ARTICLE TEMPLATE

Automated reading of residual plots with computer vision models

Weihao Li[a]

[a]Department of Econometrics and Business Statistics, Monash University, Clayton, VIC, Australia

ARTICLE HISTORY

Compiled January 23, 2024

ABSTRACT

TBD.

KEYWORDS

TBD

CONTACT Weihao Li. Email: weihao.li@monash.edu

Contents

1. Introduction

The practice of plotting residuals is commonly regarded as a standard procedure in linear regression diagnostics (see Cook and Weisberg 1982; Belsley, Kuh, and Welsch 1980). This visual assessment plays a crucial role in identifying deviations from model assumptions, such as linearity, homoscedasticity, and normality. It also helps in understanding the goodness of fit and various characteristics of the model.

Generating a residual plot in most statistical software is often as straightforward as executing a line of code or clicking a button. However, accurately interpreting a residual plot can be challenging. Consider Figure 1 as an example, the residuals display a triangular shape pointing to the left. While this might suggest heteroskedasticity, it is important to avoid over-interpreting the visual pattern. In this case, the fitted model is correctly specified, and the triangular shape is actually a result of the skewed distribution of the predictors, rather than indicating a flaw in the model.

A residual plot can exhibit various visual features, but it is crucial to recognize that some may arise from the characteristics of predictors and the inherent randomness of the error, rather than indicating a violation of model assumptions (Li et al. 2023). The concept of visual inference, as proposed by Buja et al. (2009), provides an inferential framework to assess whether residual plots indeed contain visual patterns inconsistent with the model assumptions. The fundamental idea involves testing whether the actual residual plot visually differs significantly from null plots, which are created using residuals generated from the null distribution. Typically, this is accomplished through the lineup protocol. In this approach, the real residual plot is embedded within a lineup alongside several null plots. If the real residual plot can be distinguished from the lineup, it provides evidence for rejecting the null hypothesis.

The practice of delivering a residual plot as a lineup is generally regarded as a valuable approach. Beyond its application in residual diagnostics, the lineup protocol has integrated into the analysis of diverse subjects. For instance, Loy and Hofmann (2013, 2014, 2015) illustrate its applicability in diagnosing hierarchical linear models. Additionally, Widen et al. (2016) demonstrates its utility in geographical research, while Krishnan and Hofmann (2021) explores its effectiveness in forensic examinations.

However, as pointed out by Li et al. (2023), a primary limitation of the lineup protocol lies in its reliance on human judgments. Unlike conventional statistical tests that can be performed numerically and automatically in statistical software, the lineup protocol requires human evaluation of images. This characteristic makes it less suitable for large-scale applications, given the associated high labour costs and time requirements.

There is a substantial need to develop an approach that alleviates people's workload by automating repetitive tasks and providing standardized results in a controlled environment. The large-scale evaluation of lineups ihs impractical without the use of technology and machines.

The utilization of computers to interpret data plots has a rich history, with early efforts such as "Scagnostics" by Tukey and Tukey (1985), focusing on scatterplot diagnostics. Wilkinson, Anand, and Grossman (2005) expanded on this work, introducing graph theoretic scagnostics, which encompassed nine computable measures applied to planar proximity graphs. These measures, including, but not limited to, "Outlying", "Skinny", "Stringy", "Straight", "Monotonic", "Skewed", "Clumpy", and "Striated" aimed to characterize outliers, shape, density, trend, coherence and other characteristics of the data. While this approach has been inspiring, there is a recognition (Buja et al. 2009) that it may not capture all the necessary visual features that differentiate actual residual plots from null plots. A more promising alternative entails enabling computers to learn the function for extracting visual features from residual plots. Essentially, this means empowering computers to discern the crucial visual features for residual diagnostics and determining the method to extract them. Modern computer vision models are well-suited for addressing this challenge.

Modern computer vision models often rely on deep neural networks with convolutional layers (Fukushima and Miyake 1982). These layers leverage hierarchical patterns in data, downsizing and transforming images by summarizing information in a small space. Numerous studies have demonstrated the efficacy of convolutional layers in addressing various vision tasks, including image recognition (Rawat and Wang 2017). Despite the widespread use of computer vision models in fields like computer-aided diagnosis (Lee and Chen 2015), pedestrian detection (Brunetti et al. 2018), and facial recognition (Emami and Suciu 2012), their application in reading data plots remains

