

1 Automated Residual Plot Assessment with the R Package 2 autovi and the Shiny App autovi.web

3 Weihao Li^{1,2}, Dianne Cook¹, Emi Tanaka², Susan VanderPlas³ and Klaus
4 Ackermann¹

5 Monash University, The Australian National University and University of
6 Nebraska

Summary

Visually assessing Visual assessment of residual plots is a common advice for linear model diagnostics; however this approach requires manual human evaluation and thereby is not scalable for assessing many models. Human evaluation also has the potential to produce inconsistent decisions from different analysts. Using a approach for diagnosing linear models, but it relies on manual evaluation which does not scale well and can lead to inconsistent decisions across analysts. The lineup protocol, where the residual plot is embedded which embeds the observed plot among null plots, can help to alleviate inconsistency, reduce subjectivity but requires even more human effort. This is the type of task that in In today's world we might employ a robot to do the tedious work for a human. Here we describe data-driven world, such tasks are well-suited for automation. We present a new R package that includes uses a computer vision model for automated assessment to automate the evaluation of residual plots, and an An accompanying Shiny app is provided for ease of use. For a user-provided Given a sample of residuals, it predicts a measure of the model predicts a visual signal strength (VSS) and provides a suite of offers supporting information to assist the analyst decide on the appropriateness their help analysts assess model fit.

Key words: initial data analysis; statistical graphics; data visualization; visual inference;
8 computer vision; machine learning; hypothesis testing; regression analysis;
model diagnostics

¹ Department of Econometrics and Business Statistics, Monash University, Wellington Road, VIC 3800, Australia

² Biological Data Science Institute, The Australian National University, 46 Sullivan's Creek Road, ACT 2600, Australia

³ Department of Statistics, University of Nebraska, Hardin Hall, 3310 Holdrege St Suite 340, Lincoln, NE 68583, United States

Email: patrick.li@anu.edu.au

9

1. Introduction

Regression analysis is a widely used statistical modeling technique for data in many fields. There is a vast array of software for conducting regression modeling and generating diagnostics. The package `lmtest` (Zeileis & Hothorn 2002) provides a suite of conventional tests. ~~The~~, while the `stats` package (R Core Team 2022) offers standard diagnostic plots such as residuals vs. fitted values, quantile-quantile (Q-Q) plots, and residuals vs. leverage plots. ~~Packages~~ Additional packages like `jtools` (Long 2022), `olsrr` (Hebbali 2024), `rockchalk` (Johnson 2022), and `ggResidpanel` (Goode & Rey 2019) ~~provide~~ deliver similar graphical diagnostics, often with alternative enhanced aesthetics or interactive features. ~~All of these tools deliver the types of~~ These tools collectively produce the core diagnostic plots outlined in the classical text by Cook & Weisberg (1982). The `ecostats` package (Warton 2023) incorporates extends these diagnostics by incorporating simulation envelopes into residual plots; while Meanwhile, `DHARMa` (Hartig 2022) compares empirical quantiles (0.25, 0.5, and 0.75) of scaled residuals to their theoretical counterparts. ~~DHARMa is particularly focused on detecting, with a strong focus on identifying~~ model violations such as heteroscedasticity, ~~incorrect misspecified~~ functional forms, and issues specific to generalized linear and mixed-effect models, like over/under-dispersion. It also includes provides conventional test annotations to help avoid reduce the risk of misinterpretation.

However, relying solely on subjective assessments of these plots can lead to issues such as over-interpreting random patterns as model violations. Li et al. (2024a) demonstrated that visual ~~methods~~ inference methods, particularly those using the lineup protocol (Buja et al. 2009) for assessing residuals are more useful, and also perform more practically, offer more practical and reliable assessments of residual patterns than conventional tests due to their reduced sensitivity to, as they are less sensitive to minor departures. Packages such as `nullabor` (Wickham et al. 2020), `HLMdiag` (Loy & Hofmann 2014), and `regressinator` (Reinhart 2024), enable support this approach by enabling users to compare observed residual plots with plots of samples from null distributions, generated under null hypothesis, thereby helping to quantify the significance of any detected patterns.

However, as discussed As noted in Li et al. (2024b), the lineup protocol has significant limitations in large-scale applications due to dependence its reliance on human labor. Thus To overcome this constraint, a computer vision model was developed with an associated alongside a corresponding statistical testing procedure to automate

44 ~~autmoate~~ the assessment of residual plots. ~~This model takes~~ ~~The model takes as~~
 45 ~~input~~ a residual plot and a ~~vector set~~ of auxiliary variables (such as the number of
 46 observations) ~~as inputs and outputs~~ ~~the and outputs~~ a predicted visual signal strength
 47 (VSS). This ~~strength estimates the distance~~ ~~VSS estimates the degree of deviation~~
 48 between the residual distribution of the fitted ~~regression~~-model and the reference
 49 distribution ~~assumed expected~~ under correct model specification.

50 To make the statistical testing procedure and trained computer vision model widely
 51 accessible, we developed the R package ~~autovi~~, ~~and a~~ ~~along with a companion~~ web
 52 interface, ~~autovi.web~~ ~~to make it easy for~~, ~~which allows~~ users to automatically ~~read~~
 53 ~~assess~~ their residual plots ~~with using~~ the trained computer vision model.

54 The remainder of this paper is structured as follows: Section ~~2 introduces the definition~~
 55 ~~and computation of visual signal strength. Section 3 expands on the computation of~~
 56 ~~the null and bootstrapped residuals. Section 4~~ provides a detailed documentation of
 57 the `autovi` package, including its usage and infrastructure. Section 5 focuses on the
 58 `autovi.web` interface, describing its design and usage, along with illustrative examples.
 59 Finally, Section 6 presents the main conclusions of this work.

60 2. Definition and computation of visual signal strength

61 To train a computer vision model, ~~a~~ a measure of the visible pattern in a plot is
 62 needed. We call this the **visual signal strength** (VSS) ~~measuring~~, ~~which measures~~
 63 how prominently a specific set of visual patterns appears in an image. This can be
 64 computed for a training set of data, and plots, where the generating distributions are
 65 specified.

66 In the context of regression model diagnostics, ~~it~~ ~~VSS~~ describes the clarity of visual
 67 patterns on a diagnostic plot that may indicate model violations. ~~This concept~~
 68 ~~Violations~~ can be categorized as weak, moderate, or strong. ~~However, in our study,~~ ~~but~~
 69 ~~here~~ we treat it as a continuous positive real variable. Importantly, its interpretation
 70 depends on how it is linked to a function of the data or the underlying data generating
 71 process. Consequently, the calculation of VSS can ~~be~~ vary across different model classes
 72 or within the same model, depending on the generating function.

73 VSS ~~is defined as an estimate of a distance measure that quantifies the disparity~~
 74 ~~estimates the distance~~ between the residual distribution of a fitted classical normal
 75 linear regression model and a reference distribution, ~~and more details can be found in~~

76 Li et al. (2024b). This (see Li et al. 2024b, for details). The distance measure is based
 77 on the Kullback-Leibler (KL) divergence:

$$D = \log(1 + D_{KL}),$$

78 where D_{KL} is given by:

$$\underline{D_{KL} = \int_{\mathbb{R}^n} \log \frac{p(\mathbf{e})}{q(\mathbf{e})} p(\mathbf{e}) d\mathbf{e}},$$

$$\underline{\underline{D_{KL} = \int_{\mathbb{R}^n} \log \frac{p(\mathbf{e})}{q(\mathbf{e})} p(\mathbf{e}) d\mathbf{e}}}, \quad (1)$$

80 here, $p(\cdot)$ and $q(\cdot)$ are the probability density functions of the reference residual
 81 distribution P and the true residual distribution Q , respectively.

