Middle East Journal of Science

https://dergipark.org.tr/mejs

MEJS

e-ISSN:2618-6136

Research Article

THE USE OF PARTIALLY LINEAR REGRESSION MODEL IN IMAGE PROCESSING

***Merve Bingöl Sukutli* [1]** ***Seçil Yalaz* [*2]**

[1] Dicle University, Science Faculty, Statistics Department, Türkiye
[2] Dicle University, Science Faculty, Statistics Department, Türkiye
[*] Corresponding author; syalaz@dicle.edu.tr

Abstract: *Digital imaging systems are increasingly popular in various fields such as education, industry, engineering, and healthcare. The ease of use and low cost of these systems contribute to their widespread adoption. However, the main disadvantage of digital imaging is resolution issues. In practical applications that require high resolution, dense sensors are used to obtain robust images. This method, however, increases costs and produces more data and noise due to its density. Additionally, millions of low-resolution but valuable pieces of information are lost. Image processing techniques are used to enhance resolution and preserve high-frequency information. The primary aim of this study is to comprehensively investigate the importance and effectiveness of using partially linear models in image processing applications. Partially linear regression aims to offer a new model for image enhancement without losing high-frequency information. Because many problems encountered in the field of image processing stem from resolution issue, this study aims to understand the effects of resolution on image processing processes and to demonstrate how partially linear models can be used to address these effects. Various comparison methods have been used to evaluate the effectiveness of the proposed method. These methods have been employed to objectively assess the quality difference between images, highlighting the superiority of the proposed method over traditional methods. The study's findings show that partially linear models are a significant tool in image processing applications. Future studies may aim to examine in more detail how these models perform with different types of images and conditions.*

Keywords: *Digital Imaging Systems, Image Enhancement, Image Processing, Partially Linear Regression, Resolution*

Received: August 1, 2024 *Accepted: February 15, 2025*

1. Introduction

At the core of image processing lies the conversion of images captured by an optical sensor (camera) into another form in a computer environment [1]. The analysis and classification of these transformed forms rely on regression analysis methods.

Regression analysis is a statistical method that allows for the examination of the relationship between a dependent variable and one or more independent variables [2]. This method is used particularly to understand the relationship between variables and to examine the effects of independent variables on the dependent variable. Additionally, regression analysis expresses the effects of variables on each other as a mathematical function [1].

Depending on the characteristics of the dataset, the purpose of the analysis, and the assumptions, various regression approaches exist. At the center of these approaches is the parametric regression model, which works under the assumption that the form of the relationship between independent and dependent variables is known. In a parametric model, if a linear relationship is observed between the dependent variable and a single independent variable, the model is called a simple regression model.

In image processing, the linear regression method used to understand and predict the relationship between 2D data pairs expresses each pixel in the image as an independent variable, and the features or measurements corresponding to the pixel values as the dependent variable.

In addition to linear regression, multiple linear regression analysis methods are used for models that include more independent variables. While this method is used in fields like statistics and machine learning to relate a dependent variable to multiple independent variables and to model this relationship, in image processing it is used to evaluate and model the collective effect of multiple independent variables on the dependent variable (image). Multiple linear regression analysis methods are utilized to segment images (define the boundaries of objects in the image and divide the image into sections), achieve image restoration (remove noise and repair noised areas in the image), and obtain high-resolution images from low-resolution images.

It is not correct to say that only parametric (linear) models are preferred for regression. In situations where the form of the relationship between variables is unknown, non-parametric and semi-parametric (partially linear) methods are preferred, which are used in various fields such as economics, finance, marketing, medicine, health sciences, machine learning, data mining, and image processing.

Recently, the increasing popularity of image processing has made it an intriguing topic for analysts, engineers, and statisticians. Ease of use and low cost have made method selection in image processing preferable. However, with the reduction in cost, issues such as resolution, decrease in image quality, and noticeable overlapping effects have emerged. These problems stem from the limited number of highly correlated sensors used in digital imaging devices. Increasing the number of pixels used not only raises the cost but also causes blur in the image, increasing noise [3].

The aim of this study is to comprehensively examine the importance and effectiveness of using partially linear models in image processing applications. The study aims to explore the effects of image processing on workflows and how partially linear models can mitigate these effects.

2. Materials and Methods

This study examines the impact of using regression analysis within partially linear models in image processing techniques. Literature reviews on the concept of partially linear regression have revealed general approaches for parameter estimation. It has been observed that problems related to image processing are solved using kernel and Gaussian blurring methods in the relevant regression.

The study explains the correction methods for noised images. By comparing the histogram values of the noised and corrected images, a homogeneous distribution has been obtained for the corrected image. Subsequently, it has become possible to estimate the noise level by modeling the relationship between the images.

After all processes were completed, the noised and corrected images were displayed on the screen. The study includes the steps used to process images from the specified file path and visualize the results, designed to meet the specified requirements.

2.1. Regression Models and Image Processing

Regression analysis is a statistical method used to make inferences about the regression function and is primarily used to examine the relationship between two variables. Parametric and non-parametric regression techniques approach regression analysis from two different perspectives. Parametric regression has strong assumptions, whereas non-parametric regression does not require these assumptions.

In parametric regression analysis, the parameters (β) of a pre-specified model are estimated, while in non-parametric regression analysis, the goal is to directly estimate the regression function (g) [3].

