Project 1

FYS-STK4155

Håvard Skåli, Erik Røset & Oskar Idland
University of Oslo, Department of Physics
(Dated: October 4, 2024)

We have used various regression techniques and resampling methods within the context of machine learning, and tested these on the Franke function, a synthetic benchmark used in numerical analysis, as well as on output data from a cosmological N-body simulation performed with the public GASOLINE2 code. The main objective was to analyze and compare the performance of Ordinary Least Squares (OLS), Ridge, and Lasso regression in fitting synthetic and real-world data, focusing on the bias-variance tradeoff and model generalizability. To assess model performance we employed resampling methods such as bootstrap and k-fold cross-validation, examining how they help to evaluate model accuracy under different training and test data conditions. Our study provided insights into how model complexity and regularization parameters affect the bias-variance tradeoff and prediction error, enabling us to critically evaluate each regression technique's performance using statistical and resampling approaches. rewrite when results are gathered and discussion/conclusion is finished

https://github.com/Oskar-Idland/FYS-STK4155-Projects

I. INTRODUCTION

In this work, we explore regression analysis and resampling methods in the context of machine learning, focusing on both theoretical and practical aspects. The primary goal is to develop a solid understanding of various regression techniques, including Ordinary Least Squares (OLS), Ridge, and Lasso regression, and their application to both synthetic and real-world data. We will also investigate how resampling methods such as bootstrap and cross-validation can help assess model performance.

We begin by applying the abovementioned methods to the Franke function, a widely used test function in numerical analysis, and extend the analysis to cosmological N-body simulation data made with the public version of the GASOLINE2 Smoothed Particle Hydrodynamics (SPH) code [1]. Our approach involves fitting polynomial regression models of varying complexity to the Franke function as well as the simulation data, and studying the bias-variance tradeoff, an essential concept in machine learning, in both cases. This allows us to evaluate the impact of model complexity, noise, and the size of training data on the accuracy and generalizability of the models. The overarching aim is to gain insights into how different regression methods handle overfitting, model complexity, and data variability, and to develop a framework for critically evaluating model performance using statistical and resampling techniques. rewrite?

In section II we present relevant background theory, including central concepts such as model bias, model variance and the bias-variance tradeoff, as well as the Franke function. The most important expressions introduced here are derived in appendix A. Our methodology is explained in section III, specifically regression analysis, where we focus on OLS, Ridge and Lasso regression. We also present the advantages and disadvantages of two crucial resampling methods; bootstrapping and cross-validation, both of which will be used in this work. In this section we also present our dataset, specify how we implement the methods and give an overview of our code structure. The results of our analyses are presented in section IV, and in section V we discuss our findings in light of what we would expect from the regression variants and resampling methods implemented edit after discussion is written. Lastly, in section VI we summarize and conclude the main findings of our work mention reflection?.

II. THEORY

A. Regression Analysis

Regression analysis is a fundamental statistical technique used to model the relationship between one or more independent variables (also known as predictors or features) and a dependent variable (or target). The goal of regression analysis is to find the mathematical relationship that best explains the variation in the dependent variable based on the values of the independent variables.

Given a dataset consisting of n data points $\{(\mathbf{x}_i, y_i)\}_{i=0}^{n-1}$, where $\mathbf{x}_i$ represents the input features (which is a vector in the case of multiple features), and y_i is the corresponding target value, the goal is to find a model that predicts $\tilde{y}_i$ based on x_i such that it is as close to y_i as possible. For a linear model, the relationship between x_i and $\tilde{y}_i$ can be expressed as

$$\tilde{y}_i = \mathbf{x}_i^\top \boldsymbol{\beta} = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}, \qquad (1)$$

where $\mathbf{x}_i^\top$ is the vector of input features for the i-th data point and $\boldsymbol{\beta} = [\beta_0, \beta_1, \ldots, \beta_p]^\top$ is the vector of regression

coefficients that we want to estimate. This can be written as a full-fledged matrix equation by using the so-called design or feature matrix $\mathbf{X} \in \mathbb{R}^{n\times(p+1)}$, which contains all input features and the bias term (so that the intercept β_0 is included):

$$\tilde{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta}. \tag{2}$$

write differently?

At its core, regression analysis seeks to find a mathematical function that relates the independent variables to the dependent variable. In its most basic form, the relationship between a dependent variable $\mathbf{y}$ and an independent variable $\mathbf{x}$ is modeled as

$$\mathbf{y} = f(\mathbf{x}) + \boldsymbol{\epsilon}, \tag{3}$$

where $f(\mathbf{x})$ is the function we are trying to estimate, which represents the relationship between $\mathbf{x}$ and $\mathbf{y}$. This is then what we approximate with our model $\tilde{\mathbf{y}}$. Furthermore, $\boldsymbol{\epsilon}$ is the error term, representing the part of the variation in $\mathbf{y}$ that is not explained by the model (due to noise or other unobserved factors).

1. *Ordinary Least Squares*

The Ordinary Least Squares (OLS) method is one of the most fundamental and widely used techniques in regression analysis. Its objective is to find the best-fitting line or curve for a given set of data by minimizing the sum of the squared differences between the observed values and the predicted values. These squared differences are called errors, hence the sum of squared errors (SSE):

$$\text{SSE} = \sum_{i=0}^{n-1} (y_i - \tilde{y}_i)^2 . \tag{4}$$

Minimizing this sum is equivalent to minimizing the cost function

$$C(\mathbf{X}, \boldsymbol{\beta}) = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^2 = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\beta}), \tag{5}$$

since the solution to this minimization problem gives the optimal values of $\boldsymbol{\beta}$.

To minimize $C(\boldsymbol{\beta})$, we take the derivative of the cost function with respect to $\boldsymbol{\beta}$ and set it to zero:

$$\frac{\partial C(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -2\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) = 0. \tag{6}$$

This gives the normal equation

$$\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^\top\mathbf{y}, \tag{7}$$

which, by solving for $\boldsymbol{\beta}$ yields the OLS estimator:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}. \tag{8}$$

This formula provides the best-fitting coefficients $\boldsymbol{\beta}$ that minimize the sum of squared errors. In appendix A 1 we derive the following important results: edit? maybe move up here?

$$\mathbb{E}\left[\hat{\boldsymbol{\beta}}\right] = \boldsymbol{\beta}, \tag{9}$$

$$\text{Var}\left[\hat{\boldsymbol{\beta}}\right] = \sigma^2 \left(\mathbf{X}^\top\mathbf{X}\right)^{-1}. \tag{10}$$

Here $\mathbb{E}\left[\hat{\boldsymbol{\beta}}\right]$ and $\text{Var}\left[\hat{\boldsymbol{\beta}}\right]$ are the expectation value and variance of the OLS estimator, respectively, while σ^2 is the variance of the error term $\boldsymbol{\epsilon}$.

