SoftwareX 23 (2023) 101456

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

Copatrec: A correlation pattern recognizer Python package for nonlinear relations

Siamak Khatami *, Christopher Frantz

Department of Computer Science, Norwegian University of Science and Technology, Norway

ARTICLE INFO

Article history:
Received 23 December 2022
Received in revised form 23 May 2023
Accepted 23 June 2023

Keywords:
Complexity
Nonlinear regression
Correlation pattern recognition
Python

ABSTRACT

Nonlinear regression plays a significant role as a method of analysis in different fields like statistics and simulations. While many researches focus on improving technical issues in the processes of nonlinear fitting, the absence of preprocessing, model selection, and validation is causing technical problems in various fields, i.e., statistical analysis and simulation methods. This work introduces a Python package named **Copatrec**, which facilitates preprocessing of data (i.e., data cleaning, standardization, and outlier detection), nonlinear model selection (from a group of complex behaviors' mathematical expressions), and validation (by returning a result object that contains validation metrics and visualization functions), as observed in diverse scientific fields, including human-related sciences (e.g., social, political, economics, psychology, and health), and engineering fields like computer science (e.g., computer vision), electrical engineering, as well as fundamental science like physics and chemistry, where a precise characterization of the nonlinear relationship is central to the analytical outcome.

Code metadata

Current code version	0.0.7
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-22-00443
Easy setup	pip install copatrec
Legal Code License	CC BY-NC-SA 4.0
Code versioning system	Git
Software language	Python (v ⩾ 3.6)
Compilation requirements, operating environments & dependencies	Python: **Pandas**, **Numpy**, **SciPy**, **Scikit-learn**, **Matplotlib**
developer documentation/manual	https://github.com/copatrec/copatrec
Support email for questions	khatami.sia@gmail.com
Official website and web application	copatrec.org

1. Introduction: Motivation and significance

1.1. Motivation

Regression plays a significant role in science and correlation analysis, both in technical disciplines, such as engineering and finance, as well as in the social sciences, such as political science and economics, amongst others. Regression can be used alone or in combination with any other scientific methods like simulation and artificial intelligence, potentially serving different purposes, such as the generalization of equation patterns from collected data for the purpose of engineering or technology development, the parameterization of complex models with input data, as well as to facilitate statistical analysis more broadly. The two broad categories of regressions are *linear* (i.e., $Y = \beta_0 + \beta_1 X_1^2$) and *nonlinear* (i.e., $Y = \beta_0 + (\beta_1 X_1)^2$; parameter β is affected by another function). While both aim to understand the relationship between an independent and a group of dependent variables, linear regression receives primary attention from the scientific community due to difficulties in nonlinear regression, specifically (1) preprocessing data [1] and selection of a proper mathematical model [2], (2) technical issues in fitting data [3],

* Corresponding author.
E-mail addresses: siamak.khatami@ntnu.no (Siamak Khatami), christopher.frantz@ntnu.no (Christopher Frantz).

https://doi.org/10.1016/j.softx.2023.101456

and (3) evaluation and visualization [3,4]. However, to foster a grounded understanding of natural and social phenomena and their underlying complexity, accessible mechanisms for the analysis of nonlinear dynamics promise to leverage novel insights that traditional linear approaches do not offer. However, from the three major challenges mentioned above, specifically, the technical challenges, have benefited from rapid developments in the past decade, especially with the development of *Machine Learning* (ML) techniques [5]. Moreover, in the light of the increasing availability of diverse datasets, the need for a more accurate representation of underlying data increases, e.g., to support predictive analyses in approaches such as Discrete Event Simulation [6], System Dynamics [7], and Agent-based Modeling [8], puts the spotlight on nonlinear regressions more than before.

This work responds to that demand. However, the main aim of this work is not to directly contribute to methods for solving nonlinear regression, given that there is a plethora of available packages in different programming languages that operate with nonlinear regressions [for Python, these specifically include **SciPy** [9] and **Scikit-learn** [10]]. Instead, this article introduces a novel package **Copatrec** (*correlation pattern recognizer*), which deals with challenges 1) preprocessing data, 2) Model Selection and 3) Evaluation and Visualization highlighted in the earlier section, and furthermore provides a conceptual overview and operational guidance. To date, most existing packages for the analysis of nonlinear correlations operate supervised in that they require preprocessing of data before any regression analysis and, more centrally, the prior selection of a nonlinear form as the parameters for their operation. Furthermore, **Copatrec** offers a set of embedded and callable functions related to the regression, i.e., data preprocessing like standardization, null value handling, and outlier recognition (Section 2.2) and panel regression analysis (Section 2.3) which will be discussed later in the outliers' section. Data preprocessing before model selection is crucial because the resulting dataset can show and imply different behavior. Following the data preprocessing, model selection is handled automatically based on the available complex models in the dataset. In addition to the regression estimations and statistical analysis offered, **Copatrec** offers a set of validation and visualization tools (in the form of result-object) that facilitate immediate inspection of determining equations and foster a better understanding of complex events, as well as making the output accessible to a broader scientific audience (including ones with limited data-processing skills), making it a basis for the communication and iterative exploration of findings based on the observed output. In addition, both to support the communication and exploration while establishing transparency of the execution process, **Copatrec** features transparent inspection of its operations, both in textual and visual form. These aspects combined are intended to navigate the trade-off of analytical accessibility as well as establish a basis for methodologically rigorous evaluation. All mentioned features will be discussed later in their relevant sections.

1.2. Significance

Both regressions (linear and nonlinear) can be studied using a single independent variable or multiple variables. The multivariable analysis generally implies the presence of a complex system, but single variables can likewise characterize aspects of complex systems. The simplest example of single variable complex equations is a differential equation that includes the rate of change of a variable [7]. Sterman [7] has illustrated many single variable complex systems in which *delay* can also make a complex behavior and output. This delay in simulation systems can be interpreted as differential equations in the mathematical models because of both discussions' callbacks to the *rate of change of a variable*. In this light, pair mode analysis could be an essential step of complex behavior analysis, which is the main focus of **Copatrec** in the current version. In addition, a set of mathematical expressions (Table 2) are selected as the initial set by studying the most observed complex behaviors, which are illustrated in Fig. 1. Reviewing the referenced packages with respect to their ability to accommodate parameter estimation (**Numpy** [11], **SciPy** [9], **Scikit-learn** [10], **Statsmodel** [12], and **LMFIT** [13]), **SciPy** and **LMFIT** provide functions to fit input data to a particular mathematical form of regression that is then provided as output. While other packages (i.e., [10–12]) support prediction, one challenge is that these packages predominantly rely on black-box models, in which the model itself cannot be meaningfully inspected and/or interpreted. Among others, **LMFIT** offers a set of predefined mathematical models that provides researchers the ability to trace its operation, hence making **LMFIT** methodologically valuable as far as sense-making and reproducibility are concerned. Seeking to integrate selected features from existing packages, **Copatrec** has, for instance, adopted features such as **Numpy** for vector-based calculations, **SciPy** for fitting progress, **Scikit-learn** and **Statsmodel** for specific aspects of the statistical analysis, and **Matplotlib** [14] for visualization.

