Advances in Artificial Intelligence for Data Visualization: Developing Computer Vision Models to Automate Reading of Data Plots, with Application to Predictive Model Diagnostics

Weihao Li

B.Comm. (Hons), Monash University

A milestone report submitted for the degree of PhD at

Monash University in 2022

Department of Econometrics and Business Statistics

Contents

Chapter 1

Background and Motivation

1.1 Predictive Modelling and Visual Diagnostics

Donoho (2017) in its summary of data science stated that the concept of the predictive modelling culture could be traced back to an article written by Breiman (2001). In contrast to the generative modelling culture, which aims to develop stochastic models to make inferences about the data generating process, predictive modelling emphasizes the ability of the model to make accurate predictions.

Predictive models are primarily evaluated by predictive accuracy with the use of validation and test data, but in predictive model diagnostics, especially model testing and tuning, data plots play an irreplaceable role. In these diagnostics, though numeric summaries are mostly available and some are even endorsed by finite or asymptotic properties, graphical representation of data is still preferred, or at least needed by researchers, due to its intuitiveness and the possibility for unexpected discoveries which may be abstract and unquantifiable.

However, unlike confirmatory data analysis built upon rigorous statistical procedures, e.g., hypothesis testing, visual diagnostics relies on graphical perception - human's ability to interpret and decode the information embedded in the graph (Cleveland and McGill, 1984), which is to some extent subjective. Further, visual discovery suffers from its unsecured and unconfirmed nature where the degree of the presence of the visual features

typically can not be measured quantitatively and objectively, which may lead to over or under-interpretations of the data. One such example is finding an over-interpretation of the separation between gene groups in a two-dimensional projection from a linear discriminant analysis when in fact there are no differences in the expression levels between the gene groups and separation is not an uncommon occurrence (Roy Chowdhury et al., 2015).

1.2 Visual Inference

Visual inference was first introduced in a 1999 Joint Statistical Meetings (JSM) talk with the title "Inference for Data Visualization" by Buja, Cook, and Swayne (1999) as an idea to address the issue of valid inference for visual discoveries of data plots (Gelman, 2004). Later, in the Bayesian context, data plots was systematically considered as model diagnostics by taking advantage of the data simulated from the assumed statistical models (Gelman, 2003, 2004).

It was surprising that the essential components of visual inference had actually been established in Buja, Cook, and Swayne (1999), but it was not until 10 years later that Buja et al. (2009) formalized it as an inferential framework to extend confirmatory statistics to visual discoveries. This framework redefines the test statistics, tests, null distribution, significance levels and p-value for visual discovery modelled on the confirmatory statistical testing. Figure 1.1 outlines the parallelism between conventional tests and visual discovery.

In visual inference, a visual discovery is defined as a rejection of a null hypothesis, and the same null hypothesis can be rejected by many different visual discoveries (Buja et al., 2009). For model diagnostics, the null hypothesis would be the assumed model, while the visual discoveries would be any findings that are inconsistent with the null hypothesis. The same assumed model, such as classical linear regression model, can be rejected by many reasons with residual plot, including non-linearity and heteroskedasticity as shown in Figure 1.2.

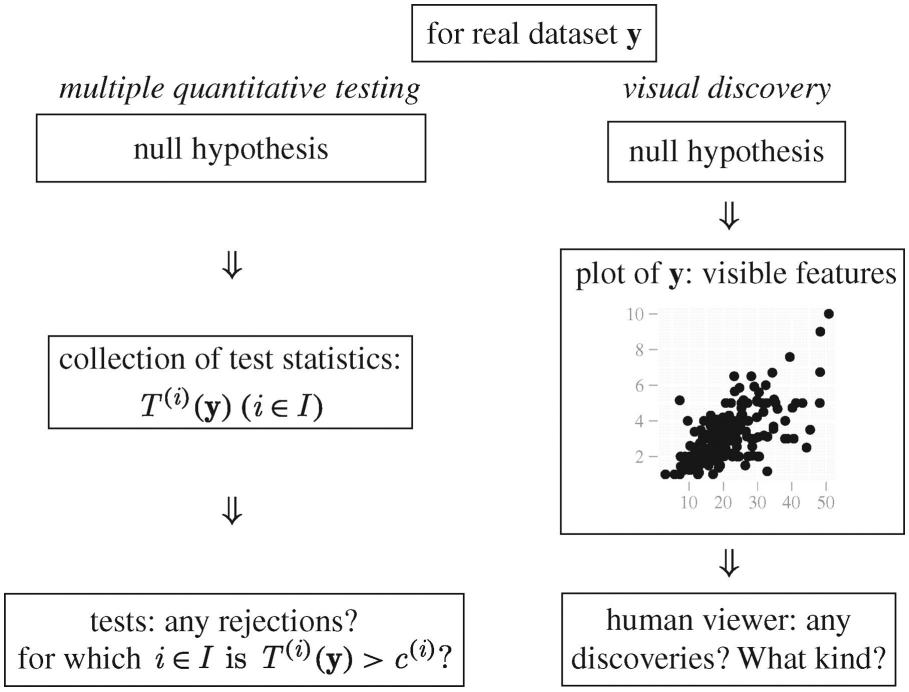


Figure 1.1: *Parallelism between multiple quantitative testing and visual discovery (Buja et al., 2009). Visible features in a plot are viewed as a collection of test statistics $T^{(i)}(\mathbf{y})$ $(i \in I)$, and any visual discoveries that are inconsistent with the null hypothesis are treated as evidence against the null. For model diagnostics, the null hypothesis would be the assumed model, and visual discoveries could be any visual features in favour of any alternatives.*

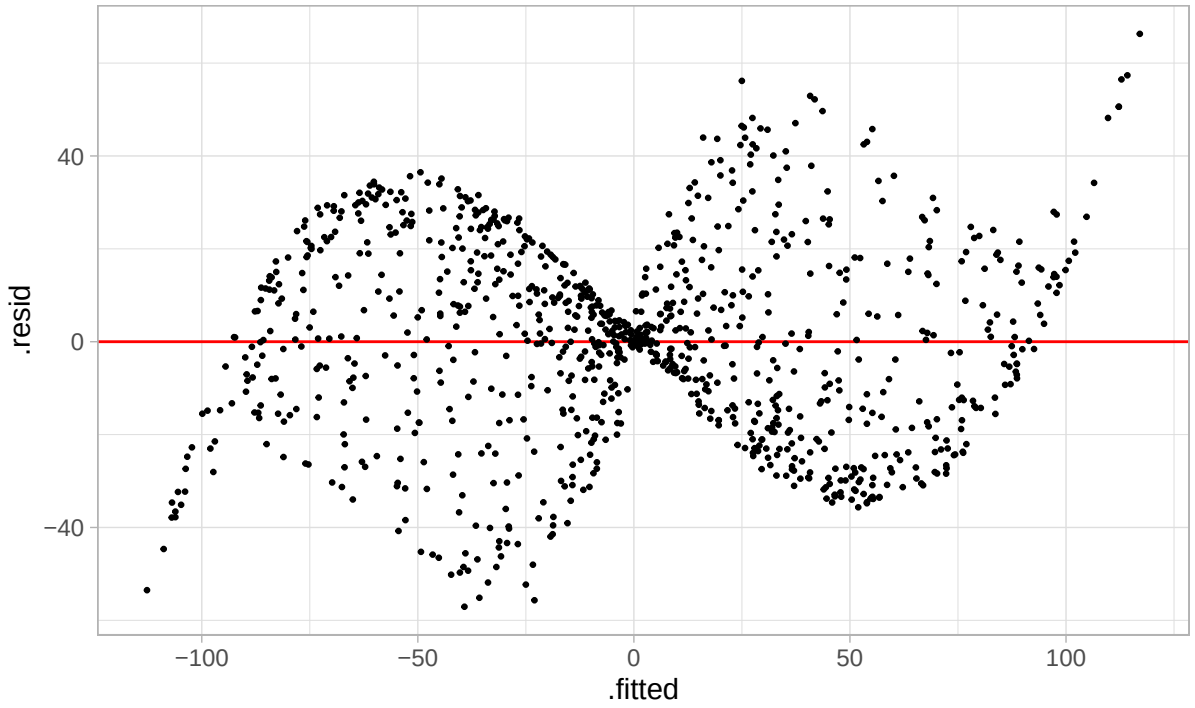


Figure 1.2: *Residuals vs. fitted values plot for a classical linear regression model. The residuals are produced by fitting a two-predictor multiple linear regression model with data generated from a cubic linear model. From the residual plot, "butterfly shape" can be observed which generally would be interpretd as evidence of heteroskedasticity. Further, from the outline of the shape, nonlinear patterns exist. Both visual discoveries are evidence against the null hypothesis, though heteroskedasticity actually does not exist in the data generating process.*

1.3 Pre-specification of Visual Discoverable Features

As discussed in Buja et al. (2009), in the practice of model diagnostics, the range of possible visual discoveries is not pre-specified. In other words, people do not explicitly specify which visual feature(s) they are looking for before the reading the diagnostic plot. In contrary, conventional hypothesis testing always requires the pre-specification of the parameter space Θ of the parameter of interest $\theta \in \Theta$ to form a valid inferential procedure. To address this issue, a collection of test statistics $T^{(i)}(\mathbf{y})\ (i \in I)$ is defined, where $\mathbf{y}$ is the data and I is a set of all possible visual features. Buja et al. (2009) described each of the test statistics $T^{(i)}(\mathbf{y})$ as a measurement of the degree of presence of a visual feature. Alternatively, Majumder, Hofmann, and Cook (2013) avoids the use of visual features and defined the visual statistics $T(.)$ as a mapping from a dataset to a data plot. Both definitions of visual test statistics are valid, but in the rest of the report the first definition will be used as it covers some details needed by the following discussion.

The size of the collection $T^{(i)}(\mathbf{y})\ (i \in I)$ depends on the size of the set I. Thus, if one can define I comprehensively, i.e, pre-specify all the visual discoverable features, the validity issue will be solved. Unfortunately, to our knowledge, there is no such a way to list all visual features. In linear regression diagnostics, possible visual features of a residual plot may be outliers, shapes and clusters. But this is an incomplete list which does not enumerate all the visual features.

Similarly, Wilkinson, Anand, and Grossman (2005) proposed the work called graph theoretic scagnostics, which adopted the idea of "scagnostics" - scatter plot diagnostics from John and Paul Tukey. It includes 9 computable scagnostics measures defined on planar proximity graphs: "Outlying", "Convex", "Skinny", "Stringy", "Straight", "Monotonic", "Skewed", "Clumpy" and "Striated" which attempts to describe outliers, shape, density, trend and coherence of the data. This approach is inspiring but it still does not give the complete list of visual discoverable features. In fact, it is possible that such a list will never be complete as suggested in Buja et al. (2009).

Thinking out of the box, Buja et al. (2009) argued that there is actually no need for pre-specification of visual discoverable features. In model diagnostics, when the null

hypothesis is rejected, the reasons for rejecting the hypothesis will also be known. This is because observers can not only point out the fact that visual discoveries have been found, but also describe the particular visual features they observed. Those features will correspond to the subset of the collection of visual test statistics $T^{(i)}(\mathbf{y})$ $(i \in I)$ which resulted in rejection. This argument helps justifies the validity of visual inference.

1.4 Sampling from the Null Distribution

In visual inference, the null distribution of plots refers to the infinite collection of plots of null datasets sampled from H_0. It is defined as the analogue of the null distribution of test statistics in conventional test (Buja et al., 2009). In practice, a finite number of plots of null datasets could be generated, called null plots.

In the context of model diagnostics, sampling data from H_0 is equivalent to sampling data from the assumed model. As Buja et al. (2009) suggested, H_0 is usually composited by a collection of distributions controlled by nuisance parameters. Since statistical models can have various forms, there is no general solution to this problem, but it sometimes can be reduced to so called "reference distribution" by applying one of the three methods: (i) sampling from a conditional distribution given a minimal sufficient statistic under H_0, (ii) parametric bootstrap sampling with nuisance parameters estimated under H_0, and (iii) Bayesian posterior predictive sampling.

The conditional distribution given a minimal sufficient statistic is the best justified reference distribution among the three (Buja et al., 2009). Suppose there exists a minimal sufficient statistic $\boldsymbol{S}(\boldsymbol{y})$ under the null hypothesis, any null datasets $\boldsymbol{y}^*$ should fulfil the condition $\boldsymbol{S}(\boldsymbol{y}) = \boldsymbol{s}$. Using the classical linear regression model as example, the minimal sufficient statistic is $\boldsymbol{S}(\boldsymbol{y}) = (\hat{\boldsymbol{\beta}}, \boldsymbol{e}'\boldsymbol{e})$, where $\hat{\boldsymbol{\beta}}$ are the coefficient estimators and $\boldsymbol{e}'\boldsymbol{e}$ is the residual sum of square. Alternatively, the minimal sufficient statistic can be constructed as $\boldsymbol{S}(\boldsymbol{y}) = (\hat{\boldsymbol{y}}, ||\boldsymbol{e}||)$, where $\hat{\boldsymbol{y}}$ are the fitted values and $||\boldsymbol{e}||$ is the length of residuals, which is more intuitive as suggested by Buja et al. (2009). Since the fitted values are held fixed, the variation can only occur in the residual space. And because the length of residual is also held fixed, residuals obtained from a null dataset has to be a random rotation of $\boldsymbol{e}$ in the residual space. With this property, null residuals can be simulated by regressing N i.i.d

standard normal random draws on the regressors, then rescaling it by the ratio of residual sum of square in two regressions.

1.5 Lineup Protocol

With the validity of visual inference being justified and the simulation of null plots being provided, another aspect of hypothesis testing that needs to be addressed is the control of false positive rate or Type I error. Any visual statistic $T^{(i)}(\mathbf{y})$ needs to pair with a critical value $c^{(i)}$ to form a hypothesis test. When a visual feature i is discovered by the observer from a plot, the corresponding visual statistic $T^{(i)}(\mathbf{y})$ may not be known as there is no general agreement on the measurement of the degree of presence of a visual feature. It is only the event that $T^{(i)}(\mathbf{y}) > c^{(i)}$ is confirmed. Similarly, if any visual discovery is found by the observer, we say, there exists $i \in I : \ T^{(i)}(\mathbf{y}) > c^{(i)}$ (Buja et al., 2009).

Using the above definition, the family-wise Type I error can be controlled if one can provide the collection of critical values $c^{(i)}$ $(i \in I)$ such that $P(\text{there exists } i \in I : T^{(i)}(\mathbf{y}) > c^{(i)}|\mathbf{y}) \leq \alpha$, where α is the significance level. However, since the quantity of $T^{(i)}(\mathbf{y})$ may not be known, such collection of critical values can not be provided.

Buja et al. (2009) proposed the lineup protocol as a visual test to calibrate the Type I error issue without the specification of $c^{(i)}$ $(i \in I)$. It is inspired by the "police lineup" or "identity parade" which is the act of asking the eyewitness to identify criminal suspect from a group of irrelevant people. The protocol consists of m randomly placed plots, where one plot is the actual data plot, and the remaining $m - 1$ plots have the identical graphical production as the data plot except the data has been replaced with data consistent with the null hypothesis. Then, an observer who have not seen the actual data plot will be asked to point out the most different plot from the lineup.

Under the null hypothesis, it is expected that the actual data plot would have no distinguishable difference with the null plots, and the probability of the observer correctly picks the actual data plot is $1/m$. If we reject the null hypothesis as the observer correctly picks the actual data plot, then the Type I error of this test is $1/m$.

This provides us with an mechanism to control the Type I error, because m - the number of plots in a lineup can be chosen. A larger value of m will result in a smaller Type I error, but the limit to the value of m depends on the number of plots a human is willing to view (Buja et al., 2009). Typically, m will be set to 20 which is equivalent to set $\alpha = 0.05$, a general choice of significance level for conventional testing among statisticians.

Further, if we involve K independent observers in a visual test, and let X be a random variable denoting the number of observers correctly picking the actual data plot. Then, under the null hypothesis $X \sim \text{Binom}_{K,1/m}$, and therefore, the p-value of a lineup of size m evaluated by K observer is given as

$$P(X \geq x) = \sum_{i=x}^{K} \binom{K}{i} \left(\frac{1}{m}\right)^{i} \left(\frac{m-1}{m}\right)^{k-i}, \tag{1.1}$$

where x is the realization of number of observers correctly picking the actual data plot (Majumder, Hofmann, and Cook, 2013).

The multiple individuals approach avoids the limit of m, while provides visual tests with p-value much smaller than 0.05. In fact, the lower bound of p-value decreases exponentially as K increases. With just 4 individuals and 20 data plots in a lineup, the p-value could be as small as 0.0001. Additionally, by involving multiple observers, variation of individual ability to read plots can be addressed to some degree as different opinions about visual discoveries can be collected.

Compared to the conventional test, whose power only depends on the parameter of interest θ, several studies (see Hofmann et al., 2012; Majumder, Hofmann, and Cook, 2013, 2014; Roy Chowdhury et al., 2015; Loy, Follett, and Hofmann, 2016) have shown the power of the visual test is subject-specific. Thus, to be able to account for individual's ability, an individual is required to evaluate multiple lineups (Majumder, Hofmann, and Cook, 2013).

Suppose individuals have the same ability and a lineup has been evaluated by multiple individuals, under the alternative hypothesis, the estimated power for a lineup can be expressed as $\hat{p} = x/K$, the estimated probability of identifying the actual data plot from

the lineup. If the individual skill needs to be taken into account, and L lineups have been evaluated by K individuals, Majumder, Hofmann, and Cook (2013) suggests that mixed effects logistic regression model can be fit as:

$$g(p_{li}) = W_{li}\delta + Z_{li}\tau_{li},$$

where $g(.)$ is the logit link function $g(p) = log(p) - log(1-p); 0 \leq p \leq 1$. $W_{li}, 1 \leq i \leq K$, $1 \leq l \leq L$, is the covariate matrix including lineup-specific elements and demographic information of individuals, and δ is a vector of parameters. Z is the random effects matrix, and τ is a vector of variables follow $N(\mathbf{0}, \sigma_\tau \boldsymbol{I}_{KL\times KL})$.

Then, the estimated power for lineup l and individual i can be calculated as $\hat{p}_{li} = g^{-1}(W_{li}\hat{\delta} + Z_{li}\hat{\tau}_{li})$ (Majumder, Hofmann, and Cook, 2013).

1.6 Limitations of Lineup Protocol and Automatic Visual Inference

Although lineup protocol has already been integrated into data analysis of various topics, such as diagnostics of hierarchical linear models (Loy and Hofmann, 2013), geographical Research (Widen et al., 2016) and forensic examinations (Krishnan and Hofmann, 2021), the involvement with human judgements limits its popularity. Similar to handicraft in pre-industrial society, lineup protocol conducted by humans is infeasible on a large scale, high in labour cost and time consuming. Moreover, it is strongly unfriendly to vision-impaired people.

