Conjugate Gradient Method for finding Optimal Parameters in Linear Regression

Vishal Menon, V. Ashwin, and G. Gopakumar

Abstract Linear regression is one of the most celebrated approaches for modeling the relationship between independent and dependent variables in a prediction problem. It can have applications in a number of domains including weather data analysis, price estimation, bioinformatics, etc. Various computational approaches have been devised for finding the best model parameter. In this work, we explore and establish the possibility of applying the Conjugate Gradient Method for finding the optimal parameters for our regression model, which is demonstrated by taking the house price prediction problem using the Boston dataset. The efficiency of the conjugate gradient method over the pseudo-inverse method and gradient descent methods in terms of computational requirement are discussed. We show that the weights obtained by the conjugate gradient are accurate and the parameter vector converges to an optimal value in relatively fewer iterations when compared to the gradient descent techniques. Hence, Conjugate Gradient Method proves to be a faster approach for a linear regression problem in ordinary least square settings.

Keywords Machine learning · Linear regression · Conjugate gradient method · Gradient descent · Boston house price prediction

1 Introduction

In recent years Machine learning has become a field of eminence. Most of the daily life challenges are being solved by machine learning algorithms. The root cause for this upshoot in the field is due to the capability of ML algorithms to go beyond human thinking. From a simple logical reasoning, ML develops into more complex patterns capable of providing solutions unimaginable by humans. One such example is Google's AlphaGo, a computer program capable of playing the popular Chinese game "Go." Researchers stated that AlphaGo was capable of doing moves which

V. Menon · V. Ashwin · G. Gopakumar (✉)
Department of Computer Science and Engineering, Amrita Vishwa Vidyapeetham, Amritapuri, India
e-mail: gopakumarg@am.amrita.edu

M. D. Borah et al. (eds.), *Big Data, Machine Learning, and Applications*, Lecture Notes in Electrical Engineering 1053, https://doi.org/10.1007/978-981-99-3481-2_15

even the world renowned champions couldn't think of [12]. In a nutshell, we can say that ML has the capability of thinking outside the box.

One such ML algorithm is Linear Regression. It is an algorithm under supervised learning. Regression models a target prediction value based on independent variables. It has various applications including weather data analysis [4], sentiment analysis [15], performance prediction [2], aerodynamics [27], price estimation [22], bioinformatics [6, 8], etc., and many variants are popular in the literature [9, 25]. In almost all practical situations, we can model the dependent variable meaningfully from the independent variables. For example, the sales of products in a super market depends upon the popularity index, season of the year, availability, festivals during the year, etc. Thus a good model predicting the sales of different products could be used by the owner to control the supply chain thereby maximizing the profit.

The model selection process has proved to be one of the main aspects of predictive modeling. Once a particular model is fixed, the best parameters that make up the model are computed by using an optimization algorithm based on several factors like time complexity, convergence, and computational requirements. The main aim of our machine learning model will be to find the best fitting parameters that can minimize the recorded cost function on the training dataset. For a linear regression model, the traditional method of computing the optimal parameters consisted of the use of the gradient descent optimization approaches. In basic setting, the batch gradient descent [16] is employed where the model parameter is updated in each epoch and it demands one pass through all training samples (Eq. 9).

In this research work, we analyze different computational techniques to find the optimal parameters using a basic linear regression model. The traditional gradient descent method proved to be less efficient in finding the optimal parameters as the number of iterations will vary depending on the initial parameter vector and learning rate. Hence, we provide the necessary theory to show that the shortcomings of linear regression can be tackled by the conjugate gradient method which requires exactly "N" steps to find the "N-D" optimal parameter vector. In practical applications, as we are looking for a parameter vector that performs decently well on the validation set, we may get a descent solution in less than "N" iterations.

The remaining sections of this manuscript are organized as follows. Section 2 discusses different computational approaches used in the literature to find the optimum model parameters for regression. Section 3 provides the theoretical background behind Linear regression and Conjugate Gradient Descent. The details of the dataset used in this study and different experiments conducted to establish the merit of these computational methods followed by results and discussion are provided in Sect. 4. Finally, the paper is concluded in Sect. 5.