2. *Ridge*

While OLS provides a foundational method for regression, it can lead to problems when the data has high multicollinearity or when there are more features than data points, leading to overfitting. OLS attempts to minimize the sum of squared errors, but it does not impose any restrictions on the model complexity. This often results in high variance when the model learns to fit noise in the training data, especially when the design matrix $\mathbf{X}$ is poorly conditioned (i.e., when columns of $\mathbf{X}$ are nearly linearly dependent). This is because the matrix $\mathbf{X}^\top\mathbf{X}$ becomes close to singular, making its inverse highly sensitive to small changes in the data. remove last sentences?

To mitigate the issue with OLS, Ridge regression introduces a regularization term to the cost function (5), penalizing large coefficients and preventing the model from becoming overly complex. The modified cost function is

$$C_{\text{Ridge}}(\mathbf{X}, \boldsymbol{\beta}) = \frac{1}{n} ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 + \lambda ||\boldsymbol{\beta}||_2^2 , \tag{11}$$

where $\lambda \geq 0$ is a hyperparameter that controls the strength of the regularization, and the subscripts 2 simply mean that we are taking the L^2 norm. This is defined as

$$||\mathbf{x}||_2 = \sqrt{\sum_i x_i^2}.$$

We see that when $\lambda = 0$, Ridge regression reduces to OLS, as the regularization term disappears. On the other hand, when λ is large, the coefficients $\boldsymbol{\beta}$ must shrink to minimize the cost function, potentially reducing overfitting. It is important to note, however, that if λ becomes too large the coefficients tend to zero, and we may end up with underfitting instead. We therefore need to determine the hyperparameter carefully in order to find the perfect balance between the two extremes. formulate differently?

To minimize this altered cost function, we again take the derivative with respect to $\boldsymbol{\beta}$ and set it to zero:

$$\frac{\partial C_{\text{Ridge}}(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -2\mathbf{X}^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) + 2\lambda\boldsymbol{\beta} = 0. \tag{12}$$

Rearranging this equation gives us the Ridge regression normal equation:

$$(\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})\boldsymbol{\beta} = \mathbf{X}^\top\mathbf{y}, \tag{13}$$

where $\mathbf{I}$ is the identity matrix. Notice the term $\lambda\mathbf{I}$ added to $\mathbf{X}^\top\mathbf{X}$, which makes the matrix invertible even when $\mathbf{X}^\top\mathbf{X}$ is poorly conditioned. This leads to the Ridge regression estimator:

$$\hat{\boldsymbol{\beta}}_{\text{Ridge}} = (\mathbf{X}^\top\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y}. \tag{14}$$

Here, the regularization term λ shrinks the coefficients and prevents them from becoming too large, thereby controlling the model's variance. remove this paragraph?

3. *Lasso*

Like Ridge regression, Lasso regression (Least Absolute Shrinkage and Selection Operator) is a regularization technique designed to improve the generalizability of the model by introducing a penalty term to the cost function. While Ridge regression uses the L^2 norm for regularization, Lasso regression employs the L^1 norm, defined as

$$||\mathbf{x}||_1 = \sum_i |x_i|.$$

This leads to a different form of regularization, since the cost function now takes the form

$$C_{\text{Lasso}}(\mathbf{X}, \boldsymbol{\beta}) = \frac{1}{n}\,||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 + \lambda\,||\boldsymbol{\beta}||_1\,. \tag{15}$$

The difference from Ridge regression has important implications, mainly because the L^2 norm makes it impossible for any of the coefficients to vanish completely. correct? This can be problematic in high-dimensional datasets where we might expect many features to be irrelevant. By using the L^1 norm, Lasso regression has the unique ability to drive some coefficients to exactly zero, effectively performing both regularization and feature selection. Thus, Lasso regression not only addresses overfitting but also simplifies the model by automatically excluding irrelevant features, making it particularly useful for sparse models where only a subset of features are truly important. double check, introduce expressions, cite?

B. Properties of Predictive Models

maybe change structure, avoid repetitiveness

1. *Predicted Values*

In the context of predictive modeling, $\tilde{\mathbf{y}}$ represents the model's predicted values of the target variable $\mathbf{y}$. These predictions are made based on the features and parameters learned from the training data. For new or unseen data, the true target values $\mathbf{y}$ are often unknown, but the model generates an estimate $\tilde{\mathbf{y}}$ that approximates these values.

The performance of the model's predictions can be analyzed by decomposing the error into several components. The error of $\tilde{\mathbf{y}}$ stems from three main sources: noise variance σ^2, model bias and model variance. The latter two are expressed as

$$\text{Bias}\,[\tilde{y}] = \mathbb{E}\left[(\mathbf{y} - \mathbb{E}\,[\tilde{\mathbf{y}}])^2\right], \tag{16}$$

$$\text{Var}\,[\tilde{y}] = \mathbb{E}\left[(\tilde{\mathbf{y}} - \mathbb{E}\,[\tilde{\mathbf{y}}])^2\right]. \tag{17}$$

Each of these represents a different aspect of the total error that affects the model's performance.

2. *Model Bias*

From the expression (16) we see that the model bias measures how far the average model prediction $\mathbb{E}[\tilde{\mathbf{y}}]$ is from the true target value $\mathbf{y}$. More specificaly, it refers to the systematic error introduced by the model's inability to capture the true underlying relationship between the independent and dependent variables. This error arises when the model makes incorrect assumptions about the data or oversimplifies the relationship. For example, a linear model trying to fit highly non-linear data will result in high bias because the model cannot flexibly represent the complexity of the data.

3. *Model Variance*

We see from the expression (17) for the model variance that it represents the variability in the model's predictions across different training sets. A high model variance indicates that the model is overly sensitive to the specific training data, leading to overfitting, where the model performs well on the training set but poorly on new, unseen data such as the test set. This happens because the model has "memorized" the noise in the training data rather than capturing the true underlying pattern. High variance typically arises in overly complex models that are capable of capturing minute details, which may not generalize well to new data.