In image processing, digital filters are used to smooth images or enhance edges. Some of the filters used include the mean filter, median filter, Gaussian smoothing filter, Kalman filter for object recognition in images, frequency filters, and kernel regression.

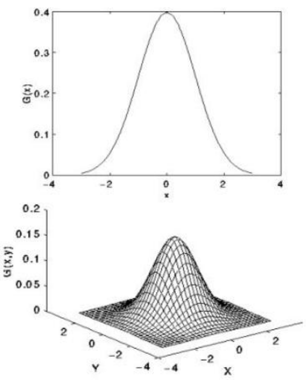
The mean filter replaces each pixel value in an image with the average value of its neighbors, including itself. This process removes pixel values that are not meaningful for the image. The mean filter is a convolution filter based on a kernel template.

The median filter, like the mean filter, considers neighboring pixels to calculate the value of each pixel. However, instead of replacing the pixel value with the average of the neighboring pixel values (mean filter), it sorts the neighboring pixels and takes the median value. If the region being examined (within the template) has an even number of pixels, the average of the two middle pixels is used as the median value.

The Gaussian smoothing operator is a two-dimensional (2D) convolution operator used to blur images by removing details and noise. It is similar to the mean filter but uses a different kernel template. This template is represented by a bell-shaped curve known as the Gaussian function. The formula for the Gaussian function that gives the bell-shaped curve can be written in both the 2D plane and the 3D space.

$$G(x) = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{x^2}{2\sigma}}$$

$$G(x) = \frac{1}{\sqrt{2\pi}\,\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



x	σ	$\frac{1}{\sqrt{2\pi}\,\sigma}$	$e^{-\frac{x^2}{2\sigma}}$	$z = \frac{1}{\sqrt{2\pi}\,\sigma} e^{-\frac{x^2}{2\sigma}}$
-2	1	0,398943084	0,135335283	0,053991075
-1,9	1	0,398943084	0,164474457	0,065615947
-1,8	1	0,398943084	0,197898699	0,078950317
-1,7	1	0,398943084	0,235746077	0,094049267
-1,6	1	0,398943084	0,2780373	0,110921058
-1,5	1	0,398943084	0,324652467	0,129517856
-1,4	1	0,398943084	0,375311099	0,149727767
-1,3	1	0,398943084	0,429557358	0,171368937
-1,2	1	0,398943084	0,486752256	0,194186446
-1,1	1	0,398943084	0,546074427	0,217852616
-1	1	0,398943084	0,60653066	0,241971212
-0,9	1	0,398943084	0,666976811	0,266085786
-0,8	1	0,398943084	0,726149037	0,289692136
-0,7	1	0,398943084	0,782704538	0,312254562
-0,6	1	0,398943084	0,835270211	0,333225274
-0,5	1	0,398943084	0,882496903	0,352066036
-0,4	1	0,398943084	0,923116346	0,368270882
-0,3	1	0,398943084	0,955997482	0,381388584
-0,2	1	0,398943084	0,980198673	0,391043481
-0,1	1	0,398943084	0,995012479	0,396953347
0	1	0,398943084	1	0,398943084
0,1	1	0,398943084	0,995012479	0,396953347
0,2	1	0,398943084	0,980198673	0,391043481
0,3	1	0,398943084	0,955997482	0,381388584

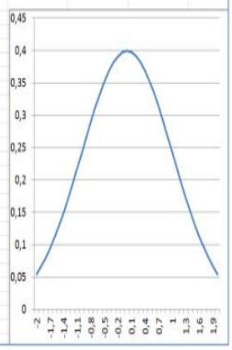


Figure 1. Gaussian filter for 2D plane and 3D spaces

In Figure 1, (σ) represents the standard deviation of the distribution. It is also assumed that the mean of the distribution is zero (i.e., centered on the line).

The purpose of the Kalman filter is to predict the multi-person pose tracking system developed and evaluate it in terms of processing time using various pose estimation algorithms. Pose estimation arises from determining the pixel positions of key points of the human skeleton viewed by the camera. The outputs of pose estimation methods associate the pixel values of all detected joint points in the image with the relevant person. Identifying individuals across successive frames in videos is particularly important for understanding their movements. This allows for determining what kind of movements individuals make and at what moments in the video [4].

Kernel regression is an image-processing method. In this method, scaling, rotation, and stretching parameters are calculated from the original data analyzed for local structures to provide directive matrices. These calculations are performed using singular value decomposition (SVD) with locally estimated gradients on local gradients. The algorithm is fed with raw pixels associated with each image. This method requires minimal preprocessing of images before classification, making it more useful than traditional training algorithms [1].

2.1.1 Partially Linear Regression

Partially linear models are a special model that includes both parametric and nonparametric methods. These models are used when part of the data can be explained with a parametric formula, but another part needs to be modeled more flexibly [1]. The formulation of the model is as follows:

$$Y = \boldsymbol{X}\beta + g(x^*) + \varepsilon, \tag{1}$$

where $\boldsymbol{X}\beta$ represents the parametric component, $\boldsymbol{X}$ is the $(n \times p)$ dimensional independent variable matrix, β is the $(p \times 1)$ dimensional coefficient vector, x^* is the nonparametric variable vector, $g(.)$ is the nonparametric function, and ε is the error term.