2 Related Works

As discussed in the previous section, linear regression is a popular ML technique that is used to model the relationship between the independent variables (features) to the output variable [5, 24, 28, 30, 31]. It has profound applications ranging from weather prediction [4], price and performance estimation [2, 18, 22, 29], to medical research [6, 8], bioinformatics [14], etc.

The most critical job in linear regression is fixing the right model. Once the model is fixed, the algorithm can give the best parameters for the model. The simplest linear regression problem tries to model the right parameter vector θ that relates feature vector X to the target variable Y. Thus we are looking for the best parameter vector θ for the relation $X\ \theta = Y$. In the literature, there are different computational techniques to find these parameters. The simplest is based on the normal equation (Eq. 12) [3, 17]. However, the normal equation-based method (Eq. 12) involves a matrix inversion. This means that the method is going to be prohibitively expensive when we need to consider large number of features, which is often the case [1, 7, 19]. The gradient descent-based techniques [13, 20, 26] are free from this issue and are increasingly popular in this field. Here, in order to find a better parameter vector (θ) that minimizes a convex cost function $J(\theta)$, we move from the current vector in the opposite direction of gradient as decided by a learning rate α (Eq. 13). The demerit of the method is that the number of iterations required to reach the optimal parameters will vary depending on the learning rate chosen and the initial parameter vector used as demonstrated in Table 3.

3 Linear Regression

Linear regression attempts to model target variable Y using the linear relation $X\ \theta = Y$, where θ is the unknown parameter vector. Hence, solving a system of equations for finding the parameter vector becomes the fundamental objective of a linear regression problem. However, often we will be dealing with an over determined system, with inherent noise in the observed features, which makes the solution non-trivial.

For establishing the applicability of different computational techniques to find the right parameter vector for the linear regression problem, we experiment on the Boston housing dataset (Sect. 4.1), where the task is to build a regression model to predict the cost of a building from several features. Our fundamental objective is to develop a relation between MEDV (final cost) and the other parameters in the dataset. This relation can be shown as

$$\begin{bmatrix} x_{11} & \dots & 1 \\ x_{21} & \dots & 1 \\ \vdots & & \\ x_{n1} & \dots & 1 \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_0 \end{bmatrix} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_n \end{bmatrix} \tag{1}$$

Equation 1 is of the form $X\ \theta = \hat{Y}$, where the $N \times D$ dimensional matrix X holds N data samples each with D features, θ is the $D \times 1$ parameter vector, and $\hat{Y}$ is the predicted value of the output variables. As the objective is to find an approximate solution to the above equation, we intend to find the parameter vector that minimizes the mean squared error (J) between the predicted and output variables, which can be written in vector form as (Eq. 2).

$$J = \frac{1}{N}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2 = \frac{\left\|Y - \hat{Y}\right\|^2}{N} \tag{2}$$

Hence, this gives us an unconstrained optimization problem. Therefore the solution for such optimization is the local minima of the cost function and the given problem can be further represented as a convex optimization problem, owing to the positive definite nature of its Hessian matrix:

$$\begin{aligned} MSE(J) &= \frac{||Y - \hat{Y}||^2}{N} \\ &= \frac{[Y - \hat{Y}]^T[Y - \hat{Y}]}{N} \end{aligned} \tag{3}$$

$$J = \frac{[Y - X\theta]^T[Y - X\theta]}{N} \tag{4}$$

The convex nature of a function can be confirmed by proving the Hessian matrix of the MSE (Eq. 4) as positive semi-definite:

$$\frac{dJ}{d\theta} = \frac{-2}{N}(Y - \hat{Y})X \tag{5}$$

$$\begin{aligned} H &= \frac{d}{d\theta}\left[\frac{-2}{N}(Y - X\theta)X\right] \\ &= \frac{2}{N}XX^T \end{aligned} \tag{6}$$

The Hessian matrix H in Eq. 6 is positive semi-definite since $z^T Hz \geq 0, \forall z$ as seen in Eq. 8

$$\begin{aligned} z^T Hz &= z^T\left(\frac{2}{N}XX^T\right)z \\ &= \frac{2}{N}\left[z^T XX^T z\right] \\ &= \frac{2}{N}v^T v = \frac{2}{N}\left\|v\right\|_2^2 \end{aligned} \tag{7}$$

