across various data types.
- It's highly sensitive to differences in distributions, even capturing subtle variations in tails and shapes.
- The test yields a simple statistic that indicates the degree of dissimilarity between distributions, facilitating easy interpretation.
- Widely available in statistical software, it's easily implemented and used across different disciplines.

Disadvantages

- The KS test's sensitivity can decrease with small sample sizes, leading to inaccurate results or low power to detect differences between distributions.
- While the KS test detects differences between distributions, it doesn't provide insights into the specific nature or causes of those differences.
- It may be less reliable when dealing with tied observations, as it does not adjust for or handle ties in a manner that preserves accuracy.
- The test assumes independence between observations, which may not hold true in all data sets, potentially leading to misleading conclusions when this assumption is violated.

2.3.2 Skewness differential

Skewness is a measurement of the distortion of symmetrical distribution or asymmetry in a data set. Skewness is demonstrated on a bell curve when data points are not distributed symmetrically to the left and right sides of the median on a bell curve. If the bell curve is shifted to the left or the right, it is said to be skewed (12).

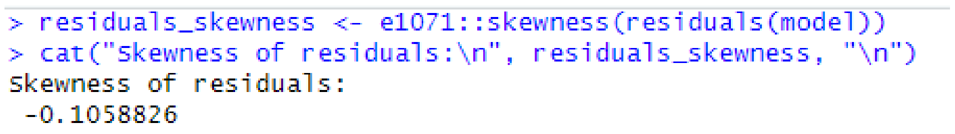
The formula to calculate Skewness differential coefficient is:

$$g_{skewness} = \left| \frac{\sum (y_i - \overline{y})^3}{(s_y)^3} - \frac{\sum \left(\overset{*}{y_i} - \overline{y^*}\right)^3}{(s_y^*)^3} \right|$$

R Script to run Skewness differential test:

```r
residuals_skewness = e1071::skewness(residuals(model))
cat("Skewness␣of␣residuals:\n", residuals_skewness, "\n")
```

Output:



```
> residuals_skewness <- e1071::skewness(residuals(model))
> cat("Skewness of residuals:\n", residuals_skewness, "\n")
Skewness of residuals:
 -0.1058826
```

Figure 2.11: Displaying the calculated Skewness differential coefficient

R Script of scatter plot with Skewness differential:

```r
ggplot(data, aes(x = fitted(model), y = residuals(model))) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE, color = "blue") +
  labs(title = "Scatter␣Plot␣with␣Skewness␣Differential")
```

Output:

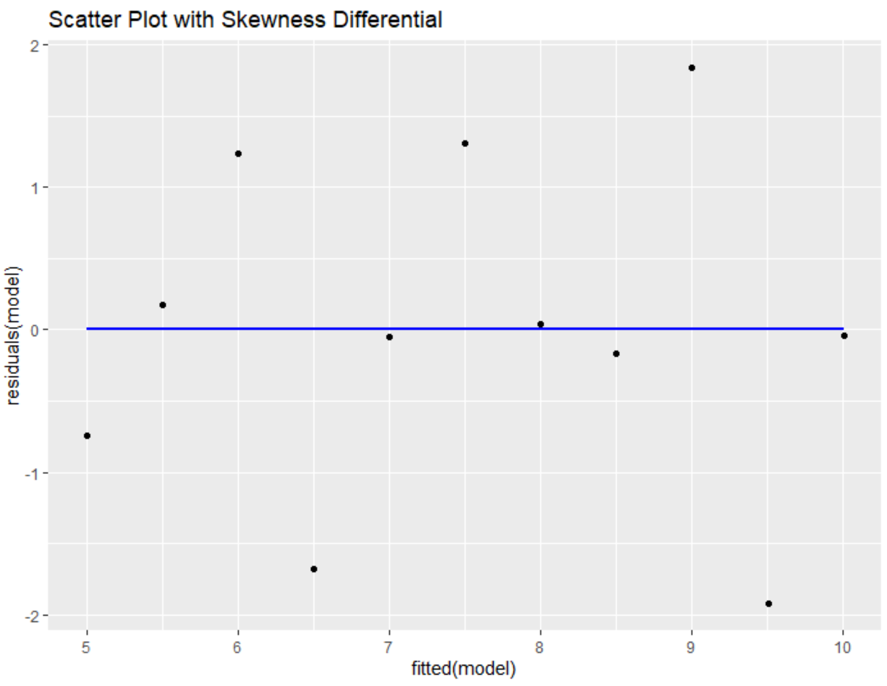


Figure 2.12: Scatter plot with Skewness differential

Advantages

- It specifically targets differences in skewness between two distributions, providing a focused analysis that can be valuable when skewness comparison is of particular interest.
- The test is effective in identifying departures from normality, which is valuable when analyzing datasets where normality assumptions may not hold.
- Results of the skewness differential test are straightforward to interpret, as they directly indicate whether the skewness of one distribution differs significantly from another.
- It can complement other statistical tests by providing additional insights into the distributional differences that may not be captured by measures of central tendency or variability alone.

Disadvantages

- The skewness differential test is primarily suited for detecting differences in skewness between two distributions and may not be suitable for other types of comparisons or analyses.
- Similar to other statistical tests, the skewness differential test's performance may be affected by small sample sizes, leading to unreliable results or low power.
- It relies heavily on the skewness measure, which may not adequately capture the entire distribution's characteristics, especially in complex or multimodal datasets.
- While the test can identify differences in skewness, interpreting these differences in practical terms or understanding their implications may not always be straightforward, particularly without additional context or analysis.

2.3.3 Cook's d statistic differential

Cook's distance, Di, is used in Regression Analysis to find influential outliers in a set of predictor variables. In other words, it's a way to identify points that negatively affect your regression model. The measurement is a combination of each observation's leverage and residual values; the higher the leverage and residuals, the higher the Cook's distance. Several interpretations for Cook's distance exist. There isn't a universally accepted rule for cut off points (13).

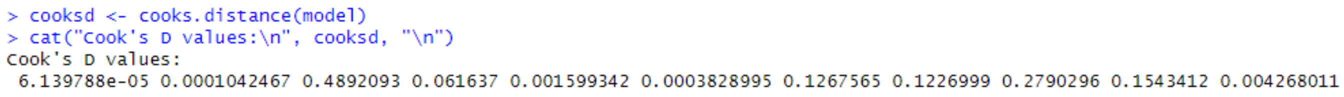
The formula to calculate Cook's d statistic differential coefficient is:

$$g_{cookd} = |max(d_i) - max(d_i^*)|$$

R Script to run Cook's d statistic differential test:

```
cooksd = cooks.distance(model)
cat("Cook's␣D␣values:\n", cooksd, "\n")
```

Output:



```
> cooksd <- cooks.distance(model)
> cat("Cook's D values:\n", cooksd, "\n")
Cook's D values:
 6.139788e-05 0.0001042467 0.4892093 0.061637 0.001599342 0.0003828995 0.1267565 0.1226999 0.2790296 0.1543412 0.004268011
```

Figure 2.13: Displaying the calculated Cook's d statistic differential coefficient

R Script of scatter plot with Cook's d statistic differential:

```r
ggplot(data, aes(x = x1, y = y1)) +
  geom_point(aes(size = cooksd)) +
  geom_smooth(method = "lm", se = FALSE, color = "red") +
  labs(title = "Linear␣Regression␣with␣Cook's␣D␣Values")
```

Output:

