https://doi.org/10.26219/heal.aueb.5567

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

SCHOOL OF INFORMATION SCIENCES & TECHNOLOGY

DEPARTMENT OF STATISTICS

POSTGRADUATE PROGRAM

Big Data Regression

By
ELENI P. KONTOU

A Thesis
Submitted to the Department of Statistics
of the Athens University of Economics and Business
in partial fulfilment of the requirements for
the degree of Master of Science in Statistics

Athens, Greece
March 2017

https://doi.org/10.26219/heal.aueb.5567

https://doi.org/10.26219/heal.aueb.5567

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

ΣΧΟΛΗ ΕΠΙΣΤΗΜΩΝ & ΤΕΧΝΟΛΟΓΙΑΣ ΤΗΣ ΠΛΗΡΟΦΟΡΙΑΣ

ΤΜΗΜΑ ΣΤΑΤΙΣΤΙΚΗΣ

ΜΕΤΑΠΤΥΧΙΑΚΟ

Παλινδρόμηση για Μεγάλα Δεδομένα

ΕΛΕΝΗ Π. ΚΟΝΤΟΥ

ΔΙΑΤΡΙΒΗ
Που Υποβλήθηκε στο Τμήμα Στατιστικής
του Οικονομικού Πανεπιστημίου Αθηνών
ως μέρος των απαιτήσεων για την απόκτηση
Μεταπτυχιακού Διπλώματος Ειδίκευσης στη Στατιστική

Αθήνα
Μάρτιος 2017

 https://doi.org/10.26219/heal.aueb.5567

ACKNOWLEDGEMENTS

First, I would first like to thank my supervisor Dimitris Karlis for his guidance and patience. The door to Prof. Karlis office was always open whenever I had a question about my thesis. Without his contribution this MSc Thesis would have never been fulfilled. Most of all, I appreciate the trust and the encouragement he offered me throughout these months.

I am also grateful to my wonderful parents, Panagiotis and Victoria, and my brother, Nikos, for my upbringing and their support not only during my studies, but also in my life. In addition, I would like to express my gratitude to my friends, Vasia, Maria and Maria, who always stand by me in every aspect of my life. I am also thankful to Dimitra and Fay for master's journey. My acknowledgements cannot end here without me referring to George, who believes in me and I wholeheartedly thank him for his unlimited patience.

https://doi.org/10.26219/heal.aueb.5567

https://doi.org/10.26219/heal.aueb.5567

VITA

I was born on January 4th, 1993 in Chalkida, where I got my High School Diploma. In 2011 I entered the department of Mathematics at the University of Athens. I graduated in 2015 and I enrolled in the Master of Science in Statistics at Athens University of Economics and Business.

https://doi.org/10.26219/heal.aueb.5567

https://doi.org/10.26219/heal.aueb.5567

ABSTRACT

Eleni Kontou

Big Data Regression

March 2017

We are entering the era of Big Data. Half a century after computers entered in our society, data has begun to be the center of attention. Not only there is massive information, which has never existed before, but also this information is proliferating faster and has a variety of forms. Therefore, Big Data brings new opportunities for the development of society and poses new challenges to data scientists. It has unique features that are not shared by the traditional data sets. Specifically, it is characterized by high dimensionality and large sample size. Due to this, ordinary statistical methods do not work. Thus, we need new effective statistical procedures and computational methods.

In the current thesis, we will study methods that deal with regression for large datasets. After we review such existing methodologies, we will examine an updating method, which estimates the regression coefficients and is faster than the ordinary least square estimation. We will also investigate the regression by orthogonalization and regression using QR decomposition. Afterwards, we will compare the methods through their respective running time that is required in order to estimate the regression coefficients. In the same way, generalised linear models are studied and an updating method for them is attempted. Finally, several proposals for deeper exploration of the previous subjects will be discussed.

https://doi.org/10.26219/heal.aueb.5567

https://doi.org/10.26219/heal.aueb.5567

ΠΕΡΙΛΗΨΗ

Ελένη Κοντού

Παλινδρόμηση για Μεγάλα Δεδομένα

Μάρτιος 2017

Έχουμε εισέλθει στην εποχή των Μεγάλων Δεδομένων. Μόλις μισό αιώνα από τότε που οι υπολογιστές εισέβαλλαν στην κοινωνία μας, τα δεδομένα έχουν αρχίσει να γίνονται το επίκεντρο της προσοχής. Όχι μόνο υπάρχει τεράστιος όγκος πληροφορίας, ο οποίος δεν υπήρχε παλιότερα, αλλά επίσης αυτή η πληροφορία πολλαπλασιάζεται γρηγορότερα και έχει μία μεγάλη ποικιλία μορφών. Γι αυτό, τα Μεγάλα Δεδομένα δίνουν νέες ευκαιρίες για την ανάπτυξη της κοινωνίας και θέτουν νέες προκλήσεις στους ερευνητές της επιστήμης των δεδομένων. Έχουν μοναδικά χαρακτηριστικά που δεν συναντάμε στα παραδοσιακά σύνολα δεδομένων. Συγκεκριμένα, έχουν συνήθως πολλές διαστάσεις και μεγάλο μέγεθος δείγματος. Εξαιτίας αυτού, οι συνηθισμένες μεθόδοι δεν δουλεύουν. Έτσι, χρειαζόμαστε νέες αποτελεσματικές στατιστικές διαδικασίες και υπολογιστικές μεθόδους.