As norm of a vector cannot be negative,

$$\|v\|_2^2 \geq 0 \therefore z^T Hz = \frac{2}{N}\|v\|_2^2 \geq 0 \tag{8}$$

Thus, MSE (Eq. 2) forms a convex function and the parameter vector that minimizes J (the global minimum) can be easily found by setting the gradient to zero:

$$\frac{\partial J}{\partial \theta} = \frac{-2}{N}\left(\sum_{i=1}^{n}(y_i - \hat{y}_i)x_i\right) = \frac{-2}{N}(Y - \hat{Y})X \tag{9}$$

$$\begin{aligned}\therefore \frac{dJ}{d\theta} &= 0 \Rightarrow (Y - \hat{Y})X \\ &\Rightarrow X^T Y - X^T X\theta = 0\end{aligned} \tag{10}$$

$$\therefore X^T b = X^T X\theta \tag{11}$$

$$\therefore \theta_o = \left(X^T X\right)^{-1} X^T b \tag{12}$$

Thus the optimal model parameter vector (θ_o) which minimizes MSE(J) can be found by pseudo-inverse (Moore-Penrose Inverse) as shown in Eq. 12.

Note that Eq. 12 involves matrix inversion, and in many practical applications we will be dealing with matrices having a large number of predictor variables [1, 7, 19]. This means that cost for finding the model parameter using Eq. 12 is going to be prohibitively high for such cases. Gradient descent-based techniques can be used to counter this problem.

The Gradient descent technique [13, 20, 26] finds the correct parameter without involving any matrix inversion. Since the cost function J is convex, the method is ensured to converge to the optimum parameter vector. The gradient descent method involves moving in the opposite direction of the gradient at each iteration, in order to find the optimal parameter as shown in Eq. 13.

$$\theta_{\text{new}} = \theta_{\text{old}} - \alpha \frac{\text{d}J}{\text{d}\theta_{\text{old}}} \tag{13}$$

In the equation above (Eq. 13), the learning rate α decides the speed with which we are moving from the current parameter vector (convergence). A higher learning rate can even lead to divergence (as shown in Fig. 1a). An effective divergence for the chosen learning rate can easily be identified by inspecting the value of the cost function across two successive iterations. The cost, being a convex function (refer Eq. 8), should always decrease for a good learning rate. Once we have chosen a good learning rate, the gradient descent will always converge irrespective of the value of the chosen α. However, a smaller learning rate can cause slower convergence (refer Fig. 1b).

The gradient computed for the function requires to pass through all training samples as shown in Eq. 9. This means that the gradient used is batch gradient descent. Although being a computationally easier technique when compared to the methods based on normal equations (the pseudo-inverse method), gradient descent for param-

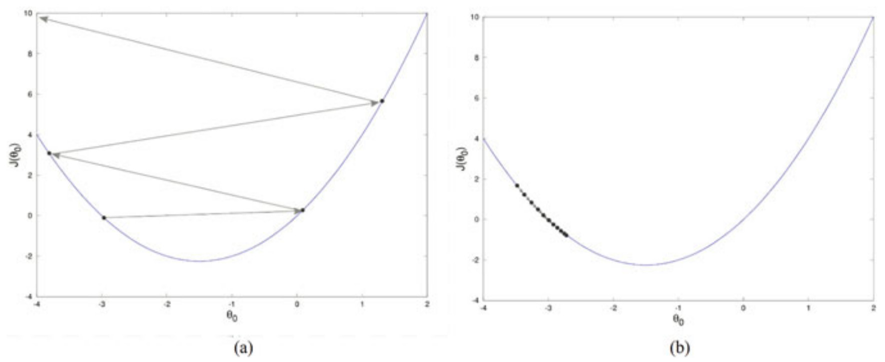

Fig. 1 **a** Large learning rate leads to drastic updates causing divergent behavior **b** Small learning rate requires many updates before reaching minima

eter updation has a drawback. In normal gradient descent, the number of iterations required to converge to the right parameter depends on the chosen initial vector and the learning rate α. For our dataset, we have experimented with different initial parameter vectors and the results are provided in Sect. 4.3.