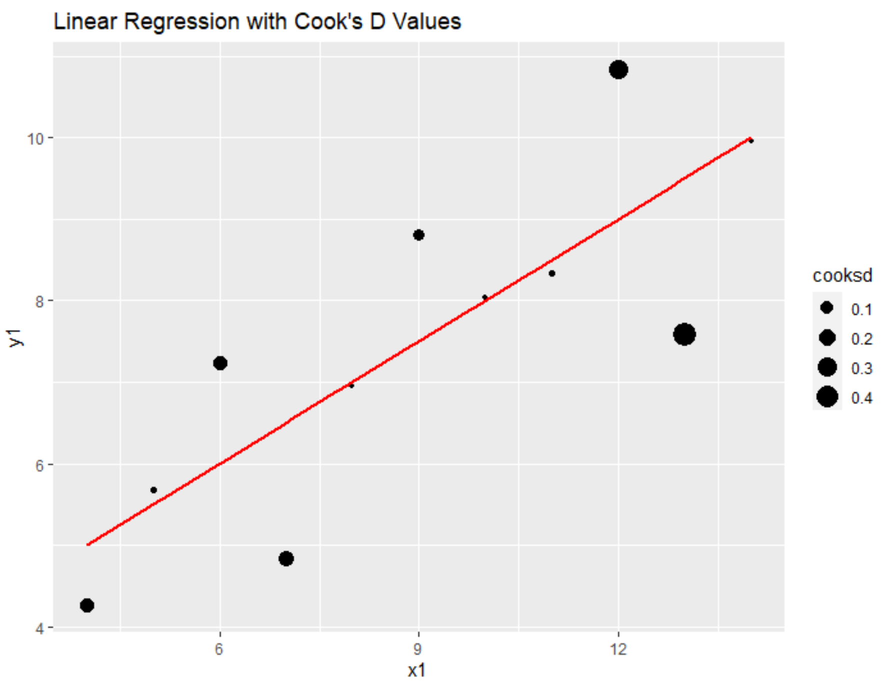


Figure 2.14: Scatter plot with Cook's d statistic differential

Advantages

- Cook's D test helps identify influential data points that have a disproportionate impact on the regression model's parameters, allowing analysts to assess their effect on the overall model.
- It provides a quantitative measure of the influence of each data point on the regression coefficients, aiding in the prioritization of data points for further investigation or potential exclusion.
- Cook's D test enables analysts to assess the robustness of their regression models by identifying influential observations that may distort parameter estimates or compromise model accuracy.
- It serves as a diagnostic tool for detecting outliers or influential observations,

facilitating data exploration and refinement of regression models to improve their predictive accuracy and reliability.

Disadvantages

- Cook's D test performance may be impacted by small sample sizes, potentially leading to unreliable results or limited power to detect influential points accurately.
- The effectiveness of Cook's D test relies on the underlying assumptions of the regression model, such as linearity, normality, and independence of errors. Violations of these assumptions can affect the accuracy and validity of the test results.
- Cook's D values do not have a straightforward interpretation. While higher values indicate greater influence, determining the significance or practical relevance of these influences may require additional context or judgment.
- Cook's D test may not distinguish between influential points that genuinely affect the regression model's fit and leverage points that merely influence the estimated coefficients without significantly altering the model's predictive performance.

Chapter 3

Applications

3.1 Coercion Towards Target Shapes

In this example, each dataset contains 182 points and are equal (to two decimal places) for the "standard" summary statistics (x/y mean, x/y standard deviation, and Pearson's correlation). Each dataset was seeded with the plot in the top left. The target shapes are specified as a series of line segments, and the shapes used in this example is shown in the figure (10).

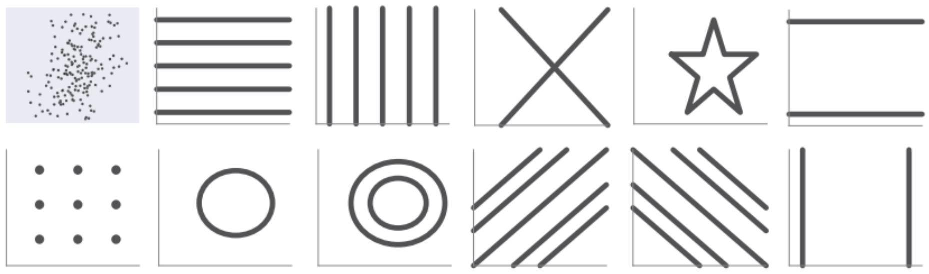


Figure 3.1: The initial data set (top-left), and line segment collections used for directing the output towards specific shapes

With this example dataset, the algorithm ran for 200,000 iterations to achieve the final results. On a laptop computer this process took about 10 minutes.