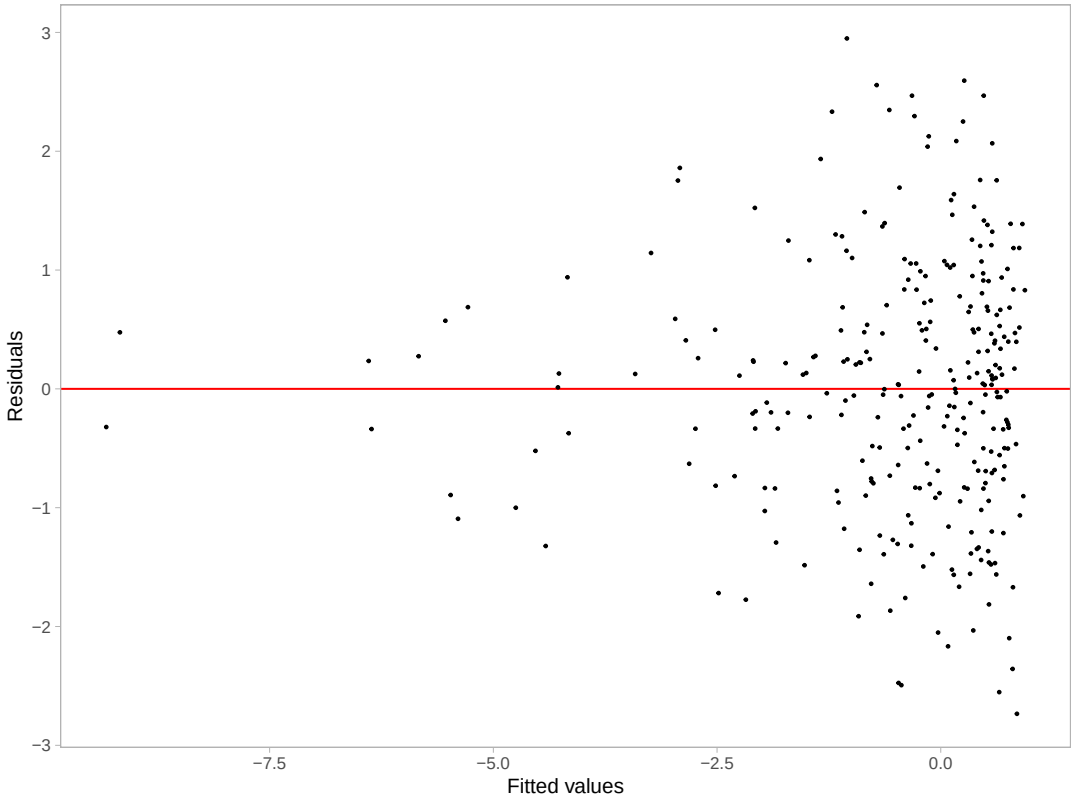

Figure 1. An example residual vs fitted values plot (red line indicates 0). The vertical spread of the data points varies with the fitted values. This often indicates the existence of heteroskedasticity.

limited. While some studies have explored the use of computer vision models for tasks such as reading 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), the application of reading residual plots with computer vision models represents a relatively new field of study.

In this paper, we develop computer vision models and integrate them into the residual plots diagnostics workflow, filling the gap of.... The paper is structured as follows: ...

2. Methodogology

2.1. *Different possible configurations of the model formula*

In addressing a problem, there exist multiple avenues for solutions. Various configurations of the computer vision model formula can effectively read residual plots, and these configurations can be categorized based on two key components of the model formula: the input and the output format.

2.1.1. Different input formats

In computer vision models, the input plays a role similar to predictors in a linear regression model. The quality and relevance of the input data greatly influence the model's capacity to generate insightful and meaningful results. A straightforward approach involves feeding the model a vector of residuals along with a vector of fitted values, essentially providing all the necessary details for creating a residuals vs fitted values plot. However, a drawback of this method is the dynamic input size, as different fitted regression models may have varying numbers of observations. In modern computer vision models created with mainstream software like TensorFlow (ref here), the input is typically a fixed-shape tensor. One solution is to pad the input vector with leading or trailing zeros if the input tensor expects a longer vector. However, this approach may fail if the input vector surpasses the designed length. Another strategy is to summarize the residuals and fitted values separately using histograms and utilize the counts as the input. By controlling the number of bins in the histograms, it becomes possible to provide a fixed-length input vector.

The primary advantage of using an image as input, as opposed to a vector input format, relates to the development of sophisticated image processing architectures over the years (see ref here). These architectures effectively capture and summarize spatial information from nearby pixels, which is less straightforward with vector input. The only considerations are the image resolution and the aesthetics of the residual plot. In general, higher resolution provides more information to the model but comes with the trade-off of increased complexity and greater difficulty in training. As for the aesthetics of the residual plot, a practical solution is to consistently present residual plots in the

same style to the model. This implies that the model can not accept arbitrary images as input but requires the use of the same preprocessing software to convert provided residuals and fitted values into a standardized style for the residual plot.