The "steam engine" of visual inference needs to be developed for relieving people's workload by automating repeating tasks and providing standard result in a control environment. Large-scale evaluation of visual tests is not possible without the use of technology and machines.

Modern computer vision model could be a promising solution to this problem. As a sub-field of AI, computer vision with the modern deep learning architectures solved numerous critical problems in automation. Inspired by the vision processing in living organisms,

the convolutional neural network (CNN) was introduced by Fukushima and Miyake (1982). Soon, this architecture was applied to hand-written number recognition trained with back-propagation by LeCun et al. (1989). This was one of the earliest attempts which human successfully extract information from digital images via self-learning algorithms. Modern computer vision model is typically built on the deep neural network with convolutional layers (Fukushima and Miyake, 1982). Convolutional layers take advantage of the hierarchical pattern in data and provide regularized versions of fully-connected layers. It downscales and transforms the image by summarising information in a small space. Numerous studies have shown that it can be used to effectively tackle vision tasks, such as image recognition (Rawat and Wang, 2017). With the development of graphics processing units and the spread of high-performance personal computers, researches in computer vision become a new hype in the 21st century. Achievements such as computer-aided diagnosis (Lee and Chen, 2015), pedestrian detection (Brunetti et al., 2018) and facial recognition (Emami and Suciu, 2012) had a significant impact on our daily life.

Using computer vision models to read data plots is not a general choice. some fields have adopted this idea by applying computer vision models to read recurrence plots for time series regression (Ojeda, Solano, and Peramo, 2020), time series classification (Chu et al., 2019; Hailesilassie, 2019; Hatami, Gavet, and Debayle, 2018; Zhang et al., 2020), anomaly detection (Chen, Su, and Yang, 2020) and pairwise causality analysis (Singh et al., 2017). However, evaluating lineups with computer vision model is a new field of study.

1.7 Structure of the Report

The remainder of the report is organised as follows. Chapter 2 outlines the three research problems. Chapter 3 gives an overview of the thesis including descriptions of three projects. Chapter 4 presents the prototype of the automatic visual inference system along with the results of two human subject experiments. Chapter 5 introduces the general-purpose automatic visual inference system. Chapter 6 describes the deployment of the automatic system. Chapter 7 presents the timeline of the three projects.

Chapter 2

Research Problem

The main objective of this research is to build an automatic visual inference system for the purpose of conducting large-scale visual test. The research will focus on three projects as follows:

1. Develop a prototype of automatic visual inference system for evaluating lineups of residual plots of the classical normal linear regression model and study factors that affect the performance of the system with comparison to human subjects.
2. Extend and enhance the automatic visual inference system developed in the first project to general lineup protocol problems by adopting image similarity assessed by computer vision models.
3. Deploy the automatic visual inference system as an adaptive online application by applying online machine learning.

Chapter 3

Overview of the Thesis

The first project develops a prototype of automatic visual inference system for evaluating lineups of residual plots, with a focus on the classical normal linear regression model given it is one of the simplest predictive models. The automatic system is identical to the lineup protocol except evaluators are replaced by computer vision models. The computer vision models are trained by using data simulated from linear models with violations of different classical assumptions, such as linearity and homoscedasticity. Data of human performance on evaluating residual plots generated under the same simulation setting is collected by conducting online human subject experiments, such that the comparison between the power of human subject-assessed visual tests and the system-assessed visual tests can be made. Moreover, factors that affect the performance of the automatic system are studied for improving the architecture and design of the computer vision models.

The second project aims to extend the automatic system to lineup protocol with no assumptions on the type of the data plot and the data generating process. It only requires the protocol to asks the evaluators to select the most different plot(s) from the lineup. The automatic system will be built upon image similarity produced by computer vision models.

The third projects aims to package and deploy the automatic system as an adaptive online application such that empirical data can be collected and used in improving the system.

Online machine learning technique will be used to update the weights of computer vision models dynamically.

Chapter 4

Automatic Visual Statistical Inference for linear regression diagnostics

This project focuses on building a prototype of automatic visual statistical inference system for evaluating residual plots of classical normal linear regression model. To set up a comparison between the computer vision models and humans, human subject experiments were conducted to understand the ability of human reading residual plots.

Because the first project is still ongoing, we consider the result we have is not enough to write as a paper at the moment. Hence, the materials provided in this chapter are two main parts of the draft paper that will contain enough details for understanding the project.

4.1 Human Subject Experiments

To collect data of human performance on reading residual plot of linear regression model with non-linearity and heteroskedasticity defects, two human experiments were conducted. Participants of both experiments were recruited using an online platform called Prolific (Prolific, 2022).

Prolific provides an international participant pool with the option to apply flexible pre-screening filters. In this study, we recruited 62 participants who was fluent in English with at least 10 previous submissions and 98% approval rate in other Prolific studies for quality control. Further, balance sample across gender was imposed to prevent gender bias.

In Prolific, researchers can either approve or reject submissions based on the quality of the responses. If a submission is approved by the researcher, the participant will be paid a certain amount of money per hour of time spent on the experiment. To assess the quality of the responses, two attention checks were given to each participant during the experiment, where at least one of them was required to pass for the approval of submission.

Throughout the experiments, participants were requested to complete a short survey and evaluate 20 lineups on a website in an hour. Each lineup consists of one actual residual plot and 19 null residual plots produced by plotting null residuals simulated from the residual rotation distributions. Among the 20 lineups, there were two extremely easy lineups used as attention checks which everyone should get correct. The short survey was intended to collect information about participant that might affect their ability in reading data plot including age, highest level of education, preferred pronoun and previous experience in similar study. For the evaluation of the lineup, participant first needed to select one or multiple most different data plot(s) from the lineup by clicking the corresponding image. Then, the primary reason for choosing the plots needed to be provided by picking one of the given options - “outlier(s)”, “cluster(s)”, “shape” and “other”. Explanations about these options were provided in the training page and the mouseover text. Table 4.1 gives the detailed explanations about these reasons. If the option “other” was selected, text input for the specified reason would be collected. Lastly, the degree of difference between their chosen data plot(s) and other data plots needed to be selected among five levels - “not at all”, “slightly”, “moderately”, “very” and “extremely”. Notably, if participants could not tell the difference between the data plots, there was an option to skip the evaluation of the lineup. However, to prevent participants abusing this option, warnings were given at the beginning of the study that skipping too many lineups might lead to rejection of submission since it demonstrated clear low-effort throughout the experiment.

Table 4.1: *Explanations about reasons for choosing data plots from a lineup*

Reasons	Explanations
outlier(s)	In a data plot, an outlier is a point that differs significantly from other points.
cluster(s)	In a data plot, a cluster is a group of points positioned closely together. And usually, there will be gaps between different clusters.
shape	It could be any common shapes we would see in real life, like triangle shape, U-shape, butterfly shape, etc.
other	Patricipants needs to give their reasons in the text input.

The study website was powered by Flask (Grinberg, 2018), a web framework written in Python 3 (Van Rossum and Drake, 2009), hosted on PythonAnywhere (PythonAnywhere, 2022), a web hosting service provider. Due to the limit of the storage on PythonAnywhere, lineup images needed for the website were hosted separately on Github Pages (Gtihub, 2022b), a static site hosting service provided by Github (Gtihub, 2022a), stored in private Github repositories. The uniform resource locator (URL) to lineup images were set to be unique random strings such that images can not be accessed by general approaches without knowing the correct URL. This avoided participants seeing the lineup beforehand. The front-end of the website was built using a JavaScript (Flanagan, 2006) library jsPsych (De Leeuw, 2015) which is specialized in creating online behavioural experiments. One of the reasons we chose to use this library was it had a modularized but highly customizable template which could record participant's response time automatically. This is essential for us to confirm the quality of the data by checking exceptionally fast or slow responses. Figure 4.1 summarizes the online experiment set-up.

The first two pages of the website are the explanatory statement and the consent form. Participants were asked to read the documents and agree with the terms to advance to the short survey. With the completion of the survey, the next page is the training page, which contained instructions on how to interact with the webpage. Followed by the training page, there are 20 lineups each on a single page. Figure 4.2 illustrates the layout of the website. At the end of the experiment, participants would be redirected back to Prolific and waited for researchers' responses.

Next we will discuss the simulation set-up for this study. The experiment data was simulated by the use of the programming language `R` (R Core Team, 2021). For the ease of reproducibility, functions to build models, simulate data from models, produce lineup with data, allocate stimulus for subjects and evaluate subject responses from this study are bundled in the package `visage` with a unique object-oriented programming system built

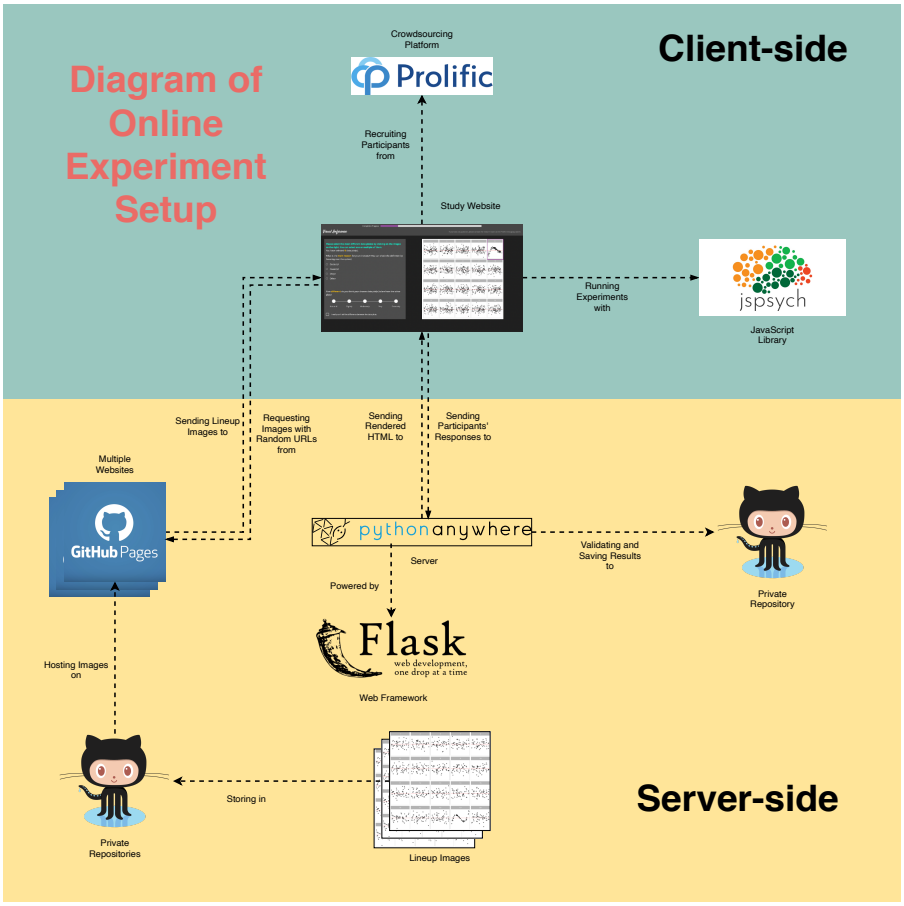


Figure 4.1: *Diagram of online experiment setup. The server-side of the study webiste used Flask as backend hosted on PythonAnywhere. And the client-side used jsPsych to run experiment.*

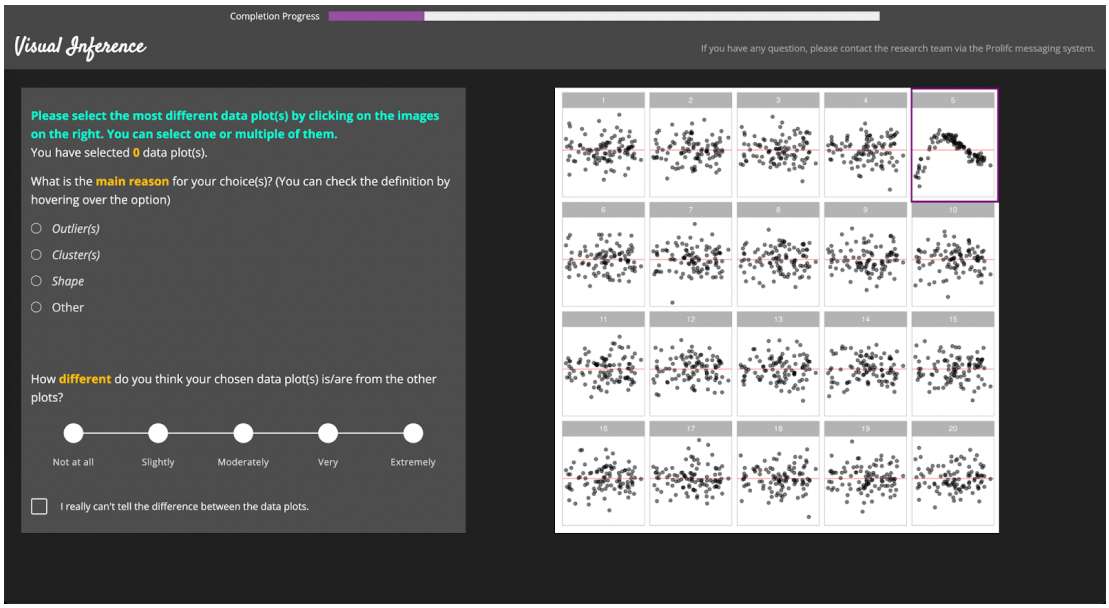


Figure 4.2: *Layout of the study website. Participants needed to choose the most different plots on the right and select their reasons and confidence levels on the left.*

upon the environment feature of R. In the description of the simulation, corresponding functionalities of the package will be introduced.

Two models were used in the study, where both were linear models with some degree of violations of classical assumptions.

4.1.1 Cubic Model

The first model was a cubic linear model with two regressors, which can be expressed by:

$$\boldsymbol{Y} = 1 + (2-c)\boldsymbol{X} + c\boldsymbol{Z} + a[(2-c)\boldsymbol{X}]^2 + a(c\boldsymbol{Z})^2 + b[(2-c)\boldsymbol{X}]^3 + b(c\boldsymbol{Z})^3 + \boldsymbol{\varepsilon}, \tag{4.1}$$

where $c \in (0,2)$, $a \in (-3,3)$, $b \in (-3,3)$, $\boldsymbol{\varepsilon} \overset{iid}{\sim} N(\boldsymbol{0}, \sigma^2\boldsymbol{I})$, $\boldsymbol{Y}$, $\boldsymbol{X}$ and $\boldsymbol{Z}$ are $n \times 1$ matrices.

This defines a cubic relationship between $\boldsymbol{Y}$, $\boldsymbol{X}$ and $\boldsymbol{Z}$. Meanwhile, to create non-linearity defect, the null model followed the assumptions of the classical normal linear regression model (CNLRM), fitted by OLS is:

$$\boldsymbol{Y} = \beta_0 + \beta_1\boldsymbol{X} + \beta_2\boldsymbol{Z} + \boldsymbol{u}, \tag{4.2}$$

where $\boldsymbol{u} \sim N(0, \sigma_u^2\boldsymbol{I})$.

Clearly, omitted-variable bias will present since the null model leaves out the quadric and cubic terms.

Lemma 4.1 (Distribution of residuals produced by the cubic model)**.** *Given the data generating process in Equation (4.1), and null model in Equation (4.2). Let* $\boldsymbol{X}_a = [\boldsymbol{1}, \boldsymbol{X}, \boldsymbol{Z}]$ *denotes the set of regressors in matrix form. Then, the residuals obtained from the null model are*

$$\boldsymbol{e} \sim N(\boldsymbol{R}_a\boldsymbol{X}_b\boldsymbol{\beta}_b, \sigma^2\boldsymbol{R}_a),$$

where $\boldsymbol{R}_a = \boldsymbol{I} - \boldsymbol{X}_a(\boldsymbol{X}_a'\boldsymbol{X}_a)^{-1}\boldsymbol{X}_a'$, $\boldsymbol{X}_b = [\boldsymbol{X}^2, \boldsymbol{Z}^2, \boldsymbol{X}^3, \boldsymbol{Z}^3]$ *and* $\boldsymbol{\beta}_b = (a(2-c)^2, ac^2, b(2-c)^3, bc^3)'$.

Proof. Using the Frisch–Waugh–Lovell theorem, the residuals obtained by the null model are

$$\boldsymbol{e} = \boldsymbol{R}_a \boldsymbol{Y} = \boldsymbol{R}_a(\boldsymbol{X}_a \boldsymbol{\beta}_a + \boldsymbol{X}_b \boldsymbol{\beta}_b + \boldsymbol{\varepsilon}),$$

where $\boldsymbol{R}_a = \boldsymbol{I} - \boldsymbol{X}_a(\boldsymbol{X}_a'\boldsymbol{X}_a)^{-1}\boldsymbol{X}_a'$, $\boldsymbol{\beta}_a = (1, 2-c, c)'$, $\boldsymbol{X}_b = [\boldsymbol{X}^2, \boldsymbol{Z}^2, \boldsymbol{X}^3, \boldsymbol{Z}^3]$ and $\boldsymbol{\beta}_b = (a(2-c)^2, ac^2, b(2-c)^3, bc^3)'$.

Because $\boldsymbol{R}_a\boldsymbol{X}_a = \boldsymbol{0}$, we have $\boldsymbol{e} = \boldsymbol{R}_a(\boldsymbol{X}_b\boldsymbol{\beta}_b + \boldsymbol{\varepsilon})$. Since $\boldsymbol{\varepsilon} \sim N(0, \sigma^2\boldsymbol{I})$, it follows that $\boldsymbol{e} \sim N(\boldsymbol{R}_a\boldsymbol{X}_b\boldsymbol{\beta}_b, \sigma^2\boldsymbol{R}_a)$. □

Lemma 4.1 shows that the expectation of the residuals is clearly a function of $\boldsymbol{X}$ and $\boldsymbol{Z}$, which indicates the residuals and the fitted values $\hat{\boldsymbol{Y}} = \boldsymbol{X}_a(\boldsymbol{X}_a'\boldsymbol{X}_a)^{-1}\boldsymbol{X}_a'\boldsymbol{Y}$ are associated. Hence, it is expected that visual discoveries can be found in the residuals vs. fitted values plot. Let X_i and Z_i, $i = 1, ..., n$, be independent random variables follow uniform distribution $U(-1, 1)$. Given the expectation of the residuals, we could plot the expected values of residuals against the observed values. Figure 4.3, 4.4 and 4.5 illustrate the shape of residuals and their expected values under different parameter settings.

From the Figure 4.3, it can be observed that with fixed σ and c, a and b are controlling the 2D projection of a hypersurface, and seemingly performing some rotations along different axes. Figure 4.4 shows that with fixed a, b and σ, c is controlling the contribution of $\boldsymbol{X}$ and $\boldsymbol{Z}$ to $\boldsymbol{Y}$. As c moves toward 0 or 2, one regressor will dominate another, which will mitigate some joint effects and resemble a typical cubic function. In Figure 4.5, a, b and c are held fixed, and σ is controlling the noises around the expected values. The underlying shape is clearer for smaller σ.