4. *Bias-Variance Tradeoff*

Measures of a predictive model's total error and how well it is likely to predict future samples are commonly expressed in terms of the mean-squared error (MSE) and

the score function, respectively. These are given by

$$\mathrm{MSE}(\mathbf{y},\tilde{\mathbf{y}}) = \frac{1}{n}\sum_{i=0}^{n-1}(y_i-\tilde{y}_i)^2, \tag{18}$$

$$R^2(\mathbf{y},\tilde{\mathbf{y}}) = 1 - \frac{\sum_{i=0}^{n-1}(y_i-\tilde{y}_i)^2}{\sum_{i=0}^{n-1}(y_i-\overline{y})^2}, \tag{19}$$

where $\overline{y}$ is the mean value of $\mathbf{y}$ and n is the number of data points. In appendix A 2 maybe move up here? we show that the MSE alternatively can be expressed in terms of the model bias, model variance and noise variance as

$$\mathbb{E}\left[(\mathbf{y}-\tilde{\mathbf{y}})^2\right] = \mathrm{Bias}[\tilde{y}] + \mathrm{Var}[\tilde{y}] + \sigma^2. \tag{20}$$

Since the noise variance adds an irreducible contribution to the total model error, an ideal model strikes a balance between bias and variance, where it is flexible enough to capture the underlying patterns in the data but not so flexible that it fits the noise. This balance is referred to as the bias-variance tradeoff. It is a fundamental concept in machine learning, specifically when building predictive models, and will be studied in great detail throughout this work.

C. Resampling Methods

Resampling methods are statistical techniques used to generate additional data samples from the available data. These methods are particularly useful in machine learning and data analysis when the dataset is limited, and we want to better assess the performance of a model. The primary goal of resampling is to estimate the accuracy of a model by splitting the data into different subsets or generating new samples of the data, repeatedly fitting the model, and evaluating its performance on different sets of data. In this work we implement two commonly used resampling techniques: bootstrap and cross-validation.

1. *Bootstrap*

The bootstrap method is a powerful resampling technique used to estimate the uncertainty and variability of a model by repeatedly drawing random samples, with replacement, from the original dataset. This resampling creates multiple "bootstrapped" datasets, each the same size as the original, but with some samples appearing multiple times and others potentially omitted. By computing the average performance of the multiple models created with the bootstrapped datasets it becomes easier to estimate the model's bias and variance.

The process of bootstrap resampling approximates the underlying distribution of a statistic, whether it's model performance, a parameter estimate or prediction error, without requiring strong parametric assumptions about the data. This makes it particularly useful when the theoretical distribution of a statistic is unknown or difficult to calculate. Bootstrap is especially convenient when the dataset is small, or when no explicit train-test split is available, as it makes the most out of the available data. It is also good for understanding the bias-variance tradeoff since it can be used to determine a model's variability, and it allows for the estimation of confidence intervals for predictions and model parameters. The downside is that it involves training a model multiple times, which can be computationally expensive, especially with large datasets or complex models. shorten or cite?

2. *Cross-Validation*

Cross-validation is another resampling technique that involves partitioning the dataset into several distinct subsets (or "folds"), and then systematically training the model on one subset while testing it on another. A common form is k-fold cross-validation, where the dataset is divided into k equally-sized folds. The model is trained on $k-1$ folds and tested on the remaining fold. This process is repeated k times, with each fold serving as the test set exactly once.

Since all observations are used for both training and validation, cross-validation maximizes the use of available data, hence it is especially useful in the case of limited datasets. It also helps in managing the bias-variance tradeoff by evaluating the model's performance on different subsets of data. By averaging the results across multiple folds, cross-validation reduces the variance of the model's predictions, as it minimizes overfitting to any particular subset of the data. Additionally, by testing the model on unseen data, it ensures that the bias is not too high, making it a good indicator of model generalization. It does require multiple model trainings however (e.g., 10-fold cross-validation requires 10 models to be trained), making it computationally expensive for large datasets or complex models. shorten or cite?

D. The Franke Function

The Franke function is a widely used two-dimensional synthetic function in numerical analysis and computational mathematics, particularly in the fields of interpolation, regression analysis, and surface fitting. It is defined over the unit square $[0,1]\times[0,1]$ and expressed as

$$\begin{aligned} f(x,y) =& \frac{3}{4}\exp\left\{\left(-\frac{(9x-2)^2}{4}-\frac{(9y-2)^2}{4}\right)\right\} \\ &+\frac{3}{4}\exp\left\{\left(-\frac{(9x+1)^2}{49}-\frac{9y+1}{10}\right)\right\} \\ &+\frac{1}{2}\exp\left\{\left(-\frac{(9x-7)^2}{4}-\frac{(9y-3)^2}{4}\right)\right\} \\ &-\frac{1}{5}\exp\left\{\left(-(9x-4)^2-(9y-7)^2\right)\right\}, \end{aligned} \tag{21}$$

which leads to the surface shown in fig. 1. We will use the function multiple times throughout this work, as it produces a surface with both smooth and non-smooth regions, making it an excellent benchmark for testing regression and resampling algorithms. It simulates real-world data that may contain both complex variations and noise, mimicking scenarios that we will encounter as we move on to implementing the algorithms on actual simulation data.

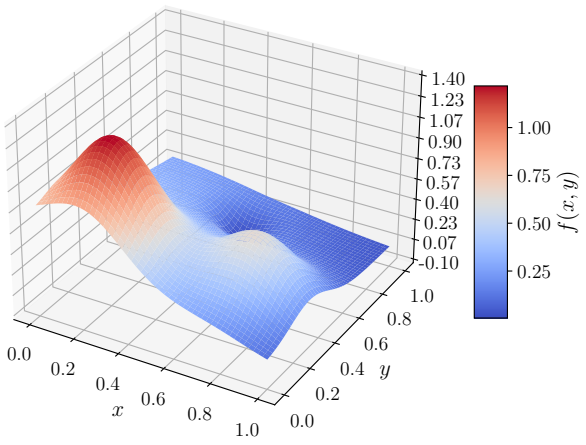

FIG. 1: Visualization of the two-dimensional Franke function expressed in (21).