Other possible input formats include, but are not limited to, a pair of residual plots, a triplet, and a lineup. Existing computer vision models designed for image comparison can assess whether a pair of images are similar or dissimilar (as seen in LeCun's 2005 paper). Applied to our specific problem, we can define null plots of a fitted model to be similar to each other, while considering actual residual plots to be distinct from null plots of any fitted model. A triplet constitutes a set of three images, denoted as $image_1$, $image_2$ and $image_3$. In computer vision models, triplets are often used to predict whether $image_2$ or $image_3$ is more similar to $image_1$, proving particularly useful for establishing rankings between samples. In the context of our problem, we can apply the same criteria to define similarity between images. However, it's important to note that these two approaches usually require additional considerations regarding the loss function and, at times, non-standard training processes due to shared weights between different convolutional blocks.

Presenting a lineup to a model and asking it to predict which residual plot is the most different one aligns closely with the lineup protocol. However, if the lineup consists of around 20 plots, similar to previous human subject experiments (reference here), the resolution of the input image could become quite large, posing challenges in training the model. In fact, we experimented with this approach in a pilot study, and the model's performance was suboptimal.

We did not explore all the mentioned input formats due to the considerable costs associated with data preparation and model training. Considering the implementation cost and the interpretability of the model, we settled on the single residual plot input format.

2.1.2. Different output formats

Given that the input is a single residual plot represented as a fixed-resolution image, the computer vision model's output can take one of two formats: binary or numeric. This choice determines whether the model is a classification model or a regression

model. The binary outcome might indicates whether the input image is a null plot or not, or whether the input image would be rejected in a visual test conducted by humans. The latter option requires data from prior human subject experiments, posing challenges in controlling the quality and quantity of data due to variations in experimental settings across different studies. Additionally, some visual inference experiments are unrelated to linear regression models and residual plots, resulting in a limited amount of available training data.

Alternatively, the output could be a meaningful numeric measure, such as the average distance to good residual plots. This approach necessitates defining a distance measure between residual plots, which may be a non-trivial task. However, once a meaningful distance measure is established, the computer vision model's prediction becomes an interpretable value with tangible significance. Studies have demonstrated that defining a proper distance between images can enhance the matching accuracy in image search compared to a binary outcome model (ref here).

The following section will propose the distance measure used for training the computer vision model.

2.2. *Distance from the good residual plots*

In a visual test, the observer will be asked to choose one or more plots that stand out as most distinct from others in a given lineup. To develop a computer vision model for evaluating residual plots within the visual inference framework, it is important to precisely define a numerical measure of "difference" or "distance" between plots. This distance can take the form of a basic statistical operation on pixels, such as the sum of square differences. Alternatively, it could involve established image similarity metrics like the Structural Similarity Index Measure (Wang et al. 2004). The challenge lies in the fact that metrics tailored for image comparison may not be suitable for evaluating data plots, where only essential plot elements require assessment (Chowdhury et al. 2018). Furthermore, scagnostics mentioned in Section 1 could be used to construct distance metrics for data plots comparison, but the functional form still needs to be carefully refined to accurately reflect the extent of the violations.

2.2.1. Residual distribution

The distance metrics proposed in this paper takes into account the fact that we try to measure how different a residual plot is from a good residual plot, or in other words, how different a given fitted model is from a correctly specified model. For the classical normal linear regression model, residuals are derived from the fitted values $\hat{\boldsymbol{y}}$ and observed values $\boldsymbol{y}$. Suppose the data generating process is known and the model is correctly specified, by the Frisch-Waugh-Lowell theorem (Frisch and Waugh 1933), residuals $\boldsymbol{e}$ can also be written as a linear transformation of the error $\boldsymbol{\varepsilon}$ formulated as $\boldsymbol{e} = \boldsymbol{R}\boldsymbol{\varepsilon}$, where $\boldsymbol{R} = \boldsymbol{I}_n - \boldsymbol{X}(\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'$ is the residual operator, $\boldsymbol{X}$ is the design matrix, $\boldsymbol{I}_n$ is a n by n identity matrix, and n is the number of observations.

One of the assumptions of the classical normal linear regression model is the error $\boldsymbol{\varepsilon}$ follows a multivariate normal distribution with zero mean and constant variance, i.e., $\boldsymbol{\varepsilon} \sim N(\boldsymbol{0}_n, \sigma^2\boldsymbol{I}_n)$. It can be known that residuals $\boldsymbol{e}$ also follow a certain probability distribution transformed from the multivariate normal distribution, which will be denoted as Q. This reference distribution Q summarises what good residuals should follow given the predictors are known and fixed.

