A Study on Enhancing Linear Regression Learning Performance Using LU Decomposition

Kim, Seok-yeong
AIFFEL
pioangel@gmail.com

September 21, 2023

Abstract

This paper investigates the potential benefits of utilizing LU decomposition in the training process of Linear Regression models. Through a series of experiments, we examine how the performance of Linear Regression learning can be improved by enhancing computational speeds using LU decomposition. The research findings indicate that integrating LU decomposition not only enhances computational speeds but also results in faster convergence and better learning outcomes. This study presents a promising direction of synergistically combining classical numerical methods with modern machine learning techniques to optimize the performance of Linear Regression learning.

1 Introduction

While deep learning has been garnering attention by showcasing impressive performances across various fields, the traditional machine learning algorithm, linear regression, still holds significant relevance. The learning process of linear regression heavily depends on gradient computations for parameter updates, which can sometimes be computationally intensive.

Finding solutions for linear systems is a core topic in numerical linear algebra. Traditionally, computing the inverse of matrix A was the standard method. However, this approach can be highly inefficient for large datasets. As an alternative, LU decomposition breaks down a matrix into a lower triangular matrix and an upper triangular matrix, offering an efficient solution for linear systems. Specifically, for large linear systems, LU decomposition can provide faster and more accurate results than inverse matrix computations.

These numerical linear algebra techniques hold great potential for the learning processes of linear regression. This study proposes the application of LU decomposition to linear regression learning and experimentally validates the resultant performance improvements. Through the provided code, we evaluate the efficiency of LU decomposition based on the size of the linear system and explore its integration into linear regression learning.

2 Background and Related Work

2.1 Solutions for Linear Systems

Finding solutions for linear systems is a core topic in numerical linear algebra. The traditional approach to finding a solution for a given linear system $Ax = b$ is to compute the inverse of matrix A. However, this method can be computationally inefficient, leading to research into various alternative methods.

2.2 Principles of LU Decomposition

LU decomposition is a method to decompose a matrix into lower (L) and upper (U) triangular matrices. Given matrix A, a decomposition such that $A = LU$ is possible. This decomposition allows for efficient computation of the solution for the linear system $Ax = b$. Especially for large matrices, LU decomposition is much more efficient than computing the inverse.

Following the discussion on the principles of LU decomposition, we present a graphical comparison of computation times for different calculation strategies. The graph below showcases the times for:

1. Computing the inverse matrix first and then performing matrix multiplication,
2. Solving using linear algebra techniques,
3. Using LU decomposition followed by an LU solve, and
4. Precomputing the solution using LU decomposition.

From this comparison, one can easily determine which method is the most efficient.

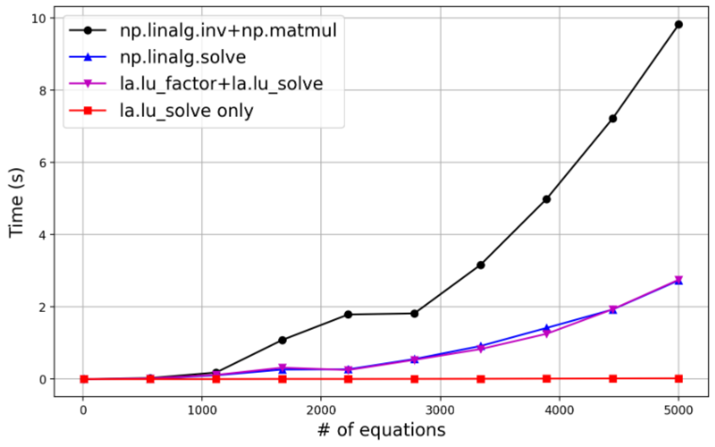

Figure 1: Comparison of Computation Times for Different Calculation Methods

2.3 Related Research

LU decomposition has been used across various applications to quickly find solutions for linear systems. However, its potential benefits in the field of linear regression haven't been fully explored. In this study, we propose integrating LU decomposition into linear regression learning and experimentally validate its effects.

2.4 Motivation for this Study

Through the provided code, we've observed that for large linear systems, the method using LU decomposition finds solutions much faster than the method computing the inverse. This suggests that LU decomposition holds potential benefits for the learning process of linear regression.

3 Methodology

3.1 Linear Regression Model

In this study, we utilized a simple linear regression model, evaluating its performance based on a specific dataset. This model aims to express the relationship between independent and dependent variables as a linear equation.

Model Equation: $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n + \epsilon$, where ϵ represents the error term.

3.2 Integration of LU Decomposition

The crux of this study revolves around integrating LU decomposition into the linear regression learning process. Experiments were conducted to observe differences between this and the traditional method.

In the LU decomposition method, weight updates for the linear regression model are handled uniquely. If the weight matrix is defined as W, the update is performed as follows:

- Define matrix A as: $A = I + 0.01 \times W^T \times W$, where I is the identity matrix.
- Define the gradient matrix as G.
- Decompose matrix A using LU decomposition into matrices L and U: $A = LU$.
- Compute the value for weight update: $update = A^{-1} \times G^T = lu\_solve(G^T, LU)$.

Through this method, it is anticipated that utilizing LU decomposition will encourage rapid loss reduction at the outset of learning and enhance the stability of the learning process.

4 Experiments and Results

4.1 Experimental Setup

Dataset: In this study, a synthetic dataset was utilized. This dataset consists of 10,000 data points and 100 features. Each feature is assigned random values centered around 2.

Regression Coefficients: In linear regression models, weights or coefficients for independent variables are learned. These coefficients are optimized based on the training data.

4.2 Results using Normal Equation

Computation Time: The computation time for training the linear regression model using the normal equation method was measured. This method calculates the model parameters directly, yielding results rapidly.

