Starting Right: The impact of random distribution sampling of initial parameters for curve fitting learning curves

Darie Dan-Vlad

Supervisor(s): Tom Viering[1], Cheng Yan[1], Taylan Turan[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
January 26, 2025

Name of the student: Darie Dan-Vlad
Final project course: CSE3000 Research Project
Thesis committee: Tom Viering, Arie van Deursen, Cheng Yan, Taylan Turan

Abstract

Learning curves are used to evaluate the performance of a machine learning (ML) model with respect to the amount of data used when training. Curve fitting finds the unknown optimal coefficients by minimizing the error prediction for a learning curve. This research analyzed the effect of parameter initialization on the performance of curve fitting. Our focus was on comparing the performance of sampling the initial parameters from 2 random distributions: **uniform** and **normal** on the curve fitting process for different parametric models. Moreover, we looked into the effect of changing the parameters for these 2 random distributions and drew conclusions about potential best initial guesses.

Finally, we arrived at the conclusion that, after choosing parameters that maintain similar data distribution, uniform and normal distribution sampling parameter initializations perform similarly during the curve-fitting process on learning curves. Moreover, our studies highlight the sensitivity of the Levenberg-Maquardt curve fitting method's sensitivity to bad initial guesses.

1 Introduction

Learning curves illustrate how a system's performance on a task evolves as a function of the resources devoted to solving it [1]. These resources can vary from case to case, such as the time spent by the system interacting with an environment or the number of iterations the learner spent in that environment. Performance is a term used for the measure that successfully captures how well the system's task is done. This is usually assessed using an error rate metric. This concept has multiple applications, from comparing different learners to determining the optimal data used to successfully train a learner. In our case, we are interested in training sets' sizes learning curves, which plot the number of training instances against the error rate. Our work will focus on a particularity of the curve fitting process. Curve fitting is a mathematical technique that seeks to find the most suitable function to represent a given set of data points, effectively modeling a real-world phenomena, and providing a comprehensive description of their behavior [2]. The objective of this is to establish a clear relationship between dependent and independent variables. Despite many previous works on learning curves such as [3],[4], there is not much publicly available information on curve fitting and the potential effects of the data on the curve fitting process. A question that hasn't been explored yet is how sampling the initial parameters of the objective function from a random distribution impacts the curve-fitting process. This paper aims to tackle this question, by assessing the performance difference between sampling from different distributions and aims to optimize the initial guesses using random distribution for better performances.

1.1 Related work

Our work is based upon previous relevant works such as [4], [3], [5]. Viering and Loog [3] show multiple ways on how to estimate learning curves, how to plot them and finally information about curve fitting. Firstly, it is suggested not including a bias term in a parametric model as a pitfall, indicating us that we should run our experiments also on more complex parametric models. In pitfall E, it is suggested that extrapolation and interpolation should be done on previously **unseen** data and that mean squared error is more suitable for non-linear curve fitting as an error metric. Moreover, this paper exhibits the previous empirical learning curve fitting studies, section relevant for finding out what model functions would be used best for our experiment. However, the conclusion is that there may not exist an universal parametric model, and that the best parametric model depends on the shape of the learning curve. Studies concerning basic relations between parameters for the exponential and power functions and their conclusions are presented. Finally, a section on how to robustly fit learning curves is presented, arriving at the conclusion that there are no studies about probabilistic models that closely match the data of the learning curve.

Mohr et al [4] introduces the Learning Curves Database, a database storing data of learning curves for 24 classification algorithms in 246 datasets. Besides important concepts used in curve fitting like interpolation and extrapolation, the performance of parametric model is compared, arriving at the conclusion that the more complex and more parameters the model has, the better will be its performance. Moreover, Mohr [4] states that objective model functions similar to the power function seem to obtain the lowest error after curve fitting. In our experiments, we will use learning curves only from the Learning Curve Database.

Kim et al [5] focuses on curve fitting techniques for learning curves and the optimization of initial parameters using K-means clustering. Moreover, it explains the difference between curve fitting methods such as Levenberg-Marquadt and Newton, 2 local optimization methods. He arrives at the conclusion that despite simple function models being easier to fit, they also get worse extrapolation values. Despite it being very critical about the results obtained and proving that KMI initialization is better than random initialization, the experiments were not done for a considerable number of iterations. Moreover, for the random intialization tests it uses only the random distribution, which is equivalent to the uniform distribution and it does not specify the parameters used for sampling, which we assume it means it only used the default parameters. This can cause initial bad guesses, which result in suboptimal fits.

Bhaskaran et al [6] studies the effect of hyper parameter tuning on learning curves. One problem he investigates is the difference between hyper parameter tuned models against default models with regards to the process of curve fitting. He arrives at the conclusion that there are not significant differences in terms of extrapolation between the tuned and the default version of the models. It critically shows the results obtained for multiple objective functions on 6 anchor percentages, concluding that the the parametric curve fits of the default and tuned learner perform quite similarly. However,

the experiments were done on KNN classifier and Decision Trees.