82 This distance measure requires depends on knowledge of the true residual distribution,
 83 which is typically unknown. Therefore, a unknown in practice. To compute D_{KL}
 84 for the training samples, Equation 1 takes different forms depending on the specific
 85 model violations. For instance, where necessary higher-order predictors, \mathbf{Z} , and their
 86 corresponding parameter, $\boldsymbol{\beta}_Z$, are omitted from the fitted linear model, the distance
 87 measure can be expanded as follows:

$$\underline{\underline{D_{KL} = \frac{1}{2} (\boldsymbol{\mu}_z^\top (\text{diag}(\mathbf{R}\sigma^2))^{-1} \boldsymbol{\mu}_z)}},$$

88 where $\boldsymbol{\mu}_z = \mathbf{R}\mathbf{Z}\boldsymbol{\beta}_z$, $\mathbf{R} = \mathbf{I}_n - \mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top$ and \mathbf{X} is the design matrix of the
 89 regression model.

90 The computer vision model approximates this mapping from a set of residuals to
 91 its corresponding distance measure. It is trained on a large number of synthetic
 92 regression models, each designed to simulate specific violations of classical linear
 93 regression assumptions. These models incorporate non-linearity through Hermite
 94 polynomial transformations of predictors, heteroskedasticity by making the error
 95 variance a predictor-dependent function, and non-normality by drawing residuals
 96 from distributions such as discrete, uniform, and lognormal. Both simple and
 97 multiple linear regression structures are used, with controlled parameters to generate

98 diverse and complex residual patterns. Since the data-generating process is known,
99 the distance measure D can be explicitly calculated, enabling supervised training.
100 The computer vision model was used for estimating the residual distribution
101 (see Li et al. 2024b, for more details). takes a residual plot as input and outputs the
102 corresponding distance measure, learning to quantify model violations directly from
103 visual patterns. Additional details are provided in Li et al. (2024b).

104 3. Definition and simulation of null and bootstrapped residuals

105 In the subsequent sections, we will frequently refer to null residuals and bootstrapped
106 residuals, so it is helpful to first define and explain how they are generated.

107 **Null residuals** are used to generate null plots within the lineup protocol framework,
108 serving as the foundation for the statistical testing in our automated residual plot
109 assessment. Specifically, they represent residuals generated under the null hypothesis
110 that the model is correctly specified. A common method for simulating null residuals
111 in linear regression involves sampling from a normal distribution with mean zero and
112 variance equal to the estimated variance of the error term. These simulated residuals
113 and their corresponding plots depict what one would expect from a correctly specified
114 model. If the true residual plot exhibits noticeable deviations from these null plots, it
115 may suggest model misspecification.

116 Our computer vision model is trained to assign lower VSS to null plots and higher VSS
117 to plots that display distinct patterns. Accordingly, statistical testing is performed by
118 computing the proportion of null plots whose VSS equals or exceeds that of the
119 observed residual plot. This proportion serves as a p-value for a one-sided hypothesis
120 test.

121 **Bootstrapped residuals** are obtained by refitting the model on bootstrap samples,
122 which are generated by sampling individual observations with replacement from the
123 original dataset. The residual plots obtained from these refitted models are evaluated
124 using the same computer vision model. The predicted VSS from the bootstrapped
125 plots provide an empirical estimate of the variation in the VSS of the observed
126 residual plot. By examining the proportion of bootstrapped plots that also exhibit
127 significant violations, we can assess whether the original conclusion is robust to
128 sampling variability.

129

4. R package: autovi

130 The main purpose of `autovi` is to provide rejection decisions and p -values for testing
 131 the null hypothesis (H_0) that the regression model is correctly specified. The package
 132 provides automated interpretation of residual plots using computer vision. The name
 133 `autovi` stands for **a**utomated **v**isual **i**nference.

134 ~~There are two ways to access the package, directly using R. This functionality can be~~
 135 ~~accessed through the R package `autovi`, or through a web interface, `autovi.web`. The~~
 136 ~~web interface has the advantage that it can be used without installing Python, R and~~
 137 ~~the relevant packages, which enables users to perform analyses without installing R,~~
 138 ~~Python, or their associated dependencies~~ locally.

139 4.1. Motivation~~for usage~~

140 Figure 1 shows three sets of plots of residuals against fitted values. The simulated
 141 example in (a) might be interpreted as a heteroscedastic pattern, however the
 142 automated reading would predict this to have a visual signal strength (VSS) of
 143 1.53, with a corresponding p -value of 0.25. This means it would be interpreted as
 144 a good residual plot, that there is nothing in the data to indicate a violation of
 145 model assumptions. Skewness in the predictor variables is generating the apparent
 146 heteroscedasticity, where the smaller variance in residuals at larger fitted values is
 147 due to smaller sample size only. The Breusch-Pagan test (Breusch & Pagan 1979) for
 148 heteroscedasticity would also not reject this as good residual plot.

149 The data in (b) is generated by fitting a linear model predicting `mpg` based on `hp`
 150 using the `datasets::mtcars`. It is a small data set, and there is a hint of nonlinear
 151 structure not captured by the model. The automated plot reading would predict a
 152 VSS of 3.57, which has a p -value less than 0.05. That is, the nonlinear structure is
 153 most likely real, and indicates a problem with the model. The conventional test, a
 154 Ramsey Regression Equation Specification Error Test (RESET) (Ramsey 1969) would
 155 also strongly detect the nonlinearity.

156 The third example is generated using the `surreal` package (Balamuta 2024)~~where~~
 157 ~~structured residuals are hidden in data, to be revealed if the correct model is specified.~~
 158 ~~Here a quote based on Tukey is used as the residual structure, where structure~~
 159 ~~residuals are embedded in the data. In this case, a quote inspired by Tukey, “visual~~
 160 ~~summaries focus on unexpected values”, is used to define the residual structure.~~ The
 161 automated plot reading predicts the VSS to be 5.87, with a p -value less than 0.05. ~~This~~

162 ~~structure is blindingly obvious visually~~ Visually, the structure is strikingly clear, but a
 163 RESET test for nonlinear structure would not report a problem. (It would be detected
 164 by a Breusch-Pagan for heteroscedasticity and also Shapiro-Wilk test (Shapiro & Wilk
 165 1965) for non-normality.)

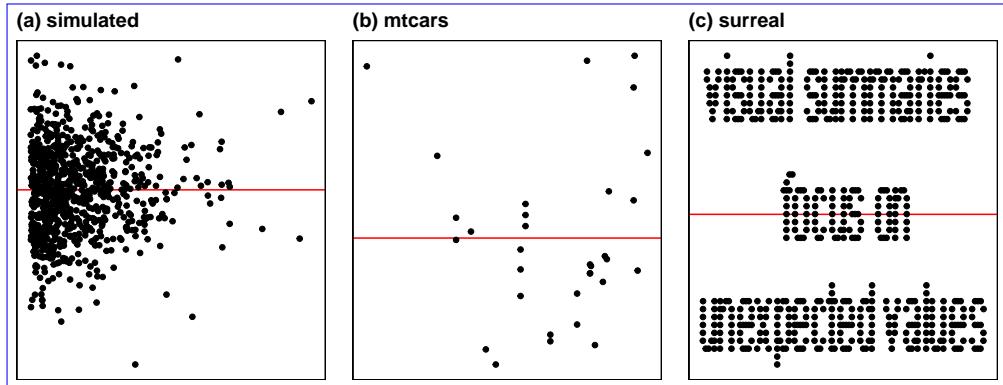


Figure 1. Reading residual plots can be a difficult task, particularly for students new to statistical modeling. The `autovi` package makes it easier. Here are three examples of residual plots, which may appear to have structure. According to `autovi`, the visual signal strengths (VSS) of these three examples are approximately (a) 1.53, (b) 3.57, (c) 5.87, resulting in (b), (c) being significant violations of good residuals, but (a) is consistent with a good residual plot.

