Journal of Machine Learning Research 23 (2024) 1-29 Submitted 12/24; Revised 12/24; Published 12/24

Coordinated Minima Search: An Efficient Approach for Optimizing Linear Regression Models

Rifat Hassan H.RIFAT1609@GMAIL.COM
Department of Electrical and Electronic Engineering
Bangladesh University of Engineering and Technology
Dhaka-1205, Bangladesh

Editor: Kevin Jamieson

Abstract

In this paper, we introduce a novel regression optimization technique called coordinated minima search (CMS), designed to iteratively minimize the squared error loss function of a linear regression model by coordinating over each weight and bias variable. Unlike traditional gradient-based approaches, CMS simplifies the complex optimization of linear regression models by treating each weight variable's loss function as a parabolic equation. It then iterates to discover the global minima for each. Thus this method enhances the ability to model linear relationships within complex datasets without the hassle of choosing any hyperparameter or involving any complex mathematical computation. It efficiently identifies the best-fitting model by considering the squared error loss function as a parabolic function of each weight variable while minimizing the squared error loss. The results demonstrate that CMS provides a flexible and precise alternative to traditional regression techniques, in datasets with linear dependencies.

Keywords: coordinated minima search, linear regression, optimization, squared error loss, parabolic function

1 Introduction

Regression analysis is one of the foundational tools for modeling relationships between dependent and independent variables in statistical and machine-learning applications. Traditional approaches to regression, such as ordinary least squares (OLS), coordinate descent methods, and gradient-based methods like stochastic gradient descent (SGD), focus on minimizing a loss function to optimize the model's weights. These techniques are highly effective for building linear models but often involve complex computations or get stuck in suboptimal solutions due to inappropriate choice of hyperparameters.

In this paper, we propose a novel approach, coordinated minima search (CMS), which is designed to overcome the limitations of traditional regression by allowing linear feature transformations while iterating over each weight variable's loss function individually. The CMS approach provides a flexible and scalable solution for both small and moderately sized datasets. It yields a superior model fit for complex data.

2 Related Work

Regression analysis is one of the most fundamental techniques used for modeling relationships between dependent and independent variables. Hastie et al. (2009) has provided an in-depth look at linear and non-linear regression, in their book *The Elements of Statistical Learning.* Various optimization techniques for machine learning applications have been discussed here. In this section, we will review the related work on optimization techniques.

2.1 Ordinary Least Squares (OLS) Methods

Ordinary least squares (OLS) is a foundational method in regression analysis which is extensively discussed in both classical and contemporary literature. It is a commonly used method to find the best-fitting line for a linear regression model by minimizing the sum of squared differences between the predicted and actual values. The solution is calculated using the formula $\beta = (X^T X)^{-1} X^T y$, where $X^T X$ is the matrix of the independent variables, y is the dependent variable, and $\boldsymbol{\beta}$ are the coefficients. *The Elements of Statistical Learning* by Hastie et al. (2009) provides a comprehensive exploration of OLS within the broader context of machine learning. *Applied Linear Statistical Models* by Kutner et al. (2005) offers a detailed treatment of linear regression which includes diagnostics and practical applications. For introductory-level readers, *An Introduction to Statistical Learning* by James et al. (2021) simplifies the theoretical aspects of OLS with illustrative examples. Chatterjee and Hadi (2013) in the book *Regression Analysis by Example* focused on real-world applications. It bridged theoretical concepts with practical problem-solving. Puntanen and Styan (1989) presented the development of several conditions for the ordinary least squares estimator to be the best linear unbiased estimator in a historical perspective, including various characterizations of the conditions using generalized inverses and orthogonal projectors. de Souza and Junqueira (2005) proposed the ordinary least squares (OLS) method for testing the linearity of calibration curves. OLS works efficiently for smaller datasets and provides unbiased, consistent results when the data meets assumptions like no multicollinearity and constant variance in errors. However, it struggles with large datasets because computing the inverse of $X^T X$ can be slow, especially when there are many features. It is also sensitive to outliers and multicollinearity (Efeizomor, 2023). The coordinated minima search (CMS) method also looks for the least squared error possible and as a result, they result in the exact same solution. However, it takes a different approach by optimizing each weight one at a time. It treats the loss function as a parabolic function for each variable, and therefore, unlike OLS, CMS does not require matrix inversion. This makes it better suited for large or complex datasets. CMS also extends naturally to non-linear polynomial models and handles noisy or incomplete data more robustly. In short, while OLS and CMS result in exactly the same outputs and OLS is faster for simple linear problems, CMS offers greater flexibility and stability for high-dimensional tasks.

2.2 Gradient-Based Optimization Techniques

Gradient descent (GD) is the most widely used optimization technique in machine learning for minimizing loss functions and training models. It is particularly used in linear regression and neural networks. The fundamental idea behind GD is to update model parameters (weights and biases) iteratively in the direction of the negative gradient of the loss function. It thus minimizes the error between predicted and actual values. The formula for parameter updates in GD is:

$$\theta = \theta - \eta \nabla_\theta J(\theta)$$

where θ represents the model parameters (weights and biases), η is the learning rate, and $\nabla_\theta J(\theta)$ is the gradient of the loss function with respect to the parameters θ (Bottou, 2010). Stochastic gradient descent (SGD) and mini-batch gradient descent are two widely used variants of GD. SGD updates the parameters using a single training example at a time. It introduces noise but accelerates the training process and helps escape local minima (Ruder, 2017). Mini-batch gradient descent strikes a balance by using small random subsets of the dataset to compute the gradients. It improves both speed and stability (Kingma and Ba, 2017).

Despite its advantages, GD has limitations. It includes the risk of getting stuck in local minima or saddle points in non-convex loss landscapes, especially in high-dimensional data (Goodfellow et al., 2016). Moreover, careful hyperparameter tuning, such as selecting the learning rate, is required to ensure effective convergence (Ruder, 2017). Nonetheless, GD remains a foundational method in machine learning, widely applied in regression, classification, and deep learning models (Bottou, 2010; Kingma and Ba, 2017).

Compared to coordinated minima search (CMS), GD is more flexible in scalability for non-linear models but may suffer from instability or slow convergence without proper tuning. On the other hand, CMS provides precise and deterministic results for linear regression problems by optimizing each parameter sequentially without the risk of local minima. However, CMS can sometimes be more computationally expensive when the dataset is too large, making it less scalable than GD.

2.3 Coordinate Descent Methods

Coordinate descent (CD) is a well-established optimization method that works by optimizing each parameter (or coordinate) individually, one at a time while keeping the others fixed. This approach is often favored for its simplicity and efficiency. This is particularly true in high-dimensional spaces or in problems with sparse data (Wright, 2015). In such settings, CD methods can be computationally more tractable than gradient-based methods because they require fewer calculations per iteration. CD has been applied successfully in many machine learning and statistical problems, such as linear regression, Lasso regression, and logistic regression, where the number of features (coordinates) can be very large (Wu and Lange, 2008).

Despite its efficiency in many applications, Coordinate descent has some well-known drawbacks. In non-convex optimization problems, where the loss function is not guaranteed to have a single global minimum, CD methods can sometimes get stuck in local minima or suboptimal solutions. This is especially true when the relationship between parameters is highly non-linear (Boyd and Vandenberghe, 2004). In these situations, CD might converge more slowly or fail to find the global optimum (Wu and Lange, 2008). Various modifications to CD have been proposed to overcome these challenges, such as randomized or parallel coordinate descent. They aim to balance the simplicity of coordinate updates with the need for faster convergence and more robust optimization performance (Wright, 2015).

Although CD methods are often faster and simpler to implement than Gradient Descent (GD), they can be slower in dense problems, where the optimization landscape is smooth and the gradient of the loss function provides useful information for updating multiple parameters simultaneously (Wright, 2015). Moreover, for non-convex functions, CD's sequential update of one parameter at a time can fail to reach the global minimum.

In this context, coordinated minima search (CMS) offers an alternative approach that addresses some of the shortcomings of traditional coordinate descent. Unlike CD, CMS optimizes the loss function by treating each parameter's loss as a parabolic curve. This allows for a more systematic and precise update without the risk of falling into local minima. CMS does not rely on sequential updates, which are characteristic of CD, and instead uses an iterative process to find the optimal values of weights and biases that minimize the error function. This makes CMS particularly attractive for tasks where linear regression is involved, as it can provide more accurate solutions without the computational complexities associated with matrix inversion or the instability of gradient-based methods. However, like CD, CMS can become computationally expensive as the number of features increases or when a very big dataset is involved.

2.4 Neural Networks

Neural networks (NNs), particularly deep neural networks (DNNs), have gained significant attention in recent years for their ability to model complex, non-linear relationships between input features and target variables. NNs are composed of layers of interconnected neurons. Here each layer performs weighted transformations on the input data and passes it through an activation function. The backpropagation algorithm is used to calculate gradients for each weight, which allows the network to update its parameters by applying gradient descent methods (Rumelhart et al., 1986).

Deep learning models use multiple hidden layers. They have achieved state-of-the-art performance in a variety of domains, such as image classification, speech recognition, and natural language processing (LeCun et al., 2015). The optimization of these models is typically performed using gradient descent-based methods, including Adam and RMSProp. They adaptively adjust the learning rate for each parameter to stabilize convergence and improve training speed (Kingma and Ba, 2017). While NNs provide great flexibility and accuracy in capturing complex relationships, training deep networks is computationally expensive, requiring large datasets and powerful hardware. Additionally, deep networks are vulnerable to issues such as overfitting and the vanishing/exploding gradient problem. These happen especially when training takes place on smaller datasets or without proper regularization techniques (Pascanu et al., 2013).