Gavin et al [7] defines the **Levenberg Marquardt** method, defining the 2 algorithms used in it, gradient descent and Newton method. It is stated that in non-linear least squares problems, this algorithm is sensitive to initial guesses, fact which will help us arrive at conclusions of our experiments.

1.2 Research questions

My main research question is "How does the distribution sampling of the initial parameters of the model function affect the performance of curve fitting in the context of well-behaved learning curves?". The focus of this study will be on the effects of setting random distributions, such as normal and uniform, to the initial parameters of curve fitting function using Levenberg-Marquadt method [7]. In order to answer this, I will split this question into multiple subquestions:

1. What is the performance difference between uniform and normal distribution sampling of initial parameters in LM algorithm curve fitting?
2. How does increasing the upper bound in uniform distribution sampling impact LM algorithm in curve fitting?
3. How does increasing the mean and standard deviation in normal distribution sampling impact LM algorithm in curve fitting?
4. How does tuning random distributions' parameters with regards to the fitting data used impact LM algorithm in curve fitting ?

1.3 Overview

In the following sections, we will discuss the motivations behind this study in 2, my contributions and ideas in 3 and 5, the experimental setup needed to replicate the experiments in 4 and the results in both conclusion 7 and experimental results and discussions 5 section. Moreover, further steps in order to arrive at more conclusions will be hinted in detail in 5. In section 8, it will be demonstrated that our research experiment and paper comply with the principles of responsible research.

2 Background

In an evolving machine learning environment, models are becoming more and more complex, therefore many may come to the conclusion that more training data used would improve the performance of the trained model. However, this is not necessarily true. In the context of machine learning, learning curves plot the error rate of a model against the number of training sizes used. This can be used in order to derive the optimal number for the training size used for that model. Well-behaved learning curves are curves that show improved performance with more data [3]. An example of well-behaved learning curve can be seen in figure 1.

2.1 Curve fitting

In the context of learning curves, curve fitting aims to optimize the objective function's parameters in order to minimize

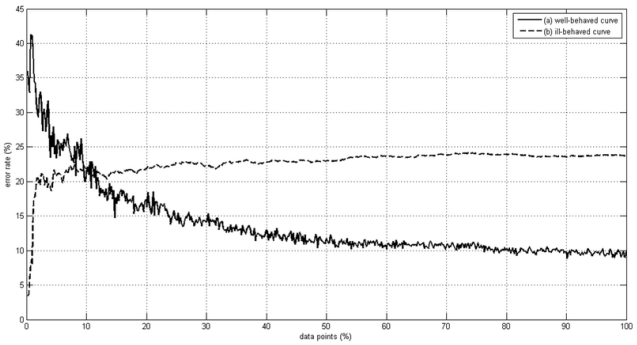

Figure 1: Example of well-behaved vs ill-behaved learning curve from [8]

the error between the predicted learning curve and the actual learning curve. Mathematically, this is expressed as:

$$\min_{\theta} \sum_{i=1}^{n} \left(y_i - f(x_i;\theta)\right)^2, \tag{1}$$

where: x_i and y_i are the training set sizes and error rate r, respectively, $f(x_i;\theta)$ is the parametric model used to fit the data, θ represents the set of parameters to be optimized, n is the total number of data points used for fitting and $\left(y_i - f(x_i;\theta)\right)^2$ is the squared error for the i-th data point. Therefore, our initial assumption is that the initial values used for the parameters of the objective function play an important role in the final values obtained after applying the curve fit function. The most common method used for optimizing the parameters is **Levenberg-Marquardt algorithm** [7], an algorithm which uses gradient descent and Gauss-newton minimization methods.

2.2 Random distribution sampling

In order to test our hypotheses in an objective manner, we will initialize the initial parameters of the curve fit function with random values sampled from **normal** and **uniform** distribution. When sampling from a uniform distribution, all the values from the 2 intervals are equally probable to be sampled[9]. The default parameters of the uniform distribution sampling used in python are [0,1] where the first value is the lower bound of the interval and the second value is the upper bound of the interval. In contrast, the normal distribution samples values with regards to the mean and standard deviation, values closer to the mean having a higher probability of being selected. In figure 2, we can see a visual representation of the normal distribution with the mean 0 and the standard deviation 1, showing that most of the values will be drawn from the interval [-3,3], with the probability of a value being drawn being higher if the value is closer to the mean. Gonzales et al [10] states that 97% of the normal distribution's sampled values are in the interval [mean-3*standard deviation,mean+3*standard deviation]. The default python function used for normal distribution uses the parameters of the standard normal distribution sampling. However, because of

the the range of sampling for the normal distribution, it is possible that negative values are also sampled, which is not the case for uniform distribution.