166 4.2. Implementation

167 The `autovi` package is built on the `bandicoot` object-oriented programming (OOP)
 168 system (Li 2024), marking a departure from R's traditional S3 generic system. This
 169 OOP architecture enhances flexibility and modularity, allowing users to redefine key
 170 functions through method overriding. ~~While similar functionality could be achieved
 171 using R's S3 system with generic functions, the OOP framework offers a more
 172 structured and extensible foundation for the package.~~

173 The `autovi` infrastructure effectively integrates multiple programming languages and
 174 libraries into a comprehensive analytical tool. It relies on five core libraries from
 175 Python and R, each playing a critical role in the analysis pipeline. In Python, `pillow`
 176 (Clark et al. 2015) handles image processing tasks such as reading and resizing PNG
 177 files of residual plots, then converting them into input tensors for further analysis. ~~The~~
 178 `TensorFlow` (Abadi et al. 2016) ~~library~~, a key component of modern machine learning,
 179 is used to predict the VSS of these plots ~~through using~~ a pre-trained convolutional
 180 neural network.

181 In the R environment, `autovi` utilizes several libraries. `ggplot2` (Wickham 2016)
 182 generates the initial residual plots, saved as PNG files for visual input. ~~The `cassowaryr`~~
 183 ([Mason et al. 2022](#)) ~~library~~ computes scagnostics (scatter plot diagnostics), providing
 184 numerical features that capture statistical properties of the plots. These scagnostics
 185 complement the visual analysis by offering quantitative metrics as secondary input to
 186 the computer vision model. ~~The `reticulate` (Ushey, Allaire & Tang 2024) package~~
 187 ~~bridges R and Python, enabling enables~~ seamless communication between ~~the two~~
 188 ~~languages and supporting the integrated infrastructure~~[R and Python](#).

189 **4.3. Installation**

190 The `autovi` package is available on CRAN. It is actively developed and maintained,
 191 with the latest updates accessible on GitHub. ~~The code discussed in this paper is~~
 192 ~~based on~~ [This paper uses](#) `autovi` version 0.4.1.

193 [2.](#) The package includes internal functions to check the current Python environment
 194 used by the `reticulate` package. If the necessary Python packages are not installed in
 195 the Python interpreter, an error will be raised. If you want to select a specific Python
 196 environment, you can do so by calling the `reticulate::use_python()` function before
 197 using the `autovi` package.

198 We recommend using the Shiny app `autovi.web` if users encounter installation
 199 problems.

200 **4.4. Usage**

201 **4.4.1. Numerical summary**

202 Three steps are needed to get an automated assessment of a set of residuals and fitted
 203 values:

- 204 1. Load the `autovi` package using the `library()` function.
- 205 2. Create a checker object with a linear regression model.
- 206 3. Call the `check()` method of the checker, which, by default, predicts the VSS for
 207 the true residual plot, 100 null plots, and 100 bootstrapped plots, ~~storing~~. [The](#)
 208 ~~method stores~~ the predictions internally. ~~A concise report of the check results~~
 209 ~~is then printed~~ and prints a concise results report.

210 The code to do this is:

```
library(autovi)
checker <- residual_checker(lm(dist ~ speed, data = cars))
checker$check()
```

211 It produces the following summary:

```
212
213
214
215
216 --<AUTO_VI object>
217 <AUTO_VI object>
218 Status:
219 - Fitted model: lm
220 -- Keras model: UNKNOWN
221 - Keras model: (None, 32, 32, 3) + (None, 5) -> (None, 1)
222 - Output node index: 1
223 - Result:
224 -- Observed visual signal strength: 3.162 (p-value = 0.0396)
225 - Observed visual signal strength: 3.16 (p-value = 0.0396)
226 - Null visual signal strength: [100 draws]
227 - Mean: 1.274
228 - Quantiles:
229
230
231 25%   50%   75%   80%   90%   95%   99%
232 0.8021 1.1109 1.5751 1.6656 1.9199 2.6564 3.3491
233
234 0.802 1.111 1.575 1.666 1.919 2.657 3.348
235
236 - Bootstrapped visual signal strength: [100 draws]
237 - Mean: 2.786 (p-value = 0.05941)
238 - Mean: 2.795 (p-value = 0.05941)
239 - Quantiles:
240
241 25%   50%   75%   80%   90%   95%   99%
242 2.452 2.925 3.173 3.285 3.463 3.505 3.652
```

```

243 2.455 2.941 3.177 3.300 3.474 3.537 3.668
244
245 Likelihood ratio: 0.7275 (boot) / 0.06298 (null) = 11.55
246 - Likelihood ratio: 0.7333 (boot) / 0.06284 (null) = 11.67

```

The summary includes observed VSS of the true residual plot and associated p -value of the automated visual test. The p -value is the proportion of null plots (out of the total 100) that have VSS greater than or equal to that of the true residual plot. The report also provides sample quantiles of VSS for null samples and bootstrapped data plots, providing more information about the sampling variability and a likelihood of model violations. The likelihood is computed from the proportion of values greater than the observed VSS in both the bootstrapped data values and the simulated null values.

255 4.4.2. Visual summary

256 Users can visually inspect the original residual plot alongside a sample null plot using
 257 `plot_pair()` or a lineup of null plot `plot_lineup()`. This visual comparison can
 258 clarify why H_0 is either rejected or not, and help identify potential remedies.

```
checker$plot_pair()
```

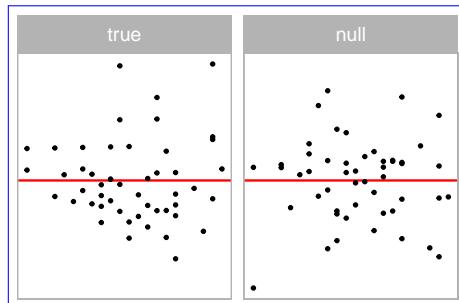


Figure 2. True plot alongside one null plot, for quick comparison.

259 The `plot_pair()` method (Figure 2) displays the true residual plot on the left and a
 260 single null plot on the right. If a full lineup was shown, the true residual plot would
 261 be embedded in a page of null plots. Users should look for any distinct visual patterns
 262 in the true residual plot that are absent in the null plot. Running these functions
 263 multiple times can help any visual suspicions, as each execution generates new random
 264 null plots for comparison.