Despite these challenges, NNs remain highly effective for tasks that involve large-scale data and non-linear relationships. However, they require careful architecture design, hyperparameter tuning, and regularization to avoid overfitting and computational inefficiencies (Goodfellow et al., 2016). In comparison to gradient descent, which is applicable to both linear and non-linear problems, NNs excel in modeling highly complex patterns, but they require more computational resources and are more sensitive to the choice of architecture and training settings (Bengio, 2009; LeCun et al., 2015).

In comparison, CMS provides an advantage in linear regression by treating the loss function as an individual equation for each parameter. This allows for precise, systematic optimization. While neural networks excel in handling complex, non-linear problems where CMS may struggle to adapt efficiently due to its reliance on feature transformations and its reduced scalability for non-linear data, CMS excels in scenarios where the relationship between features and target variables is linear.

2.5 Non-Linear Regression and Specialized Optimization Techniques

In cases where the relationship between variables cannot be expressed as a simple linear function, non-linear regression techniques are used. Methods like Levenberg-Marquardt and Trust-Region Methods are often used in these cases. These methods are adaptations of gradient descent tailored for non-linear problems. They offer a balance between the speed of gradient descent and the accuracy of second-order methods. Nocedal and Wright (2006) in the book *Numerical Optimization* covered various optimization techniques, including methods for non-linear problems. Theoretical backgrounds for methods like Levenberg-Marquardt have been provided there.

2.6 Coordinated Minima Search and Parabolic Minimization

The coordinated minima search (CMS) method builds upon the idea of parabolic minimization. By focusing on one parameter at a time and breaking the squared error loss function into a parabolic form, CMS seeks to find the precise values of the weight variables that result in minimum error using the formula $x = \frac{-\beta}{2\alpha}$, where α,β are derived from the quadratic form of the error function. CMS is particularly efficient in solving problems where traditional gradient methods struggle with convergence or accuracy. Compared to gradient-based methods, CMS is independent of limitations like slow convergence and local minima in non-convex problems. Additionally, CMS offers a clear advantage over traditional coordinate descent methods in terms of precision. By calculating the minimum value for each weight in the form of a quadratic equation, CMS reduces the likelihood of getting trapped in suboptimal solutions.

CMS also extends beyond linear models as it allows for polynomial, trigonometric, logarithmic, and even exponential transformations of the feature variables. Thus it accommodates a wide range of non-linear regression models. The ability to handle such diverse transformations makes CMS a flexible and powerful tool for regression problems that can be applied to various domains like finance, healthcare, engineering, etc.

2.7 Gaps in the Literature and Contribution of CMS

Despite the extensive research in optimization and regression methods, there is a gap in methods that can effectively handle regression models while avoiding common limitations like slow convergence or local minima, along with complex computations like matrix inversion. The coordinated minima search method addresses these gaps by:

1. Providing a precise parabolic minimization for each weight variable and ensuring accurate convergence.

2. Allowing for flexible transformations of the feature variables enabling linear regression models to be built.

3. Avoiding computational inefficiencies of gradient-based methods in high-dimensional spaces and computational complexities of matrix inversion of ordinary least squares methods.

3 Methodology

The CMS method is designed to minimize the squared error loss function through coordinated minimization of the weight and bias terms. For a given regression model, the loss function is treated as a parabolic equation for each individual weight variable while holding the others constant. By iterating over each variable and applying backward iteration to update the weight variables, CMS efficiently converges to the optimal values that minimize the loss. In this section, we will first discuss the full overview of the coordinated minima search algorithm. Then we will focus on how this algorithm can be involved in building non-linear polynomial regression models. We will finish by providing a complete guide to each step of the algorithm.

3.1 Details of the Coordinate Minima Search Algorithm

In a data set with n independent variables $x_1, x_2, x_3, \dots, x_n$, and one dependent variable y, the linear regression model that establishes the best-fit relationship between the dependent and independent variables looks like this.

$$y = w_1x_1 + w_2x_2 + w_3x_3 + \cdots + w_nx_n + b \tag{3.1.1}$$

Here, $w_1, w_2, w_3, \dots, w_n$ are the weight variables and b is the bias variable.
The squared error loss function will be,

$$J = \sum(w_1x_1 + w_2x_2 + w_3x_3 + \cdots + w_nx_n + b - y)^2 \tag{3.1.2}$$

An expression involving the square of the algebraic sum of multiple terms can be expanded as follows,

$$\begin{aligned}(x_1 + x_2 + x_3 + \cdots + x_l + x_m + x_n)^2 = {} & (x_1^2 + x_2^2 + x_3^2 + \cdots + x_l^2 + x_m^2 + x_n^2) \\ & + 2x_1(x_2 + x_2 + \cdots + x_l + x_m + x_n) \\ & + 2x_2(x_3 + \cdots + x_l + x_m + x_n) \\ & + \dots \\ & + 2x_l(x_m + x_n) \\ & + 2x_mx_n\end{aligned}$$

That is,

$$\left(\sum_{i=1}^{n} x_i\right)^2 = \sum_{i=1}^{n} x_i^2 + 2\sum_{i<j} x_ix_j \tag{3.1.3}$$

By using (3.1.3), (3.1.2) can be expanded into,

$$\begin{aligned}
J &= \sum (w_1x_1 + w_2x_2 + w_3x_3 + \cdots + w_nx_n + b - y)^2 \\[1em]
&= \sum \big\{ (w_1^2x_1^2 + w_2^2x_2^2 + \cdots + w_n^2x_n^2 + b^2 + y^2) \\
&\quad + 2w_1x_1(w_2x_2 + \cdots + w_nx_n + b - y) \\
&\quad + 2w_2x_2(w_3x_3 + \cdots + w_nx_n + b - y) \\
&\quad + \dots \\
&\quad + 2w_nx_n(b - y) \\
&\quad + 2b(-y) \big\} \\[1em]
&= (w_1^2 \sum x_1^2 + w_2^2 \sum x_2^2 + \cdots + w_n^2 \sum x_n^2 + nb^2 + \sum y^2) \\
&\quad + (2w_1w_2 \sum x_1x_2 + \cdots + 2w_1w_n \sum x_1x_n + 2w_1b \sum x_1 - 2w_1 \sum x_1y) \\
&\quad + (2w_2w_3 \sum x_2x_3 + \cdots + 2w_2w_n \sum x_2x_n + 2w_2b \sum x_2 - 2w_2 \sum x_2y) \\
&\quad + \dots \\
&\quad + (2w_nb \sum x_n - 2w_n \sum x_ny) \\
&\quad - 2b \sum y \\[1em]
&= (\sum x_1^2)w_1^2 \\
&\quad + (2w_2 \sum x_1x_2 + \cdots + 2w_n \sum x_1x_n + 2b \sum x_1 - 2 \sum x_1y)w_1 \\
&\quad + w_2^2 \sum x_2^2 + \cdots + w_n^2 \sum x_n^2 + nb^2 + \sum y^2 \\
&\quad + 2w_2w_3 \sum x_2x_3 + \cdots + 2w_2w_n \sum x_2x_n + 2w_2b \sum x_2 - 2w_2 \sum x_2y \\
&\quad + \cdots + 2w_nb \sum x_n - 2w_n \sum x_ny - 2b \sum y
\end{aligned} \tag{3.1.4}$$

This whole loss function J can be considered a function of only w_1.

$$J(w_1) = \alpha_1 w_1^2 + \beta_1 w_1 + \gamma_1 \tag{3.1.5}$$

where,

$$\alpha_1 = \sum x_1^2 \tag{3.1.6}$$

$$\beta_1 = 2w_2 \sum x_1x_2 + \cdots + 2w_n \sum x_1x_n + 2b \sum x_1 - 2 \sum x_1y \tag{3.1.7}$$

$$\begin{aligned}\gamma_1 = w_2^2 \sum x_2^2 + \cdots + w_n^2 \sum x_n^2 + nb^2 + \sum y^2 \\ + 2w_2w_3 \sum x_2x_3 + 2w_2w_n \sum x_2x_n + \cdots \\ + 2w_2b \sum x_2 - 2w_2 \sum x_2y + \cdots \\ + 2w_nb \sum x_n - 2w_n \sum x_ny - 2b \sum y\end{aligned} \tag{3.1.8}$$

Equation (3.1.5) is analogous to the parabolic equation $y = \alpha x^2 + \beta x + \gamma$. As explained in standard calculus textbooks (Stewart, 2007), it has a maximum or minimum value when its first derivative is set to zero. Moreover, it has a maximum value if the second derivative is less than zero and a minimum value if the second derivative is greater than zero.

Here, $\frac{dy}{dx} = \frac{d}{dx}\left(\alpha x^2 + \beta x + c\right) = 2\alpha x + \beta$.

When $\frac{dy}{dx} = 0$, $x = -\frac{\beta}{2\alpha}$

Therefore, it has a maximum or minimum value at $x = -\frac{\beta}{2\alpha}$.

Since its second derivative $\frac{d^2y}{dx^2} = \frac{d}{dx}(2\alpha x + \beta) = 2\alpha = 2\sum x_1^2 > 0$, it has only a minimum value.

