Polynomial Regression with Adam Optimisation Algorithm

Wenbo Duan[†]

[†] *Department of Electrical and Electronic Engineering*
pv19120@bristol.ac.uk

December 17, 2021

Abstract

In this mini-research project, a new nonlinear optimization algorithm *Adam* was studied and deployed in a house pricing perdition problem. We first analyzed the mathematical meanings behind the algorithm and introduced an application on a house price estimation problem. From the training of a polynomial loss function with the Boston house price data set, we obtained a two factors estimation function. The analysis of the loss curve and test data data set shows a successful prediction, but a more nonlinear relationship could be considered in the future to get a more precise result.

1 Introduction

Among many variants of gradient-based algorithms, different ways of route searching could lead to different problems in terms of convergence speed and accuracy. Steepest Descent (SD) takes local gradient as the direction vector at each point while this lead to a serious oscillatory trajectory as it approaches to the minimum; the Conjugate Gradient (CG) method overcomes this limitation by retaining a component of the direction vector parallel to the previous search direction, but the Hessian Matrix that used to determine the consecutive search direction, on the other hand, demonstrates that it should stay unchanged to keep the conjugation of the search direction, and hence, it is not amenable to a stochastic approximation of the gradient [4]. However, in comparison to deterministic descent, stochastic/mini-batch gradient descent is more computationally efficient for large-scale data and high-dimensional objective function.

In stochastic/mini-batch gradient descent, Momentum [3] is a technique used to relieve oscillations by computing the exponentially weighted average of the gradient to update the current position. Additionally, the RMSProp [1] algorithm proposed an adaptive technique that conducts greater updates for infrequent parameters and smaller updates for frequent parameters. In 2015, the Adaptive Moment Estimation method Adam [2] was developed, which combines the advantages of the Momentum and RMSprop approaches and quickly became one of the most popular optimization methods in the deep learning community.

In this mini-research project, the author investigated the evolution of the Adam algorithm and had a 'vanilla' trial on a Supervised Learning problem. Concretely, with the data sets of the Boston house prices from the Kaggle website, the author aims to formulate a house price prediction function and optimize it by the self-build Adam optimizer. The following section will start with the Adam algorithms analysis, introducing the process of optimizing the house price prediction function and end with comparison of the experiment result.

2 Algorithm

2.1 Momentum

Suppose in the stochastic gradient descent, the objective function is $J(\theta)$. The way to update the position θ on this scalar field to is take the local gradient as the position vector and use the predefined step length α at each iteration to find the converge point, where at time t:

$$\theta_t = \theta_{t-1} - \alpha \nabla_\theta (J(\theta_{t-1})) \tag{1}$$

However, the ill-conditioned Hessian matrices usually result in the high curvature of the scalar field, leading to the violent oscillation but slow convergence speed during the line search. Momentum method, proposed in 1985, uses exponential weighted average of the previous gradient to smooth the direction vector:

$$v_t = \beta_1 v_{t-1} + \alpha \nabla_\theta (J(\theta_{t-1})) \tag{2}$$

Where β is a friction coefficient which is usually set as 0.9, and therefore, the position at time t is:

$$\begin{aligned} \theta_t &= \theta_{t-1} - v_{t-1} \\ &= \theta_{t-1} - \alpha \sum_{i=0}^{t} \beta_1^{t-i} \nabla_\theta J(\theta_i) \end{aligned} \tag{3}$$

2.2 RMSProp

Apart from using Momentum method to relieve oscillation and accelerate convergence, RMSprop method achieves the similar result by properly tailoring the step lengths for parameters in different dimensions. To smooth the violent oscillation, the RMSProp method first calculate the moving average weighting on the square of the gradient, where the 'smoothed' gradient for a given dimension is:

$$\begin{aligned} s_t &= \beta_2 s_{t-1} + (1-\beta_2) \nabla_\theta (J(\theta_{t-1})) \odot \nabla_\theta (J(\theta_{t-1})) \\ &= \sum_{i=1}^{t} \beta_2^{t-i} \nabla_\theta (J(\theta_{t-1})) \odot \nabla_\theta (J(\theta_{t-1})) \end{aligned} \tag{4}$$