The residuals used in these three figures are simulated from the cubic models built using the `cubic_model()` function from the package `visage`. `cubic_model()` is a cubic model class constructor, which takes arguments `a`, `b`, `c`, `sigma`, `x` and `z`, where the first four are numeric values defined above, and `x` and `z` are random variable instances created by the random variable abstract base class constructor `rand_var()`.

If we would like $\boldsymbol{X}$ and $\boldsymbol{Z}$ to be random uniform variables ranged from -1 to 1, it can be achieved by using the random uniform variable class constructor `rand_uniform()`

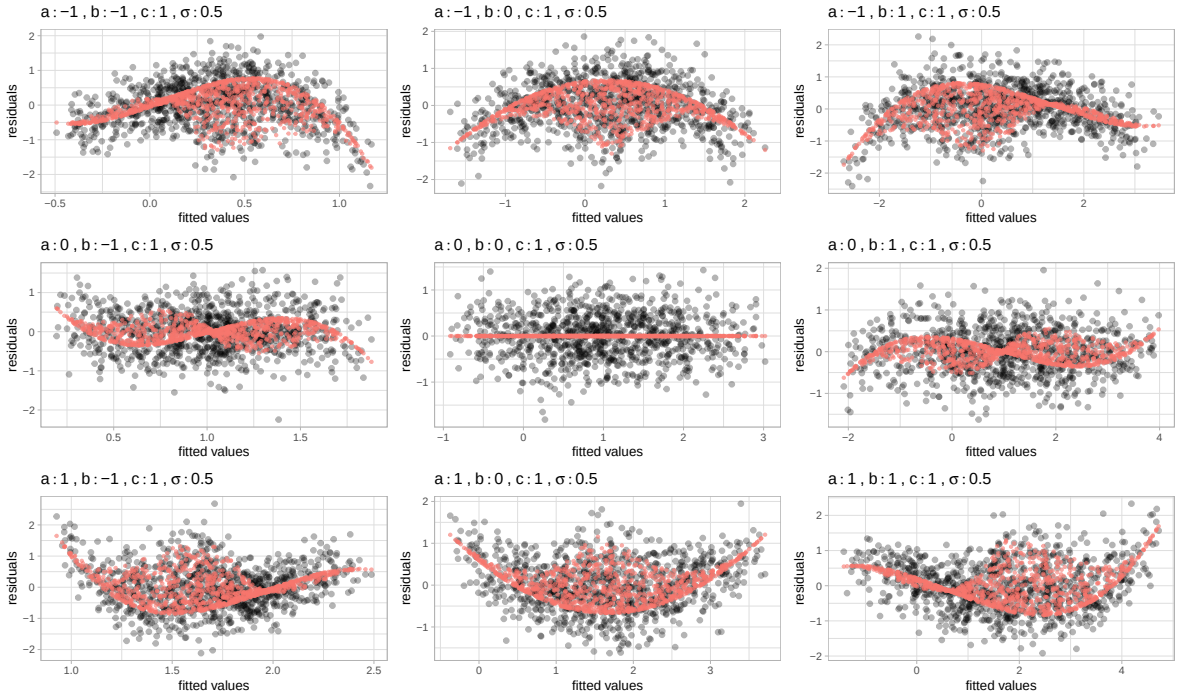


Figure 4.3: *A matrix of residuals vs. fitted values plot under different parameter settings of the cubic model. Resdiauls are drawns in black and the expected values are drawn in red. The plots show that if σ and c are held fixed, a and b rotate a two-dimension projection of a hypersurface.*

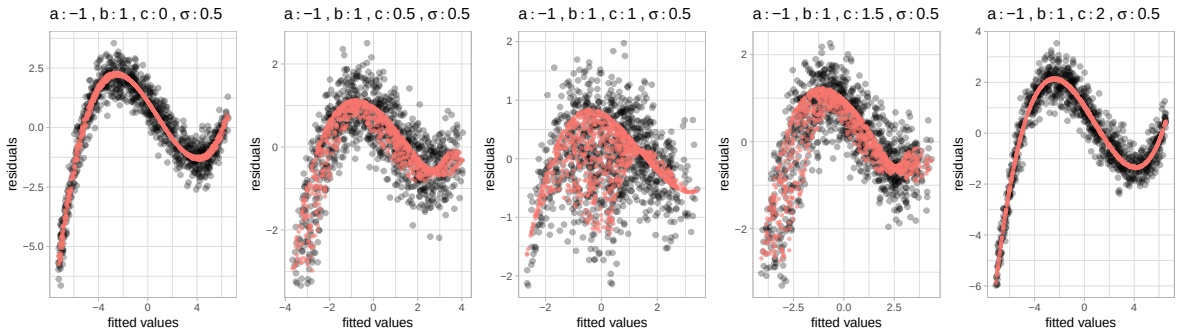


Figure 4.4: *A matrix of residuals vs. fitted values plot under different parameter settings of the cubic model. Resdiauls are drawns in black and the expected values are drawn in red. The plots show that if a, b and σ are held fixed, c controls the contribution of $\boldsymbol{X}$ and $\boldsymbol{Z}$ to $\boldsymbol{Y}$.*

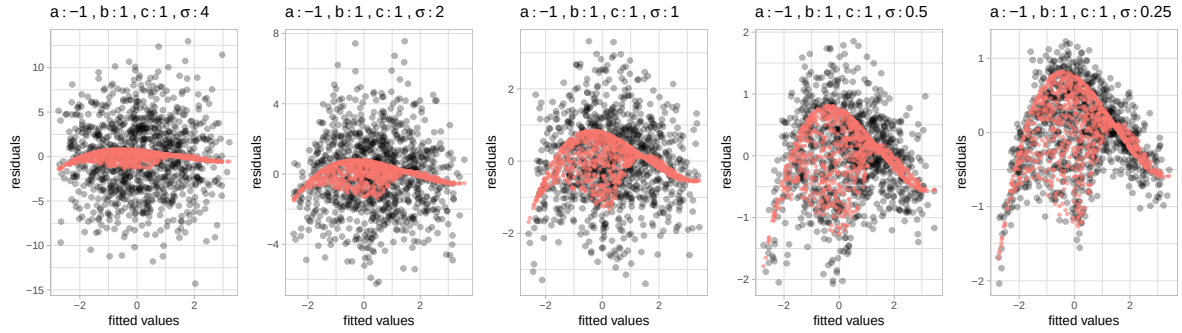


Figure 4.5: *A matrix of residuals vs. fitted values plot under different parameter settings of the cubic model. Resdiauls are drawns in black and the expected values are drawn in red. The plots show that if a, b and c are held fixed, σ controls the strength of the signal.*

inherited from the random variable abstract base class. It only takes two arguments which are the lower bound and the upper bound of the support.

```
library(visage)
mod <- cubic_model(a = -3, b = -3, c = 1, sigma = 0.5,
                   x = rand_uniform(-1, 1), z = rand_uniform(-1, 1))
```

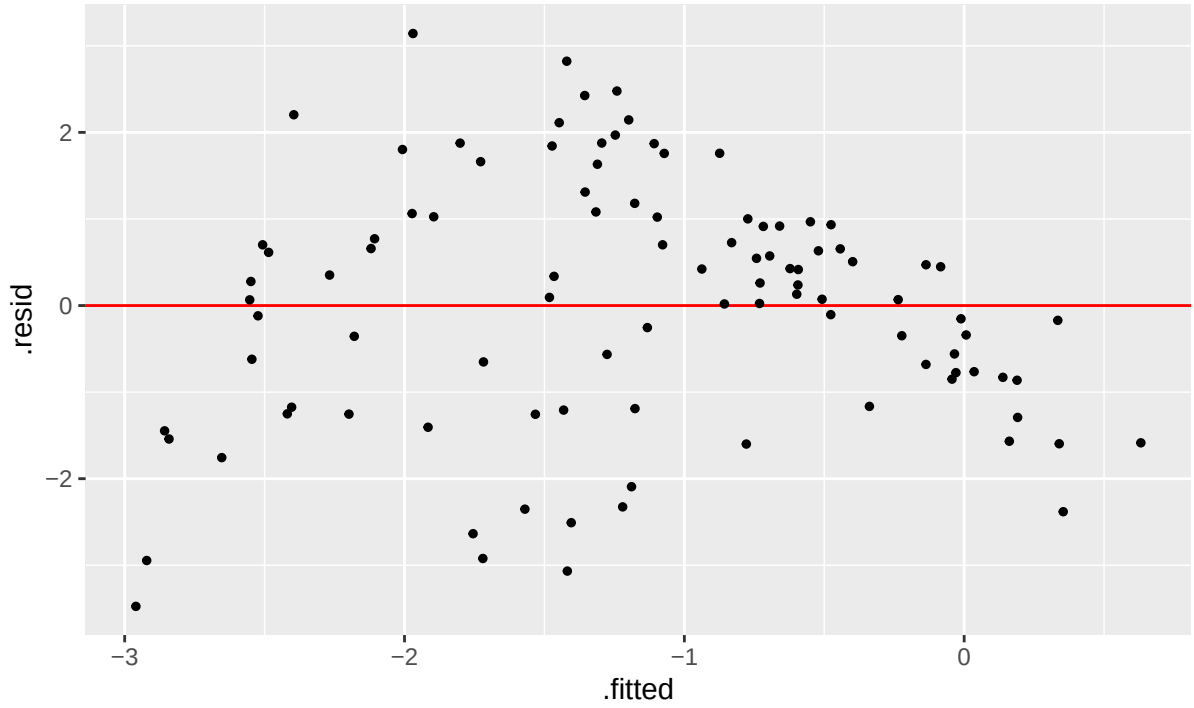
An instance of cubic model class contains methods of simulating data and making residual plot. Method `mod$gen()` returns a data frame containing realizations of X, Z, Y and ε simulated from the model. The number of realizations depends on the integer argument `n`. In addition, if argument `fit_model = TRUE`, a null model will be fitted using the simulated data and residuals and fitted values will be included in the returned data frame.

```
mod$gen(n = 5, fit_model = TRUE)
```

```
##            y          x          z           e     .resid    .fitted
## 1 -0.8263671 -0.4988275  0.5569648 -0.06142887  1.2340651 -2.0604322
## 2  0.6134661  0.1762335  0.6512424  0.99654342  1.1758970 -0.5624310
## 3  0.2285071 -0.2814395 -0.1511556 -0.10996690 -0.3757603  0.6042674
## 4 -4.6320382 -0.5813456  0.9251409 -0.60826338 -1.2664307 -3.3656075
## 5  0.3148180  0.5647967  0.4406211  0.64596536 -0.7677711  1.0825892
```

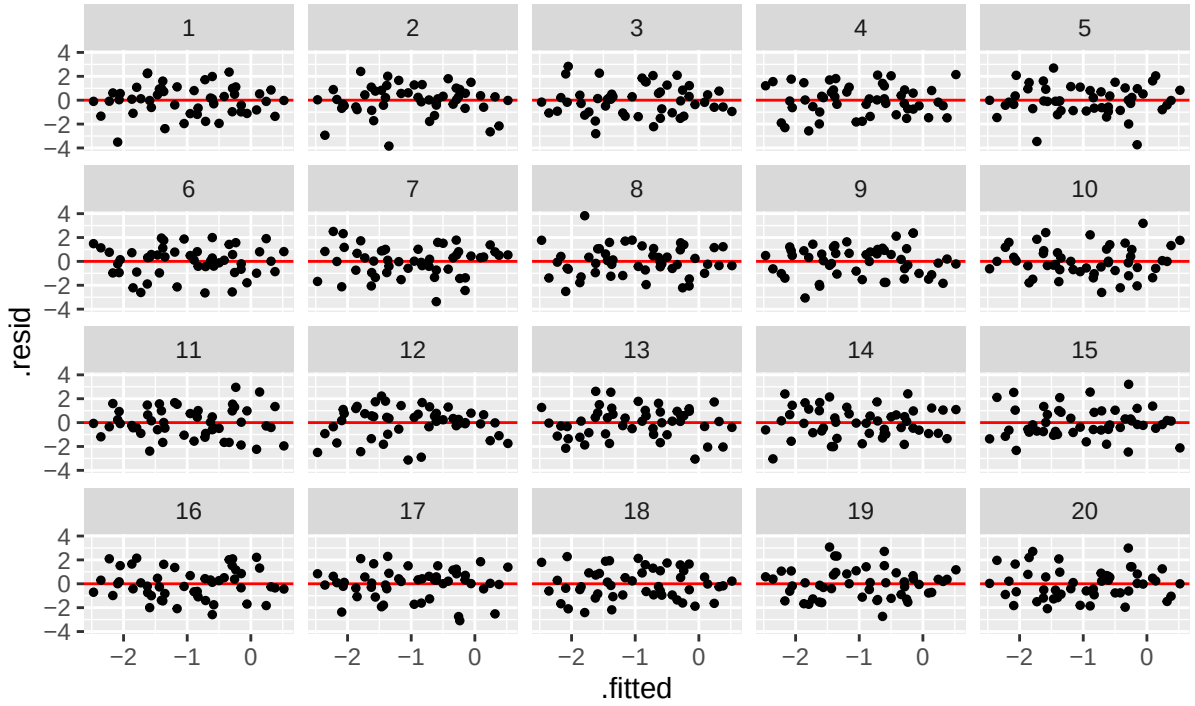
Method `mod$plot()` produce a `ggplot` (Wickham, 2011) object. It takes a data frame containing columns `.resid` and `.fitted` as input, along with a character argument `type` indicating the type of the data plot, and other aesthetic arguments such as `size` and `alpha` to control the appearance of the plot.

```
mod$plot(mod$gen(n = 100, fit_model = TRUE), type = "resid", size = 1)
```



Lineup is a matrix of residual plots which can be produced by using the methods `mod$gen_lineup()` and `mod$plot_lineup()`. Method `mod$gen_lineup` takes the number of realizations `n` and the number of plots in a lineup `k` as inputs. And the method `mod$plot_lineup()` has the same user interface as `mod$plot()`.

```
mod$plot_lineup(mod$gen_lineup(n = 50, k = 20), type = "resid", size = 1)
```



The cubic model class also provides method to compute the expected values of residuals. Method `mod$E()` takes a data frame with columns `x` and `z` as input, and returns a vector of expected values of residuals.

```
mod$E(mod$gen(n = 5))
```

```
## [1] -0.092183951 -0.003912092 -0.001531723  0.148353478 -0.050725712
```

Since we know that under the null hypothesis, the residual $\boldsymbol{e} \sim N(\mathbf{0}, \sigma^2 \boldsymbol{R}_a)$. Thus, the difference between the expected values $\boldsymbol{R}_a \boldsymbol{X}_b \boldsymbol{\beta}_b$ and $\mathbf{0}$ represents the direct impact of the parameters a and b on the residuals. It is expected that the larger the magnitude of the expected value relative to the variance and covariance, the easier the human to spot the patterns in the residual plot.

To obtain a measure of the impact of a and b on the residuals adjusted for variance and covariance, we need to address several properties of the residuals. First, the variance of the residuals $\sigma^2 \boldsymbol{R}_a$ is not an identity matrix. This can be fixed by standardizing the residuals by their variance-covariance matrix. Second, the difference between $\boldsymbol{R}_a \boldsymbol{X}_b \boldsymbol{\beta}_b$ and $\mathbf{0}$ could be negative, which is not ideal for comparison. Thus, the magnitude needs to

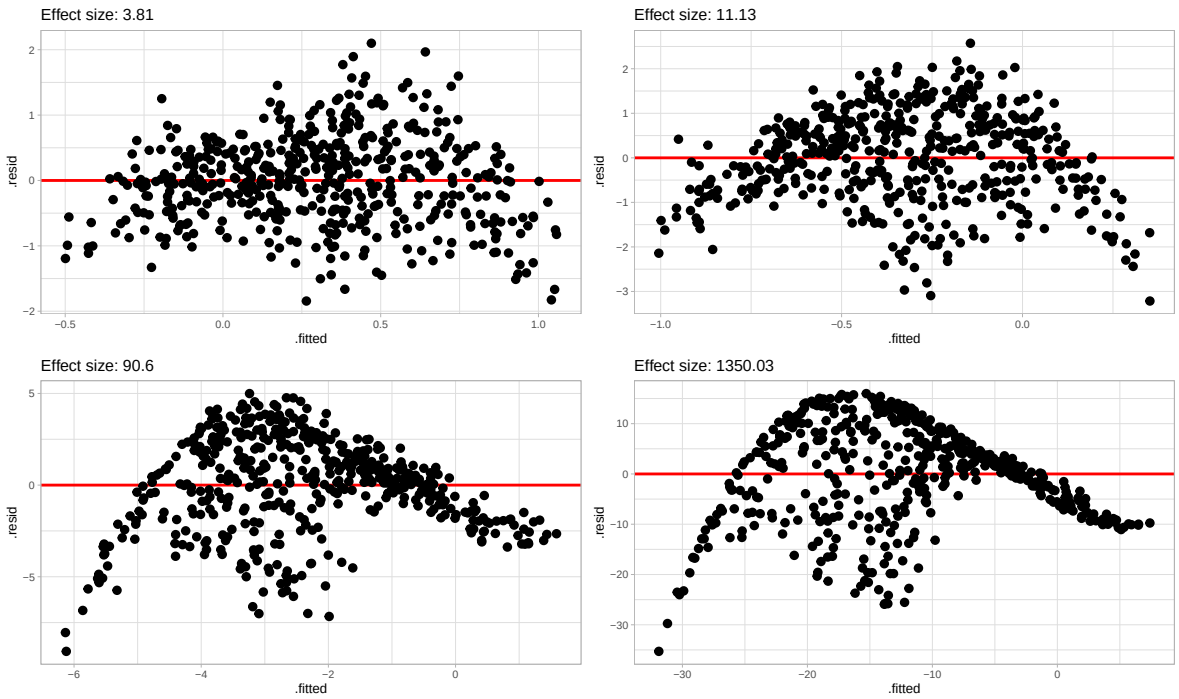


Figure 4.6: *Cubic model residual plots under different effect sizes. The larger the effect size, the stronger the signal.*

be squared. Third, the measure needs to be a scalar. We could apply a weighted average operator $\boldsymbol{W}$ on the transformed expected residuals to obtain a single numeric value. For simplicity, we set $\boldsymbol{W} = n^{-1}\mathbf{1}$. Considering the high time complexity of computing the square root of $\boldsymbol{R}_a$, off-diagonal elements of $\boldsymbol{R}_a$ are set to be zeros. This gives the effect size:

$$ES = n^{-1}||\sigma^{-1}\boldsymbol{R}_a^{-\frac{1}{2}}\boldsymbol{R}_a\boldsymbol{X}_b\boldsymbol{\beta}_b||^2 = n^{-1}\sigma^{-2}||\boldsymbol{R}_a^{\frac{1}{2}}\boldsymbol{X}_b\boldsymbol{\beta}_b||^2 \approx n^{-1}\sigma^{-2}||diag(\boldsymbol{R}_a)^{\frac{1}{2}}\boldsymbol{X}_b\boldsymbol{\beta}_b||^2,$$

where $diag(\boldsymbol{R}_a)$ is the diagonal matrix constructed from the diagonal elements of $\boldsymbol{R}_a$.