Putting $x = -\frac{\beta}{2\alpha}$ at $y = \alpha x^2 + \beta x + \gamma$,

$$\begin{aligned} y_{\min} &= \alpha\left(-\frac{\beta}{2\alpha}\right)^2 + \beta\left(-\frac{\beta}{2\alpha}\right) + \gamma \\ &= \gamma - \frac{\beta^2}{4\alpha}\end{aligned} \tag{3.1.9}$$

Equation (3.1.9) states that the loss function $J(w_1)$ mentioned in (3.1.5) has its minimum value at the point $w_1 = -\frac{\beta_1}{2\alpha_1}$, which can be computed by the following formula,

$$\min J(w_1) = \gamma_1 - \frac{{\beta_1}^2}{4\alpha_1}$$

Putting the values of α_1, β_1, and γ_1 from (3.1.6), (3.1.7), and (3.1.8) respectively, we get,

$$\begin{aligned}\min J(w_1) &= (w_2^2 \sum x_2^2 + \cdots + w_n^2 \sum x_n^2 + nb^2 + \sum y^2 \\ &\quad + 2w_2w_3 \sum x_2x_3 + \cdots + 2w_2w_n \sum x_2x_n \\ &\quad + 2w_2b \sum x_2 - 2w_2 \sum x_2y + \ldots \\ &\quad + 2w_nb \sum x_n - 2w_n \sum x_ny - 2b \sum y) \\ &\quad - \frac{(2w_2 \sum x_1x_2 + \cdots + 2w_n \sum x_1x_n + 2b \sum x_1 - 2\sum x_1y)^2}{4\sum x_1^2} \\ &= \alpha_2w_2^2 + \beta_2w_2 + \gamma_2 \\ &= J(w_2) \\ &= f(w_2, w_3, ..., w_n, b)\end{aligned} \tag{3.1.10}$$

$J(w_2)$ has again a maximum or minimum value at $w_2 = -\frac{\beta_2}{2\alpha_2}$. It will have a minimum value if $\frac{d^2J(w_2)}{dw_2^2} > 0$, or a maximum value if $\frac{d^2J(w_2)}{dw_2^2} < 0$. Here,

$$\begin{aligned}\frac{d^2J(w_2)}{dw_2^2} &= 2\alpha_2 \\ &= 2\frac{(\sum x_1^2 \sum x_2^2) - (\sum x_1 x_2)^2}{\sum x_1^2}\end{aligned}$$

According to the Cauchy-Schwarz inequality (Boyd and Vandenberghe, 2004), the expression $(\sum x_1^2 \sum x_2^2) - (\sum x_1 x_2)^2$ is always non-negative. It states that for any vectors x and y in an inner product space,

$$(\sum x_1 x_2)^2 \leq (\sum x^2 \sum y^2)$$

With equality if and only if x and y are linearly dependent.

Since two independent variables x_1 and x_2 can not be linearly dependent, therefore, $(\sum x_1^2 \sum x_2^2) - (\sum x_1 x_2)^2 > 0$.

Also $\sum x_1^2 > 0$.

Therefore, $\frac{d^2J(w_2)}{dw_2^2} = 2\frac{(\sum x_1^2 \sum x_2^2) - (\sum x_1 x_2)^2}{\sum x_1^2} > 0$.

So $J(w_2)$ has a minimum value at $w_2 = -\frac{\beta_2}{2\alpha_2}$. From (3.1.9), the minimum value is,

$$\min J(w_2) = \gamma_2 - \frac{{\beta_2}^2}{4\alpha_2}$$

Putting the values of α_2, β_2, and γ_2, we will again get,

$$\begin{aligned}\min J(w_2) &= \alpha_3 w_3^2 + \beta_3 w_3 + \gamma_3 \\ &= J(w_3) \\ &= f(w_3, ..., w_n, b)\end{aligned}$$

The sequence goes on like this.

This is the key aspect of the algorithm. Starting from the weight variable w_1, the loss function J associated with each weight variable w_i has a single minimum. While this has been demonstrated for the loss functions of w_1 and w_2, the same holds true for all weight variables. This is because the loss functions in this case are square functions, and as per standard calculus principles, a square function has a single global minimum.

The minimum value lies at the point $w_i = -\frac{\beta_i}{2\alpha_i}$ that follows the parabolic equation $y = \alpha x^2 + \beta x + \gamma$ of the next weight variable w_{i+1} and is a function of all the following weight variables, w_{i+1} to w_n and the bias variable b [Similar to (3.1.10) that shows that the $\min J(w_1) = J(w_2) = f(w_2, w_3, ..., w_n, b)$]

$$\begin{aligned}\min J(w_i) &= J(w_{i+1}) \\ &= \alpha_{i+1} w_{i+1}^2 + \beta_{i+1} w_{i+1} + \gamma_{i+1} \\ &= f(w_{i+1}, ..., w_n, b)\end{aligned} \tag{3.1.11}$$

In the case of w_2, $\min J(w_2)$ is a parabolic loss function of w_3 and is a function of w_3 to w_n and b.

$$\begin{aligned}\min J(w_2) &= J(w_3)\\ &= \alpha_3 w_3^2 + \beta_3 w_3 + \gamma_3\\ &= f(w_3, ..., w_n, b)\end{aligned}$$

For w_3,

$$\begin{aligned}\min J(w_3) &= J(w_4)\\ &= \alpha_4 w_4^2 + \beta_4 w_4 + \gamma_4\\ &= f(w_4, ..., w_n, b)\end{aligned}$$

Thus, for the last weight variable w_n, $\min J(w_n)$ will be a loss function of only the bias variable b.

$$\begin{aligned}\min J(w_n) &= J(b)\\ &= \alpha_b b^2 + \beta_b b + \gamma_b\\ &= f(b)\end{aligned} \tag{3.1.12}$$

Using $\min J(b) = \gamma_b - \frac{{\beta_b}^2}{4\alpha_b}$ on (3.1.12), we can find out the numerical minimum value of $J(b)$. This is the required minimum value of the loss function J (mentioned in (3.1.2)) we are searching for.

$$\begin{aligned}\min J(b) &= \gamma_b - \frac{{\beta_b}^2}{4\alpha_b}\\ &= J_{min}\end{aligned}$$

At the same time, using $b = -\frac{\beta_b}{2\alpha_b}$ on (3.1.12), we can find out the value of the bias variable b at which this minimum value occurs. This is the required value of the bias variable b of the regression model of (3.1.1).

$$b = -\frac{\beta_b}{2\alpha_b}$$

According to (3.1.11), the loss function $J(w_i)$ of each weight variable w_i is a function of all the following weight variables, w_{i+1} to w_n and the bias variable b. Therefore, the loss function $J(w_n)$ of the last weight variable w_n is a function of only w_n and b.

$$\begin{aligned}J(w_n) &= \alpha_n w_n^2 + \beta_n w_n + \gamma_n\\ &= f(w_n, b)\end{aligned} \tag{3.1.13}$$

Putting the value of b on (3.1.13), it can be turned into a function of only w_n.

$$\begin{aligned}J(w_n) &= \alpha_n w_n^2 + \beta_n w_n + \gamma_n\\ &= f(w_n)\end{aligned} \tag{3.1.14}$$

Here, α_n, β_n, and γ_n are numerical values.

Therefore, using $w_n = -\frac{\beta_n}{2\alpha_n}$ on (3.1.14), we can find out the value of the last weight variable w_n at which the minimum value of the loss function occurs. This is the required value of the last weight variable w_n of the regression model of (3.1.1).

Thus, we can compute all the required values of the weight variables w's from w_n to w_1 through a backward iteration. From w_n to w_1, the value of each weight variable w_i can be calculated using the values of w_{i+1} to w_n and the bias variable b. As the loss function $J(w_i)$ is a function of the $w's$ from w_i to w_n and b (according to (3.1.11)), by putting the numerical values of w_{i+1} to w_n and b, it can be left as a function of only w_i of the form $J(w_i) = \alpha_i w_i^2 + \beta_i w_i + \gamma_i$, where α_i, β_i, and γ_i are numerical values.

$$\begin{aligned} J(w_i) &= \alpha_i w_i^2 + \beta_i w_i + \gamma_i \\ &= f(w_i, w_{i+1}, ..., w_n, b) \end{aligned} \tag{3.1.15}$$

Putting the values of w_{i+1} to w_n and b on (3.1.15), it can be turned into a function of only w_i.

$$\begin{aligned} J(w_i) &= \alpha_i w_i^2 + \beta_i w_i + \gamma_i \\ &= f(w_i) \end{aligned} \tag{3.1.16}$$

Here, α_i, β_i, and γ_i are numerical values.

Now putting $w_i = -\frac{\beta_i}{2\alpha_i}$ on (3.1.16), the required value of wi can be precisely calculated.

This process will give us the precise values of all the weight variables w's and the bias variable b that are required to make the value of the loss function mentioned in equation 3.1.2 minimum.

3.2 Non-Linear Variable Transformations

CMS can be used not only for predicting the precise values of w's and b in a linear regression model but also to build models where the variables follow non-linear architectures but the parameters follow linear ones, by effectively minimizing the value of the squared error loss function.

For linear regression models, the squared error loss function mentioned (3.1.4) was:

$$\begin{aligned} J = \sum \{ & (w_1^2 x_1^2 + w_2^2 x_2^2 + \cdots + w_n^2 x_n^2 + b^2 + y^2) \\ & + 2w_1x_1(w_2x_2 + \cdots + w_nx_n + b - y) \\ & + 2w_2x_2(w_3x_3 + \cdots + w_nx_n + b - y) \\ & + \ldots \\ & + 2w_nx_n(b - y) \\ & + 2b(-y) \} \end{aligned}$$

Here, in place of directly putting the independent variables x_1, x_2, ..., x_n in the model, any of their functions like $f_1(x_1)$, $f_2(x_2)$, or so on, can be put. In that case, the loss function equation will be like this:

$$
\begin{aligned}
J = \sum \Big\{ & \left(w_1^2(f_1(x_1))^2 + w_2^2(f_2(x_2))^2 + \cdots + w_n^2(f_n(x_n))^2 + b^2 + y^2\right) \\
& + 2w_1(f_1(x_1))\,(w_2(f_2(x_2)) + \cdots + w_n(f_n(x_n)) + b - y) \\
& + 2w_2(f_2(x_2))\,(w_3(f_3(x_3)) + \cdots + w_n(f_n(x_n)) + \cdots + b - y) \\
& + \dots \\
& + 2w_n(f_n(x_n))(b - y) \\
& + 2b(-y)\Big\}
\end{aligned} \tag{3.2.1}
$$

All other processes remain the same. We will get the precise values of the w's and b that will result in the minimum value of the loss function for any combination of given functions of the independent variables.