2. Software description

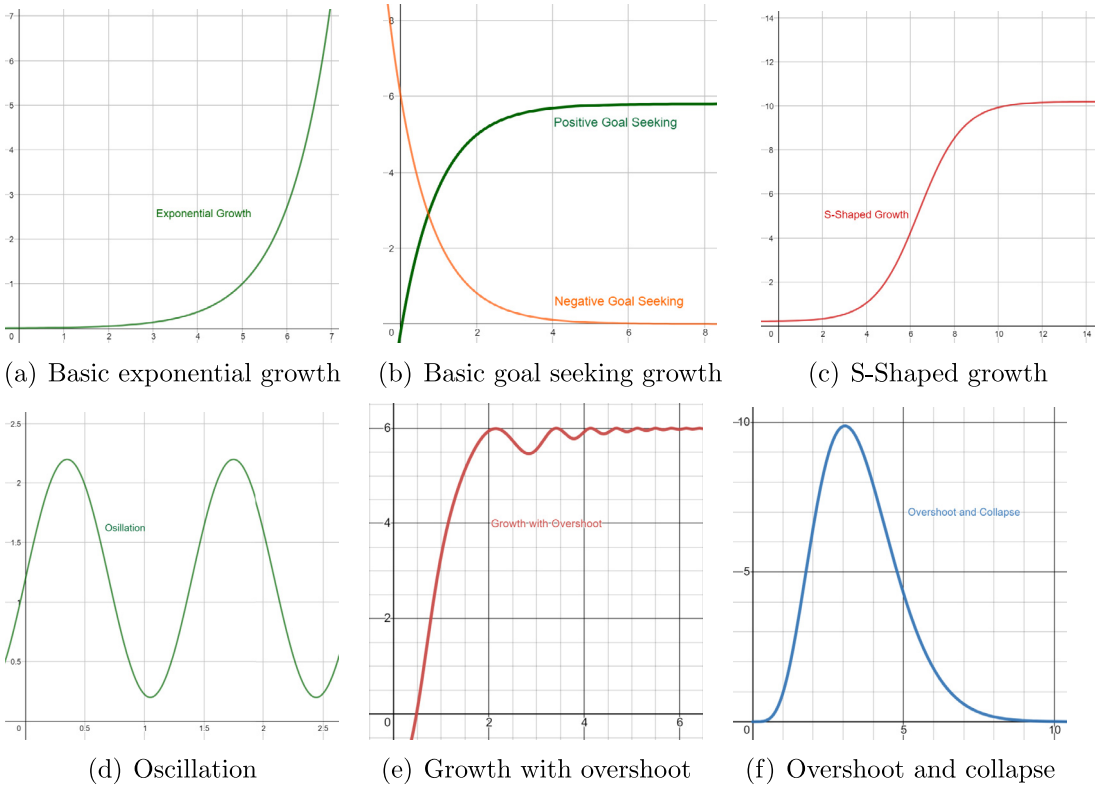
Copatrec extends nonlinear regression analysis direction by drawing on a wide and generalizable set of mathematical expressions (Fig. 1 and Table 2) that have been identified in the literature, theoretically or mathematically [7,8,15,16]. While general in kind, it remains important to state that all mentioned tools cannot replace the understanding of the underlying phenomenon, which extends to result in interpretation, thus making an explicit call for rigor and transparency. This naturally poses a challenge for any approach that estimates parameters for subsequent analytical applications, an aspect that **Copatrec** aims to address. In addition to the mathematical models, preprocessing the data is a crucial step before analysis. This involves a range of preprocessing with different functionalities (i.e., standardization, re-scaling data, e.g., to extrapolate differences more explicitly. [see M.panel_outliers in Listing 2].), however, studying outliers can be a crucial step especially if the data is panel data and each category can have different behavior. The diversity of behaviors within different categories can make exceptional groups in which studying them with others can make analysis blurry, challenging the achievement of a general pattern for a complex system. Following that, **Copatrec** has a set of outlier functions that helps to recognize potential outliers and study them separately.

Following this, the overall features of **Copatrec**, which will be discussed in the next section, can be summarized as follows:

1. Pre-processing functions (standardization and outlier recognition) [Listing 2].
2. Automated nonlinear model selection functions (to detect optimum nonlinear behavior) [Listing 3].
3. Validation and visualization of the produced results (the Result-object) [Table 1].

2.1. Software architecture and functionalities

Following the listed main features of **Copatrec** above, this section will discuss them briefly. **Copatrec** has 6 analytical functions which have additional functions built-in to plot and observe results. The overall architecture of **Copatrec** can be seen in Fig. 2. To use **Copatrec**, first, an object of **Copatrec** class should be generated (Listing 1). To generate **Copatrec** object, the input data (pandas data frame format), the column name for the dependent



(a) Basic exponential growth (b) Basic goal seeking growth (c) S-Shaped growth

(d) Oscillation (e) Growth with overshoot (f) Overshoot and collapse

Fig. 1. Basic dynamic behaviors [based on [7]].

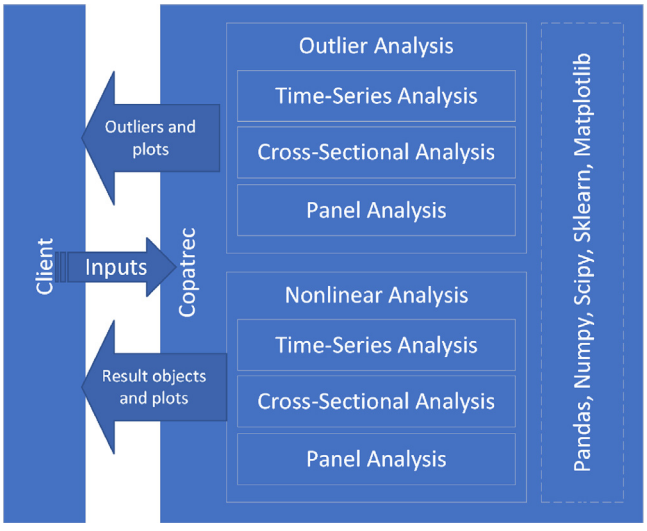


Fig. 2. Software Architecture of **Copatrec**.

variable, time, and categories should be passed. In addition, there are two parameters to control the reporting progress. Besides, if the *independent_vars* parameter (which is the list of possible independent variables in the data) is not specified, the package will consider all other columns as independent variables to be analyzed. After generating the object, both outlier and nonlinear regression functions can be called, which will be discussed further.