The interpretation of this effect size is the impact of parameter a and b on the squared deviation of the standardized expected residual per observation. It is not directly related to the shape, or the pattern human observed from the residual plot, but it is a reasonable approximation of the degree of the visual deviation from the null residuals under our cubic model setting. Figure 4.6 shows four residual plots with different effect sizes. As effect sizes increases, the strength of the signal become stronger.

Under the cubic model setting, there is an exact conventional test for testing the non-linearity defect, which is F-test. For F-test, the null hypothesis is $H_0 : a = b = 0$, and the alternative hypothesis is H_1 : at least one of them $\neq 0$. During the simulation of the lineup data, the F-statistic and the p-value will be recorded for comparison between the power of conventional test and visual test.

4.1.2 Heteroskedasticity Model

Another model used in the experiments was a heteroskedasticity model with one regressor, which can be expressed by:

$$Y_i = 1 + X_i + \varepsilon_i,\ i = 1, ..., n, \tag{4.3}$$

where $a \in \{-1, 0, 1\}$, $b \in (0, 32)$ and $\varepsilon_i \overset{iid}{\sim} N(0, 1 + b(2 - |a|)(X_i - a)^2)$.

To create heteroskedasticity defect, OLS was used to fit the null model:

$$\boldsymbol{Y} = \beta_0 + \beta_1 \boldsymbol{X} + \boldsymbol{u}, \tag{4.4}$$

where $\boldsymbol{u} \sim N(\boldsymbol{0}, \sigma_u^2 \boldsymbol{I})$.

In this case, estimators of β_0 and β_1 are unbiased, but the error term has non-constant variance.

Lemma 4.2 (Distribution of residuals produced by the heteroskedasticity model)**.** *Given the data generating process in Equation (4.3) and null model in Equation (4.4). Let* $\boldsymbol{X}_a = [\boldsymbol{1}, \boldsymbol{X}]$ *denotes the set of regressors in matrix form. The residuals obtained from the null model are*

$$\boldsymbol{e} \sim N(\boldsymbol{0}, \boldsymbol{R}_a \boldsymbol{V}),$$

where $\boldsymbol{R}_a = \boldsymbol{I} - \boldsymbol{X}_a(\boldsymbol{X}_a'\boldsymbol{X}_a)^{-1}\boldsymbol{X}_a'$ *and* $\boldsymbol{V}$ *is a diagonal matrix with* $V_{ii} = 1 + b(2 - |a|)(X_i - a)^2$, $i = 1, ..., n$.

Proof. Using the Frisch–Waugh–Lovell theorem, the residuals obtained by the null model are $\boldsymbol{e} = \boldsymbol{R}_a \boldsymbol{Y} = \boldsymbol{R}_a(\boldsymbol{X}_a \boldsymbol{\beta}_a + \boldsymbol{\varepsilon})$, where $\boldsymbol{R}_a = \boldsymbol{I} - \boldsymbol{X}_a(\boldsymbol{X}_a'\boldsymbol{X}_a)^{-1}\boldsymbol{X}_a'$ and $\boldsymbol{\beta}_a = (1, 1)'$.

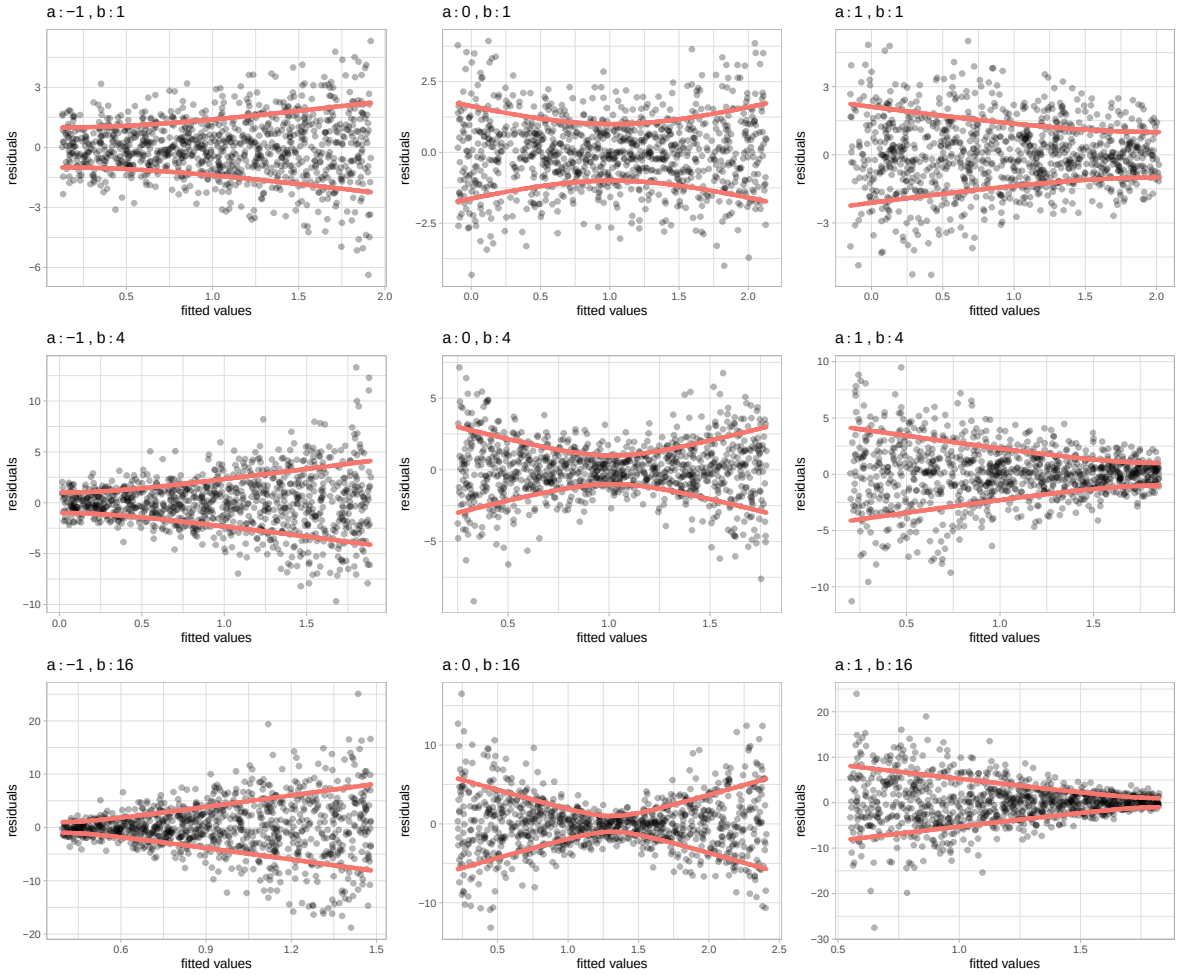


Figure 4.7: *A matrix of residuals vs. fitted values plot under different parameter settings of the heteroskedasticity model. Resdiauls are drawns in black and the one standard deviation around zero is drawn in red. The plots show that a controls the shape and direction of the resdiaul plot, while b controls the strength of the signal.*

Because $\boldsymbol{R}_a\boldsymbol{X}_a = \boldsymbol{0}$, we have $\boldsymbol{e} = \boldsymbol{R}_a\boldsymbol{\varepsilon}$. Hence, the residuals $\boldsymbol{e}$ follow $N(\boldsymbol{0}, \boldsymbol{R}_a\boldsymbol{V})$, where $\boldsymbol{V}$ is a diagonal matrix with $V_{ii} = 1 + b(2 - |a|)(X_i - a)^2$, $i = 1, ..., n$. □

Lemma 4.2 shows that variance and covariance of the residuals depend on $\boldsymbol{X}$. We could plot the one standard deviation around the residuals to indicate the region about 68% of the residuals should landed on. Figure 4.7 illustrates different shapes of residuals under various values of a and b. From the plot, it can be observed that if $a = 0$, the residual plot looks like the shape of butterfly. If $a = \pm 1$, it looks like a triangle, and the sign of a determines the direction of the shape. Parameter b is controlling the strength of the signal. As b increases, the pattern becomes less noisy.

Similar to the cubic model, the heteroskedasticity model could also be built by the heteroskedasticity model class constructor `heter_model()`. This function takes three arguments as inputs, which are `a`, `b` and `x`. `a` and `b` are numeric parameters defined in Equation (4.3). `x` needs to be a random variable object.

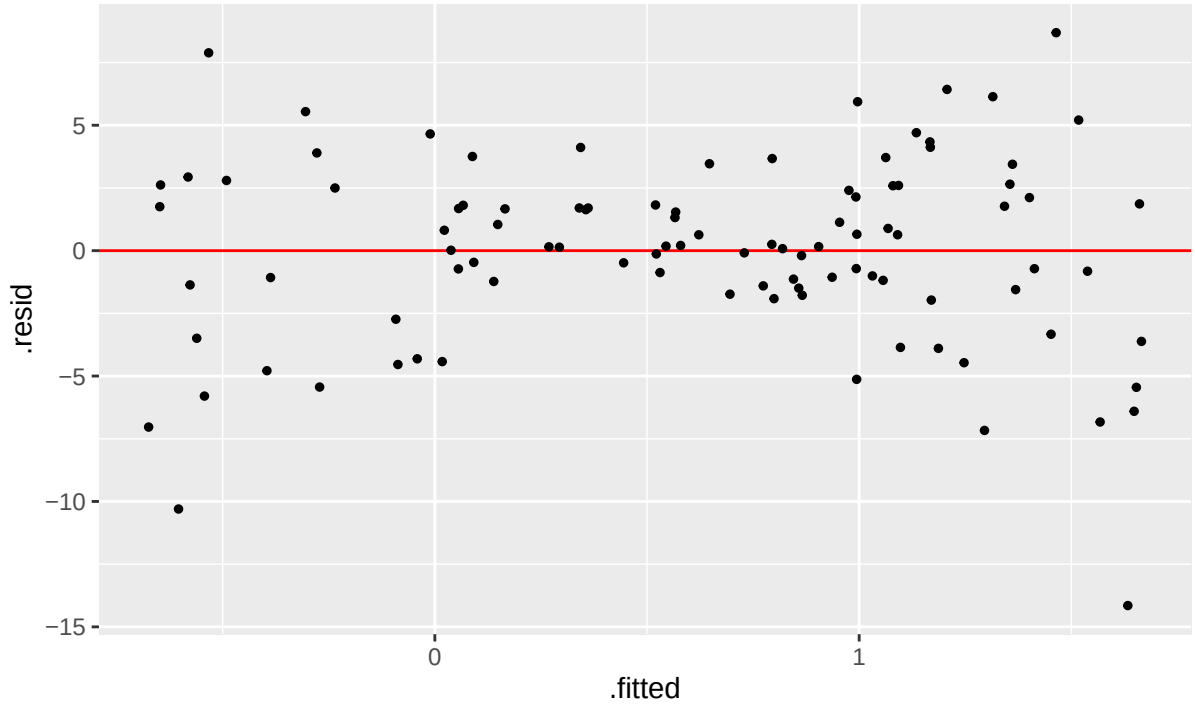
```
library(visage)
mod <- heter_model(a = 0, b = 16, x = rand_uniform(-1, 1))
```

Since both the cubic model class and the heteroskedasticity class are inherited from the visual inference model class, which has defined methods of simulating data, making residual plot and producing lineup, heteroskedasticity model object can be used in a similar way as cubic model object. The following codes give examples of the use of the object.

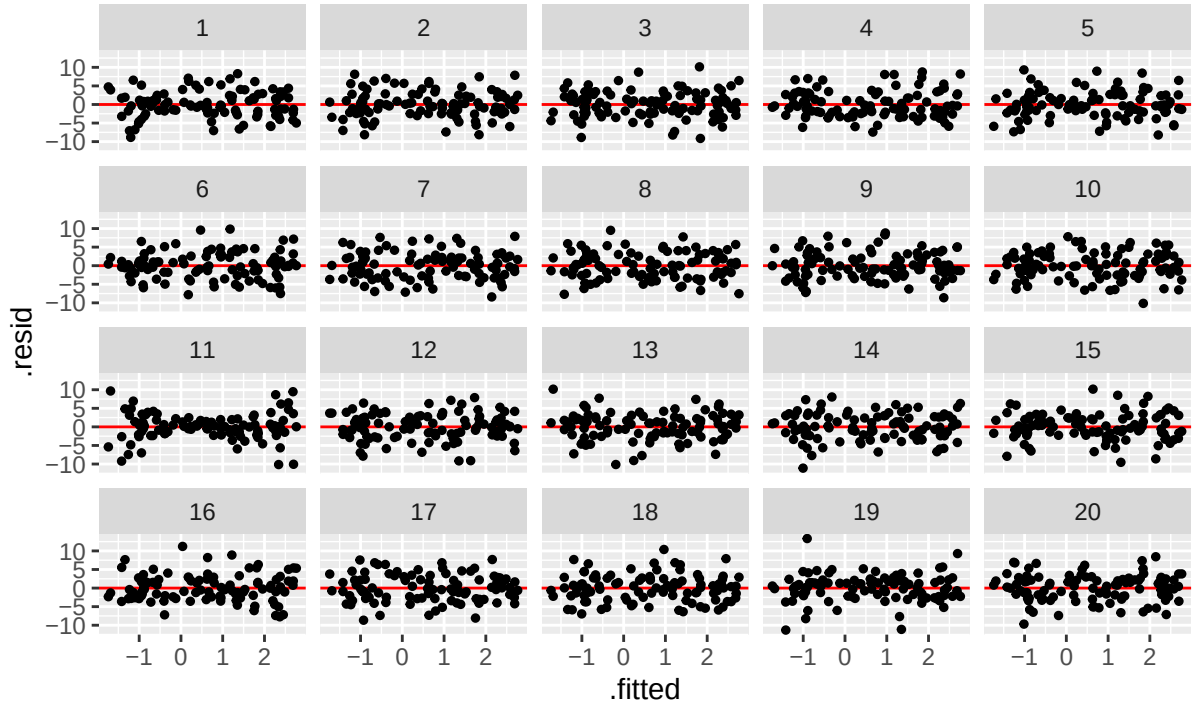
```
mod$gen(5, fit_model = TRUE)
```

```
##            y          x    sigma          e     .resid    .fitted
## 1 -7.6931740  0.8335218 4.819987 -9.5266958 -5.7469121 -1.9462619
## 2  7.0213194 -0.7458772 4.336202  6.7671966 -0.9130225  7.9343418
## 3 -2.1856528  0.9509873 5.471751 -4.1366401  0.4954643 -2.6811172
## 4  4.0402811  0.4961018 2.979219  2.5441792  3.8756685  0.1646126
## 5  0.8686919  0.7494172 4.355690 -0.8807253  2.2888017 -1.4201098
```

```
mod$plot(mod$gen(100, fit_model = TRUE), size = 1)
```




```
mod$plot_lineup(mod$gen_lineup(100), size = 1)
```



According to Lemma 4.2, when $b = 0$, the matrix V collapses to an identity matrix. Assume the shape of butterfly and triangle have identical visual impacts, the only factor affects human to recognize the pattern is the strength of the signal, where parameter a

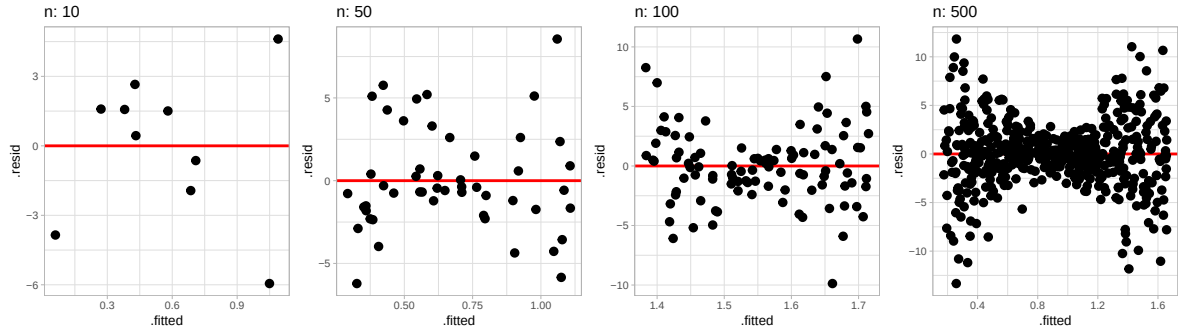


Figure 4.8: *Residuals generated from the same heteroskedasticity model but with different sample size. As sample size increases, the shape of butterfly becomes more obvious.*

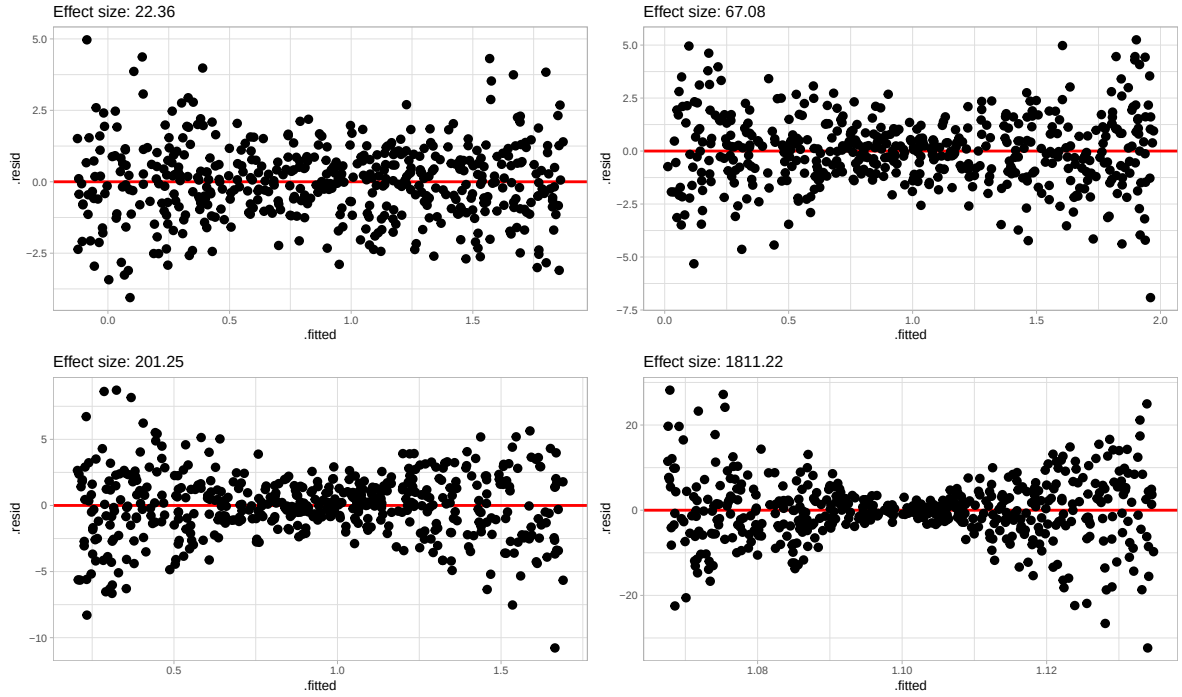


Figure 4.9: *Residuals of heteroskedasticity model with different effect size. As effect size increases, the shape of butterfly becomes less noisy.*

has no role to play. In addition, sample size has a huge impact on the chance of human recognizing the pattern. As shown in Figure 4.8, as sample size increases, the pattern becomes easier to observe. However, this effect is less noticeable with large sample size as the outline of the shape has been drawn and residuals have less probability to land outside of it. Thus, the effect size of this model can be expressed by $ES = b\sqrt{n}$. The square root operator is used for addressing the large sample issue. Figure 4.9 shows the effectiveness of the effect size.