In a data set with n independent variables from x_1 to x_n and one dependent variable y, if we consider only polynomial equations up to power r, a total of r^n combinations is possible. The number is such because each independent variable here can take a r number of possible functions, from x to x^r. One of these 2^n models will construct the best-fit model.

A program can be written in any familiar language (most probably in Python or R) that will follow the procedure mentioned in Subsection 3.1 and find out the minimum value of the loss function of (3.2.1) for each of the r^n combinations. By examining the minimum values of all the possible combinations, the best-fit combination can be chosen.

After choosing the best-fit model, again, the backward propagation procedure mentioned in Subsection 3.1 can be used to find out the weight variables w's and the bias b of the model.

Not necessary to say that CMS can include any possible function in its way to choose the best-fit model possible. It can include algebraic functions like x, x^2, trigonometric functions like $\sin(x)$, $\cos(x)$, logarithmic functions like $\ln(x)$ or $\log(x)$, or even exponential function like $\exp(x)$. But it is suggested to keep the number of possible functions included in the experiment as small as possible as with the increase of the number of functions, the number of total possible combinations increases in an exponential way (r^n), where r is the number of included functions and n is the number of independent variables.

3.3 Algorithm Steps

1. Initialization: First of all, according to (3.1.4), the whole squared error loss function J has to be considered as a function of w_1 only.

$$J(w_1) = \alpha_1 w_1^2 + \beta_1 w_1 + \gamma_1$$

which is in the form $y = \alpha x^2 + \beta x + \gamma$, where the values of α_1, β_1, and γ_1 can be extracted from the equations 3.1.6, 3.1.7, and 3.1.8 respectively.

$$\alpha_1 = \sum x_1^2$$

$$\beta_1 = 2w_2 \sum x_1 x_2 + \cdots + 2w_n \sum x_1 x_n + 2b \sum x_1 - 2 \sum x_1 y$$

$$\begin{aligned}\gamma_1 = w_2^2 \sum x_2^2 + \cdots + w_n^2 \sum x_n^2 + nb^2 + \sum y^2 \\ + 2w_2w_3 \sum x_2x_3 + 2w_2w_n \sum x_2x_n + \cdots \\ + 2w_2b \sum x_2 - 2w_2 \sum x_2y + \cdots \\ + 2w_nb \sum x_n - 2w_n \sum x_ny - 2b \sum y.\end{aligned}$$

2. Analyzing the Loss Function: Next, the equation is analyzed using $\min J(w_1) = \gamma_1 - \frac{{\beta_1}^2}{4\alpha_1}$ to find out the minimum value of the loss function $J(w_1)$. Equation (3.1.10) illustrates the result. The minimum value of the loss function $J(w_1)$ is the loss function of w_2.

$$\begin{aligned}\min J(w_1) =& J(w_2) \\ =& \left(w_2^2 \sum x_2^2 + \cdots + w_n^2 \sum x_n^2 + nb^2 + \sum y^2\right) \\ &+ 2w_2w_3 \sum x_2x_3 + \cdots + 2w_2w_n \sum x_2x_n \\ &+ 2w_2b \sum x_2 - 2w_2 \sum x_2y + \ldots \\ &+ 2w_nb \sum x_n - 2w_n \sum x_ny - 2b \sum y \\ &- \frac{(2w_2 \sum x_1x_2 + \cdots + 2w_n \sum x_1x_n + 2b \sum x_1 - 2 \sum x_1y)^2}{4 \sum x_1^2}.\end{aligned}$$

3. Forward Iteration to Build the Minimum Loss Functions: This process is continued forward to build the equations of the loss functions $J(w)$ for all the w's.

4. Ending up with Minimum Loss Value and Bias: Continuing with the process, the minimum value of the loss function of the last weight function of w_n becomes a function of only b.

$$\begin{aligned}\min J(w_n) &= J(b) \\ &= \alpha_b b^2 + \beta_b b + \gamma_b \\ &= f(b)\end{aligned}$$

Here, α_b, β_b, γ_b all are numerical values.

Using $\min J(b) = \gamma_b - \frac{{\beta_b}^2}{4\alpha_b}$ on this function, its minimum value is calculated. This is the value of the total squared error loss of the model.

Also, by using $b = -\frac{\beta_b}{2\alpha_b}$, the value of the bias variable b where this minimum error occurs is calculated. This is the required bias variable b of the model.

5. Backward Iteration to Calculate the Weight Variables: Finally, a backward propagation is used to compute all the required values of the weight variables w's from w_n to w_1. Each weight variable w_i is calculated using the previously computed values of w_{i+1} to w_n and the bias variable b.

The loss function $J(w_i)$ is a function of the $w's$ from w_i to w_n and b.

$$\begin{aligned}J(w_i) &= \alpha_i w_i^2 + \beta_i w_i + \gamma_i \\ &= f(w_i, w_{i+1}, ..., w_n, b)\end{aligned}$$

Putting the already generated values of w_{i+1} to w_n and b in it, it is turned into a function of only w_i.

$$\begin{aligned}J(w_i) &= \alpha_i w_i^2 + \beta_i w_i + \gamma_i \\ &= f(w_i)\end{aligned}$$

Here, α_i, β_i, and γ_i are numerical values.

Now using $w_i = -\frac{\beta_i}{2\alpha_i}$, the required value of wi is calculated.

This process generates the precise values of all the weight variables w's and the bias variable b that build the regression model.

4 Experimental Setup and Results

To evaluate the performance of CMS, we took the initiative to experiment with a real-world dataset. A suitable real-world dataset was chosen to break down each step of constructing a linear regression model with CMS and discuss it in detail. In this section, we have accomplished this. We have also discussed how to use CMS to build non-linear polynomial regression models up to a specific index of independent variables and choose the best one from them. We then evaluated the constructed linear model with multiple metrics and analyzed the results in detail. At the end of this section, a comparison between the results obtained by CMS and other regression methods has been presented, to highlight the advantages of using CMS over other methods.

4.1 Dataset and Experimental Setup

Though CMS can handle a large number of features at a time, especially with advanced languages like python, we chose a dataset with a small number of independent variables so that the analysis of each independent variable can be shown in full detail. The selected dataset has four independent variables to predict one dependent variable. The number of data points, however, is not very small. A total number of 9568 data points are there. The full dataset is available here. Tüfekci (2014) used this dataset to successfully predict the full-load electrical power output of a base-load operated combined cycle power plant using machine learning methods.

The data points were collected from a combined cycle power plant over 6 years (2006-2011) when the plant was set to work with a full load. The four independent variables of the dataset are respectively ambient temperature (AT), exhaust volume (V), ambient pressure (AP), and relative humidity (RH). They are used to predict the values of a dependent variable power output (PE).

4.2 Linear Regression Model with CMS

The linear regression model is shown here. The equation of the best-fit straight line is similar to the one shown in equation 3.1.1, with the number of independent variables $n = 4$.

$$y = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b$$

The squared error loss function is similar to the one shown in (3.1.4), with $n = 4$.

$$\begin{aligned} J = {} & w_1^2 \sum x_1^2 + w_2^2 \sum x_2^2 + w_3^2 \sum x_3^2 \\ & + w_4^2 \sum x_4^2 + Nb^2 + \sum y^2 \\ & + 2w_1w_2 \sum x_1x_2 + 2w_1w_3 \sum x_1x_3 \\ & + 2w_1w_4 \sum x_1x_4 + 2w_1b \sum x_1 - 2w_1 \sum x_1y \\ & + 2w_2w_3 \sum x_2x_3 + 2w_2w_4 \sum x_2x_4 \\ & + 2w_2b \sum x_2 - 2w_2 \sum x_2y + 2w_3w_4 \sum x_3x_4 \\ & + 2w_3b \sum x_3 - 2w_3 \sum x_3y + 2w_4b \sum x_4 \\ & - 2w_4 \sum x_4y - 2b \sum y. \end{aligned} \tag{4.2.1}$$

Putting the corresponding values of $\sum x_1^2$, $\sum x_2^2$, $\sum x_3^2$, $\sum x_4^2$, N, $\sum y^2$, $\sum x_1x_2$, $\sum x_1x_3$, $\sum x_1x_4$, $\sum x_1$, $\sum x_1y$, $\sum x_2x_3$, $\sum x_2x_4$, $\sum x_2$, $\sum x_2y$, $\sum x_3x_4$, $\sum x_3$, $\sum x_3y$, $\sum x_4$, $\sum x_4y$, $\sum y$ in place in equation (4.2.1), we get a parabolic loss function of w_1.

$$\begin{aligned} J(w_1) = {} & (4226228.0792)w_1^2 \\ & + (21951075.3908w_2 + 380602167.3846w_3 + 26438023.385399997w_4 \\ & + 376045.95999999996b - 168554686.0126)w_1 \\ & + (29762163.2391w_2^2 + 9823745224.5548w_3^2 + 53459782.0308w_4^2 \\ & + 9568b^2 + 1978076968.9819002 + 1052377528.7871999w_2w_3 \\ & + 75074091.27w_2w_4 + 1039195.8600000001w_2b - 468564745.7976w_2 \\ & + 1421606173.7234w_3w_4 + 19389725.72w_3b - 8811018334.0838w_3 \\ & + 1402840.5999999999w_4b - 639260166.9092w_4 - 8694728.82b) \end{aligned} \tag{4.2.2}$$

Considering (4.2.2) as a function of w_1 of the form $J(w_1) = \alpha_1 w_1^2 + \beta_1 w_1 + \gamma_1$, the minimum value $\min J(w_1) = \gamma_1 - \frac{{\beta_1}^2}{4\alpha_1}$ (from (3.1.9)) can be calculated. It returns a parabolic loss function of w_2.