When fitting a linear regression model, the solver will force $\sum_{i=1}^{n} e_i = 0$, making any residual value to be a linear combination of the remaining $n-1$ residuals. This effectively means $rank(\boldsymbol{R}) = n - 1 < n$ and Q becomes a degenerate multivariate distribution. To capture the characteristics of Q, such as moments, we can simulate a large numbers of $\boldsymbol{\varepsilon}$ and transform it to $\boldsymbol{e}$ to get the empirical estimates. For simplicity, we replaced $\boldsymbol{R}$ with a full-rank diagonal matrix $diag(\boldsymbol{R})$, where $diag(.)$ set the non-diagonal entries of a matrix to zeros. The resulting distribution for $\boldsymbol{Q}$ is $N(\boldsymbol{0}_n, diag(\boldsymbol{R}\sigma^2))$.

Distribution Q is derived from the correctly specified model. However, if the model is misspecified, then the actual distribution of residuals denoted as P, will be different to Q. For example, if the data generating process contains variables correlated with any column of $\boldsymbol{X}$ but not included in $\boldsymbol{X}$, causing an omitted variable problem, P will be different to Q because the residual operator obtained from the fitted model will not be the same as $\boldsymbol{R}$. Besides, if the $\boldsymbol{\varepsilon}$ follows a non-normal distribution such as a multivariate lognormal distribution, the empirical residual distribution will usually be

skewed and has a long tail.

2.2.2. Kullback-Leibler divergence of P from Q

Define a proper distance between distributions is usually easier than define a proper distance between data plots. Given the actual residual distribution Q and the reference residual distribution P, we used a distance metric based on Kullback-Leibler divergence (Kullback and Leibler 1951) to quantify the difference between two distributions

$$D = log\left(1 + KL\right), \tag{1}$$

$$KL = \int_{\mathbb{R}^n} log\frac{p(\boldsymbol{e})}{q(\boldsymbol{e})}p(\boldsymbol{e})d\boldsymbol{e}, \tag{2}$$

where $p(.)$ is the probability density function for distribution P, and $q(.)$ is the probability density function for distribution Q.

This distance metric was first proposed in Li et al. (2023). It was mainly designed for measuring the effect size of non-linearity and heteroskedasticity in a residual plot. Li et al. (2023) showed that, for a classical normal linear regression model that omits a necessary higher-order predictors $\boldsymbol{Z}$, and incorrectly assumes $\boldsymbol{\varepsilon} \sim N(\boldsymbol{0}_n, \sigma^2\boldsymbol{I}_n)$ while in fact $\boldsymbol{\varepsilon} \sim N(\boldsymbol{0}_n, \boldsymbol{V})$, Q can be represented as $N(\boldsymbol{RZ\beta}_z, diag(\boldsymbol{RVR'}))$. Note that the variance-covariance matrix is replaced with the diagonal matrix to ensure it is a full-rank matrix.

Since both P and Q are adjusted to be multivariate normal distributions, equation 2 can be further expanded to

$$KL = \frac{1}{2}\left(\log\frac{|\text{diag}(\boldsymbol{W})|}{|\text{diag}(\boldsymbol{R}\sigma^2)|} - n + \text{tr}(\text{diag}(\boldsymbol{W})^{-1}\text{diag}(\boldsymbol{R}\sigma^2)) + \boldsymbol{\mu}_z'(diag(\boldsymbol{W}))^{-1}\boldsymbol{\mu}_z\right), \tag{3}$$

where $\boldsymbol{\mu}_z = \boldsymbol{RZ\beta}_z$, and $\boldsymbol{W} = \boldsymbol{RVR'}$. The assumed error variance σ^2 is set to be $tr(\boldsymbol{V})/n$, which is the expectation of the estimated variance.

2.2.3. Evaluation of Kullback-Leibler divergence for non-normal P

For non-normal error $\boldsymbol{\varepsilon}$, the actual residual distribution P is unlikely to be a multivariate normal distribution. Thus, equation 3 given in Li et al. (2023) will not be applicable to models with non-normality violations.

To evaluate the Kullback-Leibler divergence of non-normal P from Q, the fallback is to solve equation 2 numerically. However, since $\boldsymbol{e}$ is a linear transformation of non-normal random variables, it is very common that the general form of P is unknown, meaning that we can not easily compute $p(\boldsymbol{e})$ using a well-known probability density function. Additionally, even if $p(\boldsymbol{e})$ can be calculated for any $\boldsymbol{e} \in \mathbb{R}^n$, it will be very difficult to do numerical integration over the n dimensional space, because n could be potentially very large.

In order to evaluate equation 2 in a practically computable manner, the elements of $\boldsymbol{e}$ are assumed to be independent of each other. This assumption solves both of the issues mentioned above. First, we no longer need to integrate over n random variables. The result of equation 2 is now the sum of the Kullback-Leibler divergence evaluated for each individual residual thanks for the independence assumption. Second, it is not required to know the joint probability density $p(\boldsymbol{e})$ any more. Instead, the evaluation of Kullback-Leibler divergence for an individual residual relies on the knowledge of the marginal density $p_i(e_i)$, where e_i is the i-th residual for $i = 1, ..., n$. This is much easier to estimate through simulation.