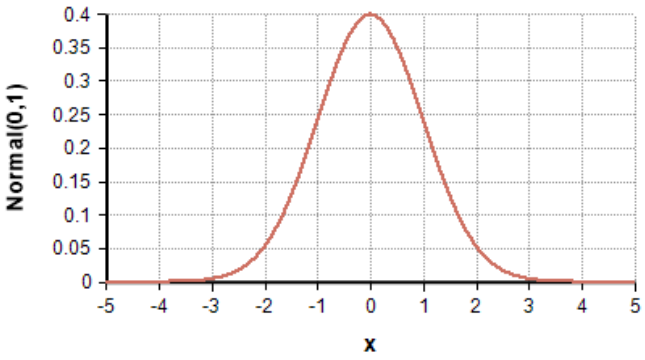

Figure 2: Normal Distribution Curve with mean 0 and standard deviation 1 from [11]

3 Methodology

In this section we will present our methodology and the initial assumptions we derived. Firstly, we will explain the error function we are using and on which part of the data it will be used. Then we will explain the way we decided to tackle each subquestion and explain what results we initially expected.

3.1 Interpolation and extrapolation

In the context of curve fitting for machine learning, **interpolation** estimates model performance within observed training data to validate the fit, while **extrapolation** predicts performance within unseen data. While we will initially look on how the error on unseen data differs so that it does not overfit. However, we will assess the error on the data used for fitting, in case the results for extrapolation are similar. We will consider good fits those for which curve fitting correctly describes the behavior of the curve on both seen and unseen data.

The measure used to compute the interpolation and extrapolation error is the mean squared error [12], a method generally used for generalisation problems. In the following sections, we may abbreviate it as **'MSE'**.

3.2 Models used for curve fitting

In table 1, we can find the mathematical formulas of the objective functions described. The models used for our experiments are the exponential and power functions, 3 parameters for the first experiment and with 2 parameters for the other 2 subquestions. Viering et al[3] suggests that well-behaved learning curves favor exponential and power-law shapes, meaning that for our example these 2 objective function would perform best.In the beggining, we will make assumptions about the 2-parametric models, in order to assess potential differences for functions which may not always yield a curve fit. However, Viering et [3] suggests that not including the bias term in the parametric model is a pitfall, displaying the example of failed fits of EXP2 function on exponential shaped curves.

Table 1: Formulas for Selected Objective Model Functions

Model	Formula
POW2	an^{-b}
POW3	$an^{-b} + c$
EXP2	$a\exp(-bn)$
EXP3	$a\exp(-bn) + c$

3.3 Levenberg-Marquardt

Gavin et al [7] states that Levenberg-Marquardt algorithm combines two numerical minimization algorithms: the gradient descent method and the Gauss-Newton method. This method method acts more like a gradient-descent method when the coefficients are far from their optimal value, and acts more like the Gauss-Newton method when the coefficients are close to their optimal value. The gradient descent method uses gradient descent method, which updates coefficient values in the "downhill" direction. The Gauss-Newton method is a method for minimizing a sum-of-squares objective function. It presumes that the objective function is approximately quadratic in the coefficients and typically converges faster than gradient-descent methods. Finally, Gavin et al [7] highlights the limitations of the method, stating that the algorithm is sensitive to initial guesses.

3.4 Initial assumptions about the differences between the effect of distributions

We believe that normal distribution with default parameters sampling produces worse results especially for these objective functions. This is due to the probability of the initial value of the second and first parameter being negative, resulting in an negative or error above the threshold. In order to completely investigate our hypothesis, we will run our pipeline on uniform distribution with a negative lower bound, in order to compare its performance to the normal distribution sampling with the mean 0. Viering et al[3] states that if the initial quantity parameter 'a' is negative, then we can also set the decay 'b' to a value negative, we can still get non-negative values until a certain value of n, for 3 parametric model. This results in us potentially getting better results than expected for the interpolation of 3 parametric models for normal distribution, since there exists the probability of 0.25 of sampling 2 random negative values for 'a' and 'b' parameters. Therefore, we expect the results for normal distribution sampling for at least POW3 to not be that relevant for interpolation. Finally, due to the curve fitting sensitivity to initial bad guesses, we expect that most of the curve fitting runs for each learning curve to not find the optimal value and to be stuck at a local minimum and the convergence matrix to not be able to be approximated. In order to find the most optimal values, we will try to increase the probability of getting a reliable initial guesses. Firstly,we thought that for uniform distribution, we will increase the upper bound from 1 to a larger value. For normal values, we will also increase the mean/peak of the normal distribution and choose a standard deviation such

that the probability of getting positive values would be close to 100 percent.

Kim et al [5] suggests values for bounds of parameters for the exponential function with 3 parameters, with the lower bound being -10 for all of the 3 parameters and the upper bound 100. However, from our experiments, we believe that a negative lower bound would affect negatively the performance of curve fitting for this model. Therefore, we would initialize all the parameters with regards to the upper bound presented, 100.