Conjugate gradient method [11, 23] is a special technique that can be used to solve linear system of equations $A\,\theta = b$, if A is symmetric positive definite (SPD). It can be shown that the solution vector θ is going to be the parameter vector that minimizes the convex optimization function given in Eq. 14 [13, 20]. The one-line proof for the same is given in Eq. 15 where we have used the SPD nature of the matrix A.

$$f(\theta) = \frac{1}{2}\theta^T A\theta - \theta^T b + c \tag{14}$$

$$\frac{df}{d\theta} = 0 \Rightarrow A\theta - b = 0 \Rightarrow A\theta = b;\ \text{if } A^T = A \text{ and } A > 0 \tag{15}$$

The conjugate gradient method [11, 23] can converge to the optimal solution in exactly "D" steps for a D dimensional parameter vector [11] and it makes use of the best learning rate in each iteration [21]. The pseudo-code for the algorithm is shown in Algorithm 1.

As shown in the pseudo-code, the parameters are updated in each iteration using the optimum learning rate α.

$$\theta_{(k+1)} = \theta_{(k)} + \alpha_k d_{(k)} \tag{16}$$

We propose to use conjugate gradient method for finding the optimal parameter vector since $X^T X$ is symmetric ($(X^T X)^T = X^T X$) and positive definite matrix (in fact semi-definite by definition of semi-definite matrices) for practical applications.

Algorithm 1 Conjugate Gradient Method

```
1: Set $k = 0$ and select initial parameter vector $\theta_0$
2: $g_{(0)} = \nabla f(\theta_{(0)}) = A\theta_{(0)} - b$
3: if $g_0 = 0$ then
4:     stop
5: else
6:     $d_0 = -g_0$
7: end if
8: $\alpha_k = -\frac{g_{(k)}^T d_{(k)}}{d_{(k)}^T A d_{(k)}}$
9: $\theta_{(k+1)} = \theta_{(k)} + \alpha_k d_{(k)}$
10: $g_{(k+1)} = \nabla f(\theta_{(k+1)})$,
11: if $g_{(k=1)} = 0$ then
12:     stop
13: end if
14: $\beta_k = \frac{g_{(k+1)}^T A d_{(k)}}{d_{(k)}^T A d_{(k)}}$
15: $d_{(k+1)} = -g_{(k+1)} + \beta_k d_{(k)}$
16: Set $k = k + 1$, go-to Step 8
```

The regression problem given in Eq. 11 can thus be reformulated into $A\theta = b$ form, where $X^T X$ is A and $b = X^T Y$.

4 Results and Discussion

This section summarizes the results of the experiments conducted using different computational techniques to find the optimal model parameters in Linear Regression. A brief description of the dataset and features used in this study are provided in Sects. 4.1, and 4.2, respectively followed by the experimental outcome in Sect. 4.3.

4.1 Boston Housing Dataset

The Boston Housing Dataset [10] consists of housing values in suburbs of Boston. The dataset has 506 instances and 13 continuous, binary valued attributes. The dataset doesn't have any missing value. More details on the dataset are given in Table 1.

Table 1 Overview of the Boston Dataset

Total number of samples	Number of features	Number of numerical features	Number of categorical variables	Number of missing features
506	13	12	1(CHAS)	0

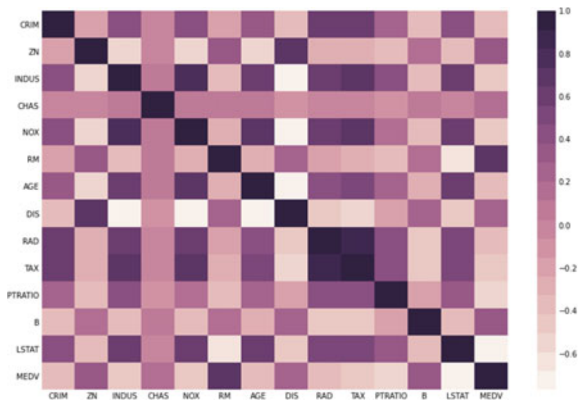

Fig. 2 Correlation Matrix of Boston House estimation dataset

4.2 Selecting Features From the Dataset

In order to find the right parameters for predicting the cost using the Boston dataset, we make use of selected features based on the correlation analysis of all the features. The correlation analysis revealed that out of 13 features, top 5 features (ZN, CHAS, RM, DIS, B) are the most important features as reflected by their high correlation values with our target variable MEDV (Median Value of owner-occupied homes in \$1000s) as shown in Fig. 2. Note that the relatively high value for the correlation is indicated by the darker shades for the features mentioned above in the row and column of the MEDV feature. We have also compared the model performance by considering all features whose results are provided in Sect. 4.3

4.3 Result Analysis

In order to compare the accuracy and computational requirements of conjugate gradient method, we do the following experiments.