The algorithm for computing equation 2 starts from simulating m sets of $\boldsymbol{\varepsilon}$ according to the error distribution. The simulated errors are stored in a matrix $\boldsymbol{A}$ with n rows and m columns. So each column of $\boldsymbol{A}$ is a set of realization values of $\boldsymbol{\varepsilon}$. Then, we can get m sets of $\boldsymbol{e}$ stored in the matrix $\boldsymbol{B}$ by applying the residual operator $\boldsymbol{B} = \boldsymbol{RA}$. Furthermore, kernel density estimation (KDE) with Gaussian kernel and optimal bandwidth selected by the Sliverman's rule of thumb (Silverman 2018) is applied on each row of B to estimate $p_i(e_i)$ for $i = 1, ..., n$. The KDE computation is done by the `density` function in R.

Since the Kullback-Leibler divergence can be viewed as the expectation of the log-likelihood ratio between distribution P and distribution Q evaluated on distribution P, we can reuse the simulated residuals in matrix $\boldsymbol{B}$ to estimate the expectation by

the sample mean. With the independence assumption, for non-normal P, KL can be estimated by

$$\hat{KL} = \sum_{i=1}^{n} \hat{KL}_i, \tag{4}$$

$$\hat{KL}_i = \frac{1}{m} \sum_{j=1}^{m} log \frac{\hat{p_i}(B_{ij})}{q(B_{ij})}, \tag{5}$$

where $\hat{KL}_i$ is the estimator of the Kullback-Leibler divergence for an individual residual e_i, B_{ij} is the i-th row and j-th column entry of the matrix B, $\hat{p_i}(.)$ is the kernel density estimator of $p_i(.)$, $q(.)$ is the normal density function with mean zero and an assumed variance estimated as $\hat{\sigma^2} = \sum_{b \in vec(B)} (b - \sum_{b \in vec(B)} b/nm)^2/(nm-1)$, and $vec(.)$ is the vectorization operator which turns a $n \times m$ matrix into a $nm \times 1$ column vector by stacking the columns of the matrix on top of each other.

2.2.4. Approximation of the distance metric

In the previous sections, we have defined a distance metric given in equation 1 for quantifying the difference between the actual residual distribution P and an ideal reference distribution Q. You may have noticed that this distance metric can only be computed when the data generating process is known. In reality, we often have no knowledge about the data generating process, otherwise we do not need to fit a linear regression model in the first place.

What we proposed in this paper is a method to approximate this distance with a residual plot. Let D be the result of equation 1 indicating the extent of the model violations, and our estimator $\hat{D}$ is formulated as

$$\hat{D} = cv(plot_{h \times w}(\boldsymbol{e}, \hat{\boldsymbol{y}})), \tag{6}$$

where $\boldsymbol{e}$ is a vector of residuals obtained from the fitted model rather than a vector of random variables, $\hat{\boldsymbol{y}}$ is the fitted values also obtained from the fitted model, $plot(.)_{h \times w}$

is a plotting function that generates a residuals vs fitted values plot with fixed aesthetic and then saves it as an image with $h \times w$ pixels and three colour channels, and $cv(.)$ is a computer vision model which takes an $h \times w$ image as input and predicts the distance in the domain $[0, +\infty)$.

With the approximated distance $\hat{D}$, we will be able to know how different the underlying distribution of the residuals is from a good residual distribution. This is meaningful way to check if a residual plot is a good residual plot, and to know the strength of the visual signal embedded in the residual plot.

The approximated distance $\hat{D}$ is not expected to be the same as the original distance D. This is largely because information contained in a single residual plot is limited and it may not be able to summarise all the characteristics of the residual distribution. For a given residual distribution P, we can generate many different residual plots. Some of them share similar visual patterns, but some of them could be visually very different from the rest, especially for models with small n. This suggests the error of the approximation will vary depends on whether the observed residual plot is representative or not.

2.3. *Statistical testing based on the approximated distance*

2.3.1. Null distribution of the approximated distance

Having a computer vision model cv which gives an approximated distance between 0 and $+\infty$ is not enough to determine if the null hypothesis that the model is correctly specified should be rejected. Theoretically, the distance D for a correctly specified model is 0, because P will be the same as Q, and $log(1 + 0) = 0$. The computer vision model may not necessary predict 0 for a null plot. Using 1 as an example, it contains a visual pattern which is an indication of heteroskedasticity. We would not expect the model to be able to magically tell if the suspicious pattern is caused by the skewed distribution of the fitted values or the existence of heteroskedasticity. Assume we only train the model with this single image, while 50% of the time the label is 0, and 50% of the time the label is some constant greater than 0. Then, a perfect model will predict a value between 0 and that constant to achieve a minimum loss. So, it

is expected that our computer vision models will also predict value greater than 0 for some null plots. It will make more sense if we interpret the model prediction as the visual signal strength of the residual plot. Some null plots could have outliers or strong visual patterns due to randomness, and the model will try to summarise those information into the prediction.