In which $\odot$ is the element-wise product for two matrices, $s_0 = 0$ and the decaying coefficient β is usually set as 0.9. As an adaptive method, the root of s_t is then applied as the denominator to normalize the step length α , increasing the step length for small gradient and decreasing the step length for big gradient:

$$\theta_t = \theta_{t-1} - \frac{\alpha}{\sqrt{s_t} + \epsilon} \odot \nabla_\theta (J(\theta_{t-1})) \tag{5}$$

Where ϵ is a small constant to avoid the zero denominator, which is usually set as $10e-8$

2.3 Adam

RMSProp method achieves the adaptive step length for different dimensions by normalizing the step length using s_t, the root of the exponentially decaying average of past squared gradients; Momentum smooths the current direction vector by considering v_t, the previous gradients using the moving exponential weighted average of past gradients. From the statistic prospect, as the gradient is the stochastic variable, v_t and s_t could be treated as the estimation of the first moment and second moment of gradient g by using the exponential moving average, where the first moment evaluation v_t controls the direction vector and second moment evaluation s_t controls the step length at time t. In Adam algorithm, both of estimation were leveraged to achieve a better optimizing performance, where:

$$\begin{aligned} v_t &= \beta_1 v_{t-1} + (1-\beta_1) \nabla_\theta (J(\theta_{t-1})) \\ s_t &= \beta_2 s_{t-1} + (1-\beta_2) \nabla_\theta (J(\theta_{t-1})) \odot \nabla_\theta (J(\theta_{t-1})) \end{aligned} \tag{6}$$

However, at the initial time steps, the author of Adam pointed out that both v_t and s_t are observed to be biased towards 0 as they initialised as the vectors of 0. [2] Hence, the bias-correction of the first and second moment estimation were used, in which:

$$\begin{aligned} \hat{v_t} &= \frac{v_t}{1-\beta_1^t} \\ \hat{s_t} &= \frac{s_t}{1-\beta_2^t} \end{aligned} \tag{7}$$

Finally, the update rule of Adam is obtained by inserting equation (7):

$$\theta_t = \theta_{t-1} - \frac{\alpha}{\sqrt{\hat{s_t}} + \epsilon} \hat{v_t} \tag{8}$$

For a complete optimization iteration, the algorithms can be implemented as the follow steps:

Algorithm 1 Adam

```
Require: $\alpha$: Step size
Require: $\beta_1, \beta_2 \in [0, 1)$
Require: $J(\theta)$: Stochastic objective function $\theta$
Require: $\theta_0$: Initial parameter vector
  $v_0 \leftarrow 0$
  $s_0 \leftarrow 0$
  $t \leftarrow 0$
  while $\theta_t$ not converged do
    $t + 1$
    $g_t \leftarrow \nabla_\theta J_t(\theta_{t-1})$
    $v_t \leftarrow \beta_1 \cdot v_{t-1} + (1 - \beta_1) \cdot g_t$
    $s_t \leftarrow \beta_2 \cdot s_{t-1} + (1 - \beta_2) \cdot g_t^2$
    $\hat{v_t} \leftarrow \frac{v_t}{1-\beta_1^t}$
    $\hat{s_t} \leftarrow \frac{s_t}{1-\beta_2^t}$
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \frac{\hat{v_t}}{\sqrt{\hat{s_t}}+\epsilon}$
  end while
  return $\theta_t$
```

3 Application

3.1 Problem Definition

Collected by the U.S. Census Bureau in the 1970s, Boston house price contains 456 cases and each case counts 13 indicators including the crime rate per capita in localtowns and the proportion of non-retail businesses. The 14th feature gives the median price of housing. We want to find a linear relationship between the house price and some of the indicators, and therefore, be able to forecast the house price with the new released indicators. The object could be expressed as the equation:

$$h_\theta(x) = \theta^T x \tag{9}$$

Where θ is a set of coefficient for different indicators. By obtaining a set of optimal coefficients can we use the hypothesis function to predict the house price with the given indicators. To evaluate the goodness of the coefficient set, a common way is to compare the predict price ($h_\theta(x)$) and the actual price y by the mead square error in a total number of m, where:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} [h_\theta(x^i) - y^i]^2 \tag{10}$$

Since equation 10 is a quadratic polynomial, it can be considered as a convex scalar field with the coefficients of house price indicator as the independent variables. Therefore, The optimal coefficients would be obtained as long as the converge point was found.