$$\begin{aligned}J(w_2) = {} & (1258631.142472323)w_2^2 \\ & + (63951595.11838591w_3 + 6414398.319104612w_4 \\ & + 62602.42210770468b - 30827294.008724272)w_2 \\ & + (1254756098.601389w_3^2 + 231139306.66392922w_3w_4 + 2456907.0540330783w_3b \\ & - 1221238793.4384289w_3 + 12112683.185908273w_4^2 + 226624.8707311789w_4b \\ & - 112046224.94852197w_4 + 1202.9424722074054b^2 - 1195806.9059104733b \\ & + 297460019.0452943)\end{aligned} \tag{4.2.3}$$

Considering (4.2.3) as a function of w_2 of the form $J(w_2) = \alpha_2 w_2^2 + \beta_2 w_2 + \gamma_2$, the minimum value $\min J(w_2) = \gamma_2 - \frac{{\beta_2}^2}{4\alpha_2}$ can be computed. It leaves a parabolic loss function of w_3.

$$\begin{aligned}J(w_3) = {} & (442404016.212824)w_3^2 \\ & + (68180124.10880119w_4 + 866478.9226434017b - 438064693.2136165)w_3 \\ & + (3940212.235580419w_4^2 + 67103.60290732619w_4b - 33493209.613271415w_4 \\ & + 424.50486616401866b^2 - 429155.26078207954b + 108698986.55693269)\end{aligned} \tag{4.2.4}$$

Considering (4.2.4) as a function of w_3 of the form $J(w_3) = \alpha_3 w_3^2 + \beta_3 w_3 + \gamma_3$, the minimum value $\min J(w_3) = \gamma_3 - \frac{{\beta_3}^2}{4\alpha_3}$ can be computed. It provides a parabolic loss function of w_4.

$$\begin{aligned}J(w_4) = {} & (1313354.6839215187)w_4^2 \\ & + (335.85406407437404b + 262479.8204115853)w_4 \\ & + (0.24011275358890316b^2 - 165.231856519414558b + 257003.43305903673)\end{aligned} \tag{4.2.5}$$

Considering (4.2.5) as a function of w_4 of the form $J(w_4) = \alpha_4 w_4^2 + \beta_4 w_4 + \gamma_4$, the minimum value $\min J(w_4) = \gamma_4 - \frac{{\beta_4}^2}{4\alpha_4}$ can be calculated. It provides a parabolic loss function of b.

$$J(b) = 0.21864141120588582b^2 - 198.79282656884664b + 243888.99090438892 \tag{4.2.6}$$

Considering (4.2.6) as a function of only b of the form $J(b) = \alpha_b b^2 + \beta_b b + \gamma_b$, its minimum value can be obtained from (3.1.9), $\min J(b) = \gamma_b - \frac{{\beta_b}^2}{4\alpha_b} = 198702.4595912306$, and the value of b that results in this minimum value will be, $b = \frac{-\beta_b}{2\alpha_b} = 454.6092743191532$. The plot of the loss function J as a parabolic function of bias b has been shown in Figure 1.

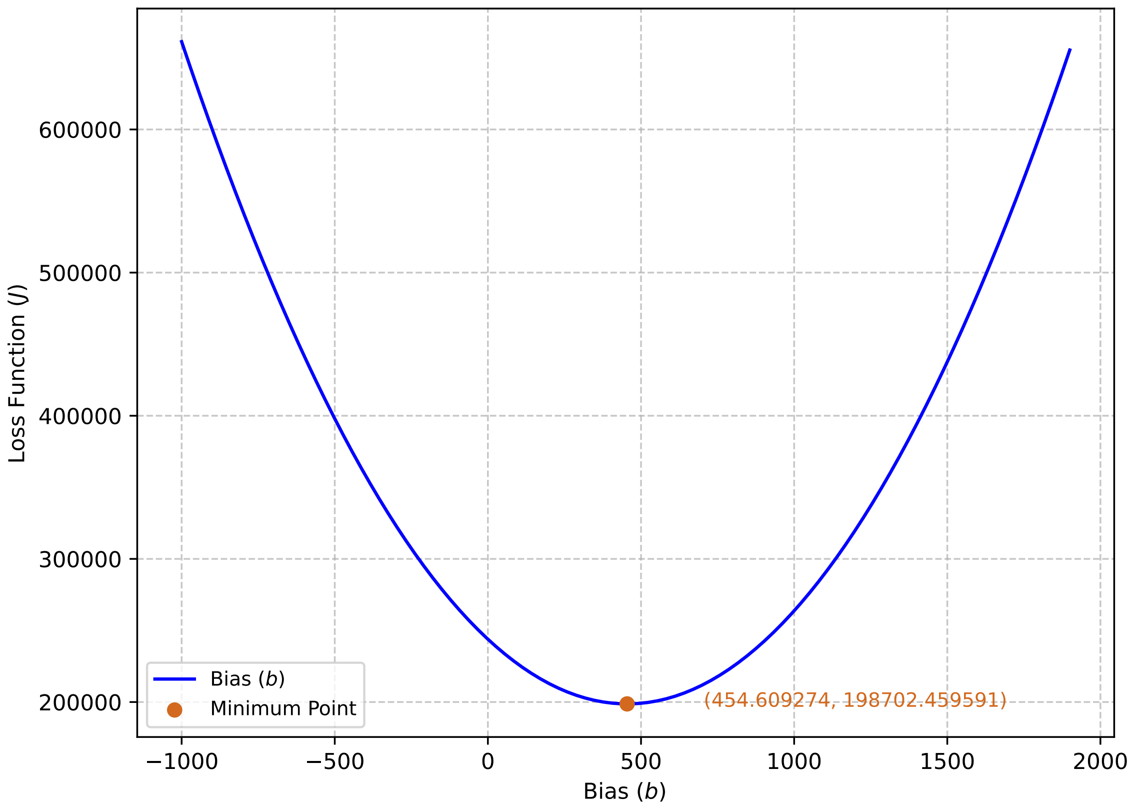


Figure 1: Squared error loss function J as a parabolic function of the bias b.

Therefore, the minimum value of the loss function is 198702.4595912306, and the value of b that results in this minimum value is 454.6092743191532. Putting $b = 454.6092743191532$ in (4.2.5), it becomes:

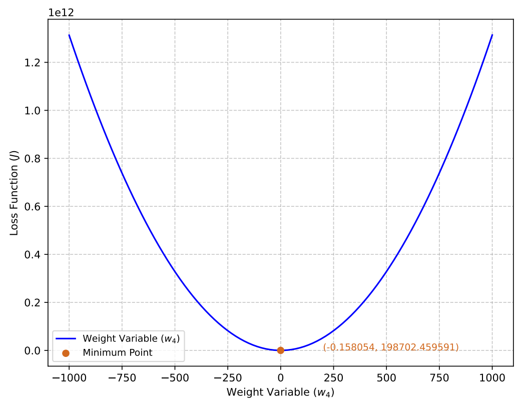
$$J(w_4) = 1313354.6839215187w_4^2 + 415162.19275757484w_4 + 231511.50356186475$$

Again, the minimum value of this function can be obtained using the formula $J_{min} = \gamma_4 - \frac{{\beta_4}^2}{4\alpha_4} = 198702.4595912306$. It appears when $w_4 = \frac{-\beta_4}{2\alpha_4} = -0.1580541029167957$. The plot of J_{min} as a parabolic function of weight w_4 has been shown in Figure 2a.

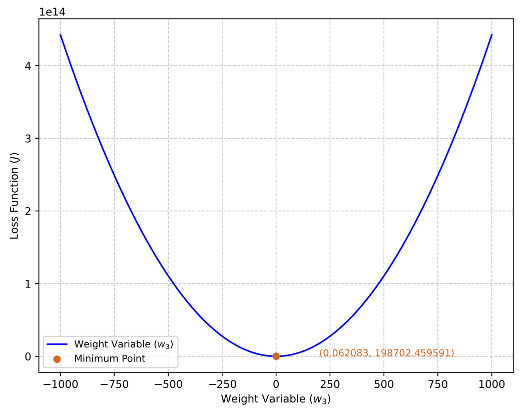
Putting $w_4 = -0.1580541029167957$ and $b = 454.6092743191532$ in (4.2.4), it becomes:

$$J(w_3) = 442404016.212824w_3^2 - 54931487.33063036w_3 + 1903856.6793619245$$

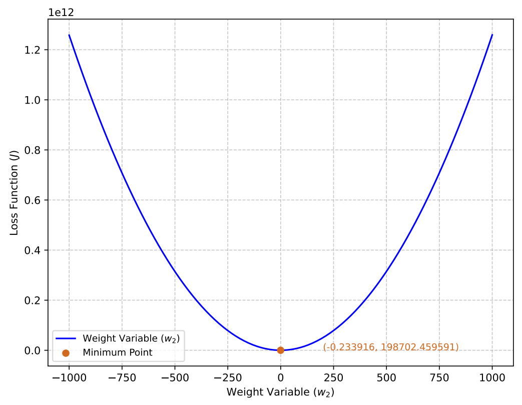
The minimum value of this function is found to be $J_{min} = \gamma_3 - \frac{{\beta_3}^2}{4\alpha_3} = 198702.4595912306$. It appears when $w_3 = \frac{-\beta_3}{2\alpha_3} = 0.062082943777125296$. The plot of J_{min} as a parabolic function of weight w_3 has been shown in Figure 2b.



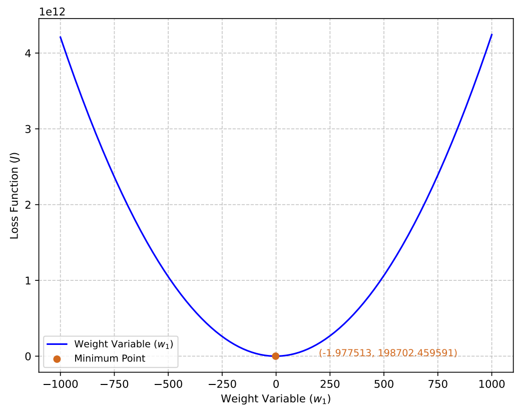
(a) Loss function J as a parabolic function of weight variable w_4.