265 The package offers a straightforward visualization of the assessment result through
 266 the `summary_plot()` function.

```
checker$summary_plot()
```

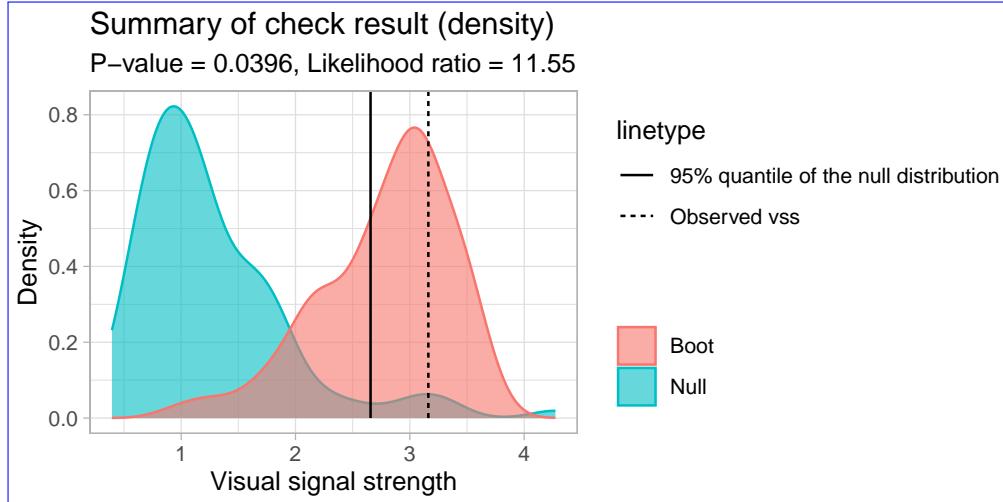


Figure 3. Summary plot comparing the densities of VSS for bootstrapped residual samples (red) relative to VSS for null plots (blue).

267 In the result, shown in Figure 3, the blue area represents the density of VSS for null
 268 residual plots, while the red area shows the density for bootstrapped residual plots.
 269 The dashed line indicates the VSS of the true residual plot, and the solid line marks
 270 the critical value at a 95% significance level. The p -value and the likelihood ratio are
 271 displayed in the subtitle. The likelihood ratio represents the ratio of the likelihood
 272 of observing the VSS of the true residual plot from the bootstrapped distribution
 273 compared to the null distribution.

274 Interpreting the plot involves several key aspects. If the dashed line falls to the right of
 275 the solid line, it suggests rejecting the null hypothesis. The degree of overlap between
 276 the red and blue areas indicates similarity between the true residual plot and null
 277 plots; greater overlap suggests more similarity. Lastly, the portion of the red area to
 278 the right of the solid line represents the percentage of bootstrapped models considered
 279 to have model violations.

280 This visual summary provides an intuitive way to assess the model's fit and potential
 281 violations, allowing users to quickly grasp the results of the automated analysis.

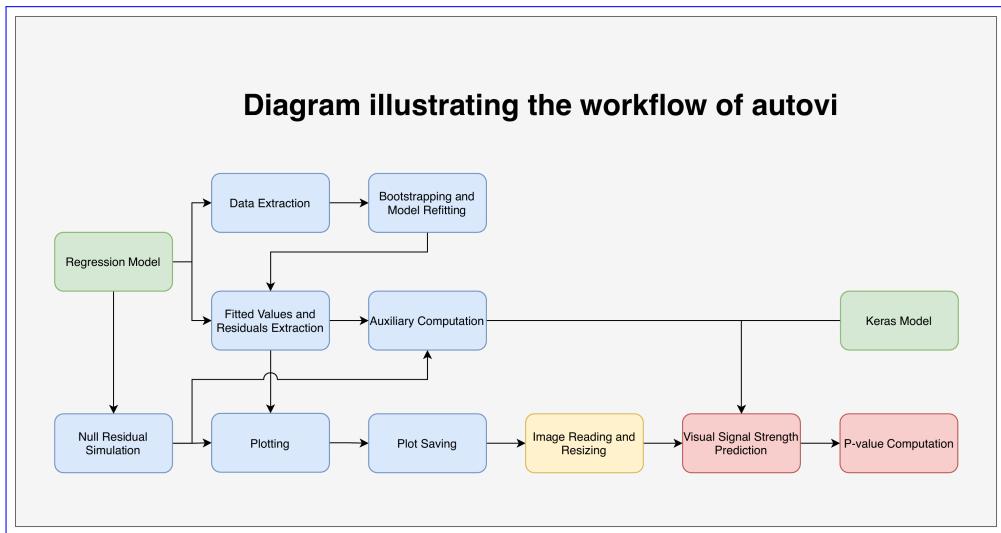
282 **4.5. Modularized infrastructure**

Figure 4. Diagram illustrating the infrastructure of the R package **autovi**. The modules in green are primary inputs provided by users. Modules in blue are overridable methods that can be modified to accommodate users' specific needs. The module in yellow is a pre-defined non-overridable method. The modules in red are primary outputs of the package.

283 The initial motivation for developing **autovi** was to create a convenient interface for
 284 sharing the models described and trained in [Li et al. \(2024b\)](#). However, recognizing
 285 that the classical normal linear regression model represents a restricted class of
 286 models, we sought to avoid limiting the potential for future extensions, whether by
 287 the original developers or other developers. As a result, the package was designed to
 288 function seamlessly with linear regression models with minimal modification and few
 289 required arguments, while also accommodating other classes of models through partial
 290 infrastructure substitution. This modular and customizable design allows **autovi** to
 291 handle a wide range of residual diagnostics tasks.

292 The infrastructure of **autovi** consists of ten core modules: data extraction,
 293 bootstrapping and model refitting, fitted values and residuals extraction, auxiliary
 294 computation, null residual simulation, plotting, plot saving, image reading and resizing,
 295 VSS prediction, and *p*-value computation. Each module is designed with minimal
 296 dependency on the preceding modules, allowing users to customize parts of the
 297 infrastructure without affecting its overall integrity. An overview of this infrastructure
 298 is illustrated in Figure 4.

299 The `modules` package takes regression models and a `Keras` model as primary inputs.
 300 `Modules` for VSS prediction and *p*-value computation are predefined and cannot
 301 be overridden, although users can interact with them directly through function
 302 arguments. Similarly, the image reading and resizing module is fixed but will adapt to
 303 different `Keras` models by checking their input shapesfixed but accessible via function
 304 arguments, using `TensorFlow` for inference and statistical testing. The image loading
 305 module is also fixed, using `PIL` to read and resize images based on the `Keras` model's
 306 `input shape`. The remaining seven modules are designed to be overridable, enabling
 307 users to tailor the infrastructure to their specific needs. overridable, allowing users to
 308 adapt the workflow as needed. The data extraction module extracts a `data.frame`
 309 containing variables used in the regression model. The bootstrapping and refitting
 310 module resamples the data and refits the model. The fitted values and residuals
 311 extraction module returns these values as a `data.frame`. The auxiliary computation
 312 module calculates scagnostics such as monotonicity. The plotting module generates a
 313 `ggplot` in a standard format, and the plot saving module exports it at the same
 314 resolution as the training images. These modules are discussed in detail on the
 315 software's website, described in detail in the package documentation.

316 4.6. Extension to Other Model Classes

317 The `autovi` R package can be extended to accommodate other classes of models
 318 beyond linear regression, such as generalized linear models (`glm`). This is achieved
 319 by substituting the relevant overridable modules, and if needed, supplying a different
 320 `Keras` model.