E. Cosmological Simulation Data

After implementing regression analysis and resampling methods using the Franke function we repeat the process using data from an N-body simulation of dark matter structure formation made with the public GASOLINE2 SPH code [1], as mentioned in section I. The initial conditions for the simulation were generated with MUSIC (MUlti-Scale Initial Conditions) [2], and standard Planck cosmology parameters $\Omega_{\rm m} = 0.3077$, $\Omega_\Lambda = 0.6923$ and $H_0 = 67.81$ km/s/Mpc were used. The simulation box has length, width and height $L = 20$ Mpc, corresponding to a $500 \times 500 \times 500$ grid, and contains 64^3 dark matter particles.

The output data includes 128 "snapshots" of the box through time, where the time paramater in the simulation is the scale factor, defined as

$$a = \frac{1}{1+z}. \tag{22}$$

Here z is the cosmological redshift, which is 0 today and thus in the 128th snapshot. We focus on the dark matter density ρ, and study a snapshot at redshift $z \approx 12.88$ in addition to the snapshot at $z = 0$, because the distribution of ρ is smoother at larger z. To get a function of two variables we average over one of the three axes so that we get a 2D grid, and because the density is so large and vastly different at different points in the box we take $\ln \rho$ to be our dependent variable instead of ρ. The resulting data is shown as intensity plots in figure

Because the Universe expands as time goes on the box does as well, and we therefore scale the coordinates of the box by multiplying with a. Similarly, space between the dark matter particles also increases with time, which is why the density is scaled by multiplying with a^{-3}. GASOLINE2 uses the rest of the scaling factors by default. move to method? explain better?

III. METHODS & IMPLEMENTATION

write in past tense from here on!

A. Regression Analysis

explain implementation on both Franke and data

1. *Creating the Design Matrix*

small paragraph, Håvard explain logic

2. *Scaling the Data*

For this study, data scaling was essential to ensure that the features were on a similar scale, which is critical for many machine learning algorithms. By utilizing the StandardScaler function from the scikit-learn library cite, we standardized our datasets by subtracting the mean and dividing the standard deviation over the data set, for each feature. Although the data generated by the Franke function and the cosmological simulation data used in subsequent analyses are inherently two-dimensional and may not exhibit significant variation in scale between features, scaling still provides consistent benefits.

The primary motivation for applying this scaling function is to enhance the performance and stability of our algorithms. Scaling can prevent features with larger magnitudes from disproportionately influencing the objective function and helps in balancing minor discrepancies between features that might affect the learning process. The performance of our regression models is significantly improved when the data is scaled, as it ensures faster convergence and enhances the overall effectiveness of the algorithms due to recreased risk of numerical instability.

3. Complexity Dependency

Tie this to the design matrix? Investigating the dependency on polynomial degree was a critical aspect of understanding how the regression models performed under varying conditions. By increasing the polynomial order, we enhanced the models' capacity to fit more complex patterns in the data. However, this also raised the risk of overfitting, where the model might capture noise along with the underlying signal. To explore this, we evaluated the performance of the models across different polynomial degrees using metrics such as the Mean Squared Error (MSE) and R^2 scores mentioned in the previous section. These evaluations allowed us to quantify how well each model balanced bias and variance.

For the Franke function-generated data, we systematically varied the polynomial degree from 1 to 6 for OLS and . We then assessed the models' performance using the MSE and R^2 scores, comparing the results to determine the optimal polynomial degree for each regression method.

Our findings indicated that while higher polynomial degrees could improve fit to the training data, there was often a trade-off in terms of generalization to unseen data. Thus, understanding polynomial degree dependency was vital for selecting an optimal degree that achieved the best predictive accuracy while maintaining generalizability across different datasets.

These

4. Hyperparameter Dependency

In addition to polynomial degree, we also examined the impact of the hyperparameter λ on the performance of our Ridge and Lasso regression models. The λ parameter controls the strength of the regularization applied to the models, penalizing large coefficients and thereby reducing overfitting. By adjusting λ, we could fine-tune the complexity of the models, striking a balance between bias and variance. We systematically evaluated the performance across a range of λ values using the same performance metrics as when exploring complexity dependency. This approach helped us understand how varying regularization strengths influenced model stability and predictive power. Our results demonstrated that appropriate tuning of λ values significantly mitigated overfitting and enhanced generalization. Therefore, analyzing λ dependency was crucial for optimizing our regression models' performance, ensuring they were robust and effective when applied to both the Franke function-generated data and actual terrain data from the database.

B. Bias-Variance Tradeoff

explain how we recreate 2.11

we expect MSE=bias+var for data since noise var is 0

1. Bootstrap With OLS

model complexity, number of data points

2. Cross-Validation

go into detail how we implement

compare with bootstrap include Ridge and Lasso

C. The Program

1. Code Structure

2. Tools

raw and smoothed surfaces, explain reason/method

IV. RESULTS

V. DISCUSSION

VI. CONCLUSION

VII. REFLECTION

ACKNOWLEDGEMENTS

REFERENCES

[1] T. R. Q. James W. Wadsley, Benjamin W. Keller, "Gasoline2: A Modern SPH Code," https://arxiv.org/abs/1707.03824 (Accessed: September 2024).

[2] O. Hahn, "MUSIC," https://www-n.oca.eu/ohahn/MUSIC/ (Accessed: September 2024).