2.2. Preprocessing

Preprocessing is one of the essential steps of the analysis process. Preprocessing can be divided into six categories: data cleaning, data standardization, data transformation, missing value imputation, data integration, and noise identification [1]. The activities related to preprocessing are challenging to generalize ex-ante and require consideration of the analytical context and hence rely on the analyst or expert in the field of study due to the inherent requirement to consider domain expertise and analytical objectives. However, while those techniques can be very critical in other fields like machine learning, some can cause problems in regression analysis. For example, missing value imputation and data integration techniques are designed to satisfy the machine learning method's lack of data and input shape problems. Since machine learning uses data to train the model and reduce the error term, these techniques can be hired under specific circumstances. But in nonlinear regressions, adding new data rows or duplicating any can make unreal scenarios or put more weight on the duplicated data raw. For example, adding new unreal data can imply impossible events if we miss a country's values. Following that, **Copatrec** covers some of the mentioned steps in its preprocessing section, including missed data cleaning (only null values because adding new data can add up extra weights,

```python
#Importing Packages
import pandas as pd
from copatrec import Copatrec

#Reading Data
data = pd.read_pickle("../data/GRIN.pkl")
# Initiating copatrec object
Time_Col = 'year' # Which column represents data for the time
Category_Column = 'countryname' # Which column represents data for categories
Dep_var = 'gdppc'  # The dependent variable (Y)

# The dataset has another column called "Government Integrity" which is used as the independent variable (X). All columns except time_col, category_Column, and Dep_var will be dealt with as independent variables.
SM = Copatrec(data=data, # the input data
              dependent_var=Dep_Var,
              independent_vars=["Government Integrity"],
              # If omitted, it will loop over all remaining columns as independent variables.
              category_col=Category_Column,
              time_col=time_Col,
              report=True, # Progress report
              report_to_file=False) # Save the report in a file?
```

Listing 1: Initializing Copatrec object

which affects the final parameter output), data standardization and normalization, data type adjustments (floating numbers), and noise identification (outliers). In this context, outliers are broadly understood as noise identification. Noise points, invalid data registration, shock points, or significantly different points can be categorized as outliers, and including or excluding them from the study relies on expert assessment. Outliers can be discarded, replaced, or, where relevant, studied in separate cases depending on the outlier type. To cover outliers, three different kinds of outlier functions (*panel*[1], *time*-series[2], and *cross-sectional*[3]) have been provided based on the possible analysis types that are listed in Listing 2. In addition, to retain access to add input data as part of the regression analysis, outliers' functions have been implemented as built-in functionality for regression analysis (Section 2.3), which can be activated by corresponding arguments described in the following sub-section.

The outlier function is set to assume a beta distribution by default. The most-used method in studying outliers by the research community is normal distribution [17]; however, this method can be used when the normal distribution of data is guaranteed, and it has a symmetrical bell shape, which in most cases is not like that. But the capability of Beta distribution (also known as the distribution of distributions) is dynamic to capture any skewness. Fig. 3 illustrates the difference between beta and normal distributions for further elaboration. As shown, intervals in the normal distribution have a symmetric distance from the mean point following the symmetry assumption of distributions in the population. However, as the data is evidently skewed, their analysis based on normal intervals is unacceptable. In contrast, beta intervals are more dynamic by focusing on the population distribution's mean, deviation, and skewness. It should be noted that in a normal symmetric distribution, both beta and normal intervals would be similar. Listing 2 shows the instruction to call outlier functions. **Copatrec** arranges the input data (Listing 1) based on the selected analysis type. For example, when the panel outliers are called, all data for all times and all categories are used to find the pattern. The data arrangement is different for time-series and cross-sectional analysis. For example, when the time-series outlier function is called, **Copatrec** divides the initial dataset into the new datasets considering the various categories (based on the category column passed as input when the **Copatrec** object was generated (Listing 1)). Following this, it applies the outlier function on each new dataset differently. The returned dictionary has a relevant structure which has been commented on in Listing 2. All three outlier functions check for outliers on all variables.

2.3. Nonlinear regression and model selection

Like outliers, **Copatrec** has three nonlinear regression functions following their data structure (panel, time-series, and cross-sectional analysis). All functions have a set of parameters to facilitate outlier analysis and exclusion, data standardization, and plotting. It should be noted that **Copatrec** functions will follow the initial setup of **Copatrec** object. So, considering any of the regression functions, **Copatrec** iterates over all columns passed in the independent_vars parameter (if not specified, all other columns as an independent variable) corresponding to the initialized dependent variable. For each pair of dependent and independent variables, it tries to fit existent mathematical models and returns a dictionary of best-fitted regressions per independent variables (based on *Standard Error of Regression Estimation* (SE)). The result of nonlinear analysis functions (Listing 3) is three dictionaries (optimum mathematical form selected from the nonlinear analysis between the available dependent and all independent variables in pair modes (i.e., Eq. (1)), all mathematical forms between available dependent and all independent variables in pair mode, and error terms for all checked forms for each pair relation). The structure of all three outputs are commented in the Listing 3.

2.4. Result-object

The available packages in Python for regression analysis offer different regression evaluation and validation metrics. However, to facilitate the analysis progress, **Copatrec** returns a result-object as the result of each regression analysis, which contains many metrics (Table 3) that can be accessed throughout *ResultObject.report()* and *ResultObject.result_items()* functions. In addition to the evaluation metrics and variables, the result object has some built-in visualization and reporting functions (Table 1).

[1] Panel data refers to a collection of data for a variable (e.g., population) that includes different category groups (e.g., countries) for different time frames (e.g., 2000–2020).

[2] Time-series data refers a collection of data for a single category during different points in time, e.g., the population of Denmark from 2000 to 2020.

[3] Cross-sectional data refers to a collection of data at a specific point in time for different categories, e.g., the population of a group of countries in 2020.

```
# The panel_outliers for the current dataset will plot three figures.
# Figure 4 (a) => First figure as result of panel_outlier
# Figure 4 (b) => Third figure

intervals, outliers = SM.panel_outliers(
                            method='beta',
                            plot_pairs=True,
                            plot_hists=True,
                            plot_outliers_name=True)
# structure for panel analysis:
# intervals['independent variable']
################################################
# Additional
# Outlier analysis for a time-series setup
intervals, outliers = SM.time_series_outliers(
                            method='beta',
                            plot_pairs=True,
                            plot_hists=True,
                            plot_outliers_name=True)
# structure for panel analysis:
# intervals['variable']['category']
# i.e., intervals['GDPPC']['Norway']
################################################
# Outlier analysis for a time-series setup
intervals, outliers = SM.cross_sectional_outliers(
                            method='beta',
                            plot_pairs=True,
                            plot_hists=True,
                            plot_outliers_name=True)
# structure for panel analysis:
# intervals['variable']['time']
```