To align with the principle of visual inference, we need to provide a valid null distribution for the test statistic. If we treat the approximated distance $\hat{D}$ as a test statistic, then the null distribution of this statistic can be approximated by the empirical distribution of $\hat{D}$ for a large amount of null plots of given a fitted model. The procedure involves applying the residual rotation technique (Buja et al. 2009) on the fitted model to obtain null residuals. The null residuals are then used to make null plots and fed into the model. The results are used to construct an empirical distribution.

2.3.2. Estimation of quantiles of the null distribution

With the null distribution of the test statistic,

- How many null samples are needed to get stable 95% quantile
- How do we make decision based on the quantile (reject or not reject)
- p-value calculation (percentage of null vss > vss)

2.3.3. Bootstrapping the approximated distance

- What is the assumption of bootstrapping
- Any drawback: loss of information
- For each bootstrapped fitted model, we can get a bootstrapped vss and construct a new 95% quantile for the null distribution, and check if H_0 should be rejected
- However, it is very expensive to construct a new 95% quantile for each bootstrapped fitted model
- The 95% quantile constructed using the original fitted model can be considered as an estimate of those new 95% quantiles
- We borrow information from the original fitted model
- We can check how many bootstrapped fitted model should be rejected (like the

percentage)

- This gives an overall estimate of how often the assumed regression model are considered to be incorrect if the data can be obtained repetitively from the same data generating process
- If this percentage is relatively large, it gives the analyst an warning that the regression model is inappropriate to the data

2.4. *Generation of training data*

2.4.1. Data generating process

While observational data is frequently employed in training models for real-world applications, the data generating process of observational data often remains unknown, making computation for our target variable D unattainable. Consequently, the computer vision models developed in this study were trained using synthetic data. This approach provides us with precise label annotations. Additionally, it ensures a large and diverse training dataset, as we have control over the data generating process, and the simulation of the training data is relatively cost-effective.

We have incorporated three types of residual departures of linear regression model in the training data, including non-linearity, heteroskedasticity and non-normality. All three departures can be summarised by the data generating process formulated as

$$\boldsymbol{y} = \mathbf{1}_n + \boldsymbol{x}_1 + \beta_1 \boldsymbol{x}_2 + \beta_2(\boldsymbol{z} + \beta_1 \boldsymbol{w}) + \boldsymbol{k} \odot \boldsymbol{\varepsilon}, \tag{7}$$

$$\boldsymbol{z} = He_j(g(\boldsymbol{x}_1, 2)), \tag{8}$$

$$\boldsymbol{w} = He_j(g(\boldsymbol{x}_2, 2)), \tag{9}$$

$$\boldsymbol{k} = \sqrt{\mathbf{1}_n + b(2 - |a|)(\boldsymbol{x}_1 + \beta_1 \boldsymbol{x}_2 - a\mathbf{1}_n)^2}, \tag{10}$$

where $\boldsymbol{y}$, $\boldsymbol{x}_1$, $\boldsymbol{x}_2$, $\boldsymbol{z}$, $\boldsymbol{w}$, $\boldsymbol{k}$ and $\boldsymbol{\varepsilon}$ are vectors of size n, $\mathbf{1}_n$ is a vector of ones of size n, $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are two independent predictors, $He_j(.)$ is the jth-order probabilist's Hermite polynomials (Hermite 1864), the $\sqrt{(.)}$ and $(.)^2$ operators are element-wise operators, $\odot$ is the Hadamard product, and $g(., k)$ is a scaling function to enforce the support of

the random vector to be $[-k, k]^n$ defined as

$$g(\boldsymbol{x}, k) = 2k \cdot \frac{\boldsymbol{x} - x_{min}\mathbf{1}_n}{x_{max} - x_{min}} - k\mathbf{1}_n, \ for\ k > 0,$$

where $x_{min} = \underset{i \in \{1,\ldots,n\}}{min} x_i$, $x_{max} = \underset{i \in \{1,\ldots,n\}}{max} x_i$ and x_i is the i-th entry of $\boldsymbol{x}$.

The residuals and fitted values of the fitted model is obtained by regressing $\boldsymbol{y}$ on $\boldsymbol{x}_1$. This data generating process is adopted from Li et al. (2023) where it was used to simulate residual plots with non-linearity and heteroskedasticity visual patterns for human subject experiments.