Because the partially linear regression model combines both parametric and nonparametric regression functions, it is also called semiparametric regression [5]. This model offers a more flexible and adaptable analysis method by using both parametric and nonparametric components together. If the effects of the parametric variables are ignored or if these variables are not included in the analysis, the semiparametric regression model becomes a purely nonparametric regression model. This feature makes the semiparametric regression model a powerful tool for examining both linear and nonlinear relationships simultaneously.

In the partially linear model, a kernel corrector $K_h(x^*)$ applied with bandwidth h is used. This corrector, used to calculate the effect from other points at point x^*, is shown as:

$$w(x^*) = \frac{K_h(x_i^* - x^*)}{\sum_{i=1}^{n} K_h(x_i^* - x^*)} \tag{2}$$

where $K_h(x^*) = \frac{1}{h_n} K\left(\frac{x^*}{h_n}\right)$, K is the kernel function and h_n is the bandwidth parameter. Using the kernel corrector, the effect at point x^* on variables x and y is estimated using the some regulations in Nadaraya-Watson estimator (NWE) separately:

$$\hat{g}_{x,h}(x_i^*) = \sum_{i=1}^{n} w(x^*) x_i$$

$$\hat{g}_{y,h}(x_i^*) = \sum_{i=1}^{n} w(x^*) y_i$$

Using these estimates, new variables cleared of the effect are obtained by removing the effect at point x^* from the variables x and y as:

$$\tilde{Y}_i = Y_i - \hat{g}_{y,h}(x_i^*),$$

$$\tilde{X}_i = X_i - \hat{g}_{x,h}(x_i^*).$$

The parametric component is estimated by:

$$\hat{\beta} = (\tilde{X}^T \tilde{X})^{-1} \tilde{X}^T \tilde{Y}, \tag{3}$$

for the estimated variance $\hat{\sigma}^2 = (\tilde{Y}^T \tilde{Y} - \hat{\beta}^T \tilde{X}^T \tilde{Y})/(n-2)$. And the nonparametric function is estimated as:

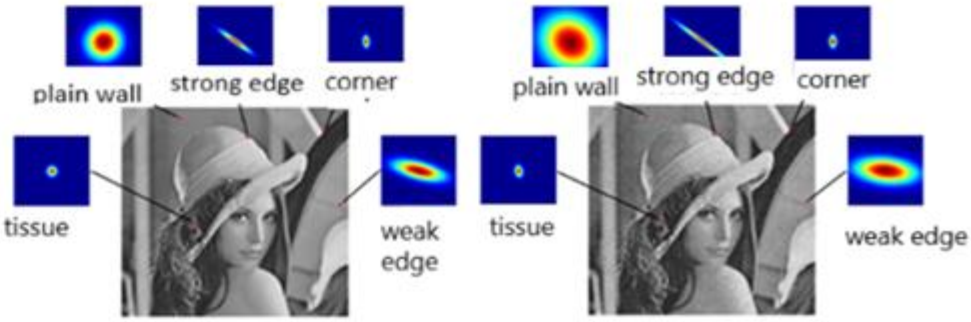
$$\hat{g}(x^*, \hat{\beta}) = \sum_{i=1}^{n} w(x^*)(y_i - x_i \hat{\beta}), \tag{4}$$

which calculates the nonparametric effect of the y variable at point x^*. This process ensures the accurate analysis of relationships between variables in image processing applications and controls the effect of nonparametric components.

3. Findings and Discussion

3.1. Use of Partially Linear Model in Image Processing

The main aim of using regression predictors in image processing is to reduce noise while preserving the edge structure when correcting the image.



(a) (b)

Figure 2. (a) Noiseless and (b) Gaussian noisy image with SNR = 5.64dB

Figure 2 shows how local structures change in predictions for noiseless and noisy images [1]. In digital image processing, some filters are needed to smooth the image, highlight objects, or recognize them. In parametric models, different pixels are more easily identified, while it is quite difficult to distinguish this in similar scenes. Smoothing at the x^* point in any data set is obtained by fitting a weighted OLS line. In image processing, after determining the color weights of the image divided into pixels, a smoothed image is obtained by fitting the least squares line at any pixel.

When performing partially linear regression analysis in image processing, a second dimension (2-D) is added to classical prediction methods (see Equation (1)) [6]:

$$y_i = \boldsymbol{x}_i\beta + g(\boldsymbol{x}_i^*) + \varepsilon_i, \quad i = 1, \dots, n. \tag{5}$$

Here, the coordinates of measured data y_i is now the (2×1) vector $\boldsymbol{x}_i^*$, $g(.)$ is the (hitherto unspecified) regression function and ε_is are the i.i.d. zero mean noise values (with otherwise no particular statistical distribution assumed), $\boldsymbol{x}^*$ has dimensions $(n \times q)$. When expanding the nonparametric function $g(\boldsymbol{x}_i^*)$ becomes

$$g(\boldsymbol{x}_i^*) = g(\boldsymbol{x}^*) + \{\boldsymbol{\nabla} g(\boldsymbol{x}^*)\}^T(\boldsymbol{x}_i^* - \boldsymbol{x}^*) + \frac{1}{2}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)^T\{\boldsymbol{\mathcal{H}} g(\boldsymbol{x}^*)\}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)+\dots$$

$$= g(\boldsymbol{x}^*) + \{\boldsymbol{\nabla} g(\boldsymbol{x}^*)\}^T(\boldsymbol{x}_i^* - \boldsymbol{x}^*) + \frac{1}{2} vec^T\{\boldsymbol{\mathcal{H}} g(\boldsymbol{x}^*)\} vec\{(\boldsymbol{x}_i^* - \boldsymbol{x}^*)(\boldsymbol{x}_i^* - \boldsymbol{x}^*)^T\} + \cdots, \tag{6}$$