3.5 Methodology for sampling parameters with regards to the fitting data

We believe that a better method would be to set the parameters of the distributions with regards to the data. A relevant example for this would be the following for exponential function with 2 parameters: the initial guess for the initial quantity parameter 'a' would be sampled from the uniform distribution with the minimum value of the fitting data as the upper bound and with the maximum value of the fitting data as the upper bound. For the decay parameter 'b' we would try to make it as close as possible to 0 such that the $e^{-b \cdot x}$ would be converge to 1. Despite not having the certainty that the fitting data would be following an uniform distribution, we believe that this way would certainly lead to an optimal value more often and would not stop at local minimum. We will also try to analytically derive formulas in a way that $e^{-b \cdot x}$ would not converge to 1, in a way that b will also be initialized with regards to the data. However, Viering et al [3] states that there is no clear relationship found between parameters. We will analyze this problem in the following order: first, we will compute 'a' initial quantity parameter by sampling it with regards to the fitting data. Mathematically, 'a' will be expressed by :

$$a \sim \mathcal{U}(\min(Y_{\text{fit}}), \max(Y_{\text{fit}}))$$

Now we will compute the decay 'b'. The other idea is to firstly sample the value for b with a value from the distribution and then try to compute 'a' and sample its value for its distribution. Due to time constraints, we will validate our hypothesis only on one model, exponential 3 in this case. First, we will write the equation for exponential 3 which is the following:

$$y = ae^{-bx} + c$$

First, in order to find a relation between parameter 'a' and parameter 'b' we will apply log to this equation, by subtracting c and then applying log to the equation, arriving at the following mathematical expression:

$$log(y - c) = log(a) - b * x$$

Finally, when trying to find a value relation between a and b, we will get rid of the c value after empirically observing from experiments on default parameters that it converges to the same value for most of the time. We will initialize 'c' with also an uniform distribution sampling with default parameters. Our final b value in this case will be

$$b = (-log(a) + log(y - c))/x$$

In our case, we will plug in the y value and the x value as the mean of the fitting points used. Our other options were to use the first point of the fitting points, however this may not yield a good approximation for b since the 'x' value would be too small, close to 0, causing value errors. For the normal distribution, however, we will sample the value similarly like in the case of uniform, following the mean of the fitting data and the standard deviation of the fitting data. Therefore, a will be represented by the following mathematical equation:

$$a \sim \mathcal{N}(\text{mean}(Y_{\text{fit}}), \text{std}(Y_{\text{fit}}))$$

4 Experimental design

In this section, we present the experimental setup, delving into the models I chose and the datasets utilized, as well as details about the evaluation methods used and the comparing methods to assess the results. It is worth mentioning that we will extensively use the functions provided by SciPy [13].

4.1 Datasets used

We will extensively evaluate the curve fitting performance of our experiments on various datasets with learning curves provided by LCDB [4], the learning curve database. We will assess the performance of our experiments only on well-behaved curves, which implies that a filtering of the available datasets will be done.

For each dataset, we will run only the learners which yield the greatest percentage of monotonicity and convexity. For the purpose of our experiment, we will only use the validation set of each learning curve. Due to the limited time and computer resources used, our experiment will be done extensively on well-behaved curves, which were manually selected after plotting them.

We will split each validation set using a 80%-20% split, the same Brumen et al [14] uses in his experiments. This means that we will use 80% of the data for fitting (interpolating) and only 20% percent for evaluating (extrapolating).

4.2 Learners used in the experiment

As previously mentioned, for our experiment we will use the learners which yield the greatest monotonicity, which are Gradient Boosting and the Linear classifier (C. Yan, personal communication, December 05, 2024).

4.3 Pipeline

In order to investigate our hypothesis with high-fidelity, we will run the following pipeline 2000 times for each dataset, learner, and distribution chosen. For each run, then, we initialize the parameters with random values sampled from the distribution chosen. Firstly, we will compute and store the error between the actual values and the objective function with the initial guesses as parameters. For each run, we will compute the curve fitting function, obtaining the full output and storing the new parameters obtained and the number of function calls obtained. After this, we compute the MSE between the actual value and the value obtained with the new parameters from curve fitting, for both interpolation and extrapolation. Mohr, Viering, Loog and van Rijn et al. [4] recommend us to discard mean squared errors higher than 100, value which will be used as a discard threshold in our experiments. To assess the significance of our results, we will use

the **Mann-Whitney U** test [15]. The null hypothesis for this test can be described as :

(H_0): There is no difference in central tendency between the two groups in the population.

4.4 Metrics used for evaluation

The metrics used in our experiments in order to compare 2 different initializations are the mean squared error obtained for interpolation and extrapolation and the number of average number of failed fits per method of sampling.

5 Experimental results

Table 2: Average failed fits for different methods of distributions' sampling for EXP3 objective model function

Distribution Type	Failed fits
Default Uniform Distribution	148.375
Standard Normal Distribution	1038.4375
Uniform Distribution with Bounds [-3,3]	1017.875
Uniform Distributions Improved	861.625
Normal Distributions Improved	985.125
Normal Distribution with Larger Positive Interval	2.8125
Uniform Distribution with Larger Positive Interval	2.1875

5.1 Performance differences between Sampling Methods on Curve fitting

What is the performance difference between uniform and normal distribution sampling of initial parameters in LM algorithm curve-fitting for well-behaved curves?