Listing 2: Outlier Analysis

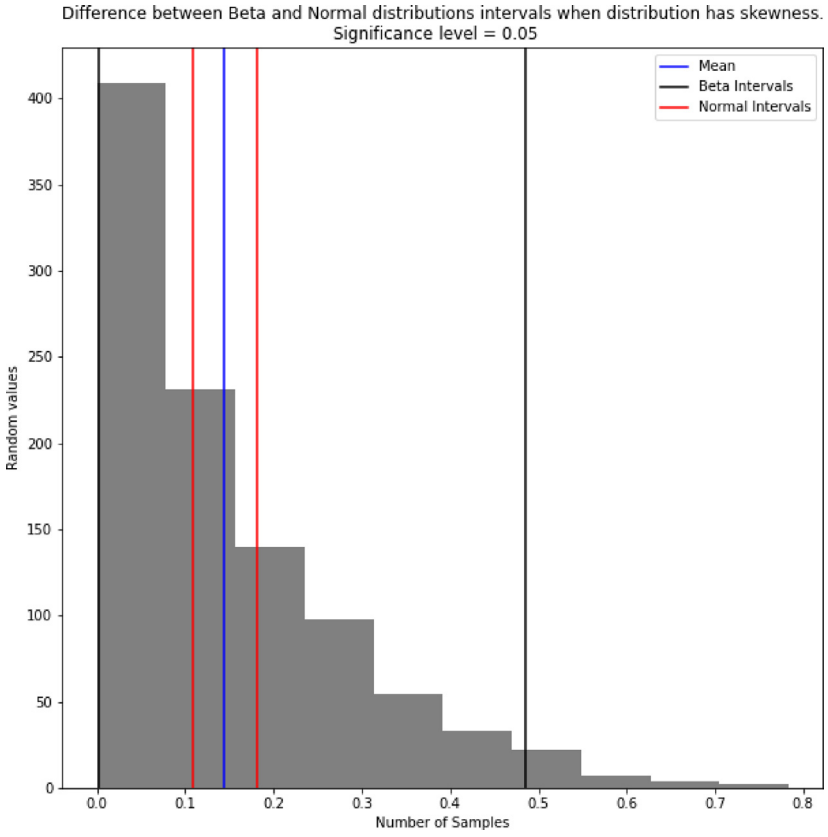


Fig. 3. Difference between beta and normal distributions.

Table 1
Copatrec Result-object functions.

Function	Description
help()	The help() function lists other available functions in the result-object.
report()	The report() function prints a nice report about the model, including all statistical validation variables. If a list of variables passes into the function, it will return the requested variables.
plot()	The plot() function plots the relevant data and fitted mathematical expressions.
predict()	The predict() function receives an array of independent variables and returns its predicted values.
result_items()	The result_items() function prints all callable variables presented in the result-object.
error_hist()	The error_hist() function prints the histogram of the error terms using the fitted equation form.
save()	The result-object can be saved using the save() function in a pickle format. Users can load it later without any need to install the **Copatrec** package.

```
# Panel regression analysis
# Figure 5 (a) => drop_outliers=True
# Figure 5 (b) => drop_outliers=False
Opt_Dict, All_Dict, Errors = SM.panel(
                                max_epochs=8000,
                                alpha=0.05,
                                standardization=True,
                                plot=False,
                                show_time_label=False,
                                show_category_label=False,
                                drop_outliers=True,
                                show_outliers=True,
                                plot_predicted_outliers=True,
                                outlier_method='beta')
# Opt_Dict['Independent Variable'] = result-object
# All_Dict['Independent Variable']['available mathematical expressions'] = result-object
# All_Dict['Independent Variable']['available mathematical expressions'] = result-object
# For the available mathematical expressions, please refer to Appendix A.
##################################################
# Time-series regression analysis
Opt_Dict, All_Dict, Errors = SM.time_series(parameters)
# Opt_Dict['Independent Variable'][category] = result-object
# All_Dict['Independent Variable'][category]['available mathematical expressions'] = result-object
# All_Dict['Independent Variable'][category]['available mathematical expressions'] = result-object
##################################################
# Cross-sectional regression analysis
Opt_Dict, All_Dict, Errors = SM.cross_sectional(parameters)
# Opt_Dict['Independent Variable'][time] = result-object
# All_Dict['Independent Variable'][time]['available mathematical expressions'] = result-object
# All_Dict['Independent Variable'][time]['available mathematical expressions'] = result-object
```

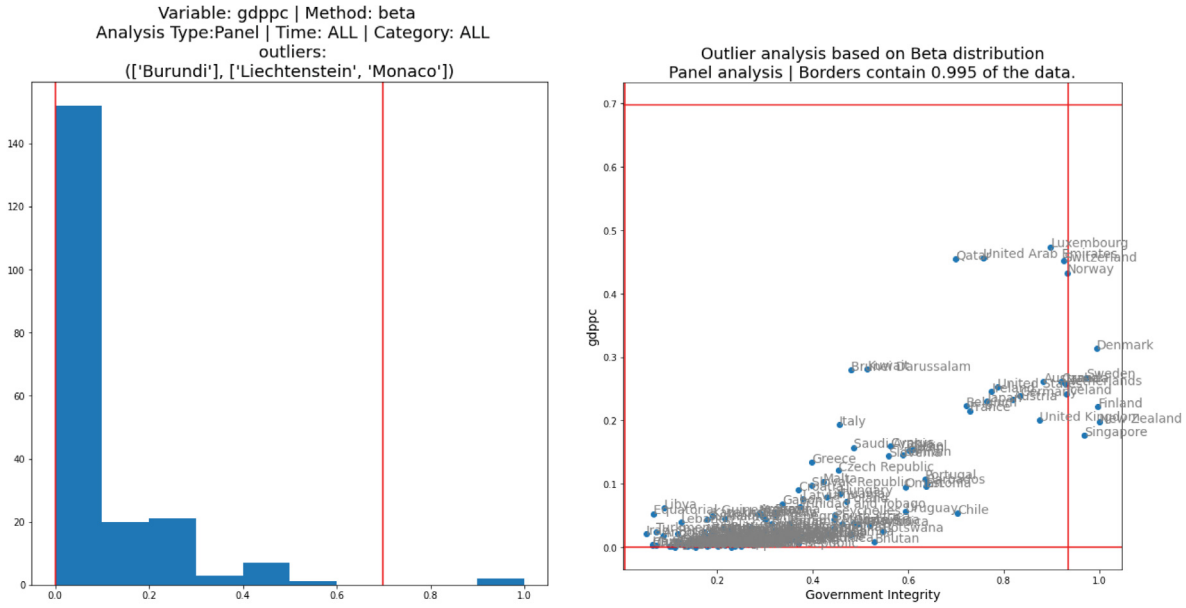
Listing 3: Regression Analysis

3. Illustrative example

In this section, we introduce the principal use of **Copatrec** capabilities using data from the area of political economy. We draw on a dataset of two variables, including *Gross Domestic Product per Capita* (**GDPPC**) [18], and *Government Integrity* [19] named *GRIN* (Gross and Integrity). GDP's status is subject to the ongoing discussion in the context of *International Political Economics* (**IPE**) [20], given the many economic and political factors that can affect it. Among all factors, Government Integrity is considered one of the crucial requirements for GDPPC [20]. It is measured based on *Corruption Perceptions Index*, which is a variable to measure insecurity and uncertainty in economic relations. Following the main purpose of this paper, the aim of selecting these variables and exploring their interaction is to show the practical application of **Copatrec** and the nuance in analysis that the tool affords (i.e., revealing insights that may not have been accessible based on linear regression alone), without extending the discussion to a substantive debate related to those variables, let alone theoretical or practical implications.