where $vec\left(\begin{bmatrix} a & b \\ b & d \end{bmatrix}\right) = [a \quad b \quad b \quad d]^T$ is a vectorization operator, $g(\boldsymbol{x}_i^*)$ represents the function's value at $\boldsymbol{x}_i^*$, $\boldsymbol{\nabla} g(\boldsymbol{x}^*)$ is the (2×1) gradient (derivative vector) at $\boldsymbol{x}^*$, and $\boldsymbol{\mathcal{H}} g(\boldsymbol{x}^*)$ is the (2×2) second derivative matrix (Hessian) at $\boldsymbol{x}^*$. The operation of vectorizing the elements of the lower triangle of a symmetric matrix is represented as $vech\left(\begin{bmatrix} a & b \\ b & d \end{bmatrix}\right) = [a \quad b \quad d]^T$ and $vech\left(\begin{bmatrix} a & b & c \\ b & e & f \\ c & f & i \end{bmatrix}\right)$=$[a \quad b \quad c \quad e \quad f \quad i]^T$.

$$g(\boldsymbol{x}_i^*) = \theta_0 + \boldsymbol{\theta}_1^T(\boldsymbol{x}_i^* - \boldsymbol{x}^*) + \boldsymbol{\theta}_2^T\, vech\{(\boldsymbol{x}_i^* - \boldsymbol{x}^*)(\boldsymbol{x}_i^* - \boldsymbol{x}^*)^T\} + \cdots \tag{7}$$

Comparing (6) and (7), $\theta_0 = g(\boldsymbol{x}^*)$ becomes the pixel values of interest and vectors $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$ becomes

$$\boldsymbol{\theta}_1 = \boldsymbol{\nabla} g(\boldsymbol{x}^*) = \left[\frac{\partial \mathrm{g}(\boldsymbol{x}^*)}{\partial \mathrm{x}_1^*}, \frac{\partial \mathrm{g}(\boldsymbol{x}^*)}{\partial \mathrm{x}_2^*}\right]^T, \tag{8}$$

$$\boldsymbol{\theta}_2 = \frac{1}{2}\left[\frac{\partial^2 \mathrm{g}(\boldsymbol{x}^*)}{\partial {\mathrm{x}_1^*}^2},\ 2\frac{\partial^2 \mathrm{g}(\boldsymbol{x}^*)}{\partial \mathrm{x}_1^*\, \partial \mathrm{x}_2^*}, \frac{\partial^2 \mathrm{g}(\boldsymbol{x}^*)}{\partial {\mathrm{x}_2^*}^2}\right]^T. \tag{9}$$

Regardless of parameter results, the main goal is to find pixel values. Therefore, the estimation of $\boldsymbol{\theta}_q$ is:

$$\underset{\{\boldsymbol{\theta}_q\}}{min} \sum_{i=1}^{n}[y_i - \boldsymbol{x}_i\beta - \theta_0 - \boldsymbol{\theta}_1^T(\boldsymbol{x}_i^* - \boldsymbol{x}^*) - \boldsymbol{\theta}_2^T vech\{(\boldsymbol{x}_i^* - \boldsymbol{x}^*)(\boldsymbol{x}_i^* - \boldsymbol{x}^*)^T\} \dots]^2 K_{\boldsymbol{H}}(\boldsymbol{x}_i^* - \boldsymbol{x}^*) \quad (10)$$

where $K_{\boldsymbol{H}}(t) = \frac{1}{\det(\boldsymbol{H})} K(\boldsymbol{H}^{-1}\boldsymbol{t})$ represents $\boldsymbol{H}$, the (2×2) smoothing matrix, and K, the 2-dimensional kernel function. Selection of $\boldsymbol{H}$ can be obtained using [1]. However, since the β parameters are also unknown here, they must first be estimated.

Let $\boldsymbol{\omega}$ be the weight matrix and $diag$ represent the diagonal matrix:

$$\boldsymbol{\omega} = diag\left[K_{\boldsymbol{H}}\left(\boldsymbol{x}_1^* - \boldsymbol{x}^*\right),\ K_{\boldsymbol{H}}\left(\boldsymbol{x}_2^* - \boldsymbol{x}^*\right), \dots, K_{\boldsymbol{H}}(\boldsymbol{x}_n^* - \boldsymbol{x}^*)\right]. \quad (11)$$

Let $\widehat{\boldsymbol{\omega}}_x = \boldsymbol{\omega}\boldsymbol{X}$ and $\widehat{\boldsymbol{\omega}}_y = \boldsymbol{\omega}y$ be the new variables stripped of the effect. For the new variables, $\tilde{y} = y - \widehat{\boldsymbol{\omega}}_y$ and $\widetilde{\boldsymbol{X}} = \boldsymbol{X} - \widehat{\boldsymbol{\omega}}_x$, the parametric component β is estimated by:

$$\hat{\beta} = (\widetilde{\boldsymbol{X}}^T\widetilde{\boldsymbol{X}})^{-1}\widetilde{\boldsymbol{X}}^T\tilde{y}. \quad (12)$$