(b) Loss function J as a parabolic function of weight variable w_3.



(c) Loss function J as a parabolic function of weight variable w_2.



(d) Loss function J as a parabolic function of weight variable w_1.

Figure 2: Squared error loss function J as parabolic functions of all weight variables (w_4 to w_1).

Putting the values of $w_3 = 0.062082943777125296$, $w_4 = -0.1580541029167957$ and $b = 454.6092743191532$ in (4.2.3), it becomes:

$$J(w_2) = 1258631.142472323w_2^2 + 588828.9883958176w_2 + 267570.8448303938$$

The minimum value of this function is found to be $J_{min} = \gamma_2 - \frac{\beta_2{}^2}{4\alpha_2} = 198702.4595912306$. It appears when $w_2 = \frac{-\beta_2}{2\alpha_2} = -0.23391642258238726$. The plot of J_{min} as a parabolic function of weight w_2 has been shown in Figure 2c.

Putting the values of $w_2 = 0.062082943777125296$, $w_3 = 0.062082943777125296$, $w_4 = -0.1580541029167957$ and $b = 454.6092743191532$ in (4.2.2), it becomes:

$$J(w_1) = 4226228.0792w_1^2 + 16714842.836514562w_1 + 16725612.851887941.$$

The minimum value of this function is found to be $J_{min} = \gamma_1 - \frac{{\beta_1}^2}{4\alpha_1} = 198702.4595912306$. It appears when $w_1 = \frac{-\beta_1}{2\alpha_1} = -1.9775131066374658$. The plot of J_{min} as a parabolic function of weight w_1 has been shown in Figure 2d.

Finally, after getting the values of all the weight variables w_1, w_2, w_3, w_4, and the bias variable b, we can conclude that the best-fit linear model for this dataset is:

$$\begin{aligned} y = & -1.9775131066374658x_1 - 0.23391642258238726x_2 + 0.062082943777125296x_3 \\ & - 0.1580541029167957x_4 + 454.6092743191532 \end{aligned} \tag{4.2.7}$$

And the total squared error loss is 198702.4595912306.
Mean square error (MSE) $= \frac{198702.4595912306}{9568} = 20.7673975325282$.

4.3 Coefficient Analysis

The coefficients reflect each predictor's effect on the dependent variable, holding others constant. For this dataset, they are:

- **Ambient Temperature (AT):** Coefficient is -1.9775. This suggests that with a unit increase in AT, the dependent variable decreases by approximately 1.98 units.
- **Exhaust Volume (V):** Coefficient is -0.2339. This indicates a reduction of about 0.23 units in the dependent variable per unit increase in V.
- **Ambient Pressure (AP):** Coefficient is 0.0621. This indicates an increase of about 0.06 units in the dependent variable per unit increase in AP.
- **Relative Humidity (RH):** Coefficient is -0.1581. This indicates a reduction of about 0.15 units in the dependent variable per unit increase in RH.

4.4 Evaluation Metrices

The model formed using the coordinated minima search (CMS) algorithm had a mean square error(MSE) of 20.7673975325282. It was further evaluated using:

- R-squared (R^2)
- Adjusted R-squared
- Residual Standard Error (RSE)
- F-statistic
- p-values of the independent variables

4.5 Results Analysis

Evaluating the model with the already mentioned metrics, we got the following results:

- R-squared: 0.9286960898122537

- Adjusted R-squared: 0.9286513430848061
- RSE: 4.558793940891878
- F-statistic: 31138.26
- p-values: close to zero for each independent variable.

The results can be evaluated like this:

4.5.1 Overall Model Fit

The **R-squared** value of the model formed by coordinated minima search (CMS) is found to be 0.9287. This suggests that the model appears to fit the data exceptionally well. 92.87% of the variance in the dependent variable is explained by the independent variables included in the model. The **adjusted R-squared** is computed to be 0.9287. This also indicates that the model complexity is justified by the explained variance.

4.5.2 Residual Analysis

Residual analysis was also performed to evaluate the performance of the model built by coordinated minima search (CMS). **Residual standard error (RSE)** of the model was calculated as 4.5588. This RSE, whereas the range of the dependent variable PE is 75.5 (420.26 to 495.76), represents about 6% of the range.

$$\frac{4.5588}{75.5} \times 100 \approx 6\%.$$

This indicates that the model's average prediction error is small relative to the scale of the dependent variable, which suggests it as a reasonably accurate model.

To investigate the behavior of the residuals more closely, we calculated their mean and standard deviation. The mean was found to be $4.7565 \times 10^{-}13$ (close to zero) and the standard deviation was computed as 4.5571. The closeness of the mean to zero suggests that the model fits good with the data, but we still had to evaluate whether the residuals follow a normal distribution or not. To evaluate it, we plotted their Q-Q plot. It has been shown in Figure 3.

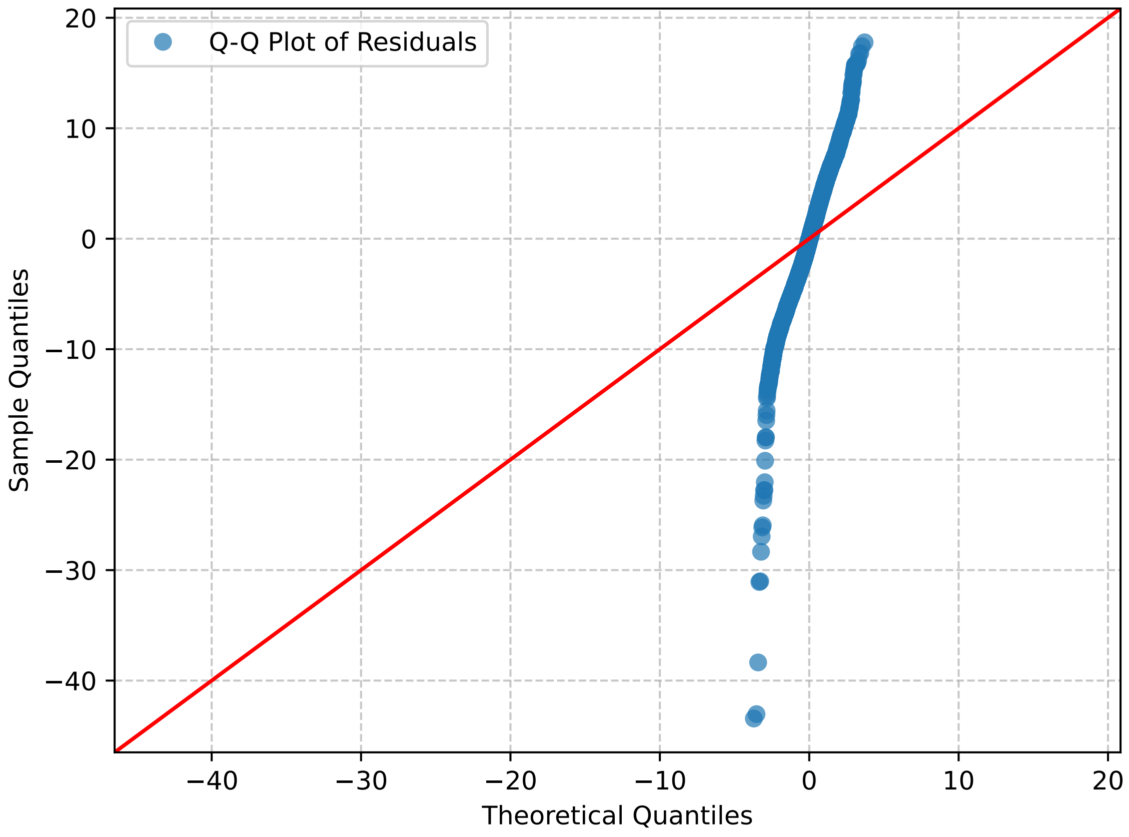


Figure 3: Q-Q plot of the residuals. This states that the residuals follow an approximately normal distribution with a small downward curve in the lower tail and an upward curve in the upper tail.

The Q-Q plot shows that the residuals have a small downward curve in the lower tail and an upward curve in the upper tail. This indicates that they do not follow a perfectly normal distribution but an approximate one with a few outliers. To diagnose this more accurately, we plotted the residuals against the fitted values of the dependent variables, $\hat{y}$. There they are found to scatter randomly around the horizontal axis $y = 0$, except few distinct outliers. This confirms that the model fits reasonably well with the available data, ignoring the distinct outliers. The plot of the residuals versus fitted values has been shown in Figure 4.