3.1. Outliers

As discussed before, handling outliers is an essential feature in the context of data preprocessing and understanding. **Copatrec** offers the *panel_outliers()* function (all data for all categories



(a) Outliers (histogram) of GDPPC (b) Outliers in pair plot mode

Fig. 4. Illustration of Outliers (for larger version see Fig. A.6(a) and Fig. A.6(b) in Appendix A).

and times in the dataset) for this purpose. In terms of visual comparison, the produced figures appear similar for all beta, normal, and IQR (*Inter-Quartile Range*) methods (Fig. 4). As the first step, the corresponding code to recognize outliers can be found in Listing 2. As is illustrated in Fig. 4, the outlier function can study outliers per each variable (Fig. 4(a)) and in pair modes as a combination of the dependent and independent variable (Fig. 4(b)). As a result, the first list in the list of outliers presents the outliers group based on the lower band analysis, while the second list presents outliers based on the upper bound intervals in a dictionary form in which keys are variables presented in the dataset columns (Listing 2).

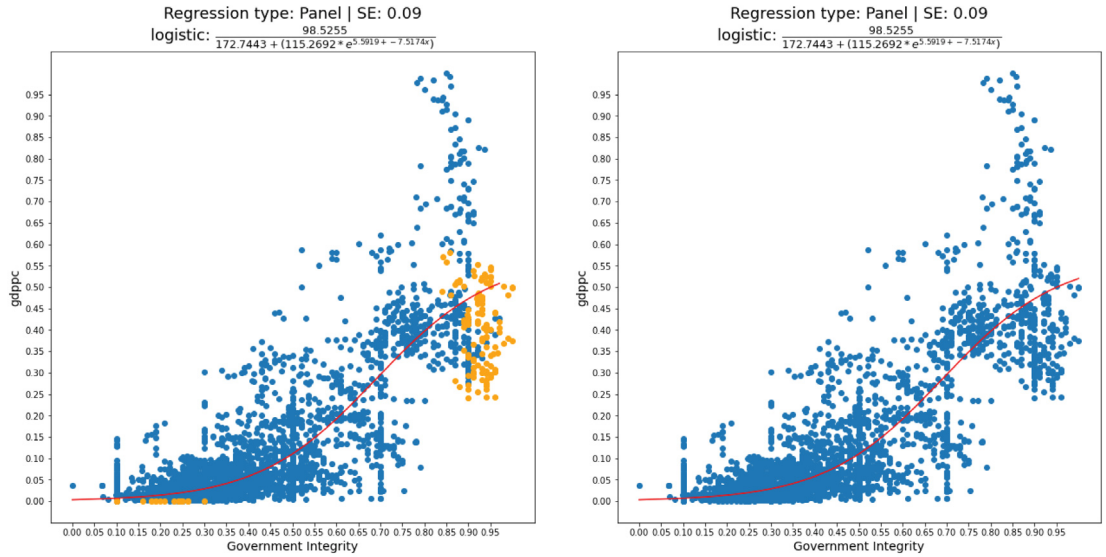
3.2. Regressions

Turning toward the actual analysis, an example of panel analysis (*panel()*) will be illustrated in this sub-section, specifically since it is commonly used in regression analyses. Fig. 5(a) has a *drop_outlier* equal to *True*; thus, the orange points have not been included in the analysis process. On the other hand, Fig. 5(b) includes all points. In this particular example, most outliers have already been dropped (i.e., *Liechtenstein*, but because of null values for *Government Integrity*), the remaining outlier points do not have a specific effect on the final equation form. For the exemplary case, the output of models in Fig. 5(b) is as follows:

$$Y = \frac{98.5255}{(172.7443 + (115.2692 * e^{5.5919 - 7.5174X}))}, \quad (1)$$

$$SE = 0.09,\ R^2 = 0.713$$

Eq. (1) (in which Y stands for the GDPPC and X stands for Government Integrity) and Fig. 5 indicate an S-shape growth relation between Government integrity and GDPPC with a standard error of 0.09 (the standard error has the same unit regarding the dependent variable. So, its high or low error amount should be interpreted by an expert, except if the data has been standardized, which can be interpreted easily considering the accepted error, i.e., 0.05). The analysis could imply that government integrity affects the GDPPC, but as most of the variables in complex systems, it is not the only factor. So, while government integrity affects GDPPC positively, it has a maximum capacity to affect the GDPPC. At the initial stage in which government integrity is low (high corruption and flawed system structure), GDPPC likewise resides at lower levels. We can observe that the growth of government integrity has exponential effects at first; however, the more it reaches its capacity of impact, the less effect it shows, converging toward a plateau. Indeed, as mentioned before, these are exemplary analyses, and the relation between government integrity and GDPPC should be studied separately in relation to their context. In this illustration, **Copatrec** highlights its potential compared to the already existing approaches like linearization and reviewed nonlinear packages. For instance, by using a linear method, like logarithmic linearization, since the logarithmic approach can only re-scale exponential behaviors to a linear form, it would only focus on a potentially present exponential growth part of the behavior, as shown in Fig. 5. The impact of this simplification may be much more significant for imbalanced datasets, i.e., datasets in which the number of data points for important features of complex behavior is underrepresented for selected categories. Thus, their weights become less pronounced, and the emerging linear regression may crowd out those aspects – an effect that the nonlinear methods avoid.



(a) Panel regression using logistic form with dropped outliers

(b) Panel regression using logistic form with all data

Fig. 5. Results of logistic equation fitted on panel data (for large version see Fig. A.7(a) and Fig. A.7(b) in Appendix A).

Table 2
Mathematical Expressions supported by **Copatrec**.