Thus, by adding weights to the least squares method to determine the $\boldsymbol{\theta}$ coefficients, the model is allowed to perform better in situations with varying variance. Consequently:

$$\widehat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{argmin} \left|\left|\boldsymbol{y} - \boldsymbol{X}\hat{\beta} - \boldsymbol{X}_{\boldsymbol{x}^*}^*\boldsymbol{\theta}\right|\right|^2 \boldsymbol{\omega} = \underset{\boldsymbol{\theta}}{argmin} \left(\boldsymbol{y} - \boldsymbol{X}\hat{\beta} - \boldsymbol{X}_{\boldsymbol{x}^*}^*\boldsymbol{\theta}\right)^T \boldsymbol{\omega}\left(\boldsymbol{y} - \boldsymbol{X}\hat{\beta} - \boldsymbol{X}_{\boldsymbol{x}^*}^*\boldsymbol{\theta}\right). \quad (13)$$

Here, $\boldsymbol{y} = [y_1, y_2, \dots y_n]^T$ represents the observation vector, $\boldsymbol{\theta} = \left[\theta_0, \boldsymbol{\theta}_1^T, \dots, \boldsymbol{\theta}_q^T\right]^T$ represents the coefficient vector, and $\boldsymbol{X}_{\boldsymbol{x}^*}^*$ represents the matrix containing the distance of each point in the data set to the point $\boldsymbol{x}^*$. Therefore:

$$\boldsymbol{X}_{\boldsymbol{x}^*}^* = \begin{bmatrix} 1 & (\boldsymbol{x}_1^* - \boldsymbol{x}^*)^T & vech^T\left\{\left((\boldsymbol{x}_1^* - \boldsymbol{x}^*)(\boldsymbol{x}_1^* - \boldsymbol{x}^*)^T\right)\right\} & \dots \\ 1 & (\boldsymbol{x}_2^* - \boldsymbol{x}^*)^T & vech^T\left\{\left((\boldsymbol{x}_2^* - \boldsymbol{x}^*)(\boldsymbol{x}_2^* - \boldsymbol{x}^*)^T\right)\right\} & \dots \\ \vdots & \vdots & \vdots & \vdots \\ 1 & (\boldsymbol{x}_n^* - \boldsymbol{x}^*)^T & vech^T\left\{\left((\boldsymbol{x}_n^* - \boldsymbol{x}^*)(\boldsymbol{x}_n^* - \boldsymbol{x}^*)^T\right)\right\} & \dots \end{bmatrix}. \quad (14)$$

To estimate the pixel values of the image, regardless of the estimator order q, we focus on the estimators of the θ_0 parameter:

$$\hat{g}(\boldsymbol{x}^*) = \hat{\theta}_0 = \boldsymbol{e}_1^T\left(\boldsymbol{X^*}_{\boldsymbol{x}^*}^T\boldsymbol{\omega}\boldsymbol{X}_{\boldsymbol{x}^*}^*\right)^{-1}\boldsymbol{X^*}_{\boldsymbol{x}^*}^T\boldsymbol{\omega}\left(y - \boldsymbol{X}\hat{\beta}\right). \quad (15)$$

Here, $\boldsymbol{e}_1$ represents a column vector where the first element is 1 and the remaining elements are 0. For $q = 0$, estimator θ_0, which is known as the NWE, is fundamentally different from using a higher-order estimator and then effectively discarding all estimated $\boldsymbol{\theta}_q$ values except for θ_0. Unlike the previous situation, the second method calculates the pixel value estimates assuming a local polynomial structure of degree q.

$\boldsymbol{X^*}_{\boldsymbol{x}^*}^T\boldsymbol{\omega}\boldsymbol{X}_{\boldsymbol{x}^*}^*$ is a $(q+1) \times (q+1)$ block matrix with the $\boldsymbol{X^*}_{\boldsymbol{x}^*}^T\boldsymbol{\omega}\boldsymbol{X}_{\boldsymbol{x}^*}^* = \begin{bmatrix} S_{11} & S_{12} & S_{13} & \cdots \\ S_{21} & S_{22} & S_{23} & \cdots \\ S_{31} & S_{32} & S_{33} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{bmatrix}$ structure in equation (15). Here the block elements are $S_{11} = \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)$, $S_{12} = S_{21}^T = \sum_{i=1}^{n}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)^T K_{\boldsymbol{H}}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)$, $S_{22} = \sum_{i=1}^{n}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)(\boldsymbol{x}_i^* - \boldsymbol{x}^*)^T K_{\boldsymbol{H}}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)$, $S_{13} = S_{31}^T = \sum_{i=1}^{n} vech^T\left\{\left((\boldsymbol{x}_1^* - \boldsymbol{x}^*)(\boldsymbol{x}_1^* - \boldsymbol{x}^*)^T\right)\right\}K_{\boldsymbol{H}}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)$, $S_{23} = S_{32}^T = \sum_{i=1}^{n}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)vech^T\left\{\left((\boldsymbol{x}_1^* - \boldsymbol{x}^*)(\boldsymbol{x}_1^* - \boldsymbol{x}^*)^T\right)\right\}K_{\boldsymbol{H}}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)$, and $S_{33} = \sum_{i=1}^{n} vech\left\{\left((\boldsymbol{x}_1^* - \boldsymbol{x}^*)(\boldsymbol{x}_1^* - \boldsymbol{x}^*)^T\right)\right\}vech^T\left\{\left((\boldsymbol{x}_1^* - \boldsymbol{x}^*)(\boldsymbol{x}_1^* - \boldsymbol{x}^*)^T\right)\right\}K_{\boldsymbol{H}}(\boldsymbol{x}_i^* - \boldsymbol{x}^*)$,… Then the weights become, for $q = 0$ $\boldsymbol{\omega}(x^*; 0) = \frac{K_{\boldsymbol{H}}(x_i^* - x^*)}{S_{11}}$ similar to equation (2), for $= 1$ $\boldsymbol{\omega}(x^*; 1) = \frac{\{1 - S_{12}S_{22}^{-1}(x_i^* - x^*)\}K_{\boldsymbol{H}}(x_i^* - x^*)}{S_{11} - S_{12}S_{22}^{-1}S_{21}}$, and for $q = 2$ $\boldsymbol{\omega}(x^*; 2) = \frac{\left\{1 - S_{12}S_{22}^{-1}(x_i^* - x^*) - S_{13}S_{33}^{-1}vech\left\{\left((x_1^* - x^*)(x_1^* - x^*)^T\right)\right\}\right\}K_{\boldsymbol{H}}(x_i^* - x^*)}{S_{11} - S_{12}S_{22}^{-1}S_{21} - S_{13}S_{33}^{-1}S_{31}}$.