The performance between the 2 distributions regarding the mean squared error in terms of mean was significant for the power function model with 2 parameters, as seen in 3. Moreover, the mean values obtained for extrapolation were identical in all cases. The power function fits the set of curves chosen well, with an average of 2 fits discarded per combination of dataset and learner, in comparison with the exponential 2 model which results in 981 average failed fits. However, the results for the exponential parametric model with 2 parameters were more decisive, showing significant differences for both interpolation and extrapolation errors. In order to obtain visible results, we scaled the y-axis of the results. In figures 4 and 5, we can observe that the uniform distribution sampling outperforms the normal distribution sampling, having values of about 70 percents smaller for interpolation and 32 percents smaller for extrapolation on average. Finally, on 3 parametric models, our initial assumptions about pow3 interpolation results 3.4 were correct in this case , the interpolation results being slightly better for the normal distribution sampling, as seen in 6. However, the results obtained for exponential 3 function remain consistent, obtaining better average fits when uniform distribution sampling with default parameters than normal distribution sampling with default parameters, as seen in picture 7.

.

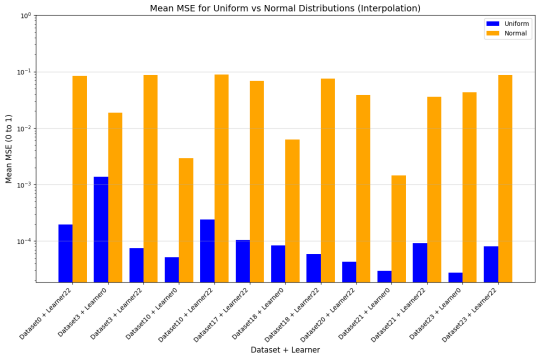

Figure 3: Interpolation difference for POW2 parametric model with normal and uniform sampling, showing a slightly better value for uniform, with the **Y-axis** scaled

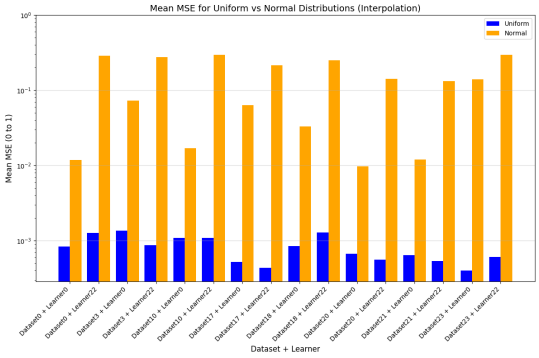

Figure 4: Interpolation difference for EXP2 parametric model with normal and uniform sampling, showing smaller values for uniform, with the **Y-axis** scaled

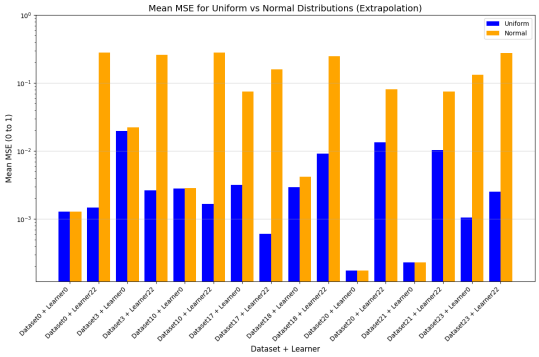

Figure 5: Extrapolation difference for EXP2 parametric model with normal and uniform sampling, showing smaller values for uniform, with the **Y-axis** scaled

5.2 Analyzing uniform distribution with negative lower bound

We initially assumed that one of the reasons why normal distribution with default parameters performs worse in practice is due to the wider spread and the random sampling of also negative values. Now, we will mimic the standard normal distribution with uniform distribution with a negative lower

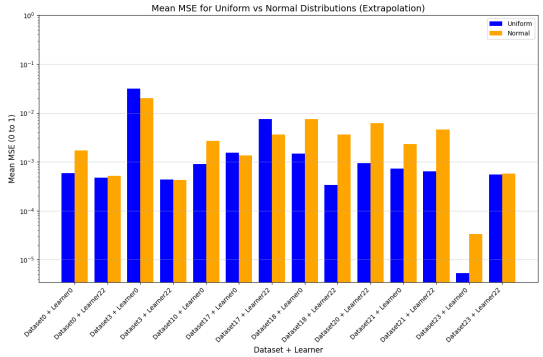

Figure 6: Interpolation difference for POW3 parametric model with normal and uniform sampling, showing similar values for uniform and normal , with the **Y-axis** scaled

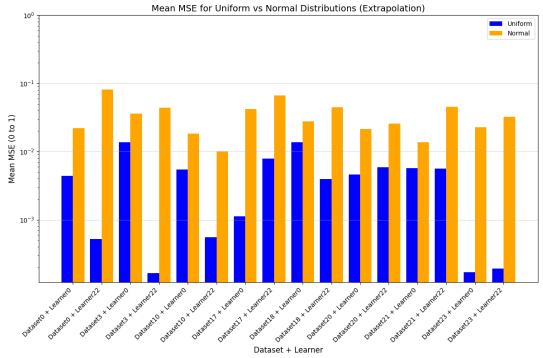

Figure 7: Extrapolation difference for EXP3 parametric model with normal and uniform sampling, showing similar values for uniform and normal, with the **Y-axis** scaled

bound. By looking at the standard normal distribution presented in figure 2, we will set the lower bound of the uniform distribution sampling to -3 and the upper bound to 3, the boundaries of the values most probable drawn for the standard normal distribution. Firstly, looking at the table 2, we can observe that the average number of failed fits was very similar, uniform distribution sampling method with negative lower bound failing also around 50 percents of the time. Moreover, in figure 8 we observe that the uniform actually performs worse when having their bounds of [-3,3]. However, when restricting this bounds of the uniform distribution sampling to [-1.5,1.5], we only got one pair for which the extrapolation mean values difference between sets was significant and it had a p-value lower than 0.05. Therefore, we can conclude that by changing the parameters accordingly, standard normal distribution produces the same results as an uniform distribution.