For the heteroskedasticity model, the conventional test we used was Breusch–Pagan test (Breusch and Pagan, 1979), which tested whether the variance of the error terms of the regression is dependent on the regressors with the auxiliary regression equation

$$\boldsymbol{e}^2 = \gamma_0 + \gamma_1 \boldsymbol{X} + \gamma_2 \boldsymbol{X}^2 + \boldsymbol{v}.$$

Majumder, Hofmann, and Cook (2013) suggested that visual test is not expected to perform equally well as conventional test especially when there exists a exact conventional test. However, in contrast to the F-test used in the cubic model, Breusch–Pagan test is an approximate test. Thus, the power of visual test may exceed the power of Breusch–Pagan test. Throughout the study, Breusch–Pagan test statistic and p-value were recorded for comparison.

4.1.3 Distribution of regressors

The model definitions given in the previous two sections does not include the specification of the regressors. In this sections, distribution of $\boldsymbol{X}$ and $\boldsymbol{Z}$ will be discussed.

The cubic model involved the use of both $\boldsymbol{X}$ and $\boldsymbol{Z}$. In the simulation, X_i, $i = 1, ..., n$, had equal chance to follow one of the following distributions: $N(0, 0.09)$, $U(-1, 1)$, $Lognormal(0, 0.36)/3 - 1$ and $-Lognormal(0, 0.36)/3 + 1$. Uniform and normal distribution were symmetric and common. Adjusted log-normal distribution and adjusted negative log-normal distribution provided right-skewed and left-skewed density respectively. These distributions were chosen such that most the realizations will fall between -1 and 1.

The distribution of Z_i, $i = 1, ..., n$, had 50% chance to be a uniform distribution ranged from -1 to 1, and 50% chance to be a discrete uniform distribution with z_n outcomes simulated from a uniform distribution ranged from -1 to 1. z_n itself was a discrete uniform distribution with outcomes $\{10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20\}$, which defined the number of possible values Z_i could take. As shown in Figure 4.10, this set-up would create discreteness in residual plot, which could enrich the pool of visual patterns.

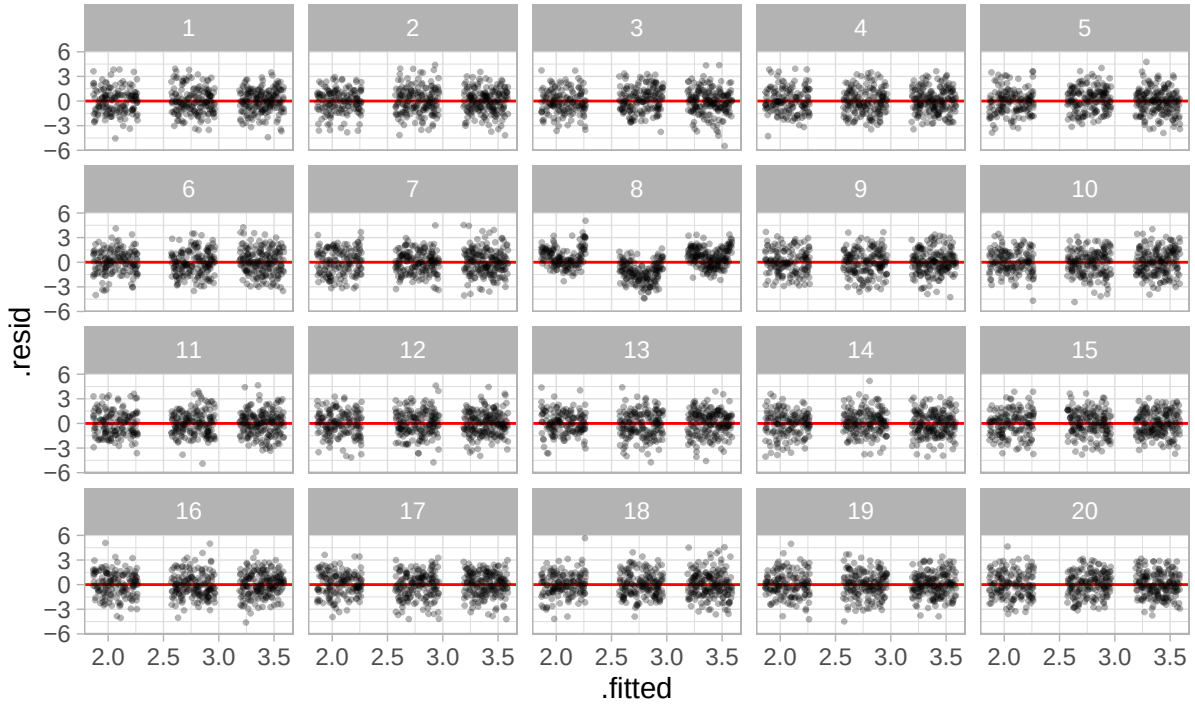


Figure 4.10: *Discreteness in residuals created by using a discrete uniform random variable as one of the regressors in a cubic model. For each residual plot of the lineup, there are three clusters because the number of possible values the regressor can take is three.*

$\boldsymbol{X}$ used in the heteroskedasticity model was a combination of $\boldsymbol{X}$ and $\boldsymbol{Z}$ used in the cubic model. It could be one of the five distributions mentioned above - normal distribution, uniform distribution, adjusted log-normal distribution, adjusted negative log-normal distribution and discrete uniform distribution.

4.1.4 Experiment I and II

In this study, multiple selection was allowed for the evaluation of a lineup. Given our desired significance level was $\alpha = 0.05$ and the number of plots in a lineup was 20, a single visual test procedure needed to involve multiple evaluations. Otherwise, the p-value of the test would be greater than 0.05 in spite of the actual data plot being selected. The number of plots selected by a user depended on the user and the difficulty of the lineup. According to the pilot study evaluated by 10 faculty members of Department of Econometrics and Business Analytics, Monash University, the number of selections would be generally smaller than 4, which suggested a visual test consisted of 5 evaluations were sufficient to yield p-value smaller than 0.05 with at least three detections. Thus, each

lineup was replicated for five times and evaluated by different participants. However, our study website was designed such that any set of lineups consisting of 20 lineups generated beforehand would be shown to only one person only once. And whether the participant would finish the experiment was unpredictable. Therefore, some lineups had insufficient number of evaluations, but some had more evaluations than expected. This slightly affected the estimate of the power of the visual test, which will be discussed in Section 4.1.5.

We called the lineup being detected if the actual data plot was one of the selections of an evaluation of a lineup. The sample size calculation was based on the detection rate of the lineup, which was closely related to the difficulty level of the lineup. With the data collected from the pilot study, two logistic regressions given in Table 4.2 and Figure 4.11 were developed to describe the relationship between the natural logarithm of effect size and the detection rate for the cubic model and the heteroskedasticity model.

Table 4.2: *Logistic regressions with detection rate as response variable and natural logarithm of the effect size as regressor fitted on the pilot data for both cubic and heteroskedasticity models.*

	Dependent variable:	
	detect	
	Cubic	Heteroskedasticity
	(1)	(2)
log(effect_size)	0.611***	0.737***
	(0.098)	(0.137)
Constant	0.457***	−2.402***
	(0.165)	(0.645)
Observations	246	289
Log Likelihood	−122.305	−147.971
Akaike Inf. Crit.	248.610	299.943
Note:	*p<0.1; **p<0.05; ***p<0.01	

Lineups were classified into three categories - easy, medium and hard. Easy lineups were those of which predicted detection rates were higher than 80%, while medium lineups were between 40% to 80% and hard lineups were less than 40%.

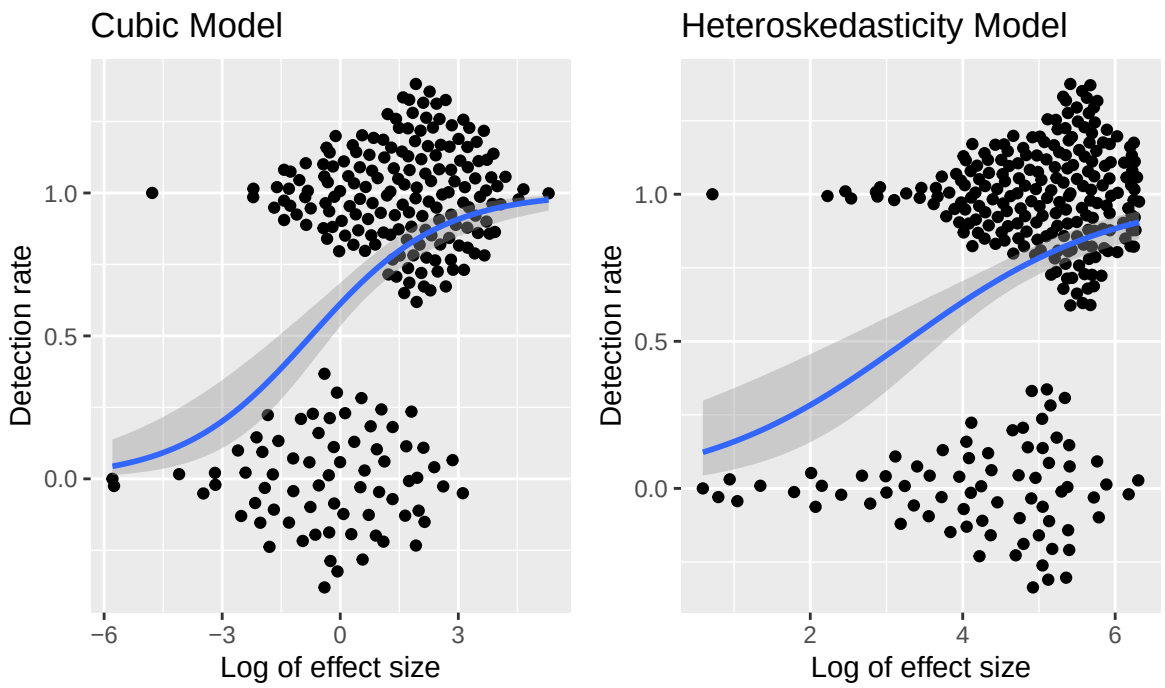


Figure 4.11: *Logistic regressions with detection rate as response variable and natural logarithm of the effect size as regressor fitted on the pilot data for both cubic and heteroskedasticity models. Points are jittered to reduce overplotting using the* `ggbeeswarm` *package. The smooth curves are the fitted regression curves. The ribbons around the curves are the 95% confidence intervals.*

For the first experiment, we would like to have at least 20 detection with chance greater than 99.99% in each category. Thus, we prepared 68 easy lineups, 120 medium lineups and 388 hard lineups with five replications. Every participant would get at least 8 easy or medium lineups. Two out of 10 extremely lineups with predicted detection rate over 90% were randomly given to every participant as attention checks. We initially planned to recruit $(68 + 120 + 388) \times 5/(20 - 2) = 160$ participants but eventually decided to only recruit 20 participants because it was the first time we launched an experiment on Prolific where factors like payment method and quality of participants were unclear at the moment. As a consequence, most of the lineups had only two evaluations and not all the lineups were used. However, data collected from the first experiment was sufficient for building a more accurate logistic regression to predict detection rate for the second experiment.

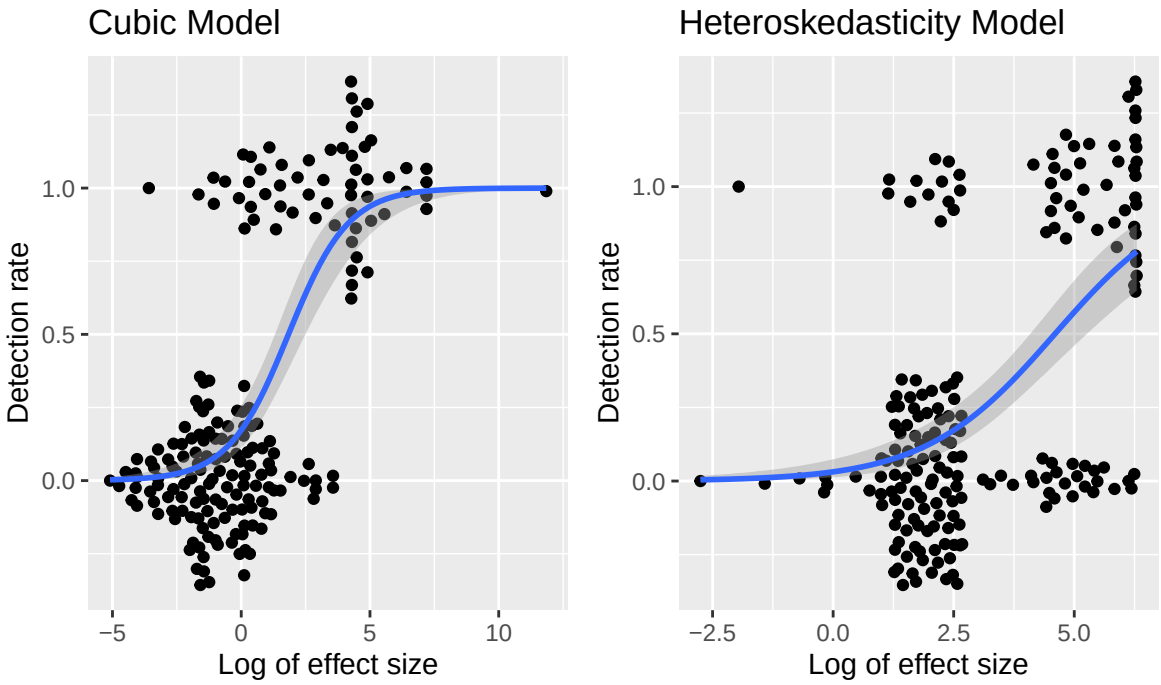


Figure 4.12: *Logistic regressions with detection rate as response variable and natural logarithm of the effect size as regressor fitted on the data collected from experiment I for both cubic and heteroskedasticity models. Points are jittered to reduce overplotting using the* `ggbeeswarm` *package. The smooth curves are the fitted regression curves. The ribbons around the curves are the 95% confidence intervals.*

Table 4.3: *Logistic regressions with detection rate as response variable and natural logarithm of the effect size as regressor fitted on the data collected from experiment I for both cubic and heteroskedasticity models.*

	Dependent variable:	
	detect	
	Cubic	Heteroskedasticity
	(1)	(2)
log(effect_size)	0.857***	0.746***
	(0.122)	(0.111)
Constant	−1.571***	−3.434***
	(0.257)	(0.462)
Observations	192	188
Log Likelihood	−64.544	−84.606
Akaike Inf. Crit.	133.089	173.212
Note:	*p<0.1; **p<0.05; ***p<0.01	

According to the new logistic regression given in Table 4.3 and Figure 4.12, it was found that the first experiment might be too difficult for participants as the detection rate was lower than what we estimated from the pilot study, especially for the heteroskedasticity model. Thus, for the second experiments, we relaxed the sample size requirements and increased the proportion of easy lineups. In addition, due to budget constraint, the maximum number of participants was 50. With the new logistic regression fitted with data collected from the first experiment, 28 easy lineups, 36 medium lineups and 100 hard lineups were simulated for the second experiment. 50 participants were recruited but 6 were withdrew.

4.1.5 P-value and Power Estimation of Visual Test Allowed for Multiple Selections

The p-value calculation for multiple selection is an extension of Equation (1.1). The assumptions about the independence still needed. Let K be the number of independent evaluations of a visual test, s_i, $i = 1, ..., K$ be the number of selections of the evaluations, and X be the random variable denoting the number of detections. Then, the p-value of the visual test is given as:

$$P(X \geq x) = \sum_{j=x}^{K} Pr(j|s_1, ..., s_K). \tag{4.5}$$

The distribution of X given $s_1, ...s_K$ can not be derived trivially, as it is a sampling without replacement problem. In practice, this distribution can be approximated by computer simulation.

Function `sim_dist()` from package `visage` is designed to approximate the distribution of number of detections of a lineup via Monte Carlo method. It takes the number of evaluations `n_eval`, the number of selections `n_sel`, and the number of plots in a lineup `n_plot` as inputs, then outputs a discrete distribution.

```
sim_dist(n_eval = 3, n_sel = c(2, 2, 3), n_plot = 20)
```

```
##       0       1       2       3
## 0.70094 0.25280 0.04340 0.00286
```

Function `calc_p_value()` from the same package calculates the p-value using the distribution returned by `sim_dist()`. It takes an additional argument `n_detect`, which is the number of detections.

```
calc_p_value(n_detect = 2, n_eval = 3, n_sel = c(2, 2, 3), n_plot = 20)
```

```
## [1] 0.0456
```

The above example shows that a visual test with two detections and three independent evaluations in which three observers select two, two, and three plots out of 20 plots respectively yields a p-value smaller than 0.05.

Assume there is a visual test V_K, where K denoting the number of evaluations. The corresponding p-value of V_K can be computed by using Equation (4.5). Meanwhile, if one evaluation is randomly deleted from V_K, the remaining evaluations can still be used to form another valid visual test V_{K-1}. In fact, considering all the possibilities, K different outcomes for V_{K-1} can be obtained. Since all outcomes occur with equal probability, the proportion of outcomes which reject the null hypothesis can be used as an estimate of the power of the visual test V_{K-1}. Similarly, if we would like to estimate the power of the visual test V_{K-j} given the evaluations of V_K, for $j < K$, we could find all the possible combinations of K elements, taken $k - j$ at a time to obtain $\binom{K}{K-j}$ different outcomes for V_{K-j}. Then, the estimated power is given as $R/\binom{K}{K-j}$, where R is the number of outcomes which reject the null hypothesis.

Function `calc_p_value_comb()` from the package `visage` can be used to compute p-values of V_{K-j}. The first argument is `detected`, which needs to be a vector of Boolean values denoting whether the observer detects the actual data plot. Desired number of evaluations and number of selections need to be provided via argument `n_eval` and `n_sel`.