321 We provide an example of defining a new checker class tailored for Poisson regression
 322 using the `glm` framework:

- 323 1. Define a new class using `new_class()` with `AUTO_VI` as the parent class.
- 324 2. Override the necessary methods using `register_method()`. In this example, we
 325 use Pearson residuals. To simulate null residuals, we assume the fitted model
 326 is correct and the estimated coefficients are accurate. New response values are
 327 generated accordingly, and a new model is fitted to this simulated response. Null
 328 residuals are then extracted from this refitted model.
- 329 3. Create an alias for the `instantiate()` method of the new class.

330 The resulting checker class can be employed analogously to the linear model case
 331 described in Section 4.4. For illustration, we fit a Poisson model in which the quadratic

332 term of the predictor x is intentionally omitted. This misspecification manifests as
 333 a pronounced U-shaped pattern in the lineup display (see Figure 5), which is also
 334 successfully identified by the computer vision model, yielding a p-value substantially
 335 below the conventional threshold of 0.05.

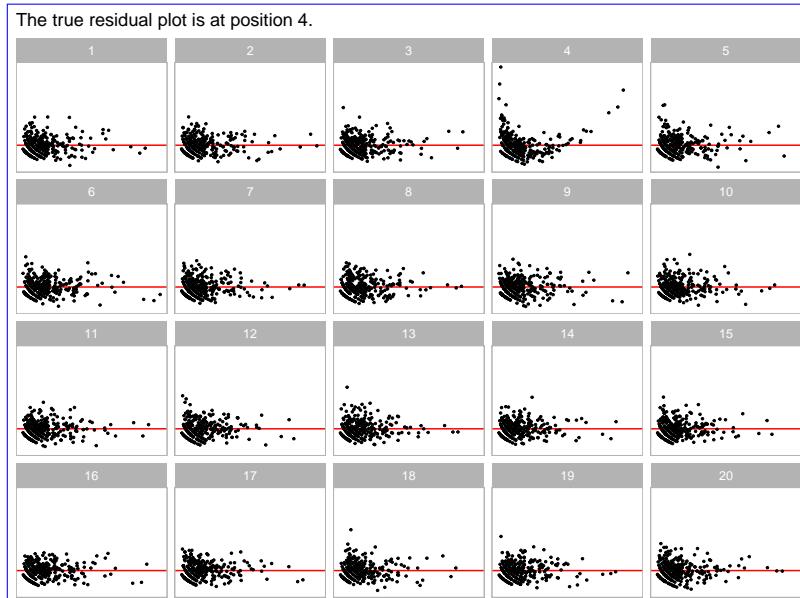


Figure 5. A lineup of residual plots from Poisson generalized linear models, with the true residual plot at position 4, which displays a distinct U-shaped pattern. In contrast, the null plots show characteristics broadly consistent with well-behaved residuals from linear regression models.

```
336 <AUTO_POIS_VI object>
337   Status:
338     - Fitted model: glm, lm
339     - Keras model: (None, 32, 32, 3) + (None, 5) -> (None, 1)
340     - Output node index: 1
341     - Result:
342       - Observed visual signal strength: 4.875 (p-value = 0.009901)
343       - Null visual signal strength: [100 draws]
344         - Mean: 1.331
345         - Quantiles:
```

```

347      25%   50%   75%   80%   90%   95%   99%
348      1.035 1.233 1.488 1.644 1.941 2.276 2.639
349
350      - Bootstrapped visual signal strength: [100 draws]
351      - Mean: 5.51 (p-value = 0.009901)
352      - Quantiles:
353
354      25%   50%   75%   80%   90%   95%   99%
355      5.330 5.505 5.698 5.735 5.830 5.903 6.013
356
357      - Likelihood ratio: 0.05096 (boot) / 0 (null) = Extremely large

```

358 It is important to note, however, that the pre-trained computer vision model included
 359 in `autovi`, such as `vss phn 32` (see `list_keras_models()` for the full list of
 360 available models), was developed specifically for diagnostics of linear regression. Its
 361 applicability to other model classes relies on the assumption that the null residual
 362 plots exhibit characteristics broadly consistent with those of well-behaved linear
 363 regression residuals, that is, residuals should be approximately randomly scattered
 364 around zero, display roughly constant variance across the range of fitted values, and
 365 exhibit no discernible structure or curvature. If these conditions are not met, or if
 366 model violations do not give rise to visually detectable patterns, the validity of the
 367 automated diagnostics may be compromised. In such cases, users are encouraged to
 368 train and apply their own `Keras` models. Detailed guidance on model training and
 369 discussion on extending the methodology to other model classes can be found in
 370 Li et al. (2024b).

371 5. Web interface: `autovi.web`

372 The `autovi.web` shiny application extends the functionality of `autovi` by offering a
 373 user-friendly web interface for automated residual plot assessment. This eliminates the
 374 common challenges associated with software installation, so users can avoid managing
 375 Python environments or handling version requirements for R libraries. The platform
 376 is cross-platform and accessible on various devices and operating systems, making it
 377 suitable even for users without R programming experience. Additionally, updates are
 378 managed centrally, ensuring that users always have access to the latest features. This
 379 section discusses the implementation based on `autovi.web` version 0.1.0.

380 5.1. Implementation

381 The interface `autovi.web` is built using the `shiny` (Chang et al. 2022) and
382 `shinydashboard` (Chang & Borges Ribeiro 2021) R packages. Hosted on the
383 `shinyapps.io` domain, the application is accessible through any modern web browser.
384 The R packages `htmltools` (Cheng et al. 2024) and `shinycssloaders` (Sali & Attali
385 2020) are used to render markdown documentation in shiny application, and for loading
386 animations for shiny widgets, respectively.

387 Determining the best way to implement the backend was difficult. In our initial
388 planning for `autovi.web`, we considered implementing the entire web application using
389 the `webr` framework (Moon 2020), which would have allowed the entire application to
390 run directly in the user's browser. However, ~~this approach was not feasible at the time
391 of writing this paper. The reason is that one of the R packages `autovi` depends on the
392 R package `webr` does not support packages which use compiled fortran code, which is
393 required by `splancs` (Rowlingson & Diggle 2023), which uses compiled Fortran code.
394 A (Rowlingson & Diggle 2023), a dependency of `autovi`. In the future, it is possible
395 that a working Emscripten (Zakai 2011) version of this package, ~~which would be
396 required for may allow full~~ `webr`, ~~was not available~~ support.~~

397 We also explored the possibility of implementing the web interface using frameworks
398 built on other languages, such as Python. However, server hosting domains that
399 natively support Python servers typically do not have the latest version of R installed.
400 Additionally, calling R from Python is typically done using the `rpy2` Python library
401 (Gautier 2024), but this approach can be awkward when dealing with language syntax
402 related to non-standard evaluation. Another option we considered was renting a server
403 where we could have full control, such as those provided by cloud platforms like
404 Google Cloud Platform (GCP) or Amazon Web Services (AWS). However, ~~correctly
405 setting up the server and ensuring a secure deployment requires significant deploying
406 and maintaining the server securely requires some~~ expertise. Ultimately, the most
407 practical solution was to use the `shiny` and `shinydashboard` frameworks, which are
408 well-established in the R community and offer a solid foundation for web application
409 development.

410 The server-side configuration of `autovi.web` is carefully designed to support its
411 functionality. Most required Python libraries, including `pillow` and `numpy`, are pre-
412 installed on the server. These libraries are integrated into the Shiny application using
413 the `reticulate` package, which provides an interface between R and Python.