5.3 Increasing uniform distribution parameters to improve curve fitting

How does increasing the upper bound in uniform sampling impact LM algorithm curve fitting for well-behaved curves?

We analyzed the effect of increasing the upper bound of the uniform distribution. Firstly, in table 2 we can observe

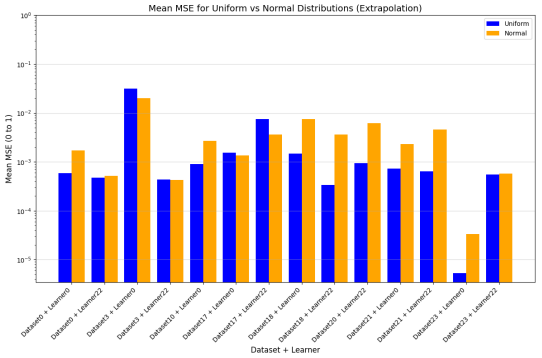

Figure 8: Extrapolation difference for EXP3 parametric model with normal and uniform sampling with bounds [-3,3], showing better values for uniform, with the **Y-axis** scaled

that it reduces the average failed fits to only 2.1875, showing that by increasing the interval from which the distributions are drawn, the changes of arriving at an optimal solution in curve fitting process are increased. However, surprisingly, the average mean mean squared error is not better than the ones shown in default uniform distribution, as seen in figure 9.

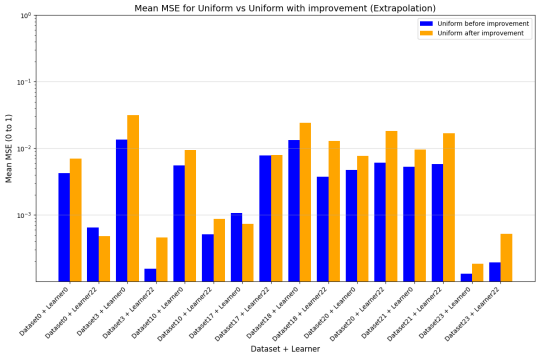

Figure 9: Extrapolation difference for EXP3 parametric model with default uniform sampling and uniform sampling with bounds [0,100], showing better values for default uniform, with the **Y-axis** scaled

5.4 Increasing normal distribution parameters to improve curve fitting

How does increasing the mean and standard deviation in normal distribution sampling impact LM algorithm curve fitting for monotonic curves in the LCDB database?

In table 2 we can observe that the number of failed fits is reduced from 1038 to 2.81 average failed fits. Moreover, in figure 10 we can observe that normal distribution with mean increased yields lower mean squared errors than the standard normal distribution for extrapolation. Therefore, we can conclude that values sampled from a normal distribution with a higher mean and a standard deviation which ensures a high probability of sampling positive values yield lower mean squared errors, despite the chance that the initial guesses can be more far from the optimal value than the method with standard normal sampling.

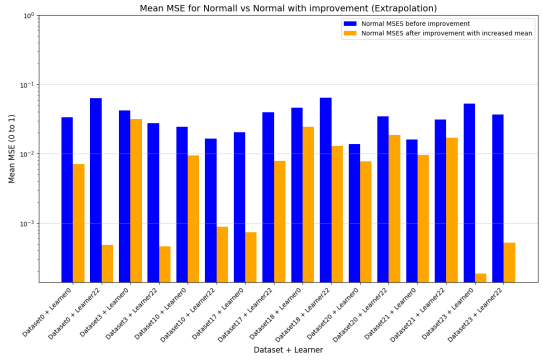

Figure 10: Extrapolation difference for EXP3 parametric model with standard normal distribution and normal distribution sampling with mean 50, standard deviation 16.66 showing lower MSE values for normal with increased mean, with the **Y-axis** scaled

5.5 Fine tuning the distributions' parameters to improve curve fitting

How does tuning random distribution parameters with regards to the fitting data used impact LM algorithm in curve fitting ?

Despite our initial assumption that adjusting the distributions' parameters with regards to the data, we can observe that our proposed method fails to significantly decrease the extrapolation's MSE. For uniform distribution, around 60% of the learning curves perform better when using the default sampling in terms of average extrapolation error, as seen in 11. The number of failed fits per distribution is significantly higher than the number of failed fits for the proposed method with increased bounds/mean, as seen in 2. Therefore, we can conclude that trying to tune parameters with regards to the fitting data might not be the best approach because the function might overfit the fitting data, giving bad extrapolation values.