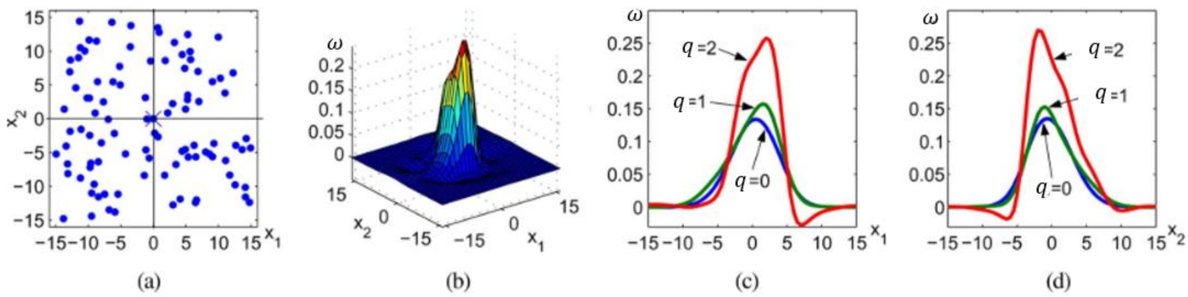


Figure 3. Comparison of kernels for different values of q: (a) irregularly sampled data set, (b) kernel corrector obtained with second-order ($q = 2$) polynomial kernel regression, (c and d) horizontal and vertical sections of correctors with different degrees ($q = 0, 1, 2$) [1].

3.1.1 Data Adapted Partially Linear Regression

In this section, data-adapted partially linear regression methods which take into account not only the spatial sampling density of the data, but also the actual (pixel) values of those samples are introduced. These more sophisticated methods lead to locally adaptive nonlinear extensions of classic kernel regression [1].

Data-adapted partially linear regression methods take into account not only the spatial location and density of the samples but also their radiometric properties. As a result, the size and shape of the regression kernel are locally adjusted to align with image features, such as edges. This approach is structured similarly to (10),

$$\min_{\{\boldsymbol{\theta}_q\}} \sum_{i=1}^{n} [y_i - \boldsymbol{x}_i\beta - \theta_0 - \boldsymbol{\theta}_1^T(\boldsymbol{x}_i^* - \boldsymbol{x}^*) - \boldsymbol{\theta}_2^T vech\{(\boldsymbol{x}_i^* - \boldsymbol{x}^*)(\boldsymbol{x}_i^* - \boldsymbol{x}^*)^T\} \ldots]^2 K_{ad}(\boldsymbol{x}_i^* - \boldsymbol{x}^*, y_i - \boldsymbol{x}_i\beta - y - \boldsymbol{x}\beta) \quad (16)$$

where the optimization problem involves a data adapted kernel function K_{ad} that is adapted to the spatial sample locations $\boldsymbol{x}_i^*$s, the sample density, and the radiometric values $y_i - \boldsymbol{x}_i\beta$ of the data.

[1] suggested bilateral kernel, steering kernel and iterative steering kernel regression for the selection of K_{ad}. While bilateral kernel is a data adapted version of NWE, steering kernel is is more appropriate for very noisy data sets, $(y_i - \boldsymbol{x}_i\beta - y - \boldsymbol{x}\beta)$s tend to be zero, and effectively useless. Because iterative steering kernel regression is founded more effective we used this approach for the selection of K_{ad}.

In image processing, partial linear regression is a powerful tool for modeling nonparametric relationships on image data. This methodology allows meaningful analysis of data obtained from image processing and plays a significant role in scientific discoveries. In this technique, relationships between parametric and nonparametric components are examined on image pixels or feature vectors. Initially, high-resolution image data is obtained, and preprocessing steps are performed to analyze this data. Relationships between dependent variables (usually grayscale values or feature vectors) and independent variables (e.g., a feature vector) in the image data can be modeled with parametric and nonparametric components. The obtained parameters and estimates are used to understand the behavior of a specific feature or variable in image processing. For example, the relationship or impact of a specific feature with other features in an image can be examined.