Στην παρούσα διπλωματική, θα μελετήσουμε μεθόδους για παλινδρόμηση σε μεγάλα σύνολα δεδομένων. Αφού αναλύσουμε τέτοιες ήδη υπάρχουσες μεθοδολογίες, θα εξετάσουμε μία μέθοδο που χρησιμοποιεί έναν αναδρομικό τύπο και κάνει συνεχή ενημέρωση. Αυτή εκτιμάει τους συντελεστές της γραμμικής παλινδρόμησης και είναι γρηγορότερη από τη συνηθισμένη εκτίμηση ελαχίστων τετραγώνων. Επίσης, θα ασχοληθούμε με τη παλινδρόμηση που χρησιμοποιεί ορθογωνοποίηση και τη παλινδρόμηση με QR παραγοντοποίηση. Έπειτα, θα συγκρίνουμε όλες αυτές τις μεθόδους με βάση τον αντίστοιχο χρόνο που απαιτείται για να εκτιμήσουν αυτούς τους συντελεστές. Με τον ίδιο τρόπο θα μελετηθούν τα γενικευμένα γραμμικά μοντέλα και θα δοκιμάσουμε να φτιάξουμε μια παρόμοια μέθοδο, όπως στα γραμμικά μοντέλα. Στο τέλος, θα συζητηθούν διάφορες προτάσεις για βαθύτερη εξερεύνηση των προηγούμενων θεμάτων.

https://doi.org/10.26219/heal.aueb.5567

https://doi.org/10.26219/heal.aueb.5567

Contents

https://doi.org/10.26219/heal.aueb.5567

List of Tables

https://doi.org/10.26219/heal.aueb.5567

https://doi.org/10.26219/heal.aueb.5567

List of Figures

https://doi.org/10.26219/heal.aueb.5567

https://doi.org/10.26219/heal.aueb.5567

Chapter 1

Introduction

1.1 What is Big Data

We have entered the world of big data. We can find it everywhere, because it is being generated by everything around us every moment. Digital processes, social media and electrical devices, such as mobile phones, are just some examples, which contain data that can be collected and analysed. Nowadays, big data is ”fashionable” and it is essential to construct new methods in order to efficiently deal with it.

Big data is the term for too large datasets, where we cannot use the traditional ways to handle them. There are a lot of definitions for Big data. For instance, TechAmerica Foundation defines it as follows: ”Big data is a term that describes large volumes of high velocity, complex and variable data that require advanced techniques and technologies to enable the capture, storage, distribution, management, and analysis of the information.” (TechAmerica Foundation’s Federal Big Data Commission, 2012)

Big data can be described by several characteristics. The most famous are the three V’s (Volume, Variety, Velocity). Volume is related to the magnitude of the data. The size determines whether a dataset is big or not. Let’s give some examples. Gandomi A. and Haider M. (2015) report that ”one

https://doi.org/10.26219/heal.aueb.5567

terabyte stores as much data as would fit on 1500 CDs or 220 DVDs, enough to store around 16 million Facebook photographs and Facebook processes up to one million photographs per second.". In addition, estimates report that Facebook stored 260 billion photos using storage space of over 20 petabytes(1 petabytes=1024 terabytes). Variety concerns the type of data. It is not all the type of data structured, such as in spreadsheets. The social media, for example, provide text, images, audio, video, which are unstructured data. Moreover, the term velocity refers to the speed at which the data is generated. The flow of data is massive and continuous. For instance, Wal-Mart processes more than one million transactions per hour (Cukier, 2010)

IBM added Veracity as the forth V. The quality of the collected data can vary and affect the accurate analysis. In other words, the veracity is another challenge of big data, which is related to reliability. SAS refer two additional characteristics. The first one is the Variability. It refers to data whose meaning is constantly changing. It can have periodic peaks. The second characteristic, according to SAS, is the Complexity. Since the data are collected from multiple sources, it is difficult to be matched, cleansed and transformed.