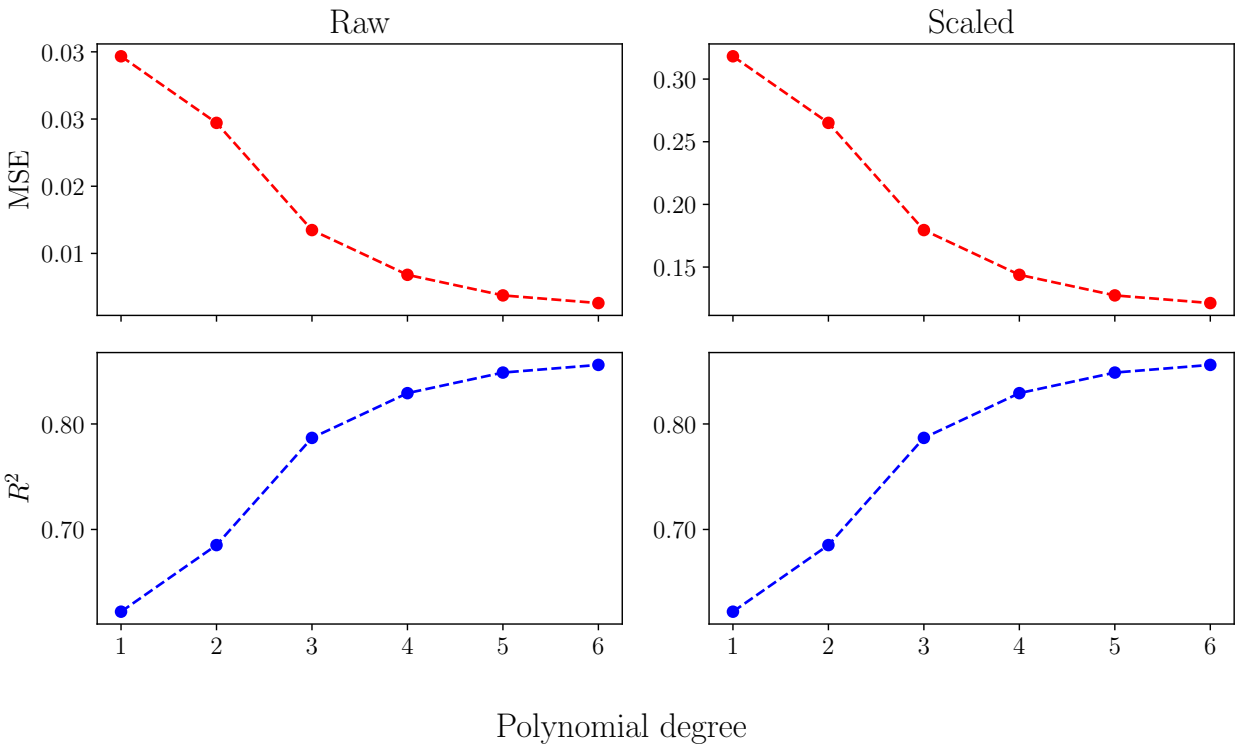

FIG. 2: caption remove figure title?

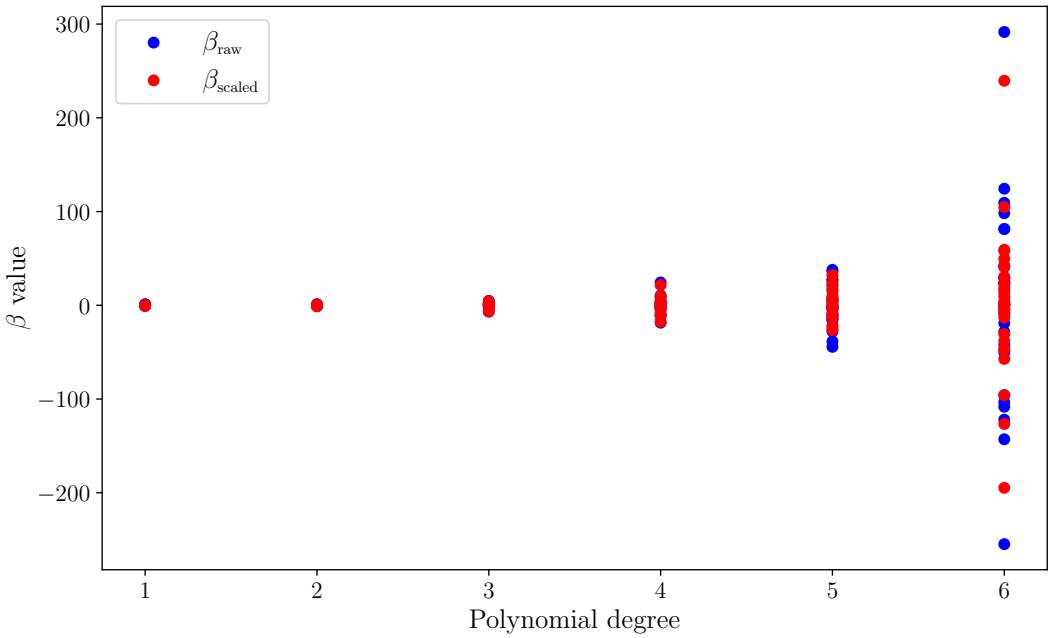

FIG. 3: caption remove figure title?