Function	Description
Euler Exponential	$\beta_0 + \beta_1 e^{(\beta_2 x+\beta_3)}$
Exponential	$\beta_0 + \beta_1 \beta_2^{(\beta_3 x+\beta_4)}$
Natural Logarithm	$\beta_0 + \beta_1 \ln(\beta_2 x)$
Logarithm	$\beta_0 + \beta_1 \log(\beta_2 x)$
Tangent	$\beta_0 + \beta_1 \tan(\beta_2 x + \beta_3)$
Sin hyperbolic	$\beta_0 + \beta_1 \sinh(\beta_2 x + \beta_3)$
Tangent hyperbolic	$\beta_0 \tanh(\beta_1 x + \beta_2)$
Logistic	$\frac{\beta_0}{\beta_1+\beta_2 e^{(\beta_3 x+\beta_4)}}$
Sin	$\beta_0 + \beta_1 \sin(\beta_2 x + \beta_3)$
Cosine	$\beta_0 + \beta_1 \cos(\beta_2 x + \beta_3)$
Growth-overshoot	$\beta_0 + ((\frac{\beta_6}{(\beta_7 e^{\beta_8+(\beta_9 x)})}) * (\frac{1}{1+e^x})) + (((\beta_1 \sin(\beta_2 x^{\beta_3} + \beta_4)^{\beta_5}) * (\frac{1}{1+e^{-x}}))$
Double Gaussian	$\beta_0 + ((\frac{\beta_2}{(\beta_6\sqrt{2\pi})}) * (e^{-(\frac{x-\beta_4}{2\beta_6})^2})) + ((\frac{\beta_3}{(\beta_6\sqrt{2\pi})}) * (e^{-(\frac{x-\beta_5}{2\beta_6})^2}))$
Cosine hyperbolic	$\beta_0 \cosh(\beta_2 x + \beta_3)$
Polynomial	$\beta_0 + \beta_1 x^{\beta_2}$

Nonlinear regression is a common technique in scientific analysis. Nevertheless, facilitating model selection, validation and visualization are contributions by **Copatrec** not present in other packages to date. The combination of equation forms and their ad-hoc visualization can be helpful in a range of scientific fields. Fig. 5, for instance, facilitates the interpretation that Government Integrity has a considerable capacity to affect GDPPC. Hence, while it is central to achieve higher Government Integrity scores for a given country, decisions to improve such metric require detailed and accurate insight into the underlying factors. In complex systems, the state of a variable, such as the GDPPC, is usually affected by multiple variables, each of which has varying influence based on their maximum effect capacity and current level. Extracting and understanding this is central to advising further analyses or interventions (e.g., by political economists). Besides

Table 3
Copatrec result object's variables.

Row	Name	Description
1	Reg_Type	Regression type
2	Independent_Var	Name of independent variable
3	Dependent_Var	Name of dependent variable
4	Cross_section_time	Time corresponding to the cross-sectional analysis
5	Time_series_category	Category corresponding to the time-series analysis
6	Time_col_name	The name of time column in the data set
7	Cat_col_name	The name of category column in the data set
8	Func	callable function
9	Func_name	Function name
10	Equation_String	Text-based equation form
11	Equation_Latex	Latex-based equation form
12	Coefficients	Estimated optimal coefficients of parameters
13	Covariance_Coefficients	The estimated covariance of parameters
14	N	Number of observations
15	Deg_Free	Degree of freedom
16	DFT	Corrected degrees of freedom total
17	Drop_outliers	Have outliers been dropped?
18	Outlier_method	Which method is used to drop outliers?
19	Intervals	List of intervals
20	Outliers	List of outliers
21	Alpha	Significance level which is used for hypothesis analysis
22	Standardization	Is data standardized in the process?
23	R2	R-squared of the equation; valid only for linear forms
24	R2adj	Adjusted R_squared
25	SE_Params	One standard deviation errors of parameters
26	T	T-values of parameters
27	T_Table	T table value for the input alpha
28	Par_P_values	Coefficients' p-values
29	Par_CI	Confidence intervals for the parameters
30	SSE[a]	Sum of squared estimate of errors
31	MSE	Mean squared error
32	R_MSE	Root mean squared error
33	MST	Mean of squares for total
34	SE	Standard error of regression/estimation
35	RSE	Relative standard error of estimation
36	SSM	Corrected sum of squares for model
37	SST	Corrected sum of squares total
38	DMF	Corrected degrees of freedom for model
39	MSM	Mean of squares for model
40	F	F_statistic value of the model
41	F_Table	Corresponding F value in the table
42	Prob_F	Probability of null hypothesis
43	Errors	All estimation error terms
44	Error_Normal_Test	Normality test of errors
45	Is_Error_Normal	Indicates whether errors are normally distributed
46	Error_Stats	Statistically analysis of errors
47	Data	The data set of initial and estimated values

[a]Also known as *residual sum of squares* (*RSS*) or *sum of squared residuals* (*SSR*).

the general patterns observed from the graphs, the derived equation helps to identify the underlying factors, their effects and capacity with greater precision.

4. Impact

This paper responds to the need to operate on nonlinear relations to capture patterns of complex behaviors in different fields like computer vision, health, social, political, and economic studies, the underlying mathematical expressions as well as statistical analyses. We have highlighted the underlying conceptual foundations of commonly found complex behaviors which, in some cases, could be produced using simulation-based methods. To provide context, existing Python packages that aim at data-driven nonlinear relations have been introduced, and their shortcomings have been identified, motivating the development of **Copatrec**, which showcases the following key features:

- **Model selection**: **Copatrec** is populated with a set of general mathematical expressions that have been collected, designed and introduced with the capabilities of capturing commonly observed complex behaviors, including growth, goal-seeking, *s*-shape, oscillators, growth with over-shoot and overshoot and collapse. Doing so, it aims to be versatile and largely independent from the application domain. The mathematical collection and automatic fitting process can be helpful to model selection problems.
- **Preprocessing**: **Copatrec** provides utility functions (outlier analysis, standardization, and data cleaning) to preprocessingthe dataset and support advanced analysis, which can be used before nonlinear analysis to calibrate or refine

the dataset. This aims at providing users with control over preprocessing and to accommodate specific analytical objectives or domain-specific considerations. The beta distribution, known as 'the distribution of distributions', has been added as a method to the outlier analysis that can be used on any population regardless of their distribution.

- **Balanced and unbalanced datasets**: **Copatrec** has specific functions to study a dataset's panel, time-series, and cross-sectional combinations separately, providing a variety of data structures for downstream processing and off-the-shelf visualization capabilities that can accommodate different forms of presentation.
- **Validation and illustration**: In contrast to existing packages, **Copatrec** iterates over all prepopulated mathematical expressions and returns best-fitted expressions specifically (based on the predefined error metric), but also other candidate expressions, as well as any relevant errors specific to the corresponding mathematical expression. This is to ensure transparency and reproducibility of the process, alternative analytical metrics (as mentioned above), and the clear indication of errors as part of the user feedback to assess the processed data prior to analysis.
- **Prediction**: Specifically, the ability of **Copatrec** to find the best-fitted function automatically and its predict() function makes it a potential tool to be used in combination with other methods and packages. For example, in demographic and simulation models, **Copatrec** can be used for synthetic population generation, i.e., to generate general distributions based on provided partial input data or to impute data for the provided form in the first place. The result of the analysis is stored in a specific object which contains different report styles, statistical analyses, and predefined visualization functions corresponding to the relevant mathematical expressions.