Big data have surely affected the field of statistics. There are predominantly two problems that we can encounter in this framework. The first one is data that might have a large number of observations, called big-n data and the second one is data that might have a large number of features, called big-p data. These problems can be time and memory consuming. They can cause multicollinearity, high standard errors, while variables that are not significant appear as significant etc. Thus, we need to overcome all of these Big Data challenges. Because of its characteristics, the ordinary data processing applications are inadequate to deal with big datasets. Therefore, new methods, techniques and algorithms have been developed.

https://doi.org/10.26219/heal.aueb.5567

1.2 Linear Regression

In several problems in statistics, it is not enough to study the characteristics of some random variables, such as mean, median, variance etc. It is also important to find how they are associated with each other. Therefore, we use regression analysis. With regression analysis we can construct models which describe or explain relationships that may exist between variables. For example, we want to study the relationship between the risk of stroke and other variables such as age, sex, blood pressure, and so on.

The two basic types of regression are the simple linear regression and the multiple linear regression. The former includes only two variables, while the latter uses more variables, such as in the previous example. Specifically, we are usually interested in just one variable, known as the response or dependent variable (notation: $\boldsymbol{Y}$). The variables that are manipulated by researchers and predict the values of the dependent variable in the model are called independent / explanatory variables or covariates (notation: $\boldsymbol{X_1}, \ldots, \boldsymbol{X_p}$). We use regression models to describe and understand the association between the variables and in order to predict future values of $\boldsymbol{Y}$.

We assume that we have n observations. So, the simple linear regression model has only one independent variable and can be written as

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2),$$

where $i = 1, \ldots, n$, β_0 is the intercept, β_1 is the slope, ϵ_i is the error variable that adds noise to the linear relationship and we assume that $\epsilon_i \sim N(0, \sigma^2)$, i.e. ϵ_i follows normal distribution with $E(\epsilon_i) = 0$ and $Var(\epsilon_i) = \sigma^2$. If there are a lot of independent variables, we have the multiple regression model

$$Y_i = \beta_0 + \beta_1 X_{1,i} + \ldots + \beta_p X_{p,i} + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2),$$

where β_j $(j = 0, \ldots, p)$ are the unknown parameters to be estimated. Be-

https://doi.org/10.26219/heal.aueb.5567

cause of the potentially large number of predictors, it is more efficient to use matrices to define the regression model. Thus, writing these n equations in matrix form, we get

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} 1 & X_{1,1} & \dots & X_{p,1} \\ 1 & X_{1,2} & \dots & X_{p,2} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & X_{1,n} & \dots & X_{p,n} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix},$$

equivalently

$$\boldsymbol{Y} = \boldsymbol{X\beta} + \boldsymbol{\epsilon}.$$

Our aim is to estimate the parameters vector $\boldsymbol{\beta}$. We will use the least square estimation. So, we have to minimize the $\sum_{i=1}^{n} \epsilon_i^2$ or $\boldsymbol{\epsilon^T\epsilon} = (\boldsymbol{Y} - \boldsymbol{X\beta})^{\boldsymbol{T}}(\boldsymbol{Y} - \mathbf{X}\boldsymbol{\beta})$(Seber and Lee, 2003). Using $\boldsymbol{\beta^T X^T Y} = (\boldsymbol{\beta^T X^T Y})^{\boldsymbol{T}} = \boldsymbol{Y^T X \beta}$ we get

$$\boldsymbol{\epsilon^T\epsilon} = \boldsymbol{Y^TY} - 2\boldsymbol{\beta^TX^TY} + \boldsymbol{\beta^TX^TX\beta},$$

and then

$$\begin{aligned} \frac{\partial \boldsymbol{\epsilon^T\epsilon}}{\partial \boldsymbol{\beta}} &= 0 \\ -2\boldsymbol{X^TY} + 2\boldsymbol{X^TX\beta} &= 0 \\ \mathbf{X^T}\boldsymbol{X\beta} &= \boldsymbol{X^TY}, \end{aligned}$$

using the below properties

$$\frac{d(\boldsymbol{\beta^T\alpha})}{d\boldsymbol{\beta}} = \boldsymbol{a} \qquad and \qquad \frac{d(\boldsymbol{\beta^TA\beta})}{d\boldsymbol{\beta}} = 2\boldsymbol{A\beta}.$$

Finally to solve this equation, it is required the matrix $\boldsymbol{X}$ has full rank,

https://doi.org/10.26219/heal.aueb.5567

namely the columns of $\boldsymbol{X}$ are linearly independent. Thus,

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T Y}.$$

It is worth mentioning the properties of least squares estimators. If we assume that the errors are unbiased, uncorrelated and have the same variance, we obtain

$$E(\boldsymbol{Y}) = \boldsymbol{X\beta} \qquad and \qquad Var(\boldsymbol{Y}) = Var(\boldsymbol{\epsilon}) = \sigma^2 \boldsymbol{I_n}.$$