3.2. Application

In image processing and computer vision systems, using partial linear models aims to reduce or completely eliminate the effects of distortions (such as blurring). This process is an optimization effort to obtain the original image from a distorted one [7]. The process is handled as follows in the study:

1. First, the distorted image and the distortion kernel used are defined. This kernel expresses how the distortion occurs.

2. Then, a convolution filter or algorithm is designed for this kernel.

3. Finally, this filter is applied to the distorted image to obtain a clearer or sharper version of the original image.

In image processing, the predicted improved pixel value for each pixel in the blurred image becomes the enhanced pixel value calculated for the parametric component. For the parametric equation $Y = \beta_0 + \beta_1 X_1$, Y represents the pixel value of the predicted image, β_0 represents the expected pixel value of the improved image when the pixel value in the noisy image is 0, and β_1 represents the change in the pixel value of the improved image when the pixel value in the noisy image increases by 1 unit.

For the nonparametric equation $Y = g(\boldsymbol{x}^*)$, Y represents the pixel value of the predicted image, and the function $g(\boldsymbol{x}^*)$ represents the kernel used to model the nonparametric relationship between the pixel values of the blurred image and the pixel values of the improved image.

3.2.1 Partially Linear Regression Algorithm for Blurred Images and Application Images Obtained

Image Creation:

- Input: Path of an image file.
- Output: A blurred black-and-white image

The image file is read, resized with a certain scaling ratio, and converted to grayscale.

Applying Gaussian Filter:

- Input: Blurred image.
- Output: Image with Gaussian filter applied.

A Gaussian kernel is created, and the Gaussian filter is applied to the image.

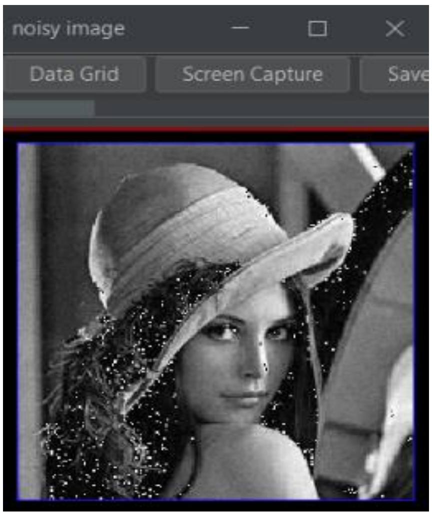


Figure 4. Noisy input image

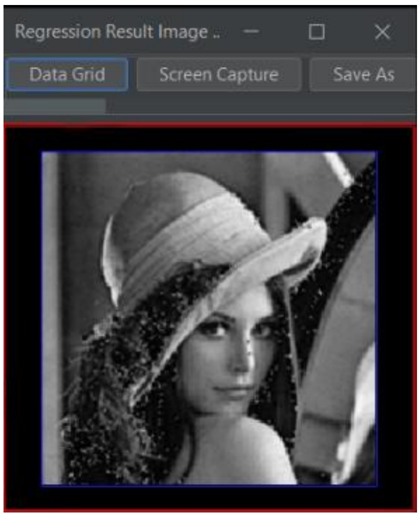


Figure 5. Image with Gaussian filter applied

Applying Histogram Equalization:

- Input: Image with Gaussian filter applied.
- Output: Image with histogram equalization applied

The histogram of the image is calculated, the cumulative distribution of the histogram is examined, and a new pixel value is created by matching each pixel value with the cumulative histogram. An improved image is obtained with new pixel values.

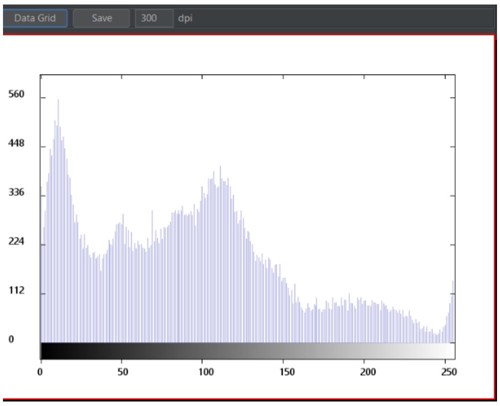


Figure 6. Histogram of the noisy image

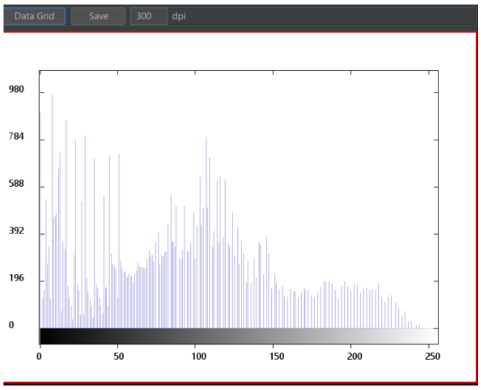


Figure 7. Histogram of the processed image

Applying Partially Linear Regression:

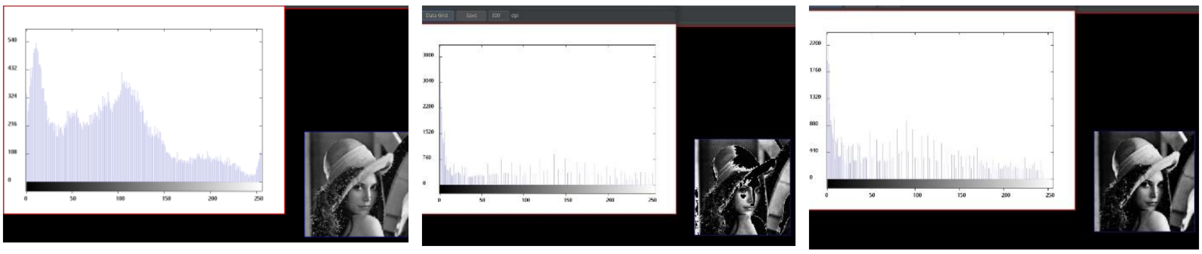
- Input: Blurred and improved images.
- Output: Regression result.