In addition to the provided analytical benefits, **Copatrec**'s feature further provides tangible, practical value for researchers from diverse fields, supporting both the understanding of the analysis and its communication as well as a transparent operation that is essential for rigorous scientific assessment. Potential use cases include:

- Sense-making and Prototyping: Researchers may use the package to explore their data and develop an initial understanding while being able to quickly reiterate the fitting process to assess the impact of their modification (e.g., preprocessing) on the outcome. A specific opportunity can lie in generalizing a mathematical model from partial data. This can support parameterization of simulations, including prototyping or testing models' sensitivity to varying mathematical input forms.
- Visualization: The package provides built-in visualization features that support the communication of research results or insights, provided by the same tool used to detect the equation patterns, and hence offering an integrated insight into both process and output.
- Methodological rigor: In addition to the visualization output, **Copatrec** provides detailed traces of its operation so that individual steps can be retraced to not only support transparent execution, but also reproducibility, and promote the integration into researchers' analytical toolchains.

A web-based application that supports the interactive use of the package will be available under https://copatrec.org. For instructions for local deployment, the latest features, as well as the codebase of the package, readers are referred to GitHub and Pip repositories listed in the code metadata table.

5. Conclusion

In this article, we introduced **Copatrec**, a tool to support the pre-processing of data and detection of correlation patterns in structured data (i.e., panel, time-series, and cross-sectional). Following the description of its essential features (e.g., preprocessing, model selection, balanced and unbalanced data set support, result object (validation and illustration)), we illustrated the application of **Copatrec** using an existing data set and drew attention to specific characteristics, such as outlier handling and equation pattern detection, that the package facilitates, but also the customizability that the package affords.

While this initial version of the package provides a range of capabilities to facilitate the study of complex data, it leaves room for further development, including support for multivariate regression, additional mathematical expressions, differential forms of expressions, as well as additional preprocessing functions.

Notwithstanding these future development directions, **Copatrec** introduces a novel and integrated approach to facilitate the deep study of data to leverage insights that conventional correlation studies based on traditional linear regression and other standard approaches are challenged to reveal, while making this ability accessible to researchers across a wide range of disciplines and interests.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

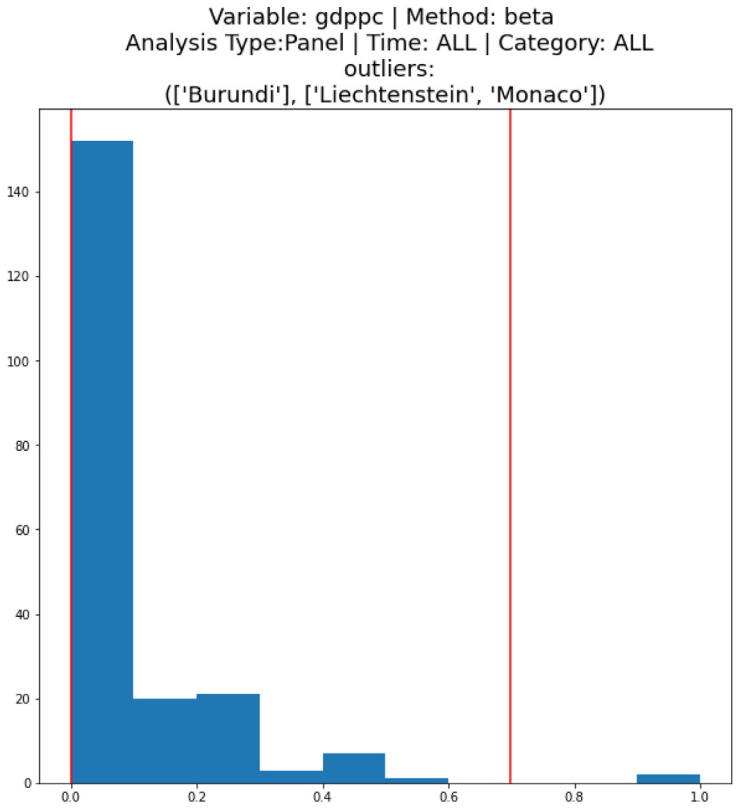
The data used in the illustrative example section is accessible via the GitHub repository.

Acknowledgments

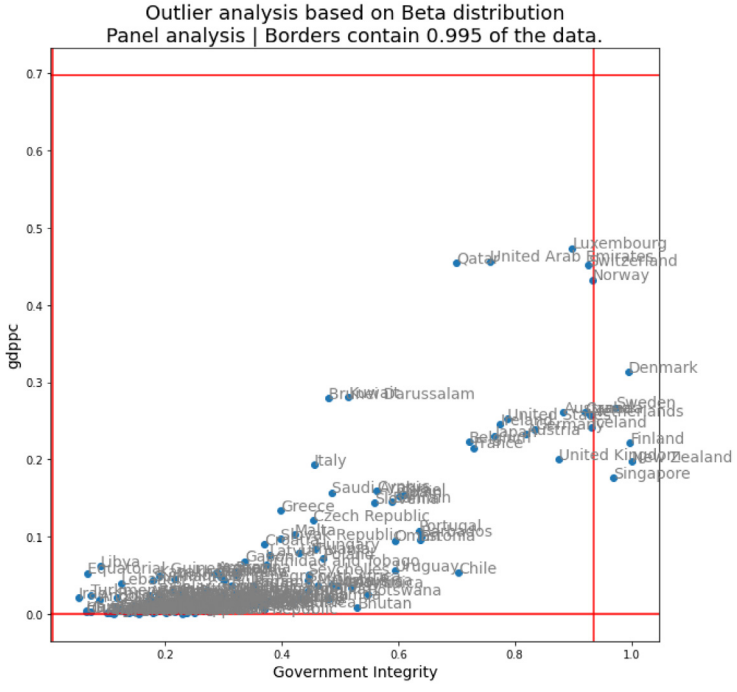
We wish to thank Jonathon Moses and Indra de Soysa for their constructive feedback on the tool and its application.

Appendix A. Enlarged images

See Figs. A.6 and A.7.