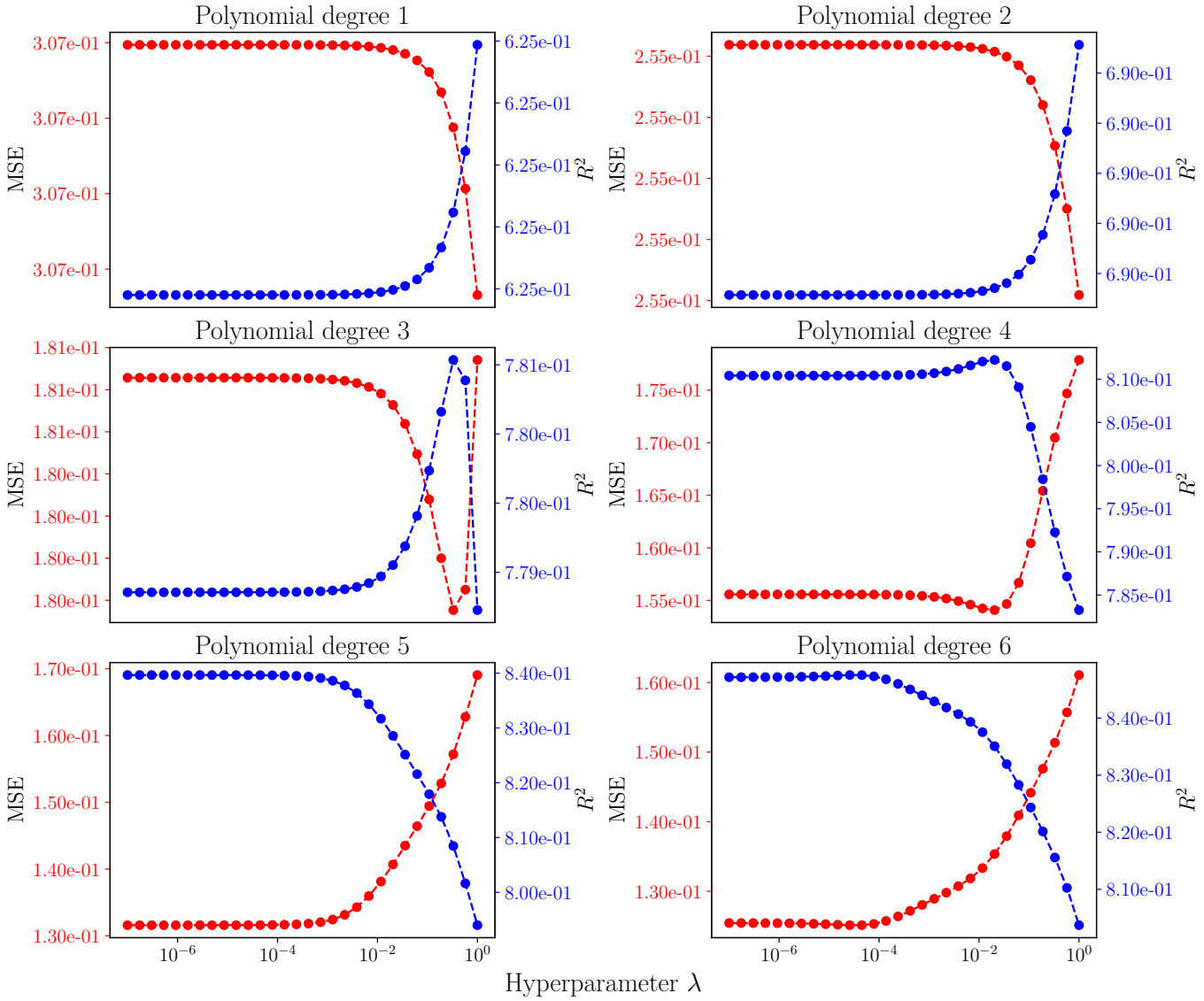

FIG. 4: caption

Appendix A: Derivations

1. Expectation Value and Variance of β_{OLS}

rewrite to fit report. use hats instead of OLS. change T

Since the error in $\mathbf{y}$ is normal distributed as $\boldsymbol{\varepsilon} \sim N(0, \sigma^2)$ we know that the expectation value and variance of the i'th element of $\boldsymbol{\varepsilon}$ is $\mathbb{E}(\varepsilon_i) = 0$ and $\mathrm{Var}(\varepsilon_i) = \sigma^2$. Thus, since we approximate $f(\mathbf{x})$ with $\tilde{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta}$ the expectation value of $\mathbf{y}$ becomes

$$\mathbb{E}(\mathbf{y}) = \mathbb{E}(\mathbf{X}\boldsymbol{\beta}) + \mathbb{E}(\boldsymbol{\varepsilon}) = \mathbb{E}(\mathbf{X}\boldsymbol{\beta}), \tag{A1}$$

which gives us the expectation value of $\mathbf{y}$ for a given element i:

$$\mathbb{E}(y_i) = \mathbb{E}\left(\sum_j X_{ij}\beta_j\right) = \sum_j X_{ij}\beta_j = \mathbf{X}_{i,*}\boldsymbol{\beta}. \tag{A2}$$

Here we have used that the sum $\sum_j X_{ij}\beta_j$ is known to be the value of $\tilde{y}_i$, hence its expectation value is itself. Moreover, we can similarly find the variance of $\mathbf{y}$ for a given element i by using that the aforementioned sum is known to be $\tilde{y}_i$

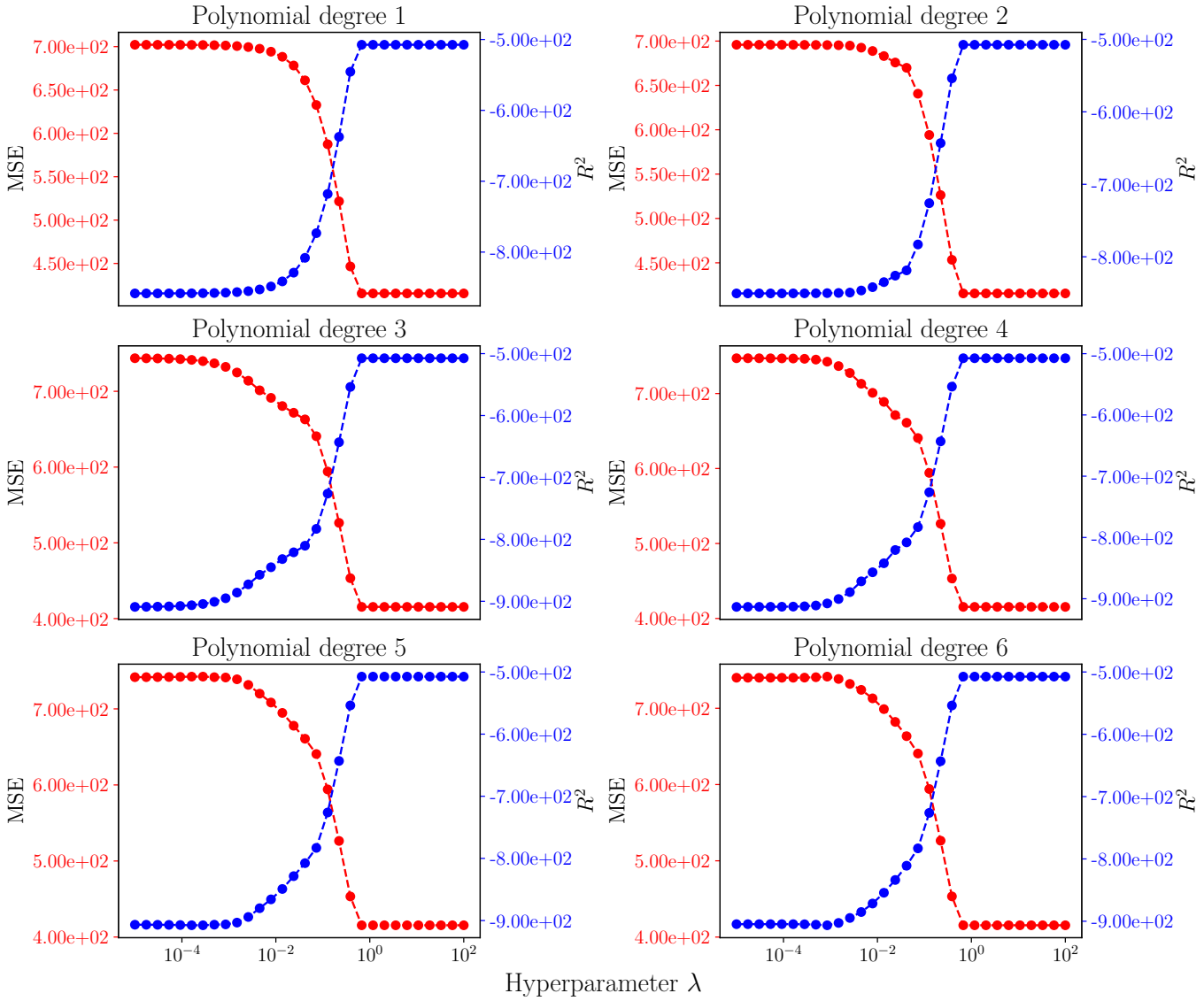

FIG. 5: caption

for all i, i.e. $\mathrm{Var}(\mathbf{X}_{i,*}\boldsymbol{\beta}) = 0$. Thus, we have

