

Report Template

FYS-STK3155 - Project X

Lars Bogner

University of Oslo

<https://github.com/larsbog/FYSSTK-Project1>

(Dated: September 25, 2025)

The basis for many data-driven methods is the optimization of numerical models to describe relations in data. With the increasing availability of large datasets and the growing complexity of models, there is an increasing need for efficient optimization techniques that can handle the computational challenges associated with big data. Furthermore, a good optimization metric can be crucial to find the best possible model to describe the data. In this paper different methods of such optimization are studied in the context of large datasets and big data with possibly noise-affected input data. To render the best possible fits a multitude of cost functions, minimization algorithms and other approaches to reduce the computation associated with the optimization process are studied and evaluated. Reducing the computational cost is also important to improve the economical and ecological footprint of training large models on big data, as the spread of datadriven methods in all areas of life is increasing rapidly.

I. INTRODUCTION

Optimizing numerical models to model a set of data has been a challenge of the natural sciences for many decades. But with the introduction of artificial neural networks in all disciplines over the last years a special focus on those optimization techniques is relevant.

In this paper different methods of such optimization[1, 2] are studied in the context of large datasets and big data with possibly noise-affected input data. To render the best possible fits a multitude of cost functions, minimization algorithms and other approaches to reduce the computation associated with the optimization process are studied and evaluated. Reducing the computational cost is also important due improve the economical and ecological footprint of training large models on big data, as the spread of datadriven methods in all areas of life is increasing rapidly.

In the following section the theoretical background of the methods used in this paper is presented. In ?? secondly the implementation of these methods is discussed. The results of applying these methods to different datasets are presented in ?. Finally, a conclusion is drawn in ? and an outlook on possible future work is given in ?.

In a first step, the dependence of the bias-variance tradeoff on the number of degrees of freedom in a model is studied. For this purpose, polynomial models of different degrees are fitted to data generated from the Runge function with significant noise. The results are then compared to different Ridge and Lasso models to study the effect of regularization on the bias-variance tradeoff. In the context of big data, the performance of different numerical minimization algorithms is studied. For this purpose, the performance of gradient descent, stochastic gradient descent and different optimization algorithms such as Adam are compared in the context of Ridge and Lasso regression. Furthermore the possible advantages and disadvantages of stochastic gradient descent techniques are

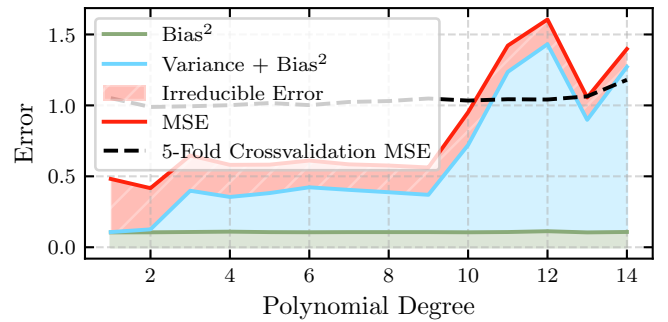


Figure 1: Bias-variance decomposition of the mean squared error (MSE) for polynomial fits of different degrees to noisy data from the Runge function using OLS. The Bias-Variance decomposition is calculated from bootstrapped samples. The MSE is also calculated using 5-fold cross-validation and bootstrap resampling techniques. $N_{\text{train}} = 80$, $N_{\text{test}} = 20$.

evaluated. Finally, the effect of resampling methods such as bootstrapping and k-fold cross-validation on the bias-variance tradeoff is studied.

II. RESULTS

To evaluate the bias-variance tradeoff of the optimized models, the MSE is decomposed into its bias and variance components as introduced in ?. The results are shown in fig. 1. To accurately depict the bias-variance tradeoff, there were two sets of test data used. To compute the bias, no noise was added to the test data, while for the rest of the computations the same noise level as in the training data was used. It is visible that for increasing polynomial degrees the variance increases, while the bias stays almost constant. This is a direct consequence of the increased flexibility of the model with increasing polynomial degree. To ensure numerical stability, Ridge regression with a very small regularization strength of