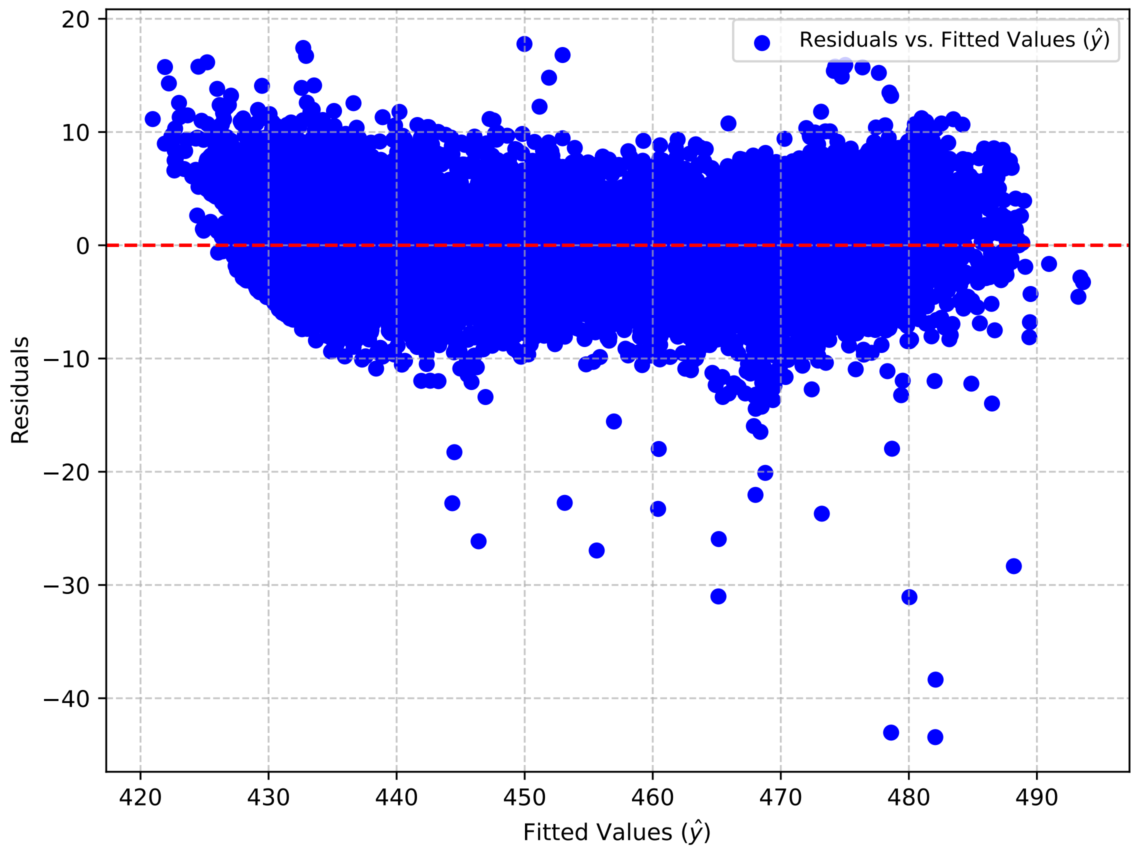
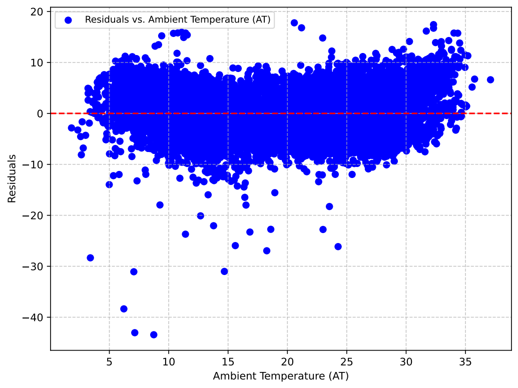
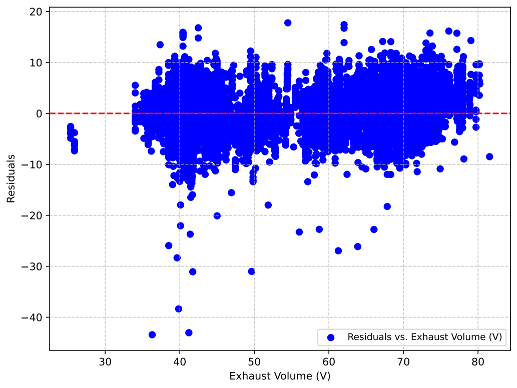


Figure 4: Residuals are plotted against the fitted values of the dependent variable $\hat{y}$. It is seen that the residuals are scattered randomly across the horizontal axis $y = 0$, ignoring a few distinct outliers. This suggests that the model fits good with the available data, ignoring the distinct outliers.

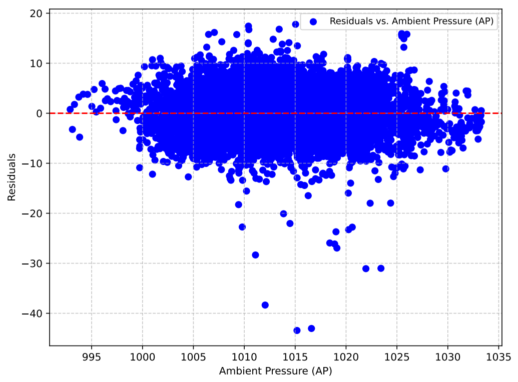
Finally, we plotted the residuals against each independent variable to evaluate how they behave. They were found to scatter randomly around $y = 0$ for each of the four independent variables, except the few distinct outliers. This confirmed that our linear regression model fits well with the data, except the distinct outliers. The plots of the residuals against the independent variables have been shown in Figure 5.



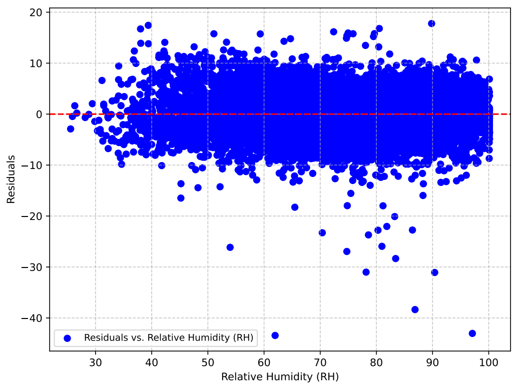
(a) Residuals plotted against the first independent variable, ambient temperature (AT).



(b) Residuals plotted against the second independent variable, exhaust volume (V).



(c) Residuals plotted against the third independent variable, ambient pressure (AP).



(d) Residuals plotted against the fourth independent variable, relative humidity (RH).

Figure 5: Residuals are plotted against each of the four independent variables. This shows that the residuals are scattered randomly across the horizontal axis $y = 0$ for each independent variable, ignoring a few distinct outliers. This confirms the model's good fit with the available data ignoring the outliers.

4.5.3 Statistical Significance

F-statistic value of the model is found to be 31138.26. This very high F-statistic demonstrates that the predictors, as a group, are significantly related to the dependent variable. This confirms the model's overall statistical significance.

Also, each predictor variable (ambient temperature, exhaust Volume, ambient pressure, relative humidity) has p-values close to 0.000. This also shows that each predictor variable has a statistically significant relationship individually with the dependent variable.

4.5.4 Summary

Overall it is seen that the model performs quite well, explaining a substantial portion of the variance. A strong fit with an R-squared of 0.9287 explains 92.87% of the variance in the dependent variable. The residual standard error (RSE) of 4.5588, which is only 6% of the range of the dependent variable, indicates a low prediction error. However, diving deep into the residual analysis suggests the presence of a few outliers that cause the normal curve of the residuals to deviate a bit upward in the upper tail and downward in the lower tail. Plots of the residuals against the fitted values of the dependent variable ($\hat{y}$) and each of the independent variables confirm the presence of the outliers, but it is seen that the model fits quite well ignoring the outliers. A high F-statistic of 20754.50 confirms the model's statistical significance, with each predictor (AT, V, AP, RH) significantly contributing. The coefficients indicate each predictor's impact. Ambient temperature has the largest effect, reducing the dependent variable by approximately 1.98 units per unit increase. Summarizing all these, it can be concluded that the regression model developed using the CMS approach by minimizing the sum of squared errors, demonstrates strong functionality and reliability across most of the evaluation metrics.

4.6 Comparison with Other Regression Methods

The independent variable coefficients of the linear regression model formed by CMS are -1.97751310663, -0.23391642258, 0.06208294377,-0.15805410291, whereas the bias value is 454.6092743191532 (from (4.2.7)). The mean square error (MSE) of the model is found to be 20.7673975325282. This is the lowest possible mean square error for this dataset. No other method can provide a model with a better result. To compare, we formed linear regression models on the same dataset using methods like ordinary least squares (OLS), stochastic gradient descent (SGD), coordinate descent (CD), and artificial neural networks (ANN). A comparative analysis of CMS with different regression methods across various metrics has been presented in Table 1.

Table 1: Comparison of Different Regression Methods Across Metrics

Metric	**CMS**	**OLS**	**SGD**	**CD**	**ANN**
MSE	20.7673	20.7673	20.7707	20.7673	25.8640
R-squared	0.9287	0.9287	0.9287	0.9287	0.9112
Adjusted R-squared	0.9287	0.9287	0.9287	0.9287	0.9112
RSE	4.5583	4.5583	4.5586	4.5583	5.0856
F-statistic	31138.2668	31138.2668	31132.8738	31138.2668	24531.20
Parameter Tuning	no	no	yes	yes	yes
Feature Scaling	no	no	yes	yes	no

We see in Table 1 that the MSE, the R-squared value, the adjusted R-squared, the RSE, and the F-statistic value of the models formed by CMS, OLS, SGD, and GD are approximately the same, only the results provided by ANN seem to deviate to some extent. But to get this result, SGD and CD require extra initiatives like parameter tuning and feature scaling. Especially without fine-tuning the hyper-parameters, it is tough to get the perfect convergence of the models; often, they get trapped within suboptimal solutions. In SGD, after several fine-tuning of the regularization parameter α, we set it to 0.001. In CD, we had to fine-tune the tolerance of convergence which was finally set to 10^-6. Moreover, the input data had to be normalized in CD and standardized using StandardScaler in SGD for better results. Furthermore, it needs to be mentioned that though SGD and CD seem to result in the same outputs compared to CMS and OLS, they are not exactly the same. For this dataset, they differ after a couple of positions after the decimal. This difference will increase with the increasing number of data points and independent variables. Only CMS and OLS provide the lowest possible mean square error (MSE), and that provided by the other models will most often become slightly larger. Lastly, though CMS and OLS resulted in exactly the same results, both without any extra hassle of parameter tuning or feature scaling, CMS produced them a bit more comprehensively compared to OLS, without involving any complex computation like matrix inversion. Therefore, it can be concluded that in the field of optimizing linear regression models, CMS offers a quite sophisticated alternative that produces the best results like OLS in a comparatively easier way.