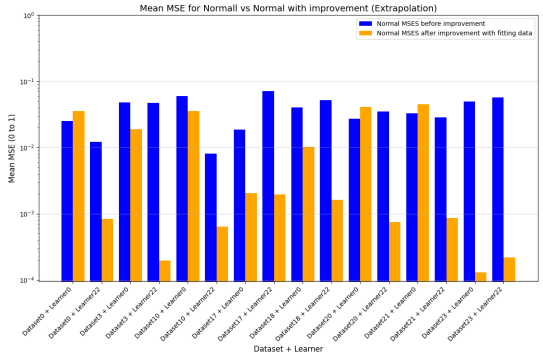

Figure 11: Extrapolation difference for EXP3 parametric model with standard normal distribution and normal distribution sampling with parameters fitted to the fitting data, with the **Y-axis** scaled

6 Discussion

As Gavin stated et al [7], LM algorithm seems to be very sensitive to initial guess also from our results. A representative

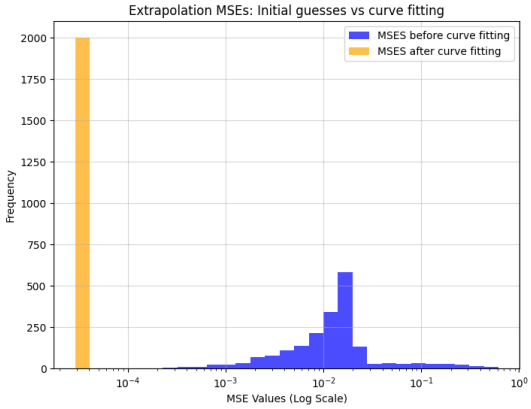

Figure 12: Extrapolation difference for EXP3 parametric model with normal and uniform sampling, showing multiple different MSE values and their frequency before sampling and the MSE values after curve fitting, which are almost identical with the **X-axis** scaled

example of the extrapolation MSE values done before and after curve fitting can be seen in figure 12. This concludes that this curve fitting method tends to constantly arrive at the same extrapolation value, showcasing that LM is prone to get stuck in local minima values found.

Initially, we obtained that the default uniform distribution performs better in terms of failed fits and average mean squared errors than standard normal distribution. However, after sampling values from a uniform distribution with a negative lower bound, we arrived at the conclusion that it is not a matter of the distribution chosen, but of the negative values which yield failed fits or suboptimal fits for power and exponential function models for well-behaved curves. However, Kim et al [5] suggests that an appropriate lower bound for all of the parameters of the exponential function with 3 parameters would be -10, a negative value which yielded worse and more failed fits in our case. Moreover, despite reducing the number of failed fits, sampling with uniform distribution with the upper bound 100 and lower bound 0 did not produce significant differences in mean squared errors for extrapolation in our setting, showing that the bounds presented can be more restrictive.

Finally, our proposed approach for finding the initial guess parameters for exponential 3 function on well-behaved curves performs better than the default method distributions. However, it performs worse in terms of extrapolation. We believe that this is due to the difference between the fitting data and the extrapolation and the models chosen tend to overfit on the data used for fitting. Since there is no previous conclusion for the relation of the parameters in the exponential 3 function in the process of curve fitting, sampling 'c' parameter from a distribution with default parameters might negatively affect the failed fits performance of our experiment.

Due to time and computational constraints, the results obtained might not be entirely accurate. One of the reasons for that is that we only tried our pipeline on well-behaved curves, omitting the ill-behaved curves in LCDB. We expect that the fits will be more difficult to realize on ill-behaved curves and that the objective model function chosen for these experi-

ments would not be fit for most of the ill-behaved curves, as Viering and Loog state et al [3]. Another shortcoming of our experiments would be split of 80% fitting data and 20% extrapolation data. Trying multiple different splits and comparing the mean of the mean squared errors and failed fits is a more sensible way to do it and could result in more accurate results.

7 Conclusions

7.1 Summary of the results

In conclusion, this paper aims to describe the effect of sampling the initial parameters of the objective model function from 2 random distributions on the curve fitting process of well-behaved curves. Moreover, it critically compares the 2 distributions used: **normal** and **uniform** in terms of **failed fits** and **mean squared errors**. Firstly, it compares their performance using the default parameters for both normal and uniform, and we initially arrived at the conclusion that uniform performs better for all the models used for experiments. However, the following experiments arrive at the conclusion that this is the case because of the negative values sampling which yielded more suboptimal and failed fits, meaning that the 2 initialization method can have similar results if their parameters are correctly chosen. Finally, we highlight the Levenberg-Marquadt sensitivity to bad initial guesses which yield more failed fits if the initial guess is very far from the optimal solution. However, by tuning the parameters of the distributions sampling functions with regards to the data, we can observe that despite getting more failed fits, we get more optimal fits on average, reducing the risk of the algorithm of getting stuck in the local minima.