414 Due to ~~the resource allocation policy of shinyapps.io's resource policy, inactive servers~~
415 ~~enter sleep mode, the server enters a sleep mode during periods of inactivity, resulting~~
416 ~~in the clearing of clearing the local Python virtual environment. Consequently, when~~
417 ~~the application "wakes up" environment. When reactivated for a new user session,~~
418 ~~these libraries need to session, libraries must~~ be reinstalled. While this ensures a clean
419 environment for each session, it may lead to slightly longer loading times for the first
420 user after a period of inactivity.

421 In contrast to `autovi`, `autovi.web` ~~does not use the native Python version of~~
422 ~~TensorFlow. Instead, it~~ leverages `TensorFlow.js`, a JavaScript library that allows
423 the execution of machine learning models directly in the browser. This choice enables
424 native browser execution, enhancing compatibility across different user environments,
425 and shifts the computational load from the server to the client-side. `TensorFlow.js`
426 also offers better scalability and performance, especially when dealing with resource-
427 intensive computer vision models on the web.

428 While `autovi` requires downloading the pre-trained computer vision models from
429 GitHub, these models in “.keras” file format are incompatible with `TensorFlow.js`.
430 Therefore, we extract and store the model weights in JSON files and include
431 them as extra resources in the Shiny application. When the application initializes,
432 `TensorFlow.js` rebuilds the computer vision model using these pre-stored weights.

433 To allow communication between `TensorFlow.js` and other components of the Shiny
434 application, the `shinyjs` R package (Attali 2021) is used. This package allows calling
435 custom JavaScript code within the Shiny framework. The specialized JavaScript
436 code for initializing `TensorFlow.js` and calling `TensorFlow.js` for VSS prediction is
437 deployed alongside the Shiny application as additional resources.

438 5.2. Usage

439 The workflow of `autovi.web` is designed to be straightforward, with numbered
440 steps displayed in each panel. There are two example datasets provided by the
441 web application. The single residual plot example uses the `dino` dataset from the
442 R package `datasauRus` (Davies, Locke & D'Agostino McGowan 2022). The lineup
443 example uses residuals from a simulated regression model that has a non-linearity
444 issue. We walk through the lineup example to further demonstrate the workflow of
445 the web application.

446 **5.2.1. Reading data and setting parameters**

447 The user can select to upload data as either a single set of residuals and fitted values
 448 in a two (or more) column CSV file or a pre-computed lineup of residuals and null
 449 datasets in a three (or more) column CSV file (i.e. multiple sets of residuals and fitted
 450 values with a column indicating the set label). Here we illustrate use with lineup
 451 example data sets (Figure 6). To use the lineup example data, click the “Use Lineup
 452 Example” button. The data status will then update to show the number of rows and
 453 columns in the dataset, and the CSV type will automatically be selected to the correct
 454 option. Since the example dataset follows the variable naming conventions assumed
 455 by the web application, the columns for fitted values, residuals, and labels of residual
 456 plots are automatically mapped such that the column named as `.fitted` is mapped
 457 to fitted values, `.resid` is mapped to residuals and if applicable, `.sample` to labels of
 458 the residual set (middle image). If the user is working with a custom dataset, these
 459 options must be set accordingly. Whenever a data containing a lineup, the user must
 460 manually select the label for the true residual plot, otherwise the web application does
 461 not provide all the results. The last step is to click the play button (right image) to
 462 start the assessment.

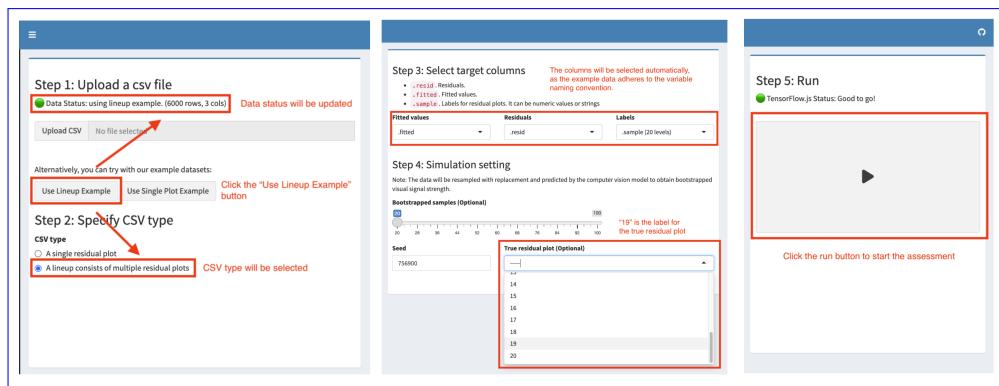


Figure 6. To begin the workflow for `autoovi` using the lineup example dataset, the user clicks the “Use Lineup Example” button (left) to load the example dataset, during which the data status and CSV type will be automatically updated. The user must manually select the label for the true residual plot (middle) to compute further results. The user initiates the assessment of the lineup example data by clicking the run button (right).

463 **5.2.2. Results provided**

464 Results are provided in multiple panels. The first row of the table Figure 7 is the most
 465 crucial to check, as it provides the VSS and the rank of the true residual plot among
 466 the other plots. The summary text beneath the table provides the *p*-value, which can

467 be used for quick decision-making. The lineup is for manual inspection, and the user
 468 should see if the true residual plot is visually distinguishable from the other plots, to
 469 confirm if the model violation is serious.

470 The density plot in Figure 8 offers a more robust result, allowing the user to compare
 471 the distribution of bootstrapped VSS with the distribution of null VSS. Finally,
 472 the grayscale attention map (right image) can be used to check if the target visual
 473 features, like the non-linearity present in the lineup example, are captured by the
 474 computer vision model, ensuring the quality of the assessment. The attention map is
475 the gradient of the model output with respect to the grayscale image input, indicating
476 the sensitivity of the output to each pixel.

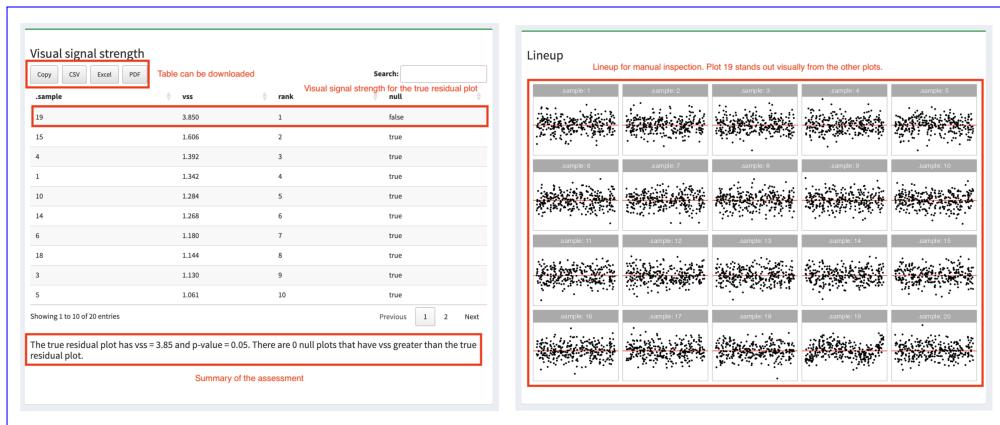


Figure 7. Results for the lineup. The VSS of the true residual plot is displayed in the first row of the table of VSS values for all the null plots (left image), with a summary text beneath the table providing the *p*-value to aid in decision-making. A lineup of residual plots allows for manual inspection (right image).