3.2 Experiment

Considering the amount of the data sets, we use the batch gradient descent with Adam algorithm to optimise the objective function $J(\theta)$. Prior to the experiment, the gradient of the object function could be derived as a linear function, supposing the coefficient for the jth price indicator is θ_j and the number of coefficients is n:

For j=0:

$$\begin{aligned}\frac{\partial}{\partial\theta_0}J(\theta) &= \frac{1}{2m}\frac{\partial}{\partial\theta_0}\sum_{i=1}^{m}[\theta_0+\theta_1 x^i+\dots\theta_n x^i-y^i]^2 \\ &= \frac{2}{2m}\sum_{i=1}^{m}[\theta_0+\theta_1 x^i+\dots\theta_n x^i-y^i] \\ &= \frac{1}{m}\sum_{i=1}^{m}[h_\theta(x^i)-y^i]\end{aligned} \tag{11}$$

For j$\neq$ 0:

$$\begin{aligned}\frac{\partial}{\partial\theta_0}J(\theta) &= \frac{1}{2m}\frac{\partial}{\partial\theta_0}\sum_{i=1}^{m}[\theta_0+\theta_1 x^i+\dots\theta_n x^i-y^i]^2 \\ &= \frac{2}{2m}\sum_{i=1}^{m}[\theta_0+\theta_1 x^i+\dots\theta_n x^i-y^i]\cdot x^i \\ &= \frac{1}{m}\sum_{i=1}^{m}[h_\theta(x^i)-y]\cdot x^i\end{aligned} \tag{12}$$

Considering the fact that not all the indicators will have a linear relation with the price, we need to reduce the number of coefficients from the given indicator. By visualizing all data through a scatter plot we got:

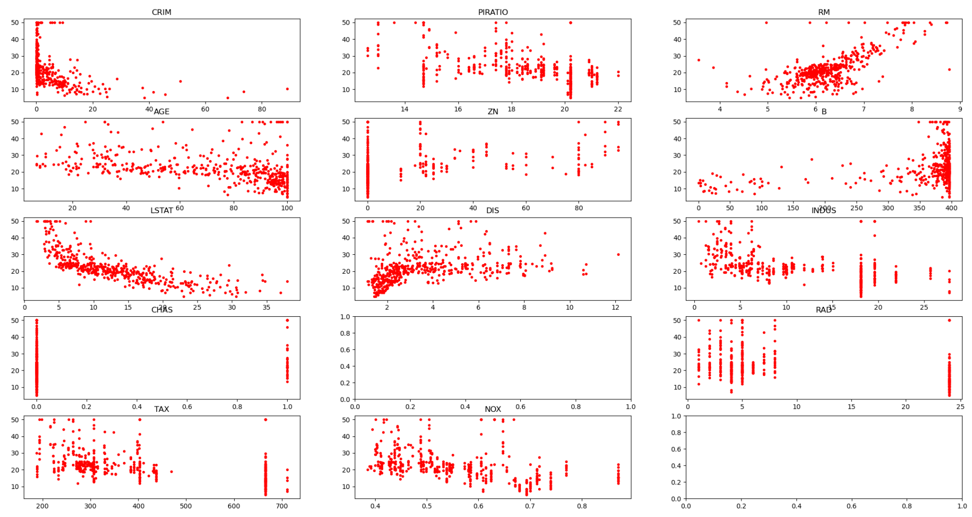


Figure 1: Scatter Plot of different indicators versus price

By observing the scatter plot, the indicator of 'LSTAT' , which is on the (3,1) of the graph, and 'RM', which is on the (1,3) of the graph are found to be linearly related to the house price. Therefore, the number of the indicators is reduced to 3 (with a constant).

During the data preparation stage, to test the accuracy of the estimate function, we separated 50 pieces of data as the test set before training, keeping the rest of data as the training set. Considering the total number of the training set, we loaded the whole data set as a batch to optimise the objective function. Additionally, the whole data sets were normalized and re-scaled to the range [0,1] to achieve a high efficient line search. All the hyper parameters were set as the author of Adam recommendation, where $\beta_1 = 0.8$, $\beta_2 = 0.999$, $\alpha = 0.02$, $\epsilon = 1e-8$ and the number of iteration is set as 1000.