For example, given a visual test V_3 where the first observer selects one plot and gets correct, the second observer selects one plot but misses, and the third observer selects two plots and gets correct. The p-value of all possible outcomes of V_2 can be computed using the following code:

```
calc_p_value_comb(detected = c(TRUE, FALSE, TRUE),
                  n_eval = 2,
                  n_sel = c(1, 1, 2))
```

```
## [1] 0.09352 0.00624 0.14454
## attr(,"combinations")
##      [,1] [,2] [,3]
## [1,]    1    1    2
## [2,]    2    3    3
```

`calc_p_value_comb()` returns a vector of p-value along with an attribute which is a matrix representing elements being used by the specific outcome. Elements in the first column indicates that the first observer and the second observer are involved in the visual test, and the corresponding p-value is 0.09352.

4.1.6 Results

We collected 400 lineup evaluations made by 20 participants in experiment I and 880 lineup evaluations made by 44 participants in experiment II. In total, 442 unique lineups were evaluated by 64 subjects. In experiment I, one of the participants skipped all 20 lineups. Hence, the submission was rejected and removed from the dataset. In experiment II, there was a participant failed one of the two attention checks, but there was no further evidence of low-effort throughout the experiment. Therefore, the submission was kept.

Power Estimation

It was expected that with larger effect size, the power of the visual test would be greater for both cubic model and heteroskedasticity model. Using the power calculation method discussed in Section 4.1.5, power of each lineup from one to five evaluations was estimated. The estimated power was further used in fitting a quasi-binomial generalized linear model with natural logarithm of effect size as the only regressor. The result of the fitted models are given in Table 4.4, Table 4.5, and Figure 4.13.

As expected, coefficients of natural logarithm of effect size of all five models were positive. From Figure 4.13, it can be observed that the fitted power of visual test increased as the number of evaluations increased for both cubic and heteroskedasticity model.

For heteroskedasticity model, this phenomenon was more obvious as the curves of visual tests with evaluations greater than two were always above the curves of visual test with evaluations smaller than two. However, this only held for large enough effect size. For small effect size, visual tests with fewer evaluations might have greater power. Note that, the expected power of visual test derived by Majumder, Hofmann, and Cook (2013) followed by the assumption that human has the ability to select the plot with the highest t-statistic from a lineup under the classical linear model regression setting showed similar properties, where visual tests with fewer evaluations were expected to perform better when the parameter values close to the null hypothesis.

For cubic model, the separation between curves was small. Fitted power of visual test with three to five evaluations were almost identical to each other in regards of effect size. In addition, all five curves peaked at one as effect size increased, suggesting that identification of non-linearity as a visual task can be completed reliably by human when the effect size is large enough.

As shown in Figure 4.13, both F-test and Breusch–Pagan test generally possessed greater power than visual test. Since the settings we were using were not ideal scenarios for visual inference, it was not expected that the visual test would be as good as the conventional test. Our goal was to quantify the ability of human reading residual plot, such that the comparison between human and computer vision model can be made. It was also found that there was a huge gap between the conventional curves and the visual test curves at the middle of the plot. We further analysed the lineup with the corresponding effect size. Figure 4.14 and 4.15 showed that human was indeed hard to identify the pattern at this level of difficulty. However, we as researchers of this project could tell which was the actual data plot by carefully comparing the plots of the lineup since we knew all possible shapes of the residual plot beforehand.

Table 4.4: *Quasi-binomial logistic regressions with estimated power as response variable and natural logarithm of the effect size as regressor fitted on the data collected from experiment I and II for cubic model. Five regressions were fitted for different number of evaluations.*

	Dependent variable:				
	hat_power				
	(1)	(2)	(3)	(4)	(5)
log(effect_size)	0.929***	1.227***	1.084***	1.057***	1.035***
	(0.049)	(0.066)	(0.074)	(0.082)	(0.087)
Constant	−1.969***	−2.603***	−1.570***	−1.501***	−1.432***
	(0.114)	(0.153)	(0.142)	(0.156)	(0.160)
Observations	630	532	456	423	371
Note:					*p<0.1; **p<0.05; ***p<0.01

Table 4.5: *Quasi-binomial logistic regressions with estimated power as response variable and natural logarithm of the effect size as regressor fitted on the data collected from experiment I and II for heteroskedasticity model. Five regression were fitted for different number of evaluations.*

	Dependent variable:				
	hat_power				
	(1)	(2)	(3)	(4)	(5)
log(effect_size)	0.821***	1.025***	0.885***	0.920***	1.063***
	(0.064)	(0.087)	(0.092)	(0.102)	(0.125)
Constant	−4.472***	−5.610***	−4.285***	−4.127***	−4.717***
	(0.320)	(0.449)	(0.461)	(0.499)	(0.607)
Observations	630	527	473	443	375
Note:					*p<0.1; **p<0.05; ***p<0.01

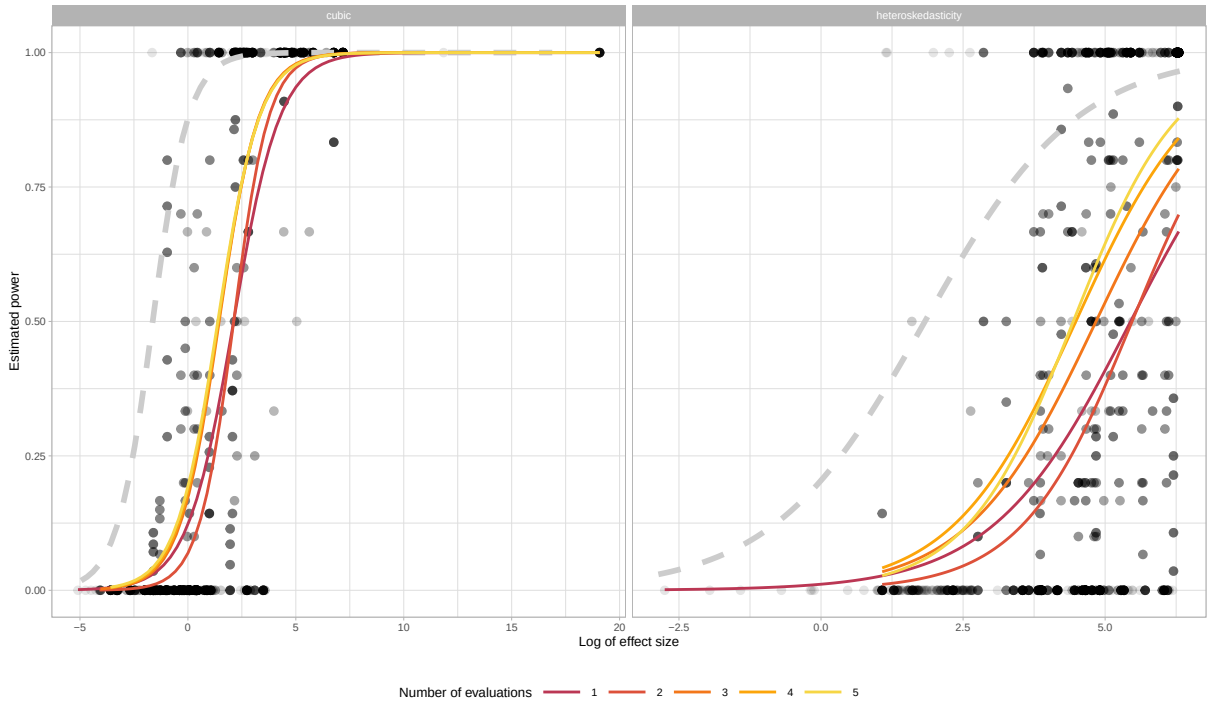


Figure 4.13: *Quasi-binomial logistic regressions with estimated power as response variable and natural logarithm of the effect size as regressor fitted on the data collected from experiment I and II for both cubic and heteroskedasticity model. Five regression were fitted for different number of evaluations shown in five different colours. Estimated power are drawn in black. The dashed curvses are the power of F-test and Breusch–Pagan test respectively.*

Another method we used to model the power was to collect all decisions made by $\binom{K}{K-j}$ different outcomes for V_{K-j}, and directly fit a binomial generalized linear model with natural logarithm of effect size as the regressor. This method skipped the calculation of the estimated power $R/\binom{K}{K-j}$, and regressed on the binary outcome with larger sample size. Results of the fitted model is given in Figure 4.16, Table 4.7 and Table 4.6. According to Figure 4.16, this method gave clearer separation between curves for heteroskedasticity model. Noticeably, fitted power of visual test with five evaluations matched the power of conventional test at large effect size.

Effect of Factors on Power of the Visual Test

The previous section focuses on the change of effect size relative to the power of the visual test. Individual factor embedded in the data simulation also needs be analysed. Two logistic regressions were fitted with decisions made by V_{K-j} as response variable. For the

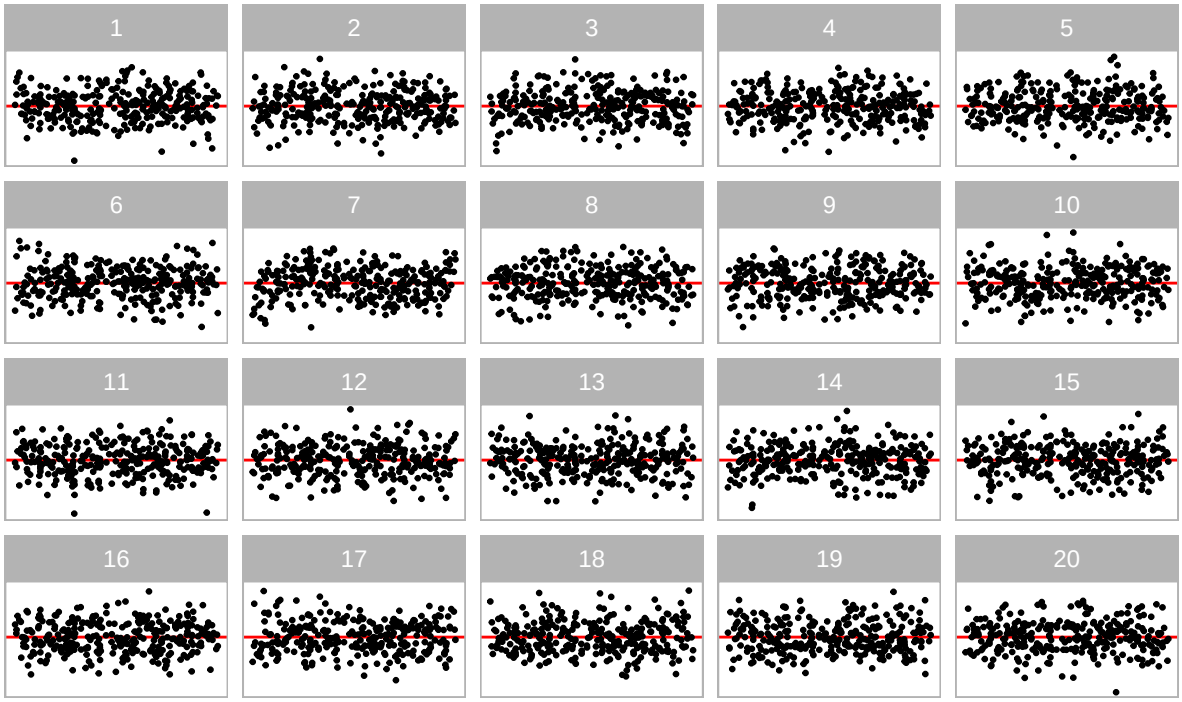


Figure 4.14: *An example lineup for the cubic model with effect size equal to 0.18. The answer to this linup is seven. It was not suprised participants could not confidently identify the actual data plot given the pattern was very weak.*

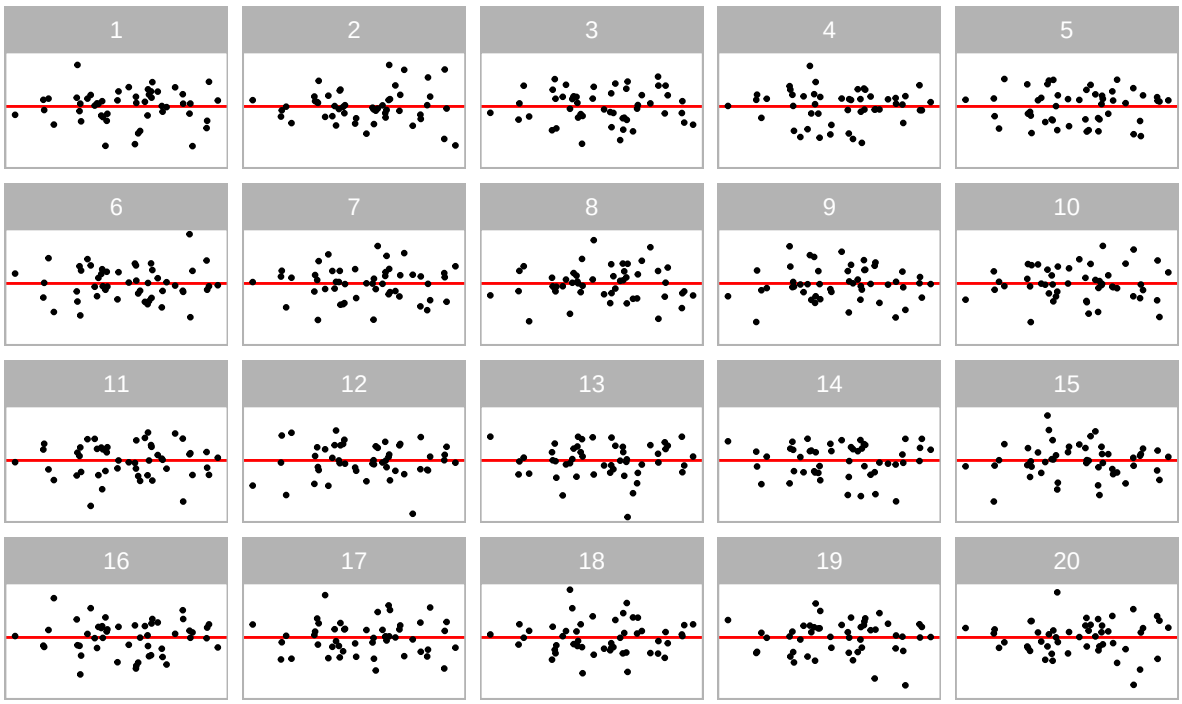


Figure 4.15: *An example lineup for the heteroskedasticity model with effect size equal to 17.5. The answer to this linup is two. The butterfly shape could merely being recognised.*

Table 4.6: *Logistic regressions with rejection as response variable and natural logarithm of the effect size as regressor fitted on the data collected from experiment I and II for cubic model. Five regression were fitted for different number of evaluations.*

	Dependent variable:				
	hat_power				
	(1)	(2)	(3)	(4)	(5)
log(effect_size)	0.872***	1.292***	1.125***	1.164***	1.315***
	(0.031)	(0.032)	(0.021)	(0.021)	(0.029)
Constant	−1.836***	−2.529***	−1.516***	−1.321***	−1.419***
	(0.077)	(0.073)	(0.046)	(0.049)	(0.072)
Observations	2,900	7,250	12,596	17,011	19,229
Log Likelihood	−752.896	−1,312.436	−2,977.445	−2,651.299	−1,463.555
Akaike Inf. Crit.	1,509.792	2,628.873	5,958.889	5,306.598	2,931.109
Note:					*p<0.1; **p<0.05; ***p<0.01

Table 4.7: *Logistic regressions with rejection as response variable and natural logarithm of the effect size as regressor fitted on the data collected from experiment I and II for heteroskedasticity model. Five regression were fitted for different number of evaluations.*

	Dependent variable:				
	hat_power				
	(1)	(2)	(3)	(4)	(5)
log(effect_size)	0.912***	1.221***	1.294***	1.525***	1.798***
	(0.044)	(0.033)	(0.027)	(0.030)	(0.042)
Constant	−4.846***	−6.396***	−6.033***	−6.771***	−7.784***
	(0.229)	(0.175)	(0.141)	(0.159)	(0.230)
Observations	2,906	7,064	11,436	13,728	13,236
Log Likelihood	−1,525.398	−3,668.325	−5,491.915	−4,828.896	−2,858.774
Akaike Inf. Crit.	3,054.796	7,340.650	10,987.830	9,661.793	5,721.548
Note:					*p<0.1; **p<0.05; ***p<0.01

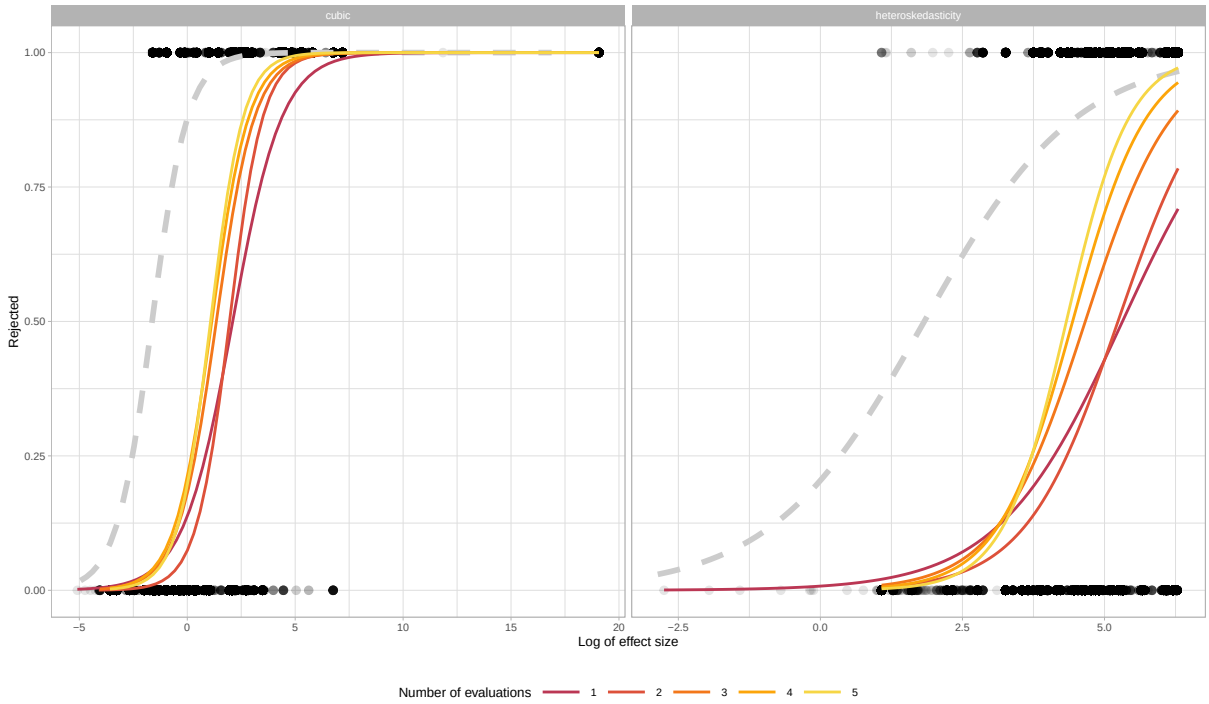


Figure 4.16: *Logistic regressions with rejection as response variable and natural logarithm of the effect size as regressor fitted on the data collected from experiment I and II for both cubic and heteroskedasticity model. Five regression were fitted for different number of evaluations shown in five different colours. Estimated power are drawn in black. The dashed curvses are the power of F-test and Breusch–Pagan test respectively.*

cubic model, regressors were $|a|$, $|b|$, $|c-1|$, σ, distribution of $\boldsymbol{X}$, distribution of $\boldsymbol{Z}$, number of observations and number of evaluations. For the heteroskedasticity model, regressors were $|a|$, b, distribution of $\boldsymbol{X}$, number of observations and number of evaluations.