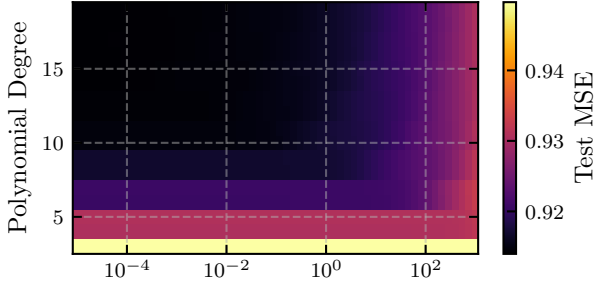


Figure 2: Mean squared error (MSE) for polynomial fits of different degrees to noisy data from the Runge function using Ridge regression with different regularization strengths λ . $N_{\text{train}} = 80\,000$, $N_{\text{test}} = 20\,000$.

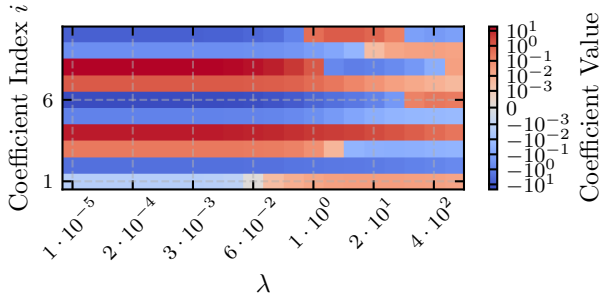


Figure 3: Parameter values for polynomial fits of different degrees to noisy data from the Runge function using Ridge regression with different regularization strengths λ . $N_{\text{train}} = 80\,000$.

$\lambda = 1 \times 10^{-10}$ has been used to compute the bias-variance decomposition in an approximation of OLS. This ensures that the matrix inversion in the computation of the model parameters is numerically stable, while the regularization term has almost no effect on the model performance. The optimal model complexity is usually found at the point where the sum of bias and variance is minimal. Usually this coincides with the point of the minimum of the mean squared error. How the MSE behaves on average for k -fold cross-validation and bootstrap resampling techniques is also shown in fig. 1. Both resampling techniques show a minimum in the MSE for the polynomial degree of 2. The crossvalidation technique shows a much more constant MSE value over the entire range of polynomial degrees, while the bootstrap technique shows a more pronounced minimum. This indicates that the bootstrap technique is more sensitive to the choice of polynomial

degree and thus provides a better estimate in the case of hyperparameter tuning at the cost of a much higher computational cost.

To counteract the negative effects of high parameter count, regularization techniques such as Ridge and Lasso regression can be used. These techniques add a penalty term to the cost function that discourages large parameter values. In Ridge regression, the penalty term is proportional to the square of the parameter values, while in Lasso regression, it is proportional to the absolute value of the parameter values. This leads to a trade-off between fitting the data well and keeping the parameter values small. With an increasing regularization strength λ the parameter values are pushed towards zero, as visible in fig. 3. This effect is especially strong for $\lambda > 0$. For very high regularization strengths the parameter values become very small, leading to a model that is almost constant. This is also visible in fig. 2, where the MSE is shown for polynomial fits of different degrees to noisy data from the Runge function using Ridge regression with different regularization strengths λ . For very high regularization strengths the MSE increases significantly, as the model is not flexible enough to fit the data well anymore. The optimal regularization strength is promoted to a hyperparameter that needs to be tuned to the problem at hand. However, with a suitable choice of λ it is possible to use models with a high number of parameters without suffering from the negative effects of overfitting.

Appendix A: Usage of AI

The code has been developed using the **Visual Studio Code** editor, which provides a wide range of features for Python development, including syntax highlighting, code completion, and debugging tools. The version control system **Git** has been used to manage the codebase and track changes. The Large Language Model (LLM) **GitHub Copilot** has been used to assist in the development of the code by providing code suggestions and autocompletions based on the context of the code being written. All LLM-generated code has been reviewed and modified as necessary to ensure its correctness and suitability for the project. The LLM ChatGPT-4 has been used to create ???. The LaTeX package **TikZ** has been used to create the figure. The LLM has been prompted to generate the TikZ code for the figure, which has then been modified as necessary to ensure its correctness and suitability for the report. The conversation can be found at the root of the GitHub repository.

[1] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer Series in Statistics (Springer, New York, 2009), URL <https://link.springer.com/book/10.1007/978-0-387-84858-7>.

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al., **12**, 2825 (2011).