4 Result

To evaluate the result, we first record the change of the $J(\theta)$ during the iteration. The smaller the value indicates a better converge point. As the Fig.2 shows: Due to the high efficiency of Adam algo-

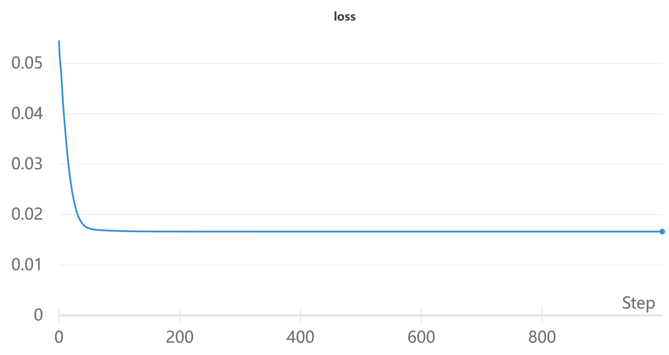
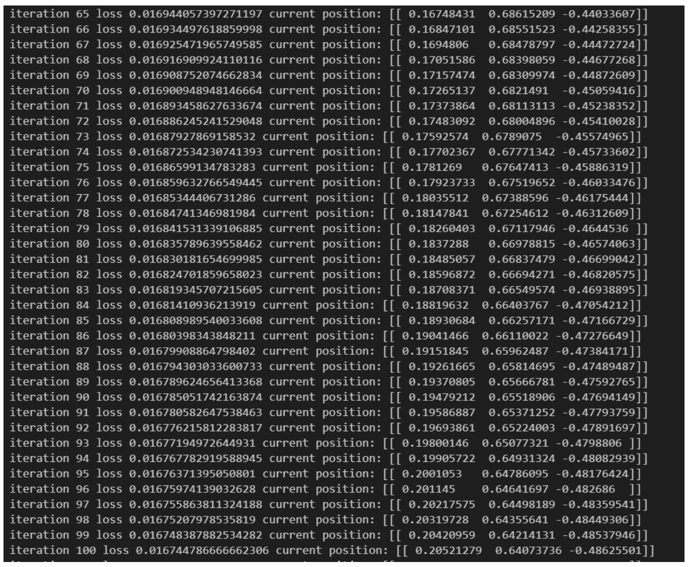


Figure 2: Values of $J(\theta)$ during the iteration

rithm and the simple objective function, the converge point is found in the first 100 iteration. As the



```
iteration 65 loss 0.016944057397271197 current position: [[ 0.16748431  0.68615209 -0.44033607]]
iteration 66 loss 0.016934497618859998 current position: [[ 0.16847101  0.68551523 -0.44258355]]
iteration 67 loss 0.016925471965749585 current position: [[ 0.1694806   0.68478797 -0.44472724]]
iteration 68 loss 0.016916909924110116 current position: [[ 0.17051586  0.68398059 -0.44677268]]
iteration 69 loss 0.016908752074662834 current position: [[ 0.17157474  0.68309974 -0.44872609]]
iteration 70 loss 0.016900948948146664 current position: [[ 0.17265137  0.6821491  -0.45059416]]
iteration 71 loss 0.016893458627633674 current position: [[ 0.17373864  0.68113113 -0.45238352]]
iteration 72 loss 0.016886245241529048 current position: [[ 0.17483092  0.68004896 -0.45410028]]
iteration 73 loss 0.01687927869158532 current position: [[ 0.17592574  0.6789075  -0.45574965]]
iteration 74 loss 0.016872534230741393 current position: [[ 0.17702367  0.67771342 -0.45733602]]
iteration 75 loss 0.01686599134783283 current position: [[ 0.1781269   0.67647413 -0.45886319]]
iteration 76 loss 0.016859632766549445 current position: [[ 0.17923733  0.67519652 -0.46033476]]
iteration 77 loss 0.01685344406731286 current position: [[ 0.18035512  0.67388596 -0.46175444]]
iteration 78 loss 0.01684741346981984 current position: [[ 0.18147841  0.67254612 -0.46312609]]
iteration 79 loss 0.016841531339106885 current position: [[ 0.18260403  0.67117946 -0.4644536 ]]
iteration 80 loss 0.016835789639558462 current position: [[ 0.1837288   0.66978815 -0.46574063]]
iteration 81 loss 0.016830181654699985 current position: [[ 0.18485057  0.66837479 -0.46699042]]
iteration 82 loss 0.016824701859658023 current position: [[ 0.18596872  0.66694271 -0.46820575]]
iteration 83 loss 0.016819345707215605 current position: [[ 0.18708371  0.66549574 -0.46938895]]
iteration 84 loss 0.01681410936213919 current position: [[ 0.18819632  0.66403767 -0.47054212]]
iteration 85 loss 0.016808989540033608 current position: [[ 0.18930684  0.66257171 -0.47166729]]
iteration 86 loss 0.01680398343848211 current position: [[ 0.19041466  0.66110022 -0.47276649]]
iteration 87 loss 0.01679908864798402 current position: [[ 0.19151845  0.65962487 -0.47384171]]
iteration 88 loss 0.016794303033600733 current position: [[ 0.19261665  0.65814695 -0.47489487]]
iteration 89 loss 0.016789624656413368 current position: [[ 0.19370805  0.65666781 -0.47592765]]
iteration 90 loss 0.016785051742163874 current position: [[ 0.19479212  0.65518906 -0.47694149]]
iteration 91 loss 0.016780582647538463 current position: [[ 0.19586887  0.65371252 -0.47793759]]
iteration 92 loss 0.016776215812283817 current position: [[ 0.19693861  0.65224003 -0.47891697]]
iteration 93 loss 0.0167719497264931 current position: [[ 0.19800146  0.65077321 -0.4798806 ]]
iteration 94 loss 0.016767782919588945 current position: [[ 0.19905722  0.64931324 -0.48082939]]
iteration 95 loss 0.01676371395050801 current position: [[ 0.2001053   0.64786095 -0.48176424]]
iteration 96 loss 0.01675974139032628 current position: [[ 0.201145    0.64641697 -0.482686  ]]
iteration 97 loss 0.016755863811324188 current position: [[ 0.20217575  0.64498189 -0.48359541]]
iteration 98 loss 0.01675207978535819 current position: [[ 0.20319728  0.64355641 -0.48449306]]
iteration 99 loss 0.016748387882534282 current position: [[ 0.20420959  0.64214131 -0.48537946]]
iteration 100 loss 0.016744786666662306 current position: [[ 0.20521279  0.64073736 -0.48625501]]
```