According to Table 4.8, for cubic model, both coefficients of $|a|$ and $|b|$ were positive, which suggested increasing the coefficients of quadric and cubic terms would result in higher visual power. The coefficient of $|c-1|$ was also positive indicating joint effect of multiple regressors on the residual plot decreased the visual power. As expected, the coefficient of σ was negative as it controlled the noise level. Lineup simulated with uniform distribution of $\boldsymbol{X}$ had the highest power, followed by negative log-normal, normal, and log-normal. Discreteness in $\boldsymbol{Z}$ decreased the visual power. The coefficient of number of observations was positive, which means larger sample size would lead to higher visual power.

Results of logistic regression for heteroskedasticity is given in Table 4.9. For heteroskedasticity model, the coefficient of $|a|$ was negative, which suggested "butter fly" shape was

easier to identify than "triangle" shape. As expected, the coefficient of b was positive. For different distributions of $\boldsymbol{X}$, the one with highest power was uniform distribution, followed by normal, discrete uniform, log-normal and negative log-normal. The coefficient of the number of observations was positive, but it was not significant.

In overall, we did not have any unexpected findings except the visual power related to different distributions of $\boldsymbol{X}$ in cubic model. We expected lineup simulated with uniform distribution would have the highest power, followed by normal, log-normal and negative log-normal, since we observed similar results in pilot study.

Table 4.8: *Logistic regressions with rejection as response variable and parameter settings as regressors fitted on the data collected from experiment I and II for cubic model.*

	Dependent variable:
	reject
|a|	1.064***
	(0.031)
|b|	0.718***
	(0.029)
|c-1|	5.443***
	(0.119)
sigma	−3.536***
	(0.050)
Distribution of X: Negative Lognormal	0.971***
	(0.073)
Distribution of X: Normal	0.754***
	(0.086)
Distribution of X: Uniform	1.751***
	(0.085)
Distribution of Z: Discrete	−2.986***
	(0.056)
n	0.005***
	(0.0002)
Number of evaluations	0.291***
	(0.017)
Constant	−2.453***
	(0.173)
Observations	58,986
Log Likelihood	−8,034.351
Akaike Inf. Crit.	16,090.700
Note:	*p<0.1; **p<0.05; ***p<0.01

Table 4.9: *Logistic regressions with rejection as response variable and parameter settings as regressors fitted on the data collected from experiment I and II for heteroskedasticity model.*

	Dependent variable:
	reject
|a|	-0.526^{***}
	(0.037)
b	0.064^{***}
	(0.002)
Distribution of X: Lognormal	-0.534^{***}
	(0.058)
Distribution of X: Negative Lognormal	-0.610^{***}
	(0.062)
Distribution of X: Normal	0.187^{***}
	(0.063)
Distribution of X: Uniform	2.980^{***}
	(0.064)
n	0.0003
	(0.0002)
Number of evaluations	0.496^{***}
	(0.014)
Constant	-2.711^{***}
	(0.062)
Observations	48,370
Log Likelihood	−13,026.190
Akaike Inf. Crit.	26,070.390
Note:	*p<0.1; **p<0.05; ***p<0.01

4.2 Automatic Visual Inference System

Lineup protocol can be used to make valid inference for visual discoveries, but one of the drawbacks of this protocol is it requires human evaluation of data plots. Such requirement is unattainable when the number of data plots need for testing is large. Even if it can be fulfilled, the cost of labour would be exceedingly high. Hence, there is a need of building an automatic visual inference system to read data plots, evaluate lineups and provide meaningful decisions. Since this desired automatic system is technically an agent intended to play the role of evaluators as humans in lineup protocol, some degree of intelligence needs to be demonstrated. This brings us to Artificial Intelligence (AI).

4.2.1 Artificial Intelligence: Four Approaches

AI is the field of research concerned with understanding and building machines who can demonstrate intelligence. As discussed in Russell and Norvig (2002), historically, there are disagreements among researchers about the definition of intelligence, which is caused by two critical questions:

1. Should AI act and think humanly or rationally?
2. Without the thought process and reasoning, are behaviours sufficient to demonstrate intelligence?

Based on the answer to the above questions, four major approaches to pursue AI have been established. These approaches can be summarized into a two by two table as shown in Figure 4.17, where the row is "Human" vs. "Rational", and the column is "Behaviour" vs. "Thought". Positioning at the top right cell, the rational agent approach aims to build agent that perform mathematically perfect acts such that the best expected outcome can always be achieved. In contrast, the "laws of thought" approach focus on understanding the logic behind the rationality. Closely related to cognitive science, the cognitive modelling approach attempts to express theories of human cognition as computer program to mimic the thought process of human. Lastly, the Turing test approach is built upon the famous Turing test proposed by Turing and Haugeland (1950). The test can be roughly described

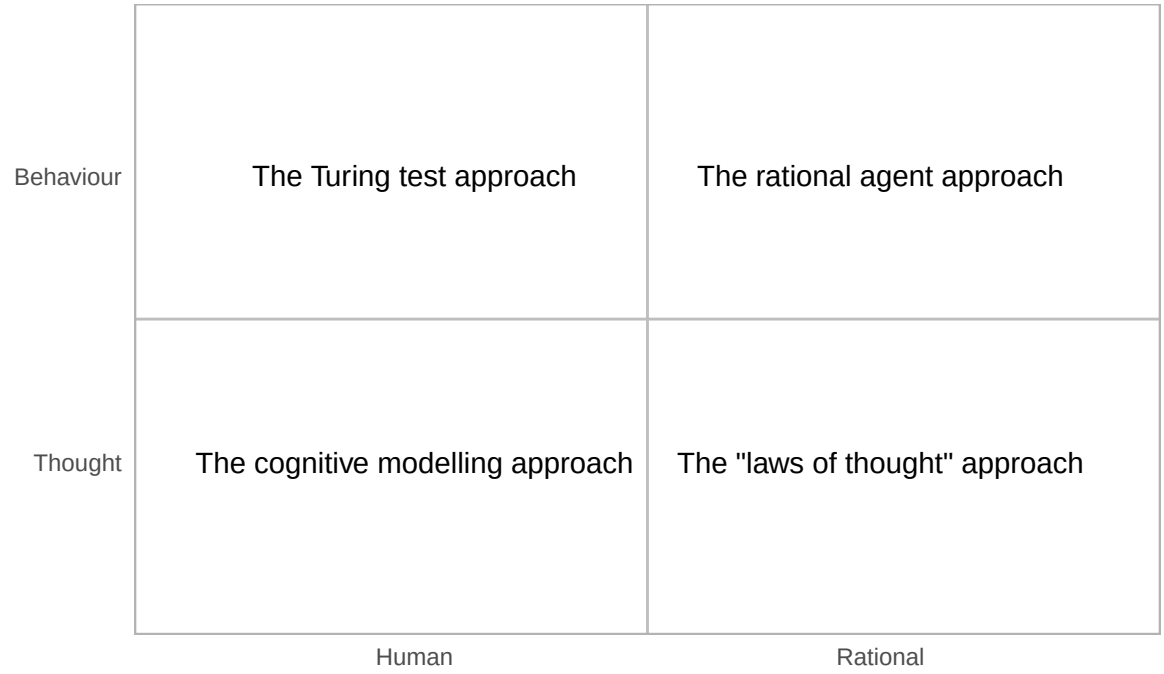


Figure 4.17: *Four possible approaches to pursue AI based on the two dimensions in AI research - human vs. rational and thought vs. behavior (Russell and Norvig, 2002).*

as, whether a human can distinguish another human from a computer with written communications only. To pass the test, several capabilities of computer are required. This includes natural language processing for communication with human, knowledge representation for encoding knowledge, automated reasoning for derivation of conclusions and machine learning for improving AI automatically through experience and data. Some researchers argued that written communication is insufficient to demonstrate intelligence, and some degree of physical simulation of a person is still necessary. One such example is the total Turing test proposed by Harnad (1991). It adds three new requirements to the list, including computer vision, speech recognition and robotics, which are response for interactions with the physical world. Notably, all seven required capabilities have become major subfields of AI today. And their development has made AI one of the fastest-growing fields in the 21st century (Russell and Norvig, 2002).

Considering there are different approaches of AI, it is necessary to clarify the type of AI we aimed to build. Understanding the human thought process or mimicking the human vision mechanism were not of interest in this research. We believed humanly behaviours

was enough for demonstrating AI for the problem we concerned. Hence, two of the capacities required by the total Turing test, namely machine learning and computer vision were adopted.

Another possible method under the rational agent approach was to define distance metrics to measure difference between data plots for making mathematically prefect decisions. This method would work if such metrics can be obtained manually or approximated algorithmically. However, we considered it was too complex to fit into our prototype automatic system. But we will revisited this method in our second project.

4.2.2 Design of the Automatic System

Depending on the desired input(s) and output(s) of the automatic system, the system can be designed in various ways. We considered the most general design would be which took inputs of multiple image files corresponding to data plots of a lineup, and outputted a p-value of the visual test conducted by the system. The human subject experiment was close to this design except the data plots are combined into a single image.

Decision also needed to be made for the input and output format of the computer vision models. If the model took multiple data plots of a lineup as inputs, and outputted the predicted probabilities of the data plot being the most different one, then the post-processing step of the system would be a mapping from a list of numeric values between zero and one to a list of one-hot encoded binary values denoting the selections. This mapping could be an indicator function, i.e. a threshold for deciding whether to select a data plot or not. It could also be a combination of the indicator function and the order statistic to select the M most likely data plots, where M was a user-defined positive integer. In fact, the choice of the mapping function was flexible.

This design was reasonable but we found that it was difficult and slow to train the computer vision model. Instead, a simpler model design was used. The model took only one data plot as input at a time, and predicted the probability of the data plot being a null plot. In this case, the post-processing step was an indicator function for deciding whether the data plot is a null plot. By evaluating all m plots of a lineup, multiple selections can be made. Five different models were included in the system to act as five evaluators, such

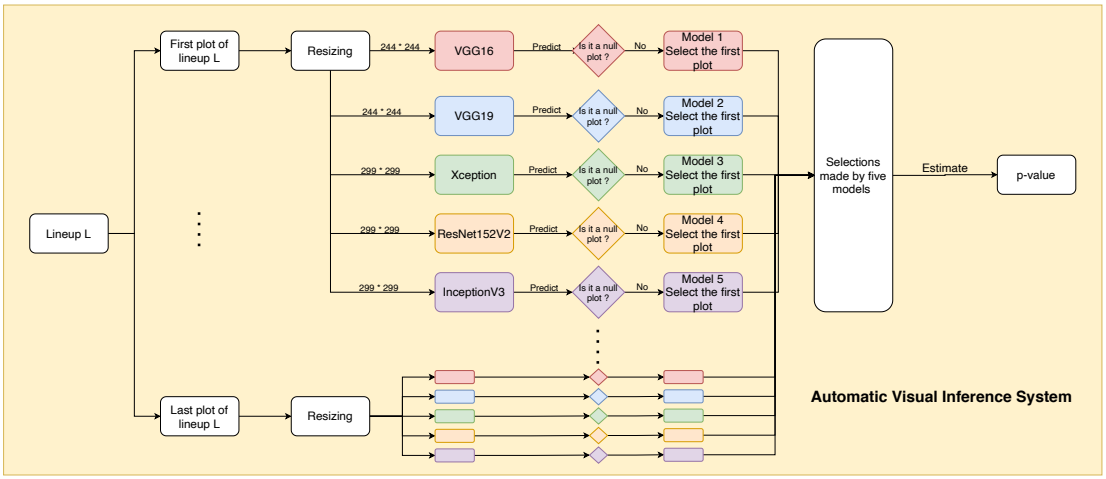


Figure 4.18: *Design of the automatic visual inference system. Every data plot of a lineup was evaluated by five models to decide whether or not to select the data plot. By combining the selections made by five models, p-value can be estimated.*

that the relationship between the number of models and the power of automatic visual test can be studied. A summary of the system design is given in Figure 4.18.

4.2.3 Model Architecture and Model Training

As mentioned in Section 1.6, modern computer vision model is built upon the deep neural network with tens of thousands of parameters. Due to limited computing resources, we did not have the luxury to train a deep neural network from scratch. A technique called transfer learning was used to leverage features learned from an existing problem on our task. It improved the performance of the model by transferring information from one domain to another related domain (Weiss, Khoshgoftaar, and Wang, 2016). The practical workflow was to take parameters from a previously trained model to a new model, and freeze them in training to prevent the knowledge preserved in the parameters from being destroyed. Then, we added some trainable layers on top of the frozen layers to solve the new problem.

By using this workflow, we were not only able to dramatically reduce the training time, but also able to gain better performance because the pre-trained models we used were trained on a large dataset called ImageNet (Deng et al., 2009) with more than 14 million images on 1000 object classes. The famous annual competition ImageNet Large Scale Visual Recognition Challenge (ILSCRV) ran annually by the ImageNet project was a

gold standard for evaluating the architecture of computer vision model. Among the leaderboard of ILSCRB, we picked architectures that were handy to implement and with low top-5 error rate. This included VGG16 (Simonyan and Zisserman, 2015), VGG19 (Simonyan and Zisserman, 2015), Xception (Chollet, 2017), ResNet152V2 (He et al., 2016) and InceptionV3 (Szegedy et al., 2015). The top-5 error rate is the average number the model failed to include the answer among its top five guesses. The overview of all five transfer learning models are given in Table 4.10. Note that the depth is the number of layers in the model.

On top of the pre-trained model, we added a 2D global average pooling layer (Lin, Chen, and Yan, 2014), followed by a fully-connected dense layer with 256 nodes, a dropout layer (Srivastava et al., 2014) with 20% dropout rate, and a output layer with softmax as activation function. A diagram of the model architecture is given in Figure 4.19.

Global average pooling layer was a replacement of traditional fully-connected dense layers in CNN which flattened the feature map into a 1D tensor. Global average pooling layer took the average of the values over all the pixels in a channel, then flattened it into a 1D tensor. This enforced the correspondences between feature maps and categories, which made the feature maps easier to be interpreted as categories confidence maps. Further, unlike the dense layer, there was no parameters that needed to be optimized in global average pooling layer which reduce the risk of overfitting (Lin, Chen, and Yan, 2014).

Dropout was primary used as a regularization technique to address overfitting. In training, the dropout layer would randomly temporarily remove some nodes from the network with fixed probability p, which in our case, we used $p = 20\%$.

Five models were trained on 10000 null plots and 10000 actual data plots simulated using the same parameter settings as the human subject experiments and balanced across cubic model and heteroskedasticity model. 20% training data was used as validation set. Image augmentation was applied to increase the diversity of the dataset during training, with random rotation angle ranged from zero to five degrees and random brightness ranged from 95% to 105%. Optimizer Adam (Kingma and Ba, 2017) was used with learning rate equal to 0.001. The loss function used in the model was binary cross-entropy, which can

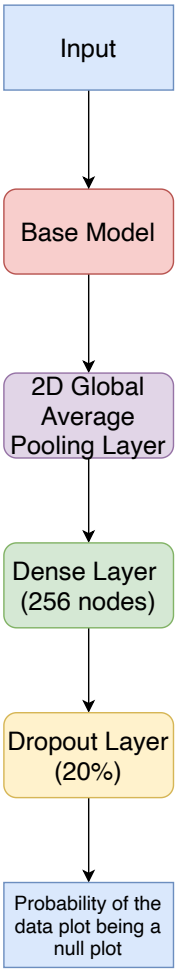


Figure 4.19: *Architecture of the computer vision model.*

be expressed by:

$$H = -N^{-1} \sum_{i=1}^{N} y_i log(\hat{p}_i) + (1 - y_i) log(1 - \hat{p}_i),$$

where N is the number of observations, y_i is a realization of a binary random variable, and $\hat{p}_i$ is the predicted probability.

Early stopping was used to reduce training time and to avoid overfitting. If the validation loss did not decrease for five iterations, then the training procedure would stop.

The trained models were evaluated on 10000 null plots and 10000 actual data plots simulated by the same way as the training data. And the system was then evaluated on 2500 cubic lineups and 2500 heteroskedasticity lineups.

Table 4.10: *Overview of the computer vision models.*

Model	Depth	Total number of parameters	Non-trainable parameters	Trainable parameters	Top-5 error rate
VGG16	20	14846530	14714688	131842	0.901
VGG19	23	20156226	20024384	131842	0.900
Xception	85	21386538	20861480	525058	0.945
ResNet152V2	311	58856706	58331648	525058	0.942
InceptionV3	193	22327842	21802784	525058	0.937

4.2.4 Results

The performance of all five models evaluated on the test set is given in Table 4.11. From the result, we observed that the average balanced accuracy for all five models were around 80%. Breaking down into null and actual data plot, it was found that models had a better performance on null plot detection. Or we could interpret it as our models tended to predict null plot as the answer. Further, models were less likely to predict null plot as the answer when the data was simulated from a cubic model, which resulted in slightly higher detection rate for actual data plot. In overall, these five model were good at detecting null plots but were not excellent in picking up residual plots with model defects especially for heteroskedasticity patterns.

Combining with the overview of models presented in Table 4.10, it was observed that models with large number of layers and parameters did worse in detecting actual data plot. One possible explanation of this phenomenon was the information was highly summarised and features corresponding to patterns might be lost in the depth of the architecture. Other than that, we did not find a strong relationship between detection rate and the meta variables of models.