477

6. Conclusions

478 This paper presents new regression diagnostics software, the R package **autovi** and
 479 its accompanying web interface, **autovi.web**. It addresses a critical gap in the current
 480 landscape of statistical software. While regression tools are widely available, effective
 481 and efficient diagnostic methods have lagged behind, particularly in the field of residual
 482 plot interpretation.

483 The **autovi** R package, introduced in this paper, automates the assessment of residual
 484 plots by incorporating a computer vision model, eliminating the need for reducing
485 reliance on time-consuming and potentially inconsistent human interpretation. This

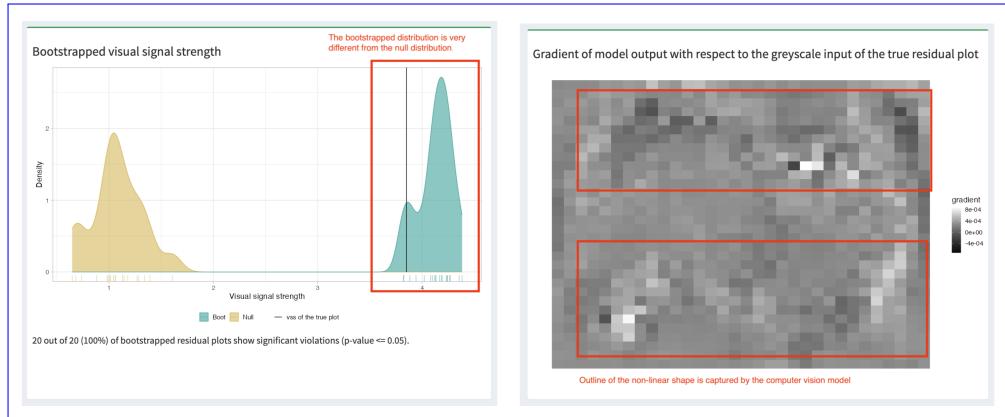


Figure 8. Summaries assessing the strength of the pattern and which elements of the plot contribute. The density plot helps verify if the bootstrapped distribution differs from the null distribution (left image). The attention map (right image) offers insights into whether the computer vision model has captured the intended visual features of the true residual plot.

486 automation improves the efficiency of the diagnostic process and promotes consistency
 487 in model evaluation across different users and studies.

488 The development of the accompanying Shiny app, **autovi.web**, expands access to these
 489 advanced diagnostic tools, by providing a user-friendly interface. It makes automated
 490 residual plot assessment accessible to a broader audience, including those who may not
 491 have extensive programming experience. This web-based solution effectively addresses
 492 the potential barriers to adoption, such as complex dependencies and installation
 493 requirements, that are often associated with advanced statistical software.

494 The combination of **autovi** and **autovi.web** offers a comprehensive solution to the
 495 challenges of residual plot interpretation in regression analysis. These tools have the
 496 potential to significantly improve the quality and consistency of model diagnostics
 497 across various fields, from academic research to industry applications. By automating
 498 a critical aspect of model evaluation, they allow researchers and analysts to focus more
 499 on interpreting results and refining models, rather than grappling with the intricacies
 500 of plot assessment.

501 The framework established by **autovi** and **autovi.web** opens up exciting possibilities
 502 for further research and development. Future work could explore the extension of these
 503 automated assessment techniques to other types of diagnostic plots and statistical
 504 models, potentially revolutionizing how we approach statistical inference using visual
 505 displays more broadly.

506

7. Resources and supplementary material

507 The current version of `autovi` can be installed from CRAN, and source
 508 code for both packages are available at github.com/TengMCing/autovi and
 509 github.com/TengMCing/autovi_web respectively. The web interface is available from
 510 autoviweb.netlify.app.

511 This paper is reproducibly written using Quarto ([Allaire et al. 2024](#)) powered by
 512 Pandoc ([MacFarlane, Krewinkel & Rosenthal 2024](#)) and pdfTeX. The full source code
 513 to reproduce this paper is available at github.com/TengMCing/autovi_paper.

514 These R packages were used for the work: `tidyverse` ([Wickham et al. 2019](#)), `lmtest`
 515 ([Zeileis & Hothorn 2002](#)), `kableExtra` ([Zhu 2021](#)), `patchwork` ([Pedersen 2022](#)),
 516 `rcartocolor` ([Nowosad 2018](#)), `glue` ([Hester & Bryan 2022](#)), `here` ([Müller 2020](#)),
 517 `magick` ([Ooms 2023](#)), `yardstick` ([Kuhn, Vaughan & Hvitfeldt 2024](#)) and `reticulate`
 518 ([Ushey, Allaire & Tang 2024](#)).

519

References

- 520 ABADI, M., AGARWAL, A., BARHAM, P., BREVDO, E., CHEN, Z., CITRO, C., CORRADO, G.S.,
 521 DAVIS, A., DEAN, J., DEVIN, M. et al. (2016). Tensorflow: Large-scale machine learning on
 522 heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* .
- 523 ALLAIRE, J., TEAGUE, C., SCHEIDECKER, C., XIE, Y. & DERVIEUX, C. (2024). Quarto. doi:
 524 10.5281/zenodo.5960048. URL <https://github.com/quarto-dev/quarto-cli>.
- 525 ATTALI, D. (2021). *shinyjs: Easily Improve the User Experience of Your Shiny Apps in Seconds*.
 526 URL <https://CRAN.R-project.org/package=shinyjs>. R package version 2.1.0.
- 527 BALAMUTA, J.J. (2024). *surreal: Create Datasets with Hidden Images in Residual Plots*. URL
 528 <https://CRAN.R-project.org/package=surreal>. R package version 0.0.1.
- 529 BREUSCH, T.S. & PAGAN, A.R. (1979). A simple test for heteroscedasticity and random coefficient
 530 variation. *Econometrica: Journal of the Econometric Society* , 1287–1294.
- 531 BUJA, A., COOK, D., HOFMANN, H., LAWRENCE, M., LEE, E.K., SWAYNE, D.F. & WICKHAM, H.
 532 (2009). Statistical inference for exploratory data analysis and model diagnostics. *Philosophical
 533 Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* **367**,
 534 4361–4383.
- 535 CHANG, W. & BORGES RIBEIRO, B. (2021). *shinydashboard: Create Dashboards with 'Shiny'*.
 536 URL <https://CRAN.R-project.org/package=shinydashboard>. R package version 0.7.2.
- 537 CHANG, W., CHENG, J., ALLAIRE, J., SIEVERT, C., SCHLOERKE, B., XIE, Y., ALLEN, J.,
 538 MCPHERSON, J., DIPERT, A. & BORGES, B. (2022). *shiny: Web Application Framework for
 539 R*. URL <https://CRAN.R-project.org/package=shiny>. R package version 1.7.3.
- 540 CHENG, J., SIEVERT, C., SCHLOERKE, B., CHANG, W., XIE, Y. & ALLEN, J. (2024). *htmltools:
 541 Tools for HTML*. URL <https://CRAN.R-project.org/package=htmltools>. R package version
 542 0.5.8.
- 543 CLARK, A. et al. (2015). Pillow (pil fork) documentation. *readthedocs* .
- 544 COOK, R.D. & WEISBERG, S. (1982). *Residuals and influence in regression*. New York: Chapman
 545 and Hall.