Figure 3: Optimisation Log

optimisation log shows, the coefficients at the 100 iteration step are $[0.20521279, 0.64073736, -0.48625501]$, mapping to the constant coefficient, RM, and LSTAT . And hence, the price estimate function could be written as:

$$estimate\ house\ price\ h = 0.20521279 + 0.64073736 \cdot \theta_{RM} - -0.48625501 \cdot \theta_{LSTAT} \tag{13}$$

To test the accuracy, the obtained estimation function is then applied to the 50 pieces of test data, which were not used in the line search:

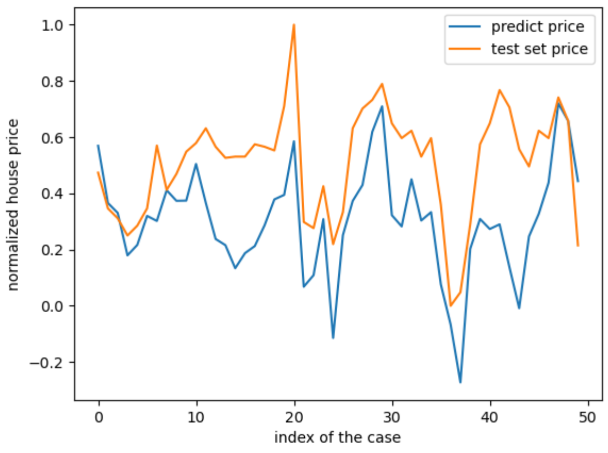


Figure 4: Comparison between estimate price and actual price in the test set

As Fig. 4 shows, both 50 predict price and the actual price are plotted on the graph. The similar changing trend and extreme values indicate that the estimation function could make a reasonable prediction according to the given parameters. However, it is clear that this is still a vague prediction. In the future, the nonlinear relationship and more indicators could be studied to estimate the price more precisely.

References

[1] G. Hinton. Lecture slides - cs.toronto.edu.

[2] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2017.

[3] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, 4(5):1–17, 1964.

[4] N. N. Schraudolph and T. Graepel. Towards stochastic conjugate gradient methods. In *Proceedings of the 9th International Conference on Neural Information Processing, 2002. ICONIP'02.*, volume 2, pages 853–856. IEEE, 2002.