Regression Techniques
FYS-STK4155
Assignment One

Ivar Haugaløkken Stangeby

September 9, 2019

1 Introduction

Regression analysis is the act of estimating relationships among variables. In this project, we study various regression methods in more detail. In particular, we compare the *ordinary least squares* (OLS) method with the *Ridge regression* and *Lasso regression* techniques. As we shall see, these three methods are all variations over the same theme. We start by testing the methods on noisy data sampled from a function known as *Franke's* bivariate test function, which has been in widespread use in the evaluation of interpolation and data fitting techniques. Finally, we run regression on real terrain data, comparing the aforementioned methods.

2 Least Squares

We start by giving an introduction to ordinary least square regression. This method is perhaps the most widely used regression technique due to its simplicity, both in its

derivation, and its implementation. The goal of regression analysis is to fit a *model function* $f(\boldsymbol{x}, \beta)$ to a set of n data points $\Omega = (\boldsymbol{x}_i, y_i)_{i=1}^n$. A simple example is that of a linear polynomial with two parameters:

$$f(x, \beta) = \beta_0 + \beta_1 x. \tag{1}$$

The *model parameters* β are determined in order to minimize a suitable *cost function* $C(\Omega, \beta)$ which measures to which extent the model function manages to capture trends in the data Ω.

One common cost function is the one which involves the sum of squared residuals (or squared errors):

$$C(\Omega, \beta) = \frac{1}{n} \sum_{i=1}^{n} \varepsilon_i^2 \ := \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{2}$$

where $\hat{y}_i \ := f(\boldsymbol{x}_i, \beta)$. The method involving the minimization of this specific cost function is known as *least squares*.

Remark 1. It is often assumed a priori that the data is infact generated from a noisy model, such that each y_i can be described as

$$y_i = f(\boldsymbol{x}_i) + \varepsilon_i \tag{3}$$

where each $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$ is normally distributed with zero mean and variance σ^2.

2.1 General linear models.

We are often interested in finding the best model function in a specific function space. Assuming this space has a basis $\mathcal{B} = \left\{ \varphi_i \right\}_{i=1}^{M}$, we may write our model function in terms of the basis functions and the model parameters as

$$f(\boldsymbol{x}, \beta) \ := \sum_{i=1}^{M} \beta_i \varphi_i(\boldsymbol{x}). \tag{4}$$

With this formulation, we can represent the approximation $\hat{\boldsymbol{y}}$ to $\boldsymbol{y}$ as a matrix product

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\beta \tag{5}$$

where $\boldsymbol{X}$ is the *design matrix* defined by components $\boldsymbol{X}_{ij} = \varphi_j(\boldsymbol{x}_i)$, and $\beta = [\beta_0, \dots, \beta_M]$ is the model parameters. With this notation in mind, we can also write the cost function as

$$C(\Omega, \beta) = \frac{1}{n}(\boldsymbol{y} - \hat{\boldsymbol{y}})^T(\boldsymbol{y} - \hat{\boldsymbol{y}}). \tag{6}$$

Optimizing the parameters of a general linear model. Assume now that our model is of the form given in Equation (4) and that our cost function is the mean squared error defined in Equation (2). We are interested in finding the parameters β that minimize the cost function $C(\Omega, \beta)$. Since $C(\Omega, \beta)$ is convex, it suffices to differentiate with respect to β and equating to zero.

We have that

$$\frac{\partial C(\Omega, \beta)}{\partial \beta} = \boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\beta), \tag{7}$$

and equating this to zero yields the familiar *normal equations*

$$\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{X}^T\boldsymbol{X}\beta. \tag{8}$$

If the matrix $\boldsymbol{X}^T\boldsymbol{X}$ is invertible, we may obtain the solution by direct numerical matrix inversion. In this case, the optimal model parameters are found directly by

$$\beta = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}. \tag{9}$$

However, these matrices may be ill-conditioned when the number of equations are very large, and it is therefore common to apply approximate solvers for the inverse, for instance using low-rank approximation based on SVD-decomposition, which we will briefly turn to in the following.

Singular value decomposition. Recall that any matrix $\boldsymbol{A} \in \mathbb{C}^{n,m}$ can be decomposed as

$$\boldsymbol{A} = \boldsymbol{U}\Sigma\boldsymbol{V}^T, \tag{10}$$

where $\boldsymbol{U}$ and $\boldsymbol{V}$ comprised of the eigenvectors of $\boldsymbol{A}\boldsymbol{A}^T$ and $\boldsymbol{A}^T\boldsymbol{A}$ respectively. As these eigenvectors are orthonormal, it follows that both U and V are unitary matrices. Furthermore, Σ is a matrix

$$\Sigma = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix}, \tag{11}$$

where Σ_1 is a square diagonal matrix of size $r \times r$ with the non-zero singular values of $\boldsymbol{A}$. The integer r is the *rank* of $\boldsymbol{A}$. As the matrix Σ is mostly containing zeros, the information stored in $\boldsymbol{A}$ is attributed to only some parts of $\boldsymbol{U}$ and $\boldsymbol{V}$. We can remove the redundant parts, and more compactly express $\boldsymbol{A}$ as $\boldsymbol{A} = \boldsymbol{U}_1\Sigma_1\boldsymbol{V}_1^T$. Here, $\boldsymbol{U}_1$ is $m \times r$ and $\boldsymbol{V}_1$ is $n \times r$.