- Analyzed the effectiveness of the model parameters in terms of the MSE and Norm of the final parameter vector for all the 3 methods: pseudo-inverse, batch gradient descent, and conjugate gradient method as shown in Table 2.
- Analyzed mean squared error and the number of iterations taken to converge to the right parameters using the batch gradient descent and conjugate gradient method considering:
  - 5 relevant features that show a good correlation with the target variables. The result is shown in Table 3.
  - All features as shown in Table 4.

Table 2 MSE and Norms of parameter vector found out using various models

Model	Considering 5 Features		Considering all Features	
	MSE	Norm of the parameter vector	MSE	Norm of the parameter vector
Pseudo-inverse	19.6389	24.7827	19.5794	23.5260
Batch gradient descent	21.6390	23.1947	29.4010	20.2504
Conjugate gradient method	21.5661	23.2704	20.7954	23.5237

Table 3 MSE and number of iteration for Batch and Conjugate Gradient methods considering five features from the dataset

Input vectors	Batch gradient descent		Conjugate gradient method	
	MSE	Number of iterations	MSE	Number of iterations
v1	21.6389	110	21.56	3
v2	21.6492	111	21.59	3
v3	21.62	112	21.57	3
v4	21.63	110	21.55	3
v5	21.61	113	21.47	3

Table 4 MSE and number of iteration for Batch and Conjugate Gradient methods considering all features from the dataset

Initial vector	Batch gradient descent		Conjugate gradient method	
	MSE	Number of iterations	MSE	Number of iterations
v1	29.40	39	20.80	8
v2	28.87	40	20.79	8
v3	29.27	40	20.80	8
v4	29.43	39	20.75	8
v5	28.81	40	20.76	8

In Table 2, we observed that the MSE using pseudo-inverse has the least value in both cases (i.e., considering 5 strongly correlated features and taking all features), MSE for the other 2 models have almost same values. Being the function convex, and since all methods resulted in identical norm and close MSE values, it is reasonable to believe that these methods converged to the same solution.

In Table 3, we have performed the analysis of these algorithms using five different random vector initialization. In all the test cases, when compared to the batch gradient descent, the cost function converged to the minimum in relatively less number of iterations when using the conjugate gradient algorithm. The average number of

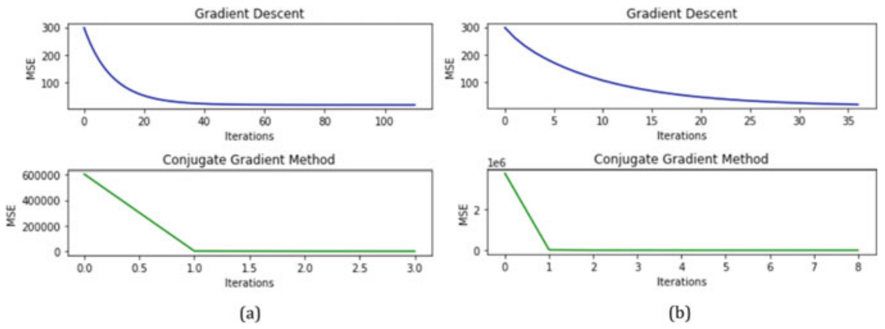

Fig. 3 **a** Graph for MSE versus Iterations (Taking most correlated features); **b** Graph for MSE versus Iterations (Taking all features)

iterations required by the batch gradient descent algorithm is 111.2 which is very high compared to the average number of iterations required by the conjugate gradient algorithms. The mean squared error obtained is also slightly less while using the conjugate gradient algorithm when compared to the batch gradient descent. Theoretically, the conjugate gradient method must converge to the optimal parameter vector in exactly 5 steps [17] for all the test cases given in Table 3, but we will get a decent solution even for lesser number of iterations. This can be seen from the results provided in Table 3, where it takes only 3 iterations to converge to the decent solution (on the validation set) irrespective of the initially chosen vector.