The pixel values of blurred and improved images are collected in pairs. Using data adaptive partially linear regression with the integrated weighting parameter q demonstrates the robustness and adaptability of the proposed approach; the relationship between these pixel pairs is found (regression coefficients are calculated). A new image is created using the calculated regression coefficients.

Displaying Results:

- Input: Images resulting from different processing steps.
- Output: Visual display of the images.

Original, noisy, blurred, improved, and regression result images, which show the effect of the q parameter for 0, 1 and 2, respectively, on image enhancement and deconvolution, are displayed.



(a) $q = 0$ (b) $q = 1$ (c) $q = 2$

Figure 8. Corrected regression images for $q = 0, 1$ and 2.

Here, Figure 8(a) demonstrates the original distorted image showing significant blur due to the distortion kernel. This is used as the basis for comparison. Figure 8(b) demonstrates an enhanced image generated using standard kernel regression without data adaptation. Although edges are slightly sharper, noise and artifacts persist, especially in areas with complex textures, and Figure 8(c) demonstrates enhanced image generated using partial linear regression model with data adaptation. With the addition of q-weighting parameter, the algorithm exhibited superior performance in edge preservation and noise reduction. Locally adaptive kernel provides a clearer and more detailed reconstruction by better adapting to image features.

The results are consistent with the theoretical framework discussed in Section 3.1.1. In this section, a data adaptive kernel function was proposed. By adding q-weight, the kernel dynamically adapts by considering both spatial sampling density and pixel features. This leads to the following results:

Improved Edge Preservation: The kernel reduces over-smoothing by adapting its shape and size according to local image features such as edges and texture.

Improved Noise Robustness: The q-parameter affects the regression by weighting regions with high variance or noise, thus preserving critical details in noisy regions while preserving smoother areas.

Iterative Refinement: The iterative guidance kernel regression approach gradually improves the enhanced image by recalculating the kernel, including the locally optimized parameters q.

4. Conclusion

It was observed that as the noise level increased, the image quality decreased. Statistical analyses of the images obtained during this process were performed. Gaussian blurring was applied to the frequency of the noisy image to reduce noise. This application aimed to eliminate noise by softening sharp edges in the image, demonstrating the effectiveness of Gaussian blurring in noise reduction.

In this study, where partially linear regression analysis was applied, the relationship modeling between blurred and improved images was successfully implemented. The operation became an effective modeling approach for processing blurred images. In noise reduction and image smoothing operations image processing techniques are very important.

The partially linear regression operations is demonstrated that each operation, such as reducing or eliminating blurring and increasing contrast, significantly improved image quality. The results obtained showed that the operation would facilitate everyday applications (e.g., medical imaging devices, security cameras, and satellite images). Testing future studies with images obtained under different conditions and larger data sets will allow for a more comprehensive evaluation of the proposed methods' general validity.

Ethical statement

The authors declare that this document does not require ethics committee approval or any special permission. Our study does not cause any harm to the environment and does not involve the use of animal or human subjects.

Acknowledgment

This study was produced from Merve Bingöl Sukutlis' master's thesis named ''The use of partially linear regression model with measurement error in image processing'' in the department of statistics at the Dicle University Institute of Science and supported by Dicle University Scientific Research Projects Unit under Project No. DÜBAP-FEN.23.023. The authors thank Musa Ataş from Siirt University for valuable contributions to conduct the research.

Conflict of interest

The authors declare that this document has not conflict of interest.

Authors' Contributions

All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by M. B. (%60). The first draft of the manuscript was written by S. Y. (%40) and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Generative AI statement

The author(s) declare that no Gen AI was used in the creation of this manuscript.

References

[1] Takeda, H., Farsiu, S., Milanfar, P., "Kernel Regression for Image Processing and Reconstruction", *IEEE Transactions on Image Processing*, 16(2), 349-366, 2007.

[2] Montgomery, D.C., Peck, E.A., Vining, G.G., *Introduction to Linear Regression Analysis*. 5th ed., John Wiley & Sons, New York, 2012.

[3] Ni, K., Nguyen, T., "Image Superresolution Using Support Vector Regression", *IEEE Transactions on Image Processing*, 16, 1596-1610, 2007.

[4] Tomasi, C., Manduchi, R., "Bilateral filtering for gray and color images", Proceeding of IEEE Int. Conf. Computer Vision, New Delhi, India, 1998, pp. 836-846.

[5] Toprak, S., Partially Linear Regression Models with Measurement Errors, PhD thesis, Dicle University, Diyarbakır, Turkey, 2015.

[6] Yalaz, S., Tez, M., "Partially linear multivariate regression in the presence of measurement error", *Communications for Statistical Applications and Methods*, 27, 511-521, 2020.

[7] Ataş, M., "Open Cezeri Library: A novel java based matrix and computer vision framework", *Computer Applications in Engineering Education*, 24(5), 736-743, 2016.