By integrating all five models into the system, the automatic visual inference system was able to make decision on lineup. The performance of the system was given in Figure 4.20. According to the result, the estimated power of the system was close to the power of a visual test evaluated by two human subjects at large effect size. And for small effect size, the estimated power of the system surpassed the power of human visual test by a large margin, but it was still less powerful than the conventional test. Patterns related to the cubic model were easy to be detected by both human and the system, so both power curves reached one at large effect size. But for heteroskedasticity model, the system could

Table 4.11: *Performance of computer vision models on null residual plot and acutal data plot of test data. Models were tested on 10000 null plots and 10000 actual data plots simulated using the same parameter settings as the human subject experiments and balanced across cubic model and heteroskedasticity model.*

	Actual data plot detection rate	Null plot detection rate	Average detection rate
VGG16			
Cubic	0.7752	0.8662	0.8207
Heteroskedasticity	0.7022	0.9038	0.8030
VGG19			
Cubic	0.8006	0.8210	0.8108
Heteroskedasticity	0.8052	0.8416	0.8234
Xception			
Cubic	0.7442	0.9182	0.8312
Heteroskedasticity	0.7298	0.9244	0.8271
ResNet152V2			
Cubic	0.7252	0.9168	0.8210
Heteroskedasticity	0.6612	0.9298	0.7955
InceptionV3			
Cubic	0.6984	0.9304	0.8144
Heteroskedasticity	0.6398	0.9494	0.7946

not wisely tell the difference between the null plot and an actual data plot. We observed the similar result from the detection rate in Table 4.11.

We also verified the type-I error of the system to see whether it was lower than the desired significance level imposed. It was done by evaluating 5000 lineups without any model defects. The results showed that the false positive rate was around 0.003, which was smaller than 0.05. This suggested that we slightly overestimated the p-value of the automatic visual test, which leaded to less rejection of the null hypothesis. A simple way to address this issue was to relax the significance level. It was found that if we set $\alpha = 0.1$, then the false positive rate was around 0.015. And if we set $\alpha = 0.2$, then the rate increased to around 0.4. The reason of the overestimation of the p-value was worth to be researched in future studies.

If we reduced the number of models used in the system, the estimated power of the automatic visual test would change. Figure 4.21 showed five power curves corresponding to number of models ranged from one to five. It was found that as the number of models increased, the estimated power of the system increased, but the amount of improvement decreased. In fact, from Figure 4.21, we would not expect the estimated power increased a lot when the number of models greater than five.

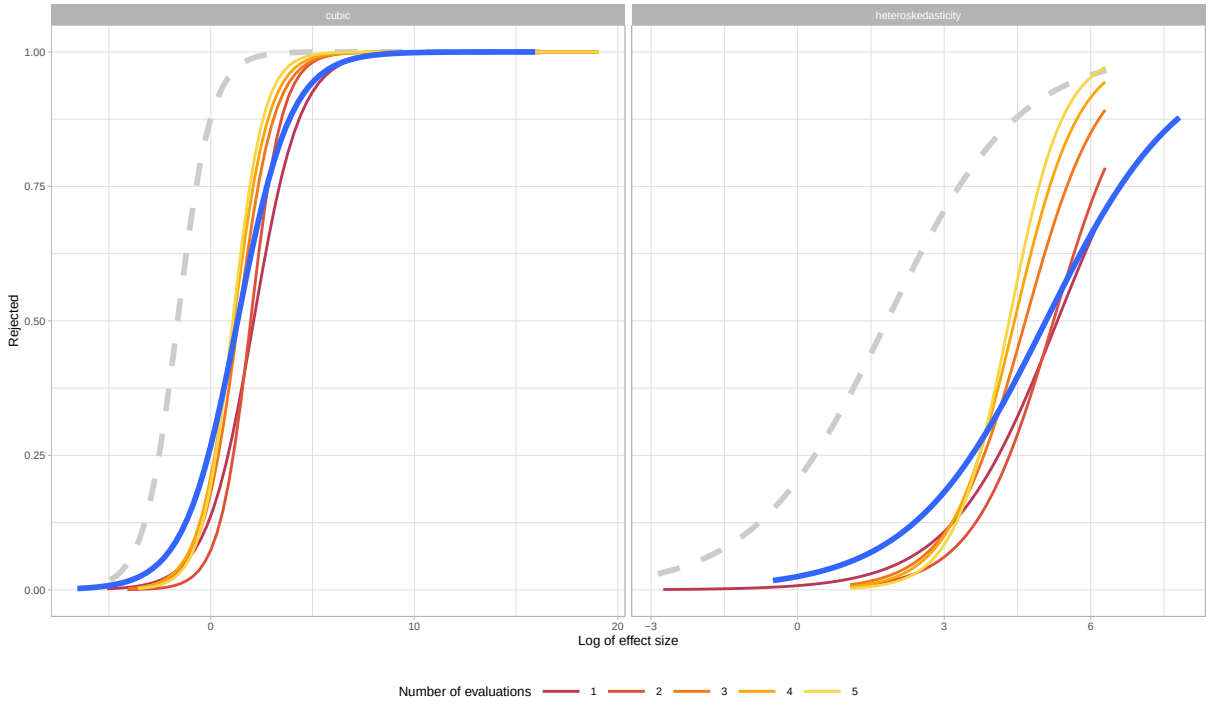


Figure 4.20: *Estimated power of the automatic visual inference system evaluated on 2500 lineups and 2500 lineups simulated using the cubic model and heteroskedasticity model respectively against the natural logarithm of the effect size. The dashed curves are power of conventional tests, the blue curves are the estimated power of the system, and the curves in other colours are the estimated power obtained from the experiments.*

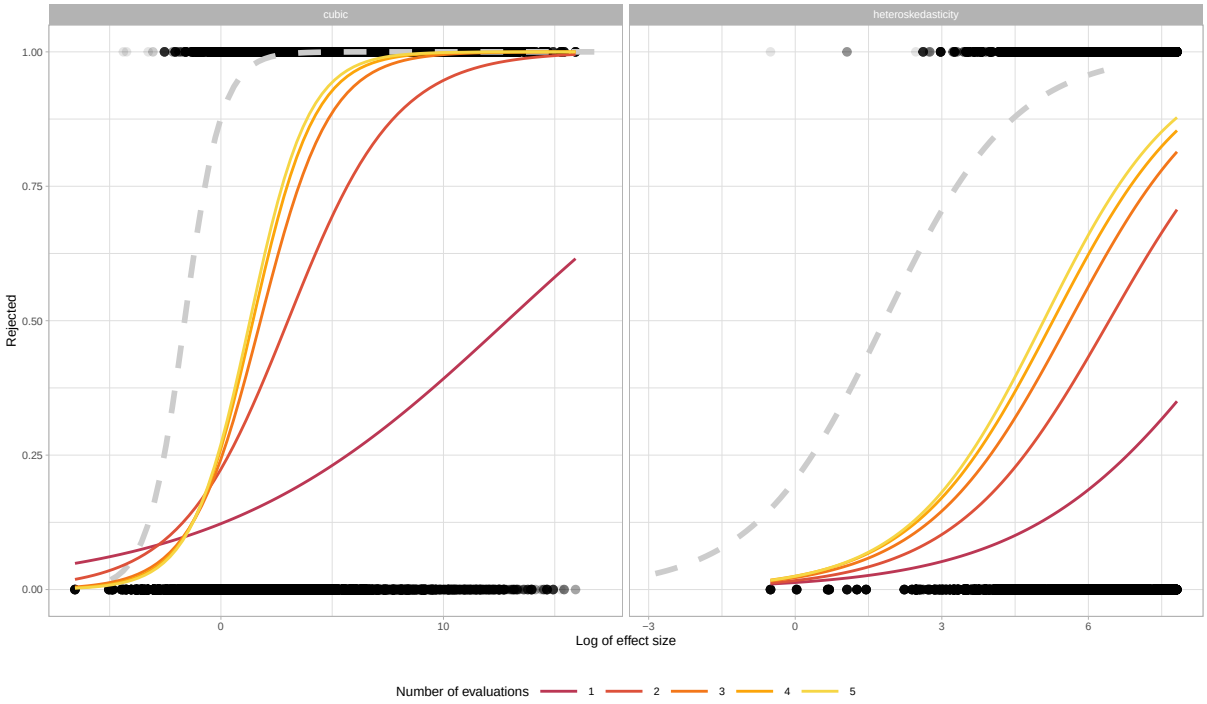


Figure 4.21: *Estimated power of the automatic visual inference system for number of evaluation ranged from one to five and evaluated on 2500 lineups and 2500 lineups simulated using the cubic model and heteroskedasticity model respectively against the natural logarithm of the effect size. The dashed curves are power of conventional tests, the curves in other colours are the estimated power of the system with different number of evaluations. Decision made the system were drawn with black marker.*

Chapter 5

General-Purpose Automatic Visual Statistical Inference System

The term content-based image retrieval (CBIR) was first introduced by Kato (1992) as a field of study which aims to retrieve image from large-scale image databases based on information derived from the image itself, such as colour, shape and texture (Rafiee, Dlay, and Woo, 2010). According to Zhou, Li, and Tian (2017), reverse image search is one of the CBIR query technique and is sometimes called query by example (QBE), which performs image search based on the image example provide by the user. Since QBE is the most intuitive query formation, it has been widely explored in the research on CBIR and search engines such as Google Image (Google, 2022) and TinEye (TinEye, 2022) provide accessible search by image services to public. In CBIR, the most common method for comparing images is using image similarity or image distance metrics, which takes into account various features extracted from the image (Zhou, Li, and Tian, 2017).

By adopting the techniques developed in query by example content-based image retrieval, we aims to build automatic general-purpose visual inference system based on image comparison in the second project. General-purpose means that we will not make any assumption about the type of the plot and the data generating process such that the system will be able to evaluate any lineups asking for selecting the most different plot(s).

Chapter 6

OAVIS: An online automatic visual inference system

Online machine learning is a technique to train machine learning model with data arriving in a sequential order. Models trained with online learning can be updated immediately for any new arrival data, which overcomes the drawbacks of the traditional batch learning where training data required to be available prior to the training. It is suitable for large-scale machine learning tasks in real-world data analytics applications [@].

The third projects aims to deploy the automatic visual inference system as an online application at a public-accessible website. The web application will evaluate lineups uploaded by users, and return the p-value with the selections made by the system. Meanwhile, as an option, users will be asked to also evaluate the lineup and give their selections. With the user selection data, the system will use online machine learning theory to update the built-in computer vision models such that the performance of the system can be improved.

Chapter 7

Timeline

Date		Event
2021		
	March - August	Design the study website and the experiments
	September	Apply for human research ethics approval
	October	Test the study website
	November	Conduct the first two human subject experiments
	December	Develop the *R* package *visage*
2022		
	January	Develop the computer vision models
	February	Prepare for the first milestone
	March	Extend the scope of violations of classical normal linear regression considered by the computer vision model
	April	Conduct human subject experiment for the extended model defects
	May	Conclude the first project: draft and submit the first paper
	June	Review previous studies in image comparison
	July	Draft a new automatic visual inference system based on image comparison
	August - November	Build the automatic system
	November - December	Conduct human subject experiment for validating the automatic system
	December	Attend the Australasian Applied Statistics Conference
2023		
	January - March	Finish the second project: prepare milestone material and submit the second paper
	April - December	Deploy the automatic visual inference system as a web application
2024		
	January - March	Finish the third project: finalize the thesis

Bibliography

Breusch, TS and AR Pagan (1979). A simple test for heteroscedasticity and random coefficient variation. *Econometrica: Journal of the econometric society*. Publisher: JSTOR, 1287–1294.

Brunetti, A, D Buongiorno, GF Trotta, and V Bevilacqua (2018). Computer vision and deep learning techniques for pedestrian detection and tracking: A survey. *Neurocomputing* **300**. Publisher: Elsevier, 17–33.

Buja, A, D Cook, H Hofmann, M Lawrence, EK Lee, DF Swayne, and H Wickham (2009). Statistical inference for exploratory data analysis and model diagnostics. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **367**(1906), 4361–4383.

Buja, A, D Cook, and D Swayne (1999). Inference for data visualization. In: *Joint Statistics Meetings, August*.

Chen, Y, S Su, and H Yang (2020). Convolutional neural network analysis of recurrence plots for anomaly detection. *International Journal of Bifurcation and Chaos* **30**(01). Publisher: World Scientific, 2050002.

Chollet, F (2017). Xception: Deep Learning with Depthwise Separable Convolutions. *arXiv:1610.02357 [cs]*. arXiv: 1610.02357.

Chu, H, X Liao, P Dong, Z Chen, X Zhao, and J Zou (2019). An automatic classification method of well testing plot based on convolutional neural network (CNN). *Energies* **12**(15). Publisher: Multidisciplinary Digital Publishing Institute, 2846.

Cleveland, WS and R McGill (1984). Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods. *Journal of the American Statistical Association* **79**(387), 531–554.

De Leeuw, JR (2015). jsPsych: A JavaScript library for creating behavioral experiments in a Web browser. *Behavior research methods* **47**(1), 1–12.

Deng, J, W Dong, R Socher, LJ Li, K Li, and L Fei-Fei (2009). Imagenet: A large-scale hierarchical image database. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp.248–255.

Donoho, D (2017). 50 Years of Data Science. *Journal of Computational and Graphical Statistics* **26**(4), 745–766.

Emami, S and VP Suciu (2012). Facial recognition using OpenCV. *Journal of Mobile, Embedded and Distributed Systems* **4**(1), 38–43.

Flanagan, D (2006). *JavaScript: the definitive guide*. " O'Reilly Media, Inc.".

Fukushima, K and S Miyake (1982). "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition". In: *Competition and cooperation in neural nets*. Springer, pp.267–285.

Gelman, A (2003). A Bayesian Formulation of Exploratory Data Analysis and Goodness-of-fit Testing*. en. *International Statistical Review* **71**(2), 369–382. (Visited on 03/06/2022).

Gelman, A (2004). Exploratory Data Analysis for Complex Models. en. *Journal of Computational and Graphical Statistics* **13**(4), 755–779.

Google (2022). *Google Image*. https://www.google.com/imghp.

Grinberg, M (2018). *Flask web development: developing web applications with python*. " O'Reilly Media, Inc.".

Gtihub (2022a). *Github*. https://www.github.com/.

Gtihub (2022b). *Github Pages*. https://pages.github.com/.

Hailesilassie, T (2019). Financial Market Prediction Using Recurrence Plot and Convolutional Neural Network.

Harnad, S (1991). Other bodies, other minds: A machine incarnation of an old philosophical problem. *Minds and Machines* **1**(1). Publisher: Springer, 43–54.

Hatami, N, Y Gavet, and J Debayle (2018). Classification of time-series images using deep convolutional neural networks. In: *Tenth international conference on machine vision (ICMV 2017)*. Vol. 10696. International Society for Optics and Photonics, pp.106960Y.

He, K, X Zhang, S Ren, and J Sun (2016). Identity Mappings in Deep Residual Networks. *arXiv:1603.05027 [cs]*.

Hofmann, H, L Follett, M Majumder, and D Cook (2012). Graphical Tests for Power Comparison of Competing Designs. *IEEE Transactions on Visualization and Computer Graphics* **18**(12). Conference Name: IEEE Transactions on Visualization and Computer Graphics, 2441–2448.

Kato, T (1992). Database architecture for content-based image retrieval. In: *image storage and retrieval systems*. Vol. 1662. International Society for Optics and Photonics, pp.112–123.

Kingma, DP and J Ba (2017). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*.

Krishnan, G and H Hofmann (2021). Hierarchical Decision Ensembles- An inferential framework for uncertain Human-AI collaboration in forensic examinations. *arXiv:2111.01131 [cs, stat]*.

LeCun, Y, B Boser, JS Denker, D Henderson, RE Howard, W Hubbard, and LD Jackel (1989). Backpropagation applied to handwritten zip code recognition. *Neural computation* **1**(4). Publisher: MIT Press, 541–551.

Lee, H and YPP Chen (2015). Image based computer aided diagnosis system for cancer detection. *Expert Systems with Applications* **42**(12), 5356–5365.

Lin, M, Q Chen, and S Yan (2014). Network In Network. *arXiv:1312.4400 [cs]*.

Loy, A, L Follett, and H Hofmann (2016). Variations of Q–Q Plots: The Power of Our Eyes! *The American Statistician* **70**(2), 202–214.

Loy, A and H Hofmann (2013). Diagnostic tools for hierarchical linear models. en. *WIREs Computational Statistics* **5**(1), 48–61.

Majumder, M, H Hofmann, and D Cook (2013). Validation of Visual Statistical Inference, Applied to Linear Models. *Journal of the American Statistical Association* **108**(503), 942–956.

Majumder, M, H Hofmann, and D Cook (2014). Human Factors Influencing Visual Statistical Inference. *arXiv:1408.1974 [stat]*.

Ojeda, SAA, GA Solano, and EC Peramo (2020). Multivariate time series imaging for short-term precipitation forecasting using convolutional neural networks. In: *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. IEEE, pp.33–38.

Prolific (2022). *Prolific*. https://www.prolific.co.

PythonAnywhere (2022). *PythonAnywhere*. https://www.pythonanywhere.com.

R Core Team (2021). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Rafiee, G, S Dlay, and W Woo (2010). A review of content-based image retrieval. In: *2010 7th International Symposium on Communication Systems, Networks Digital Signal Processing (CSNDSP 2010)*, pp.775–779.

Rawat, W and Z Wang (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation* **29**(9), 2352–2449.

Roy Chowdhury, N, D Cook, H Hofmann, M Majumder, EK Lee, and AL Toth (2015). Using visual statistical inference to better understand random class separations in high dimension, low sample size data. en. *Computational Statistics* **30**(2), 293–316.

Russell, S and P Norvig (2002). Artificial intelligence: a modern approach.

Simonyan, K and A Zisserman (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv:1409.1556 [cs]*.

Singh, K, G Gupta, L Vig, G Shroff, and P Agarwal (2017). Deep convolutional neural networks for pairwise causality. *arXiv preprint arXiv:1701.00597*.

Srivastava, N, G Hinton, A Krizhevsky, I Sutskever, and R Salakhutdinov (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* **15**(1), 1929–1958.

Szegedy, C, V Vanhoucke, S Ioffe, J Shlens, and Z Wojna (2015). Rethinking the Inception Architecture for Computer Vision. *arXiv:1512.00567 [cs]*.

TinEye (2022). *TinEye*. https://www.tineye.com.

Turing, AM and J Haugeland (1950). Computing machinery and intelligence. *The Turing Test: Verbal Behavior as the Hallmark of Intelligence*, 29–56.

Van Rossum, G and FL Drake (2009). *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.

Weiss, K, TM Khoshgoftaar, and D Wang (2016). A survey of transfer learning. en. *Journal of Big Data* **3**(1), 9.

Wickham, H (2011). ggplot2. en. *WIREs Computational Statistics* **3**(2), 180–185.

Widen, HM, JB Elsner, S Pau, and CK Uejio (2016). Graphical Inference in Geographical Research. en. *Geographical Analysis* **48**(2), 115–131.

Wilkinson, L, A Anand, and R Grossman (2005). Graph-theoretic scagnostics. In: *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.* ISSN: 1522-404X, pp.157–164.

Zhang, Y, Y Hou, S Zhou, and K Ouyang (2020). Encoding time series as multi-scale signed recurrence plots for classification using fully convolutional networks. *Sensors* **20**(14). Publisher: Multidisciplinary Digital Publishing Institute, 3818.

Zhou, W, H Li, and Q Tian (2017). Recent Advance in Content-based Image Retrieval: A Literature Survey. *arXiv:1706.06064 [cs]*.