In equation 7, predictor $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are independent of each other, so excluding $\boldsymbol{x}_2$ from the regression formula will not cause any problem. However, $\boldsymbol{z}$ and $\boldsymbol{w}$ are higher-order terms of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. If $\beta_2 \neq 0$, the regression model will suffer from non-linearity issues. Parameter j is a shape parameter controlling the number of tuning points of the non-linearity pattern. Generally, greater values of j will result in more tuning points.

Additionally, Scaling factor $\boldsymbol{k}$ directly affects the error distribution and it is correlated with $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. If $b \neq 0$ and $\boldsymbol{\varepsilon} \sim N(\mathbf{0}_n, \sigma^2 \boldsymbol{I}_n)$, the constant variance assumption will be violated. Parameter a is a shape parameter controlling the location of the smallest variance in a residual plot.

Non-normality violations are introduced by defining a non-normal distribution for $\boldsymbol{\varepsilon}$.

- distribution of x1 and x2
- summary table of all the factor values
- example plots for the violations

2.4.2. Computation of scagnostics

We have mentioned in Section 1 that scagnostics summarise some useful characteristics of the scatter plot. Although the computer vision model will learn the feature extraction function during the training process, it will be beneficial to utilize scagnostics to help computer vision model making more accurate predictions as these measure could

provide additional useful information.

For each generated residual plot, four scagnostics, namely, "Monotonic", "Sparse", "Splines" and "Striped" are computed using the `cassowaryr` R package (ref here). The computed measures along with the number of observations of the fitted model are provided as the second input of the computer vision model.

Other scagnostics are also informative, but unfortunately, they are unavailable due to a fatal bug in the compiled C program of the `interp` R package (ref here) that will crash the process in a unpredictable manner. For reproducibility, we do not include these scagnostics in the training data.

2.4.3. Crafting a balanced training set

In order to train a robust computer vision model, we intentionally control the distribution of the target variable D of the training data. Making it uniform between 0 and 7. This is accomplished by preparing 50 buckets each only accepts training samples with D between $[7(i-1)/49, 7i/49)$ for $i < 50$, where i is the index of the i-th bucket. For the 50-th bucket, any training samples with $D >= 7$ will be accepted. We have prepared 80000 training images, so each bucket can only contain $80000 \div 50 = 1600$ training samples. The simulator will repeatedly sample parameter values from the parameter sample, generate residuals and fitted values using the data generating process, compute the distance, and check if the sample can be accepted by the corresponding bucket, until all the buckets are full.

2.5. *Architecture of the computer vision model*

- optimization (MSE loss function, with equation)
- a diagram
- explain the existence of removable layers
- explain three different resolutions

2.6. *Training process and hyperparameter tuning*

- software environment (tf and keras)

- tuner (keras_tuner)
- 80% validation data, 20% training data
- 100 trials (Bayesian optimizer)
- monitor RMSE
- early stopping (50 epochs)
- learning rate reduction (10 epochs, 0.5 factor)
- max 2000 epochs (no one reach)
- restore best epoch of the best trial

2.7. *Model evaluation methods*

- RMSE for the test set
- R^2
- Mean bias deviation to understand the overall bias
- quantile loss to understand how well the model captures the entire distribution of D
- Percentage of predictions within a tolerance interval (like 0.1)

3. Results

3.1. *Best model performance*

- Metrics for model performance
- Shap values
- Heatmap

3.2. *Comparison with human visual inference*

3.2.1. Overview of the human subject experiment

3.2.2. Comparison

- power comparison
- decisions

3.3. *When the model works*

- simple examples (non-linearity, heteroskedasticity, . . .)
- datasaurus

3.4. *When the model does not work*

- human detect but model does not
- cartoon residuals?

3.5. *Workflow: how one use this model? (small showcase)*

4. Dicussion

There are other kinds of residual departures like autocorrelation that are not considered in this study. The primary goal of this paper is to establish a new way of evaluating residual plot and conducting visual test with computer vision models. Building computer vision models for other model violations and other types of diagnostic plots could be future directions of this field.

5. Conclusion

- Summary of findings
- Contributions to the field
- Future directions for research

References

Belsley, David A, Edwin Kuh, and Roy E Welsch. 1980. *Regression diagnostics: Identifying influential data and sources of collinearity*. John Wiley & Sons.

Brunetti, Antonio, Domenico Buongiorno, Gianpaolo Francesco Trotta, and Vitoantonio Bevilacqua. 2018. "Computer vision and deep learning techniques for pedestrian detection and tracking: A survey." *Neurocomputing* 300: 17–33.

Buja, Andreas, Dianne Cook, Heike Hofmann, Michael Lawrence, Eun-Kyung Lee, Deborah F

Swayne, and Hadley 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.