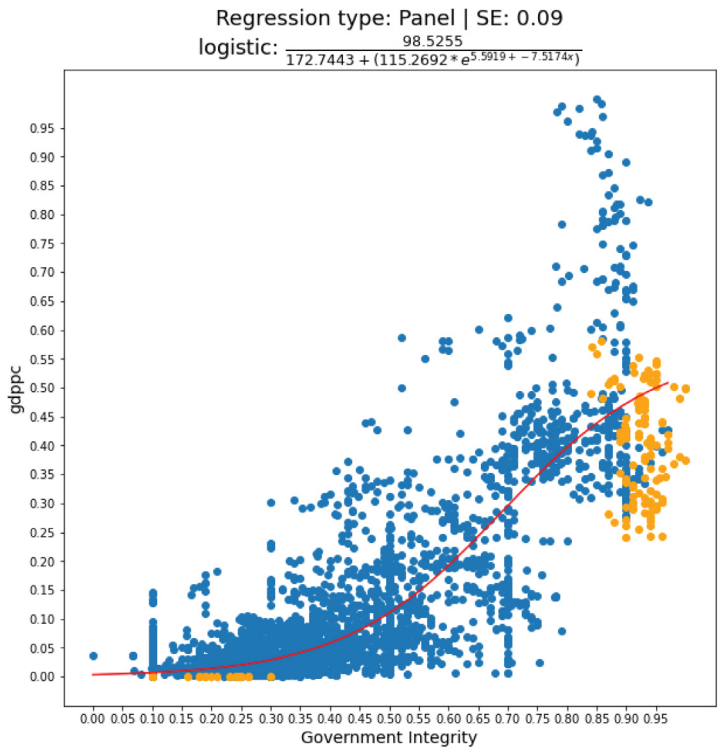


(a) Outliers (histogram) of GDPPC

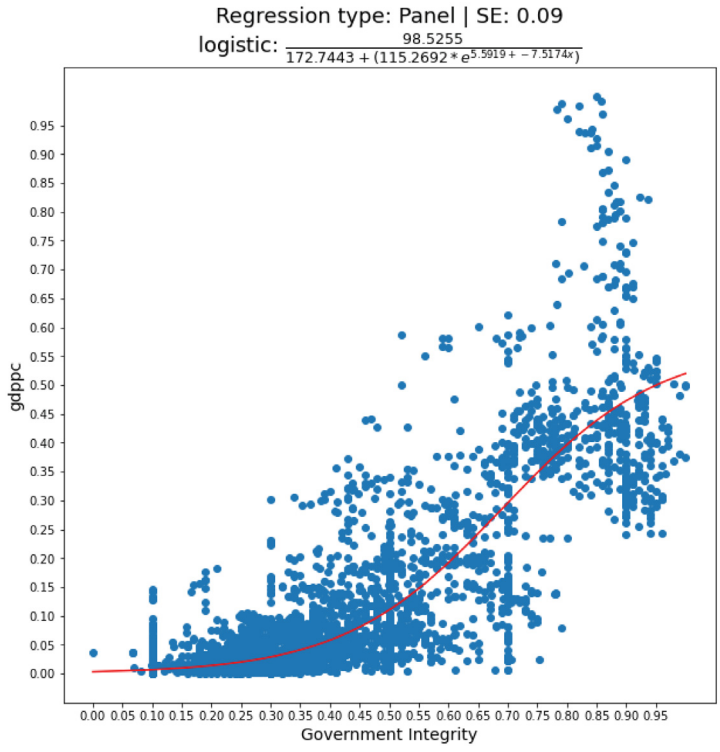


(b) Outliers in pair plot mode

Fig. A.6. Enlarged version of Fig. 4.



(a) Panel regression using logistic form with dropped outliers



(b) Panel regression using logistic form with all data

Fig. A.7. Enlarged version of Fig. 5.

References

[1] García S, Ramírez-Gallego S, Luengo J, Benítez JM, Herrera F. Big data preprocessing: Methods and prospects. Big Data Anal 2016;1(1):1–22. http://dx.doi.org/10.1186/S41044-016-0014-0.

[2] Hong X, Mitchell RJ, Chen S, Harris CJ, Li K, Irwin GW. Model selection approaches for non-linear system identification: A review. Internat J Systems Sci 2008;39(10). http://dx.doi.org/10.1080/00207720802083018.

[3] Rosipal R. Nonlinear partial least squares: An overview. In: Chemoinformatics and advanced machine learning perspectives: Complex computational methods and collaborative techniques. 2010, http://dx.doi.org/10.4018/978-1-61520-911-8.ch009.

[4] Gujarati DN. In: Sutton L, Bright A, editors. Basic econometrics. 4th ed.. Tata McGraw Hill; 2004, p. 1032.

[5] Kingma DP, Ba JL. Adam: A Method for Stochastic Optimization. In: 3rd International conference on learning representations, ICLR 2015 - Conference track proceedings. International Conference on Learning Representations, ICLR; 2014, http://dx.doi.org/10.48550/arxiv.1412.6980.
[6] Greasley A, Edwards JS. Enhancing discrete-event simulation with big data analytics: A review. J Oper Res Soc 2021;72(2). http://dx.doi.org/10.1080/01605682.2019.1678406.
[7] Sterman JD. Business dynamics : Systems thinking and modeling for a complex world. Boston: Irwin McGraw-Hill; 2000, p. 982.
[8] Edmonds B, Meyer R. In: Edmonds B, Meyer R, editors. Simulating social complexity - a handbook. Understanding complex systems, Cham: Springer International Publishing; 2017, p. 1–800. http://dx.doi.org/10.1007/978-3-319-66948-9.
[9] Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: Fundamental algorithms for scientific computing in Python. Nature Methods 2020;17(3):261–72. http://dx.doi.org/10.1038/s41592-019-0686-2.
[10] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine learning in Python. J Mach Learn Res 2011;12(85):2825–30, URL http://jmlr.org/papers/v12/pedregosa11a.html.
[11] Harris CR, Millman KJ, van der Walt SJ, Gommers R, Virtanen P, Cournapeau D, et al. Array programming with NumPy. Nature 2020;585(7825):357–62. http://dx.doi.org/10.1038/s41586-020-2649-2.
[12] Seabold S, Perktold J. Statsmodels: Econometric and statistical modeling with Python. In: 9th Python in science conference. SciPy; 2010, p. 92–6. http://dx.doi.org/10.25080/MAJORA-92BF1922-011.
[13] Newville M, Stensitzki T, Allen DB, Ingargiola A. LMFIT: Non-linear least-square minimization and curve-fitting for Python. Zenodo 2014. http://dx.doi.org/10.5281/ZENODO.11813.
[14] Hunter JD. Matplotlib: A 2D graphics environment. Comput Sci Eng 2007;9(3):90–5. http://dx.doi.org/10.1109/MCSE.2007.55.
[15] Forrester by Helen Zhu JW. Properties of damped oscillations systems. Tech. rep., 1998.
[16] Forrester JW, Agatstein KA. Oscillating systems II: Sustained oscillation. Tech. rep., 1997, p. 43.
[17] Cousineau D, Chartier S. Outliers detection and treatment: A review. Int J Psychol Res 2010;3(1). http://dx.doi.org/10.21500/20112084.844.
[18] World Bank. World development indicators. Tech. rep., Washington, D.C; 2022.
[19] Miller T, Kim AB, Roberts JM. Economic freedom. The Heritage Foundation; 2022, p. 490.
[20] Seligson MA, Passé-Smith JT, Wade RH, Banerjee AV, Duflo E. Development and underdevelopment: The political economy of global inequality, no. 5, fifth ed.. Lynne Rienner Publishers; 2013, p. 435.