Similarly in Table 4, when considering all the feature vectors, we get similar results as that obtained when using only five features. The average number of iterations taken to converge to the minima is 40 when using the batch gradient descent algorithm and is 8 when using the conjugate gradient method, this confirms that conjugate gradient method takes lesser iteration when compared to gradient descent techniques irrespective of the number of features taken. As mentioned in the last paragraph, it can be proven [17] that the conjugate gradient method will take exactly "N" steps to converge to the optimal N dimensional parameter vector [17]. For the results in Table 4, we had used 14 features and the average number of iterations to find the best parameter was 8. Clearly, this did not cross 14 iterations in any trial as indicated by the theory [23].

The above results can be further confirmed by plotting a graph for number of iterations vs the cost/error values. As we can see in Fig. 3a, the cost value is minimized at around 100 iterations for batch gradient descent, whereas the conjugate gradient method provided a decent solution even with 3 iterations. This shows the efficacy of finding regression parameters using conjugate gradient method in similar settings. Similar result can be found in Fig. 3b, where we used all features in the price prediction.

5 Conclusion

In this research work, we propose to use the conjugate gradient method for finding the optimal parameters for linear regression in an ordinary least square setting. As the conjugate gradient method demands the use of symmetric positive definite matrices, we have reformulated the linear regression problem as $X^T X\theta = X^T Y$. We have then identified that it can be reposed as $A\theta = b$ where $A = X^T X$, the symmetric positive (semi) definite matrix. The manuscript provides the necessary theory, proof, and experimental results on the Boston House Price Prediction, to show the effectiveness of the conjugate gradient method in finding the optimal parameters for the linear regression model. Unlike the Pseudo-Inverse method, the proposed approach does not involve matrix inversion which is important especially when dealing with a large number of features. Contrary to gradient descent, the proposed approach converges to the N-D parameter vector in exactly "N" iterations, irrespective of the initial parameter vector. Hence, this method proves to be a faster and an effective technique to solve linear regression problems.

References

1. Bradley JK, Schapire RE (2008) Filterboost: regression and classification on large datasets. In: Platt J, Koller D, Singer Y, Roweis S (eds) Advances in neural information processing systems, vol 20. Curran Associates Inc., pp 185–192
2. Devasia T, Vinushree TP, Hegde V (2016) Prediction of students performance using educational data mining. In: 2016 international conference on data mining and advanced computing (SAPIENCE), pp. 91–95. https://doi.org/10.1109/SAPIENCE.2016.7684167
3. Fletcher R (1968) Generalized inverse methods for the best least squares solution of systems of non-linear equations. Comput J 10(4):392–399. https://doi.org/10.1093/comjnl/10.4.392
4. Fowdur T, Beeharry Y, Hurbungs V, Bassoo V, Ramnarain-Seetohul V, Lun ECM (2018) Performance analysis and implementation of an adaptive real-time weather forecasting system. Internet Things 3–4:12–33
5. Freedman D (2005) Statistical models: theory and practice. https://doi.org/10.1017/CBO9781139165495
6. Gayathri B, Sruthi K, Menon KAU (2017) Non-invasive blood glucose monitoring using near infrared spectroscopy. In: 2017 international conference on communication and signal processing (ICCSP), pp 1139–1142. https://doi.org/10.1109/ICCSP.2017.8286555
7. Gemulla R, Nijkamp E, Haas PJ, Sismanis Y (2011) Large-scale matrix factorization with distributed stochastic gradient descent. In: Proceedings of the 17th ACM SIGKDD international conference on knowledge discovery and data mining. KDD '11, Association for Computing Machinery, New York, NY, USA, pp 69–77. https://doi.org/10.1145/2020408.2020426
8. Godfrey K (1985) Simple linear regression in medical research. N Engl J Med 313(26):1629–1636. https://doi.org/10.1056/NEJM198512263132604
9. Harikumar S, Reethima R, Kaimal MR (2014) Semantic integration of heterogeneous relational schemas using multiple l1 linear regression and svd. In: 2014 international conference on data science engineering (ICDSE), pp 105–111. https://doi.org/10.1109/ICDSE.2014.6974620
10. Harrison D, Rubinfeld DL (1978) Hedonic housing prices and the demand for clean air. J Environ Econ Manag 5(1):81–102
11. Hestenes MR (1980) Conjugate gradient algorithms. in: conjugate direction methods in optimization. Springer New York, pp 231–318