Chen, Yun, Shijie Su, and Hui Yang. 2020. "Convolutional neural network analysis of recurrence plots for anomaly detection." *International Journal of Bifurcation and Chaos* 30 (01): 2050002.

Chowdhury, Niladri Roy, Dianne Cook, Heike Hofmann, and Mahbubul Majumder. 2018. "Measuring lineup difficulty by matching distance metrics with subject choices in crowdsourced data." *Journal of Computational and Graphical Statistics* 27 (1): 132–145.

Chu, Hongyang, Xinwei Liao, Peng Dong, Zhiming Chen, Xiaoliang Zhao, and Jiandong Zou. 2019. "An automatic classification method of well testing plot based on convolutional neural network (CNN)." *Energies* 12 (15): 2846.

Cook, R Dennis, and Sanford Weisberg. 1982. *Residuals and influence in regression*. New York: Chapman and Hall.

Emami, Shervin, and Valentin Petrut Suciu. 2012. "Facial recognition using OpenCV." *Journal of Mobile, Embedded and Distributed Systems* 4 (1): 38–43.

Frisch, Ragnar, and Frederick V Waugh. 1933. "Partial time regressions as compared with individual trends." *Econometrica: Journal of the Econometric Society* 387–401.

Fukushima, Kunihiko, and Sei Miyake. 1982. "Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position." *Pattern recognition* 15 (6): 455–469.

Hailesilassie, Tameru. 2019. "Financial Market Prediction Using Recurrence Plot and Convolutional Neural Network." .

Hatami, Nima, Yann Gavet, and Johan Debayle. 2018. "Classification of time-series images using deep convolutional neural networks." In *Tenth international conference on machine vision (ICMV 2017)*, Vol. 10696, 242–249. SPIE.

Hermite, M. 1864. *Sur un nouveau développement en série des fonctions*. Imprimerie de Gauthier-Villars.

Krishnan, Ganesh, and Heike Hofmann. 2021. "Hierarchical Decision Ensembles-An inferential framework for uncertain Human-AI collaboration in forensic examinations." *arXiv preprint arXiv:2111.01131* .

Kullback, Solomon, and Richard A Leibler. 1951. "On information and sufficiency." *The Annals of Mathematical Statistics* 22 (1): 79–86.

Lee, Howard, and Yi-Ping Phoebe Chen. 2015. "Image based computer aided diagnosis system for cancer detection." *Expert Systems with Applications* 42 (12): 5356–5365.

Li, Weihao, Dianne Cook, Emi Tanaka, and Susan VanderPlas. 2023. "A Plot is Worth a Thousand Tests: Assessing Residual Diagnostics with the Lineup Protocol." *arXiv preprint arXiv:2308.05964* .

Loy, Adam, and Heike Hofmann. 2013. "Diagnostic tools for hierarchical linear models." *Wiley Interdisciplinary Reviews: Computational Statistics* 5 (1): 48–61.

Loy, Adam, and Heike Hofmann. 2014. "HLMdiag: A suite of diagnostics for hierarchical linear models in R." *Journal of Statistical Software* 56: 1–28.

Loy, Adam, and Heike Hofmann. 2015. "Are you normal? The problem of confounded residual structures in hierarchical linear models." *Journal of Computational and Graphical Statistics* 24 (4): 1191–1209.

Ojeda, Sun Arthur A, Geoffrey A Solano, and Elmer C 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)*, 33–38. IEEE.

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

Silverman, Bernard W. 2018. *Density estimation for statistics and data analysis*. Routledge.

Singh, Karamjit, Garima Gupta, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. 2017. "Deep convolutional neural networks for pairwise causality." *arXiv preprint arXiv:1701.00597* .

Tukey, John W, and Paul A Tukey. 1985. "Computer graphics and exploratory data analysis: An introduction." In *Proceedings of the sixth annual conference and exposition: computer graphics*, Vol. 85, 773–785.

Wang, Zhou, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. "Image quality assessment: from error visibility to structural similarity." *IEEE transactions on image processing* 13 (4): 600–612.

Widen, Holly M, James B Elsner, Stephanie Pau, and Christopher K Uejio. 2016. "Graphical inference in geographical research." *Geographical Analysis* 48 (2): 115–131.

Wilkinson, Leland, Anushka Anand, and Robert Grossman. 2005. "Graph-theoretic scagnostics." In *Information Visualization, IEEE Symposium on*, 21–21. IEEE Computer Society.

Zhang, Ye, Yi Hou, Shilin Zhou, and Kewei Ouyang. 2020. "Encoding time series as multi-

scale signed recurrence plots for classification using fully convolutional networks." *Sensors* 20 (14): 3818.