$$\mathrm{Var}(y_i) = \mathrm{Var}(\mathbf{X}_{i,*}\boldsymbol{\beta}) + \mathrm{Var}(\varepsilon_i) = \sigma^2, \tag{A3}$$

and consequently

$$y_i \sim N(\mathbf{X}_{i,*}\boldsymbol{\beta}, \sigma^2). \tag{A4}$$

Now, using that the optimal parameters in OLS are given by $\boldsymbol{\beta}_{\mathrm{OLS}} = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}$, their expectation values become

$$\begin{aligned}
\mathbb{E}(\boldsymbol{\beta}_{\mathrm{OLS}}) &= \mathbb{E}\left\{\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y}\right\} \\
&= \mathbb{E}\left\{\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\right\}\mathbb{E}(\mathbf{y}) \\
&= \left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{X}\boldsymbol{\beta} \\
&= \boldsymbol{\beta}.
\end{aligned} \tag{A5}$$

Furthermore, if x and y are two independent variables, the variance of their product is given by

$$\begin{aligned}
\mathrm{Var}(xy) &= \mathbb{E}(x^2y^2) - (\mathbb{E}(xy))^2, \\
&= \mathbb{E}(x^2)\mathbb{E}(y^2) - (\mathbb{E}(x))^2 (\mathbb{E}(y))^2, \\
&= \left[\mathbb{E}(x^2) - (\mathbb{E}(x))^2 + (\mathbb{E}(x))^2\right] \left[\mathbb{E}(y^2) - (\mathbb{E}(y))^2 + (\mathbb{E}(y))^2\right] - \mathbb{E}(x^2)\mathbb{E}(y^2), \\
&= \left[\mathrm{Var}(x) + (\mathbb{E}(x))^2\right] \left[\mathrm{Var}(y) + (\mathbb{E}(y))^2\right] - \mathbb{E}(x^2)\mathbb{E}(y^2), \\
&= \mathrm{Var}(x)\mathrm{Var}(y) + \mathrm{Var}(x)(\mathbb{E}(y))^2 + \mathrm{Var}(y)(\mathbb{E}(x))^2 + \mathbb{E}(x^2)\mathbb{E}(y^2) - \mathbb{E}(x^2)\mathbb{E}(y^2), \\
&= \mathrm{Var}(x)\mathrm{Var}(y) + \mathrm{Var}(x)(\mathbb{E}(y))^2 + \mathrm{Var}(y)(\mathbb{E}(x))^2,
\end{aligned}$$

so if we now set $x = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}$ and $y = \mathbf{y}$ we find that the variance of $\boldsymbol{\beta}_{\mathrm{OLS}}$ is

$$\begin{aligned}
\mathrm{Var}(\boldsymbol{\beta}_{\mathrm{OLS}}) &= \underbrace{\mathrm{Var}\left[\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\right]\mathrm{Var}(\mathbf{y})}_{0} + \underbrace{\mathrm{Var}\left[\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\right](\mathbb{E}(\mathbf{y}))^2}_{0} + \mathrm{Var}(\mathbf{y})\left(\mathbb{E}\left[\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\right]\right)^2, \\
&= \sigma^2\left[\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\right]^2, \\
&= \sigma^2\left[\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\right]\left[\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\right]^{\mathrm{T}}, \\
&= \sigma^2\left[\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\right]\left[\mathbf{X}\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\right], \\
&= \sigma^2\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}.
\end{aligned} \tag{A6}$$

Here we have used that the transpose of $\left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}$ is itself since it is square and symmetric.

2. Alternative Expression for MSE

rewrite to fit report

Substituting $\mathbf{y}$ with $f(\mathbf{x}) + \boldsymbol{\epsilon}$, and adding and subtracting $\mathbb{E}\left[\tilde{\mathbf{y}}\right]$, we find that