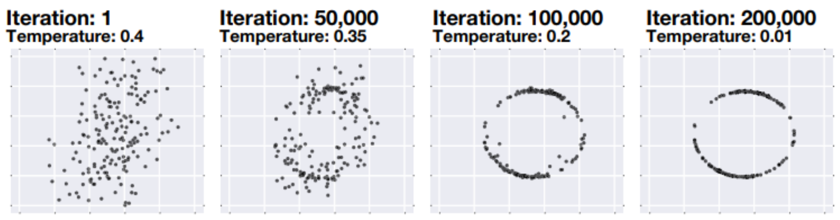


Figure 3.2: Progression of the algorithm towards a target shape over the course of the cooling schedule

3.2 Alternate Statistical Measures

In this example, the iterative process is agnostic to the particular statistical properties which remain constant between the datasets. The datasets are derived from the same initial dataset as in example first, but rather than being equal on the parametric properties, the datasets are equal in the non-parametric measures of x/y median, x/y interquartile range (IQR), and Spearman's rank correlation coefficient (10).

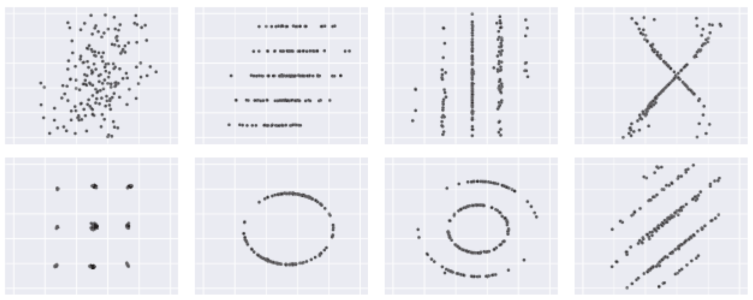


Figure 3.3: Example datasets are equal in the non-parametric statistics of x/y median (53.73, 46.21), x/y IQR (19.17, 37.92), and Spearman's rank correlation coefficient (+0.31)

3.3 Specific Initial Dataset

This example uses a very specific dataset to seed the optimization.

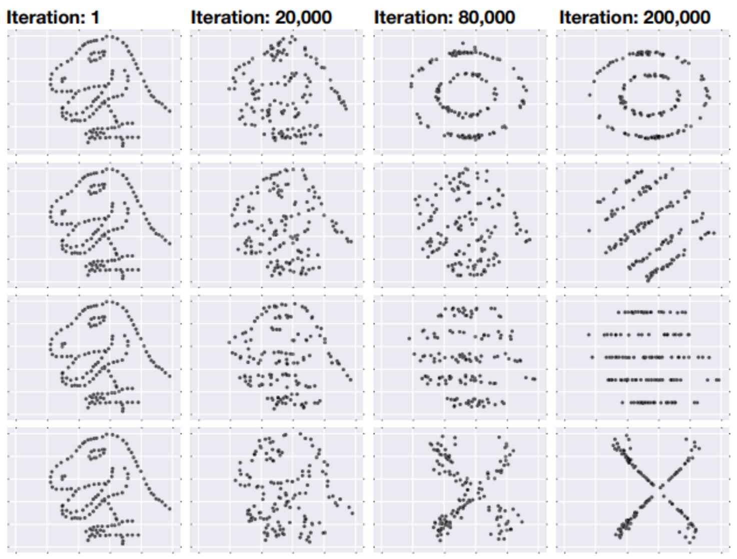


Figure 3.4: Creating a collection of datasets based on the "dinosaurus" dataset. Each dataset has the same summary statistics to two decimal places: (x=54.26, y = 47.83, sdx=16.76, sdy = 26.93, Pearson's r = -0.06)

Alberto Cairo produced a dataset called the "Datasaurus". Like Anscombe's Quartet, this serves as a reminder to the importance of visualizing your data, since, although the dataset produces "normal" summary statistics, the resulting plot is a picture of a dinosaur. In this example, we use the "datasaurus" as the initial dataset, and create other datasets with the same summary statistics (10).