7.2 Future work

To further analyze and solidify our findings, the experiment should also be conducted with ill-behaved curves and anchors percentages should be used for further experiments. This will create more reliable and potentially different results, as objective model's function performance for each anchor percentage can differ [6]. Moreover, more complex parametric models should be analyzed, since it is believed that 4 parametric models might yield a better fit [4]. Finally, different ways of tuning the distributions' parameters can help at discovering more precise ranges for good initial guesses for LM algorithm for curve fitting. Moreover, more methods for adjusting the distributions' parameters with regards to the data should be tried, methods which can give us more insight into the relation of parameters for model objective functions in learning curves. We suggest that sampling other parameters than 'a' with regards to the data for the exponential 3 function might potentially yield better results.

8 Responsible Research

In this section, we discuss the broader implications of the research conducted, ensuring that all experiments are reproducible, the data is publicly accesible and it adheres to the ethical standards required.

8.1 Reproducibility

This research study is compliant with the Netherlands Code of Conduct for Research Integrity(2018), and the FAIR(Findability, Accesibility, Interopability, and Reusability) as per [16] and integrating the research integrity vision values from TU Delft [1]. The code used to conduct the experiments will be uploaded to a public Github repository [2], an open-source platform and forwarded to the group of responsible supervisors who will assess the correctness of the code. The datasets collected are from the Learning curves data base, a database described in depth in the [4] paper. All the functions used for computing the curve fitting are from SciPy[13], an open source library from Python. The interoperability and and reusability of the experiments is assured by the complex documentation in the code and the 4 section which clearly describes the steps needed to reproduce each experiment conducted. Additionally, results from the experiment will be stored into CSV files which will also be uploaded on the Github repository. Some challenges regarding the reproducibility of the experiments may be encountered due to potential updates to the open source library used in the code. Moreover, due to the high computational power needed to conduct the experiments, some setups would not be able to support the computational processes needed to arrive at the results.

8.2 Ethical impact

The aim of our analysis is to analyze and optimize the initialization of initial parameters used for CURVE FITTING algorithm for learning curves. However, malicious actors could use these findings in trying to manipulate learning curve fitting process by selecting initial values which tend to yield an unrealistic mean squared error or optimizing deceiving models. We recommend future users of our experiments to be transparent with their experiments, reporting multiple metrics for the performance.

References

[1] Felix Mohr and Jan van Rijn. Learning curves for decision making in supervised machine learning: a survey. *Machine Learning*, 113:8371–8425, 12 2024.

[2] Leonardo Vianna, Alexandre Gonçalves, and João Souza. Analysis of learning curves in predictive modeling using exponential curve fitting with an asymptotic approach. *PLOS ONE*, 19:e0299811, 04 2024.

[3] Tom Viering and Marco Loog. The shape of learning curves: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7799–7819, June 2023.

[4] Felix Mohr, Tom J. Viering, Marco Loog, and Jan N. van Rijn. Lcdb 1.0: An extensive learning curves database for classification tasks. In Massih-Reza Amini,

[1] https://www.tudelft.nl/over-tu-delft/strategie/integriteitsbeleid/tu-delft-vision-on-integrity-2018-2024

[2] https://github.com/ddarie-wq/Initialization-of-curve-fit-parameters-with-distribution-sampling

Stéphane Canu, Asja Fischer, Tias Guns, Petra Kralj Novak, and Grigorios Tsoumakas, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2022, Proceedings*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pages 3–19. Springer, 2023.

[5] Donghwi Kim. Different approaches to fitting and extrapolating the learning curve, June 2022. Bachelor's Thesis, Supervisors: Tom Viering, Marco Loog, EEMCS, Delft University of Technology, The Netherlands.

[6] Prajit Bhaskaran. To tune or not to tune: Hyperparameter influence on the learning curve, June 2022. Bachelor's Thesis, Supervisors: Tom Viering, Marco Loog, EEMCS, Delft University of Technology, The Netherlands.

[7] Henri P. Gavin. The levenberg-marquardt algorithm for nonlinear least squares curve-fitting problems. *Department of Civil and Environmental Engineering, Duke University*, May 2024. Technical paper.

[8] Boštjan BRUMEN, Ivan Rozman, Marjan Hericko, Aleš ČERNEZEL, and Marko Hölbl. Best-fit learning curve model for the c4.5 algorithm. 25:385–399.

[9] Lih-yuan Deng. *Uniform Distribution: Definitions and Properties*. 09 2014.

[10] Beatriz Adriana Rodríguez González, Gabriela Noemí Figueroa Ibarra, Omar Guirette Barbosa, and Héctor Antonio Durán Muñoz. The use of the empirical rule in the probability class: A proposed application for university students to determine the type of statistical thinking. 22(3):521–537.

[11] Lumina Decision Systems. Normal distribution, 2024.

[12] Timothy Hodson, Thomas Over, and Sydney Foks. Mean squared error, deconstructed. *Journal of Advances in Modeling Earth Systems*, 13, 12 2021.

[13] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[14] Boštjan Brumen, Aleš Černezel, and Leon Bošnjak. Overview of machine learning process modelling. *Entropy*, 23(9), 2021.

[15] Nadim Nachar. The mann-whitney u: A test for assessing whether two independent samples come from the same distribution. *Tutorials in Quantitative Methods for Psychology*, 4, 03 2008.

[16] Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, N Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. The fair guiding principles for scientific data management and stewardship. *Scientific Data*, 3(1):1–9, 2016.