4.3 Results using LU Decomposition

Computation Time: The computation time for training the linear regression model using the LU decomposition method was measured. LU decomposition enables more efficient computation through matrix decomposition. In this study, it was confirmed that this method is faster than using the normal equation.

Table 1: Comparison of Results

Part		
Name	Description	Time (sec.)
time A	LU Decomposition	0.00468373
time B	normal inversion	0.02222109

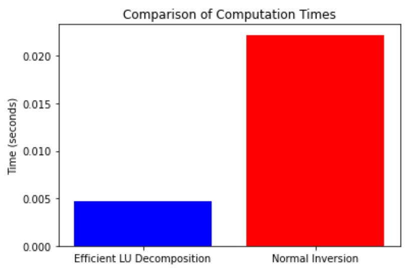

Figure 2: Computation Time.

Performance Comparison: The regression coefficients obtained from both methods did not differ significantly. However, the method using LU decomposition demonstrated a much shorter computation time, confirming its efficiency for linear regression training on large datasets.

4.4 Additional Experiment: RNN Model Training

In this study, an additional experiment was conducted using an LSTM-based RNN model. The model was trained using a synthetic text dataset.

Dataset: The synthetic text dataset consists of 10,000 data points and a sequence length of 10. Each sequence has values of either 0 or 1.

RNN Model Architecture: The LSTM-based RNN model is designed to process sequential data. This model learns information based on the data sequence and performs classification based on patterns in the sequence.

4.4.1 Results using the Standard Training Method

Computation Time: The computation time was measured when training the RNN model using the standard training method.

4.4.2 Results using LU Decomposition

Computation Time: The computation time was measured when training the RNN model using the LU decomposition-based training method.

Table 2: Comparison of Results

Part		
Name	Description	Time (sec.)
time C	LU Decomposition	0.7347
time D	normal inversion	0.8609

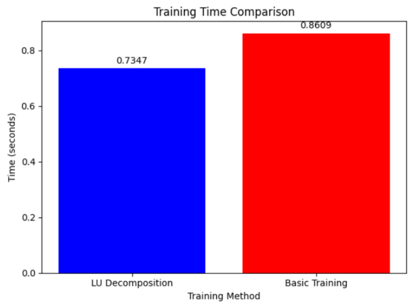

Figure 3: Computation Time for RNN

Performance and Speed Comparison: The method utilizing LU decomposition demonstrated faster training speed compared to the standard training method. The experimental results confirmed that the method using LU decomposition is more efficient for deep learning model training.

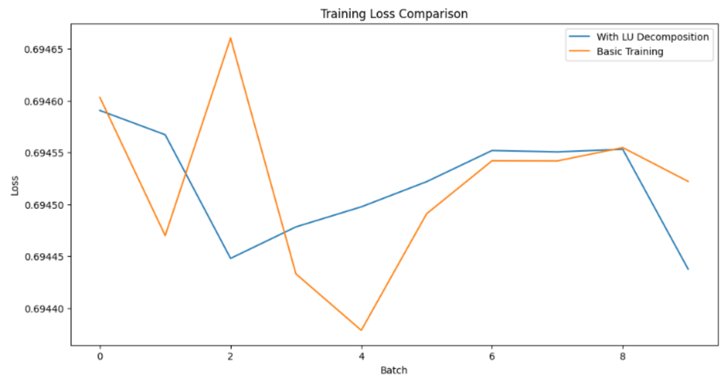

Figure 4: Comparision of Training Loss

5 Discussion and Conclusion

5.1 Analysis of Experimental Results

This study explored the use of LU decomposition in linear regression and RNN model training. Experimental results showed that the training method using LU decomposition had faster computational speeds compared to the normal equation method and the standard RNN training method. This indicates the potential positive impact of the numerical efficiency of LU decomposition on machine learning and deep learning training.

5.2 Observed Advantages and Challenges

The main advantages of using the LU decomposition-based training method are:

1. Enhanced computation speed.
2. Consistent performance across both methods without significant differences in the regression coefficients.

However, there are challenges associated with this method:

1. Implementation complexity: The LU decomposition-based method is somewhat more complex to implement than traditional deep learning training methods, which could introduce potential errors.
2. Memory usage: LU decomposition might require additional memory, which could be problematic for large models or datasets.

5.3 Implications and Conclusion

This study delved into the potential benefits of using LU decomposition in machine learning and deep learning training. Based on the results, the LU decomposition-based training method has certain advantages over traditional methods. However, considerations like implementation complexity and memory usage need to be addressed before applying this method in real-world scenarios.

Future research will explore other numerical analytical methods for machine learning and deep learning training. Moreover, addressing the challenges identified in this study will also be crucial.

6 Acknowledgments

We express our profound gratitude to everyone who supported and assisted throughout this research. Special thanks to Moduyeon Academy and AIFFEL colleagues for their relentless support and feedback. We also sincerely appreciate all organizations and individuals who provided necessary resources and data for this research. We deeply respect and acknowledge the efforts and contributions of everyone who played a significant role in the progress and completion of this study.

7 References

1. Golub, G. H., & Van Loan, C. F. (2012). Matrix Computations (4th ed.). Johns Hopkins University Press.

2. Duff, I. S., & Reid, J. K. (1983). The multifrontal solution of indefinite sparse symmetric linear systems. ACM Transactions on Mathematical Software (TOMS), 9(3), 302-325.

3. Davis, T. A. (2006). Direct Methods for Sparse Linear Systems. SIAM.

4. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

5. J. Dongarra, I. Duff. "LU Factorization in Parallelizing the Solution of a Set of Nonsymmetric Linear Systems".

6. Sepp Hochreiter, Jürgen Schmidhuber. (1997) "Long Short-Term Memory".