12. Holcomb SD, Porter WK, Ault SV, Mao G, Wang J (2018) Overview on deepmind and its alphago zero ai. In: Proceedings of the 2018 international conference on big data and education. ICBDE '18, Association for Computing Machinery, New York, NY, USA, pp 67–71. https://doi.org/10.1145/3206157.3206174, https://doi.org/10.1145/3206157.3206174
13. Háo DN, Lesnic D (2000) The cauchy problem for laplace's equation via the conjugate gradient method. IMA J Appl Math 65(2):199–217. https://doi.org/10.1093/imamat/65.2.199
14. Jung Klaus SFHM (2017) Multiple linear regression for reconstruction of gene regulatory networks in solving cascade error problems. Adv Bioinf 94–95
15. Naveenkumar KS, Vinayakumar R, Soman KP (2019) Amrita-cen-sentidb 1: Improved twitter dataset for sentimental analysis and application of deep learning, pp 1–5. https://doi.org/10.1109/ICCCNT45670.2019.8944758
16. Kershaw DS (1977) The incomplete cholesky-conjugate gradient method for the iterative solution of systems of linear equations. J Comput Phys
17. Kershaw DS (1978) The incomplete cholesky-conjugate gradient method for the iterative solution of systems of linear equations. J Comput Phys 26(1):43–65
18. Kodiyan AA, Francis K (2019) Linear regression model for predicting medical expenses based on insurance data. https://doi.org/10.13140/RG.2.2.32478.38722
19. Loh Pl, Wainwright MJ (2011) High-dimensional regression with noisy and missing data: Provable guarantees with non-convexity. In: Shawe-Taylor J, Zemel R, Bartlett P, Pereira F, Weinberger KQ (eds) Advances in neural information processing systems, vo. 24. Curran Associates Inc., pp 2726–2734
20. Lubis FF, Rosmansyah Y, Supangkat SH (2014) Gradient descent and normal equations on cost function minimization for online predictive using linear regression with multiple variables. In: 2014 international conference on ict for smart society (ICISS), pp 202–205. https://doi.org/10.1109/ICTSS.2014.7013173
21. Luenberger DG, Ye Y (2008) Conjugate direction methods. In: Linear and nonlinear programming. Springer US, New York, pp 263–284
22. Madhuri CR, Anuradha G, Pujitha MV (2019) House price prediction using regression techniques: A comparative study. In: 2019 international conference on smart structures and systems (ICSSS), pp 1–5 . https://doi.org/10.1109/ICSSS.2019.8882834
23. Polyak B (1969) The conjugate gradient method in extremal problems. USSR Comput Math Math Phys 9(4):94–112
24. Prion S, Haerling K (2020) Making sense of methods and measurements: simple linear regression. Clin Simul Nurs 48:94–95. https://doi.org/10.1016/j.ecns.2020.07.004
25. Reddy MR, Kumar BN, Rao NM, Karthikeyan B (2020) A new approach for bias-variance analysis using regularized linear regression. In: Jain LC, Virvou M, Piuri V, Balas VE (eds) Advances in bioinformatics, multimedia, and electronics circuits and signals. Springer Singapore, Singapore, pp 35–46
26. Ruder S (2016) An overview of gradient descent optimization algorithms. arxiv:1609.04747Comment: Added derivations of AdaMax and Nadam
27. Sathyadevan S, Chaitra MA (2015) Airfoil self noise prediction using linear regression approach. In: Jain LC, Behera HS, Mandal JK, Mohapatra DP (eds) Computational intelligence in data mining, vol 2. Springer India, New Delhi, pp 551–561
28. Seal HL (1967) Studies in the history of probability and statistics. xv: The historical development of the gauss linear model. Biometrika 54(1/2):1–24
29. Sharmila Muralidharan KP (2018) Analysis and prediction of real estate prices: a case of the boston housing market. Issues Inf Syst 5:109–118
30. Weisberg S (2005) Applied Linear Regression. Wiley series in probability and statistics. Wiley, New York
31. Yan X (2009) Linear regression analysis: theory and computing. World Scientific publishing Co