$$\begin{aligned}
\mathbb{E}\left[(\mathbf{y} - \tilde{\mathbf{y}})^2\right] &= \mathbb{E}\Big[(\underbrace{f(\mathbf{x})}_{\mathbf{f}} + \boldsymbol{\epsilon} - \tilde{\mathbf{y}})^2\Big], \\
&= \mathbb{E}\left[(\mathbf{f} + \boldsymbol{\epsilon} - \tilde{\mathbf{y}} + \mathbb{E}\left[\tilde{\mathbf{y}}\right] - \mathbb{E}\left[\tilde{\mathbf{y}}\right])^2\right], \\
&= \mathbb{E}\Big[\mathbf{f}^2 + \mathbf{f}\boldsymbol{\epsilon} - \mathbf{f}\tilde{\mathbf{y}} + \mathbf{f}\mathbb{E}\left[\tilde{\mathbf{y}}\right] - \mathbf{f}\mathbb{E}\left[\tilde{\mathbf{y}}\right] \\
&\quad + \boldsymbol{\epsilon}\mathbf{f} + \boldsymbol{\epsilon}^2 - \boldsymbol{\epsilon}\tilde{\mathbf{y}} + \boldsymbol{\epsilon}\mathbb{E}\left[\tilde{\mathbf{y}}\right] - \boldsymbol{\epsilon}\mathbb{E}\left[\tilde{\mathbf{y}}\right] \\
&\quad - \tilde{\mathbf{y}}\mathbf{f} - \tilde{\mathbf{y}}\boldsymbol{\epsilon} + \tilde{\mathbf{y}}^2 - \tilde{\mathbf{y}}\mathbb{E}\left[\tilde{\mathbf{y}}\right] + \tilde{\mathbf{y}}\mathbb{E}\left[\tilde{\mathbf{y}}\right] \\
&\quad + \mathbb{E}\left[\tilde{\mathbf{y}}\right]\mathbf{f} + \mathbb{E}\left[\tilde{\mathbf{y}}\right]\boldsymbol{\epsilon} - \mathbb{E}\left[\tilde{\mathbf{y}}\right]\tilde{\mathbf{y}} + \left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2 - \left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2 \\
&\quad - \mathbb{E}\left[\tilde{\mathbf{y}}\right]\mathbf{f} - \mathbb{E}\left[\tilde{\mathbf{y}}\right]\boldsymbol{\epsilon} + \mathbb{E}\left[\tilde{\mathbf{y}}\right]\tilde{\mathbf{y}} - \left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2 + \left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2\Big], \\
&= \mathbb{E}\Big[\mathbf{f}^2 + \mathbf{f}\boldsymbol{\epsilon} + \boldsymbol{\epsilon}\mathbf{f} + \boldsymbol{\epsilon}^2 - \mathbf{f}\mathbb{E}\left[\tilde{\mathbf{y}}\right] - \boldsymbol{\epsilon}\mathbb{E}\left[\tilde{\mathbf{y}}\right] - \mathbb{E}\left[\tilde{\mathbf{y}}\right]\mathbf{f} - \mathbb{E}\left[\tilde{\mathbf{y}}\right]\boldsymbol{\epsilon} + \left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2\Big] \\
&\quad + \mathbb{E}\Big[\tilde{\mathbf{y}}^2 - \tilde{\mathbf{y}}\mathbb{E}\left[\tilde{\mathbf{y}}\right] - \mathbb{E}\left[\tilde{\mathbf{y}}\right]\tilde{\mathbf{y}} + \left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2\Big] \\
&\quad + \mathbb{E}\Big[-\mathbf{f}\tilde{\mathbf{y}} - \boldsymbol{\epsilon}\tilde{\mathbf{y}} + \mathbf{f}\mathbb{E}\left[\tilde{\mathbf{y}}\right] + \boldsymbol{\epsilon}\mathbb{E}\left[\tilde{\mathbf{y}}\right] - \tilde{\mathbf{y}}\mathbf{f} - \tilde{\mathbf{y}}\boldsymbol{\epsilon} + \tilde{\mathbf{y}}\mathbb{E}\left[\tilde{\mathbf{y}}\right] + \mathbb{E}\left[\tilde{\mathbf{y}}\right]\mathbf{f} + \mathbb{E}\left[\tilde{\mathbf{y}}\right]\boldsymbol{\epsilon} + \mathbb{E}\left[\tilde{\mathbf{y}}\right]\tilde{\mathbf{y}} - 2\left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2\Big].
\end{aligned}$$

Before we move further we may note that the exact function $f(\mathbf{x})$ generally is not known, and we may therefore assume that our data is a good representation and replace $\mathbf{f}$ with $\mathbf{y}$ in the expression above. In practise this $\mathbf{y}$ is then the part of the data set that we have chosen as test set, while the model is made with the remaining data set (the training set). correct? Thus, using that $\mathbb{E}[\mathbb{E}[\mathbf{x}]] = \mathbb{E}[\mathbf{x}]$, $\mathbb{E}[(\mathbb{E}[\mathbf{x}])^2] = (\mathbb{E}[\mathbf{x}])^2$ and $\mathbb{E}[\mathbf{x}, \mathbf{y}] = \mathbb{E}[\mathbf{x}]\mathbb{E}[\mathbf{y}]$ for any

statistically independent $\mathbf{x}$ and $\mathbf{y}$, and that $\mathbb{E}[\boldsymbol{\epsilon}] = 0$ so that we can remove all first order terms in $\boldsymbol{\epsilon}$, we get

$$\begin{aligned}
\mathbb{E}\left[(\mathbf{y} - \tilde{\mathbf{y}})^2\right] &= \mathbb{E}\Big[\mathbf{y}^2 + \boldsymbol{\epsilon}^2 - \mathbf{y}\mathbb{E}\left[\tilde{\mathbf{y}}\right] - \mathbb{E}\left[\tilde{\mathbf{y}}\right]\mathbf{y} + \left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2\Big] \\
&\quad + \mathbb{E}\Big[\tilde{\mathbf{y}}^2 - \tilde{\mathbf{y}}\mathbb{E}\left[\tilde{\mathbf{y}}\right] - \mathbb{E}\left[\tilde{\mathbf{y}}\right]\tilde{\mathbf{y}} + \left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2\Big] \\
&\quad + \mathbb{E}\Big[-\mathbf{y}\tilde{\mathbf{y}} + \mathbf{y}\mathbb{E}\left[\tilde{\mathbf{y}}\right] - \tilde{\mathbf{y}}\mathbf{y} + \tilde{\mathbf{y}}\mathbb{E}\left[\tilde{\mathbf{y}}\right] + \mathbb{E}\left[\tilde{\mathbf{y}}\right]\mathbf{y} + \mathbb{E}\left[\tilde{\mathbf{y}}\right]\tilde{\mathbf{y}} - 2\left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2\Big], \\
&= \mathbb{E}\Big[\left(\mathbf{y} - \mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2\Big] + \mathbb{E}\big[\boldsymbol{\epsilon}^2\big] + \mathbb{E}\Big[\left(\tilde{\mathbf{y}} - \mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2\Big] \\
&\quad - \mathbb{E}[\mathbf{y}]\mathbb{E}[\tilde{\mathbf{y}}] + \mathbb{E}[\mathbf{y}]\mathbb{E}[\tilde{\mathbf{y}}] - \mathbb{E}[\tilde{\mathbf{y}}]\mathbb{E}[\mathbf{y}] + \mathbb{E}[\tilde{\mathbf{y}}]\mathbb{E}[\tilde{\mathbf{y}}] + \mathbb{E}[\tilde{\mathbf{y}}]\mathbb{E}[\mathbf{y}] + \mathbb{E}[\tilde{\mathbf{y}}]\mathbb{E}[\tilde{\mathbf{y}}] - 2\left(\mathbb{E}\left[\tilde{\mathbf{y}}\right]\right)^2, \\
&= \mathrm{Bias}[\tilde{y}] + \mathrm{Var}[\tilde{y}] + \sigma^2. \qquad \text{(A7)}
\end{aligned}$$

Appendix B: Additional Figures