- 546 DAVIES, R., LOCKE, S. & D'AGOSTINO McGOWAN, L. (2022). *datasauRus: Datasets from the*
 547 *Datasaurus Dozen*. URL <https://CRAN.R-project.org/package=datasauRus>. R package
 548 version 0.1.6.
- 549 GAUTIER, L. (2024). *Python interface to the R language (embedded R)*. URL <https://pypi.org/project/rpy2/>. Version 3.5.16.
- 551 GOODE, K. & REY, K. (2019). *ggResidpanel: Panels and Interactive Versions of Diagnostic Plots*
 552 *using 'ggplot2'*. URL <https://CRAN.R-project.org/package=ggResidpanel>. R package
 553 version 0.3.0.
- 554 HARTIG, F. (2022). *DHARMA: Residual Diagnostics for Hierarchical (Multi-Level / Mixed)*
 555 *Regression Models*. URL <https://CRAN.R-project.org/package=DHARMA>. R package
 556 version 0.4.6.
- 557 HEBBALI, A. (2024). *olsrr: Tools for Building OLS Regression Models*. URL <https://CRAN.R-project.org/package=olsrr>. R package version 0.6.0.
- 559 HESTER, J. & BRYAN, J. (2022). *glue: Interpreted String Literals*. URL <https://CRAN.R-project.org/package=glue>. R package version 1.6.2.
- 561 JOHNSON, P.E. (2022). *rockchalk: Regression Estimation and Presentation*. URL <https://CRAN.R-project.org/package=rockchalk>. R package version 1.8.157.
- 563 KUHN, M., VAUGHAN, D. & HVITFELDT, E. (2024). *yardstick: Tidy Characterizations of Model*
 564 *Performance*. URL <https://CRAN.R-project.org/package=yardstick>. R package version
 565 1.3.1.
- 566 LI, W. (2024). *bandicoot: Light-weight python-like object-oriented system*. URL <https://CRAN.R-project.org/package=bandicoot>.
- 568 LI, W., COOK, D., TANAKA, E. & VANDERPLAS, S. (2024a). A plot is worth a thousand tests:
 569 Assessing residual diagnostics with the lineup protocol. *Journal of Computational and*
 570 *Graphical Statistics* **33**, 1497–1511. doi:10.1080/10618600.2024.2344612.
- 571 LI, W., COOK, D., TANAKA, E., VANDERPLAS, S. & ACKERMANN, K. (2024b). Automated
 572 assessment of residual plots with computer vision models. *arXiv preprint arXiv:2411.01001*.
- 573 LONG, J.A. (2022). *jtools: Analysis and Presentation of Social Scientific Data*. URL <https://cran.r-project.org/package=jtools>. R package version 2.2.0.
- 575 LOY, A. & HOFMANN, H. (2014). *Hlmdiag: A suite of diagnostics for hierarchical linear models in*
 576 *r*. *Journal of Statistical Software* **56**, 1–28.
- 577 MACFARLANE, J., KREWINKEL, A. & ROSENTHAL, J. (2024). Pandoc. URL <https://github.com/jgm/pandoc>.
- 579 MASON, H., LEE, S., LAA, U. & COOK, D. (2022). *cassowaryr: Compute Scagnostics on Pairs of*
 580 *Numeric Variables in a Data Set*. URL <https://CRAN.R-project.org/package=cassowary>. R
 581 package version 2.0.0.
- 582 MOON, K.W. (2020). *webr: Data and Functions for Web-Based Analysis*. URL <https://CRAN.R-project.org/package=webr>. R package version 0.1.5.
- 584 MÜLLER, K. (2020). *here: A simpler way to find your files*. URL <https://CRAN.R-project.org/package=here>. R package version 1.0.1.
- 586 NOWOSAD, J. (2018). 'CARTOCOLORs' palettes. URL <https://nowosad.github.io/rkartocolor>. R
 587 package version 1.0.
- 588 OOMS, J. (2023). *magick: Advanced Graphics and Image-Processing in R*. URL <https://CRAN.R-project.org/package=magick>. R package version 2.7.4.
- 590 PEDERSEN, T.L. (2022). *patchwork: The composer of plots*. URL <https://CRAN.R-project.org/package=patchwork>. R package version 1.1.2.
- 592 R CORE TEAM (2022). *R: A Language and Environment for Statistical Computing*. R Foundation
 593 for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.

- 594 RAMSEY, J.B. (1969). Tests for specification errors in classical linear least-squares regression
 595 analysis. *Journal of the Royal Statistical Society: Series B (Methodological)* **31**, 350–371.
- 596 REINHART, A. (2024). *regressinator: Simulate and Diagnose (Generalized) Linear Models*. URL
 597 <https://CRAN.R-project.org/package=regressinator>. R package version 0.2.0.
- 598 ROWLINGSON, B. & DIGGLE, P. (2023). *splancs: Spatial and Space-Time Point Pattern Analysis*.
 599 URL <https://CRAN.R-project.org/package=splancs>. R package version 2.01-44.
- 600 SALI, A. & ATTALI, D. (2020). *shinyCSSloaders: Add Loading Animations to a 'shiny' Output
 601 While It's Recalculating*. URL <https://CRAN.R-project.org/package=shinyCSSloaders>. R
 602 package version 1.0.0.
- 603 SHAPIRO, S.S. & WILK, M.B. (1965). An analysis of variance test for normality (complete samples).
 604 *Biometrika* **52**, 591–611.
- 605 USHEY, K., ALLAIRE, J. & TANG, Y. (2024). *reticulate: Interface to 'Python'*. URL <https://CRAN.R-project.org/package=reticulate>. R package version 1.35.0.
- 606 WARTON, D.I. (2023). Global simulation envelopes for diagnostic plots in regression models. *The
 607 American Statistician* **77**, 425–431.
- 608 WICKHAM, H. (2016). *ggplot2: Elegant graphics for data analysis*. Springer-Verlag New York.
 609 URL <https://ggplot2.tidyverse.org>.
- 610 WICKHAM, H., AVERICK, M., BRYAN, J., CHANG, W., McGOWAN, L.D., FRANÇOIS, R.,
 611 GROLEMUND, G., HAYES, A., HENRY, L., HESTER, J., KUHN, M., PEDERSEN, T.L., MILLER,
 612 E., BACHE, S.M., MÜLLER, K., OOMS, J., ROBINSON, D., SEIDEL, D.P., SPINU, V.,
 613 TAKAHASHI, K., VAUGHAN, D., WILKE, C., WOO, K. & YUTANI, H. (2019). Welcome to
 614 the tidyverse. *Journal of Open Source Software* **4**, 1686. doi:10.21105/joss.01686.
- 615 WICKHAM, H., CHOWDHURY, N.R., COOK, D. & HOFMANN, H. (2020). *nullabor: Tools for
 616 Graphical Inference*. URL <https://CRAN.R-project.org/package=nullabor>. R package
 617 version 0.3.9.
- 618 ZAKAI, A. (2011). Emscripten: an llvm-to-javascript compiler. In *Proceedings of the ACM
 619 international conference companion on Object oriented programming systems languages and
 620 applications companion*. pp. 301–312.
- 621 ZEILEIS, A. & HOTHORN, T. (2002). Diagnostic checking in regression relationships. *R News* **2**,
 622 7–10.
- 623 ZHU, H. (2021). *kableExtra: Construct complex table with kable and pipe syntax*. URL
 624 <https://CRAN.R-project.org/package=kableExtra>. R package version 1.3.4.