3.4 1D Boxplots

To demonstrate the applicability of our approach to non 2Dscatterplot data, this example uses a 1D distribution of data as represented by a boxplot. The most common variety of boxplot, the "Tukey Boxplot", presents the 1st quartile, median, and 3rd quartile values on the "box", with the "whiskers" showing the location of the furthest datapoints within 1.5 interquartile ranges (IQR) from the 1st and 3rd quartiles. Starting with the data in a normal distribution and perturbing the data to the left (B), right (C), edges (D, E), and arbitrary points along the range (F) while ensuring that the boxplot statistics remain constant (10).

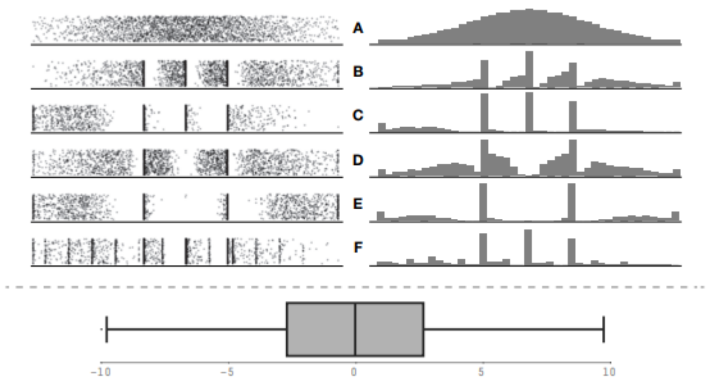


Figure 3.5: Six data distributions, each with the same 1st quartile, median, and 3rd quartile values, as well as equal locations for points 1.5 IQR from the 1st and 3rd quartiles. Each dataset produces an identical boxplot

Chapter 4

Result and Conclusion

In our experiments,we used Anscombe's four initial datasets as input data to our algorithm. Our experiments reveal that skewness and maximum of Cook's D statistic differentials consistently performed well and produced dissimilar graphics. Kolmogorov–Smirnov test differentials however did not consistently produce dissimilar scatterplots. These measures, however, were still useful when combined with other measures

Anscombe data retain their well-earned place in statistics and serve as a starting point for a more general approach to generate other datasets with identical summary statistics but very different scatter plots. It follows the principle that, before attempting to interpret and model the data or implement any machine learning algorithm, we first need to visualize the data set in order to help build a well-fit model. With this work, we provided a general procedure to generate similar data sets with an arbitrary number of independent variables that can be used for instructional and experimental purposes.

Chapter 5

Acknowledgement

We would like to express our sincere gratitude to all those who have contributed to the completion of this project titled "Visualizing Data: Beyond Anscombe's Number".

We are grateful to our course instructor, **Prof. Dr. Kanhaiya Jha** for his guidance and expertise throughout the project. His valuable feedback and suggestions have greatly enriched the content and analysis presented.

Last but not least, we would like to express our appreciation to the academic institution that supported and provided resources for this study, enabling to delve into the intricacies of statistics and share the findings along with the implementation of R programming.

Bibliography

[1] https://towardsdatascience.com/importance-of-data-visualization-anscombes-quartet-way-a325148b9fd2

[2] https://builtin.com/data-science/anscombes-quartet

[3] https://www.geeksforgeeks.org/anscombes-quartet/

[4] Chatterjee, S. and Firat, A., 2007. Generating data with identical statistics but dissimilar graphics: a follow up to the Anscombe Dataset. The American Statistician, 61(3), pp.248-254.

[5] https://www.linkedin.com/pulse/anscombes-quartet-unravels-importance-data-amitraj-yadmal/

[6] https://medium.com/analytics-vidhya/anscombes-quartet-an-importance-of-data-visualization-856b3d1bd403

[7] https://rstudio-pubs-static.s3.amazonaws.com/5238136ec82827e4b476fb968d9143aec7c4f.html

[8] https://en.wikipedia.org/wiki/Anscombe'squartet

[9] https://www.tutorialspoint.com/geneticalgorithms/geneticalgorithmsintroduction.html

[10] https://www.research.autodesk.com/app/uploads/2023/03/same-stats-different-graphs.pdfrec2hRjLLGgM7Cn2T.pdf

[11] https://arize.com/blog-course/kolmogorov-smirnov-test/

[12] https://www.investopedia.com/terms/s/skewness.asp

[13] https://www.statisticshowto.com/cooks-distance/