5 Implementation in Python

The complete implementation of the coordinated minima search (CMS) algorithm has been developed in python and made publicly available on the Python Package Index (PyPI) under the name `cms_model`. The package can be easily installed using the python `pip` package manager by running the following command in a terminal:

```
pip install cms_model
```

After installation, the package can be imported into python scripts or projects to perform regression analysis using the CMS algorithm. The package is designed to handle both linear and non-linear polynomial regression tasks efficiently. It includes functionality to optimize models based on the squared error loss function. Users can find detailed documentation and usage examples in the accompanying package repository on PyPI.

The full details of the package, including installation instructions, usage examples, and documentation, are available on its official PyPI page.

6 Discussion

To this point, we have discussed the coordinate minima search (CMS) method step-by-step in detail, illustrated each step with examples, and figured out the usefulness that CMS brings over other methods in linear regression optimization. In this section, we will discuss the advantages of using CMS over other regression methods in detail. Also, along with the advantages, CMS has some limitations that need to be addressed. After discussing the advantages, we will address these limitations, too. Finally, at the end of this section, we will focus on some works that can be accomplished in the future to turn CMS into a more powerful and sophisticated tool for solving more complex problems in the field of machine learning.

6.1 Advantages of CMS

The primary advantage of the coordinated minima search (CMS) method is that it provides the most precise output possible with a comparatively easier approach. It looks for a model with weight variables and bias that result in the least possible squared error which, no need to mention, is the best regression model possible, with the available data. Unlike gradient descent or other similar optimization methods, there is no possibility of getting trapped within local minima. Also in CMS, there is no extra hassle of tuning hyperparameters or scaling input features to improve model performance.

The major advantages of the CMS approach are demonstrated below in short:

- **Novelty of the Approach**: The coordinated minima search (CMS) method offers a completely new and innovative way of tackling regression problems of linear models. This can contribute valuable knowledge to the fields of machine learning, optimization, or statistical modeling. Not only linear regression models but also we can form any category of a model where the parameters follow linear architecture, according to the nature of the available data and the corresponding business. For example, for businesses that follow seasonal or cyclic trends, we can easily form an equation of the form $y = a\sin(\omega x + \phi) + b$ using our CMS model, where $\sin(\omega x + \phi)$ is a function $f(x)$ of the independent variable x, and a and b are the weight variable and the bias respectively that we need to find out. Similarly, models that follow hyperbolic trends like $y = \frac{m(x+a)}{x+b}$ can easily be broken into two different functions of x, $f_1(x) = \frac{mx}{x+b}$ and $f_2(x) = \frac{ma}{x+b}$, and the value of the weight variables m and ma, can be calculated precisely using CMS. Thus CMS can make valuable contributions in the fields of statistical modeling, financial analysis, business interpretation, etc.

- **Preciseness of the Algorithm**: CMS provides the most precise model possible. That is, it searches for and finds out the weight variables and the bias that results in the least squared error possible. Unlike gradient descent or similar optimization methods, there is no scope of getting trapped within local minima. Thus it comes out as one of the best approaches possible for regression problems in machine learning for both linear and non-linear cases. We saw in our experiment earlier that the total squared error loss of 198702.4595912306 or mean square error of 20.7673 was the least squared error loss possible for the available dataset. Experiments were conducted with models including OLS, SGD, CD, and ANN, but no model could provide a smaller squared error than this.

- **Robustness of the Method**: The CMS method can be termed a hyperparameter-free optimization technique. That is, the method is zero sensitive to the choice of hyperparameters, which offers more robustness to the approach. Unlike gradient descent, neural networks, or similar optimization algorithms, the CMS method does not depend on the selection of any hyperparameter, like the learning rate or similar parameters. This has made the approach easy to compute, as well as increased the possibility of getting the best possible result in exchange for a minimum effort. Simultaneously, it ensures the method's consistent performance across different datasets.

- **Data Size Independence**: The time required for the CMS algorithm to function fully does not depend on the number of data points. The loss function is formed only once, and then it iterates over each independent variable to calculate the weight variables to minimize that loss function. Therefore, the required time does not depend on the number of data points, rather it depends on the number of independent variables. However, a large number of data points may sometimes cause overflow errors.

- **Application Scope**: The technique is flexible enough to handle a variety of functions (polynomial, trigonometric, logarithmic, and exponential). In real-world applications, it will open a new window in domains like market analysis, finance, etc. where it is required to analyze a large number of data points with complex non-linear models.

6.2 Limitations

Despite its advantages, CMS has some limitations. It can become computationally expensive when applied to very large datasets. As this technique forms the first loss function taking all the available data into account, the availability of a very large number of data may surpass the machine's capacity, and throw overflow error. Also as it analyzes each independent variable individually after forming the loss function, a very large number of independent variables may sometimes slow down the process. But nevertheless, all these depend on the capacity of the machine's processor. Working with powerful processors will hardly cause any issues.

6.3 Future Work

Despite the limitations, co-ordinated minima search (CMS) brings a milestone in regular regression techniques by analyzing the loss function as a parabolic function $y = \alpha x^2 + \beta x + \gamma$ of each independent variable and finding out the weight variables that result in the least value of this loss function by fitting it perfectly. Though this paper is all about linear regression models (also about the models that follow linear structures in parameters but non-linear structures in variables), future works may extend it to more types of non-linear regression models. For non-linear models, the loss functions concerning the unknown variables do not follow the parabolic equation $y = \alpha x^2 + \beta x + \gamma$ every time and, therefore, can not be solved using CMS. For example, for hyperbolic functions like like $y = \frac{m(x+a)}{x+b}$, the values of the variables m and a can be found out by breaking down the function into two functions as $f_1(x) = \frac{mx}{x+b}$ and $f_2(x) = \frac{ma}{x+b}$ and building a loss function by (3.2.1), but the value of the unknown variable b can not be predicted this way, as the squared error loss function concerning b does not follow the parabolic equation $y = \alpha x^2 + \beta x + \gamma$. To find out the value of b, we are required to run several iterations with CMS while setting different values of b each time. At one point, we should find out the value that results in the best result. Similarly for sinusoidal functions like $y = a \sin(\omega x + \phi) + b$, the values of a and b can be predicted using CMS, as a is the co-efficient of the function $f(x) = a \sin(\omega x + \phi)$ that follows linear structure, and b is the bias variable, but the values of ω and ϕ can not be determined this way, as the loss functions concerning them do not follow the parabolic equation $y = \alpha x^2 + \beta x + \gamma$. To find out their values, we need to run several iterations while assigning different values to them each time and observing which values result in the best possible output. These limitations can be addressed by introducing an algorithm that will set the first derivative of all the loss functions concerning each unknown variable to zero, and ultimately find out the value of the unknown variable that results in the derivative's value being zero. It is known from differential calculus that the value of a function becomes maximum or minimum (minimum for the regression loss function, as it can not have a fixed upper limit) for a variable at a point where its first derivative concerning that variable becomes zero. Therefore, by setting the derivative of the loss function concerning each unknown variable to zero, the required value of the variable, along with the corresponding minimum value of the loss function, can be determined. These works are kept for the future.

By calculating the first derivative of the loss function concerning each unknown variable and setting it to zero, the method can be extended to all types of loss functions, not only the squared error loss function addressed in this paper. Loss functions that do not follow the parabolic equation $y = \alpha x^2 + \beta x + \gamma$ can also be analyzed by setting its first derivative to zero and then computing the values of the unknown variables that result in it being zero. These scopes are preserved for the future.

7 Conclusion

Coordinated minima search offers a promising alternative to traditional regression techniques by introducing a coordinated approach to minimizing the loss function in linear regression models. Through the parabolic minimization of the loss function, CMS demonstrates significant improvements in predictive accuracy and calculation ease. Given its potential applications across multiple domains and its capacity to model linear relationships with a comparatively easier approach, it represents a significant contribution to the field of regression analysis and optimization. The method's flexibility and precision make it a valuable tool for regression modeling, though future work is needed to extend its scalability.

References

Yoshua Bengio. Learning deep architectures for ai. *Foundations and Trends® in Machine Learning*, 2:1–127, 2009.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In Yves Lechevallier and Gilbert Saporta, editors, *Proceedings of COMPSTAT'2010*, pages 177–186, Heidelberg, 2010. Physica-Verlag HD.

Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

Samprit Chatterjee and Ali S. Hadi. *Regression Analysis by Example.* John Wiley & Sons, Inc., New York, NY, fifth edition, 2013.

Scheilla V.C. de Souza and Roberto G. Junqueira. A procedure to assess linearity by ordinary least squares method. *Analytica Chimica Acta*, 552(1):25–35, 2005.

Rita Obikimari Efeizomor. A comparative study of methods of remedying multicolinearity. *American Journal of Theoretical and Applied Statistics*, 12(4):87–91, 2023.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, New York, NY, second edition, 2009.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*. Springer, New York, NY, second edition, 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL `https://arxiv.org/abs/1412.6980`.

Michael H. Kutner, Chris J. Nachtsheim, and John Neter. *Applied Linear Statistical Models*. McGrwa-Hill international edition. McGraw-Hill Irwin, fifth edition, 2005.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, 2015.

Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, NY, second edition, 2006.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1310–1318, Atlanta, Georgia, USA, 2013. PMLR.

Simo Puntanen and George P. H. Styan. The equality of the ordinary least squares estimator and the best linear unbiased estimator. *The American Statistician*, 43(3):153–161, 1989.

Sebastian Ruder. An overview of gradient descent optimization algorithms, 2017. URL `https://arxiv.org/abs/1609.04747`.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

Pınar Tüfekci. Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods. *International Journal of Electrical Power & Energy Systems*, 60:126–140, 2014.

Stephen J. Wright. Coordinate descent algorithms, 2015. URL `https://arxiv.org/abs/1502.04759`.

Tong Tong Wu and Kenneth Lange. Coordinate descent algorithms for lasso penalized regression. *The Annals of Applied Statistics*, 2(1):224 – 244, 2008.