Hence,

$$\begin{aligned} E(\hat{\boldsymbol{\beta}}) &= (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T} E(\boldsymbol{Y}) \\ &= (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T X\beta} \\ &= \boldsymbol{\beta}, \end{aligned}$$

namely $\hat{\boldsymbol{\beta}}$ is an unbiased estimator of $\boldsymbol{\beta}$ and

$$\begin{aligned} Var(\hat{\boldsymbol{\beta}}) &= Var((\boldsymbol{X^T X})^{-1}\boldsymbol{X^T Y}) \\ &= (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T} Var(\boldsymbol{Y})\boldsymbol{X(X^T X})^{-1} \\ &= \sigma^2(\boldsymbol{X^T X})^{-1}(\boldsymbol{X^T X})(\boldsymbol{X^T X})^{-1} \\ &= \sigma^2(\boldsymbol{X^T X})^{-1}. \end{aligned}$$

The fitted values $(\boldsymbol{X}\hat{\boldsymbol{\beta}})$ are denoted by $\hat{\boldsymbol{Y}}$. So, the vector $\boldsymbol{e} = \boldsymbol{Y} - \hat{\boldsymbol{Y}}$, which is called residuals, gives the estimation of errors.

The linear regression models have a number of assumptions. The most important of these, which have to be checked, are: independence of errors (and Y_i), normality of errors (and Y_i), homoscedasticity of errors (and Y_i), linearity between $\boldsymbol{X}$ and $\boldsymbol{Y}$. Of course, in order to check these assumptions, we work with the residuals.

https://doi.org/10.26219/heal.aueb.5567

1.3 Purpose of the thesis

In the present thesis we will study regression for big data. As we have already mentioned, we cannot handle large datasets with the ordinary techniques. This is due to the distinct characteristics of big data, which we described previously. We will mainly focus on the problem of the huge number of observations or the large number of variables.

Let's assume that we have a linear regression model

$$Y_i = \beta_0 + \beta_1 X_{1,i} + \ldots + \beta_p X_{p,i} + \epsilon_i, \quad \epsilon_i \sim N(0, \sigma^2),$$

with $i = 1, \ldots, n$. We proved that the regression coefficients can be estimated by

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X^T X})^{-1} \boldsymbol{X^T Y}.$$

It is obvious in big datasets, the inverse of the matrix $\boldsymbol{X^T X}$ will be incredibly time consuming to compute. Moreover, if we attempt to add a new variable in the regression model, we will have to calculate the inverse of the matrix from scratch. Thus, our aim is to find a faster technique to estimate the coefficients without the calculation of the inverse from the beginning.

There are several methods for big data regression in the literature and we will describe some of them. However, we will extensively study regression by orthogonalization and regression with QR decomposition. We are interested to know if these methods of regression are computationally faster than our technique and the ordinary least square estimation.

Furthermore, we intend to be expand to generalized linear models for big data. It is known that the estimation of the coefficients uses the iteratively reweighted least squares algorithm. Hence, we have the same aim, to computationally improve this algorithm in order to estimate the coefficients of the generalized linear model faster.

 https://doi.org/10.26219/heal.aueb.5567

1.4 Structure of the thesis

Now we will report the structure of the thesis. It is organized in four additional chapters. Chapter 2 reviews methods and techniques for regression for big datasets, which exist in the literature. We will start with the most commonly used simple techniques and then proceed to the more advanced methods. Furthermore, this chapter provides a survey of existing sampling methods applicable to the data splitting problem.

Chapter 3 is the center of attention of this thesis. It presents an updating regression, proving a recursive formula in order to estimate the regression coefficients. This estimation's technique is compared with the ordinary least squares estimation. In addition, this chapter describes the regression by orthogonalization and the regression using QR decomposition. In the end, all of these methods are compared together.

Chapter 4 focuses on the generalized linear models. At first, it includes an introduction to GLMs, sparsity and relevant methods. Furthermore, a subsampling method for the estimation of the coefficients of logistic regression is described. Afterwards, the iteratively reweighted least squares algorithm and our aim, which is to try to apply the updating formula in order to be faster than the previous algorithm, are explained in detail. We conclude this thesis with a short discussion and a description of some topics that can be investigated for further research in Chapter 5.

https://doi.org/10.26219/heal.aueb.5567

https://doi.org/10.26219/heal.aueb.5567

Chapter 2

Review of Regression for Big Data

2.1 Regression methods for Big Datasets

It is difficult to analyze big data. Because of the large volume of data, the traditional and the typical methods rarely work and the majority of computers cannot store all the data in their RAM. Therefore, there are several techniques in the literature, which handle the big data and overcome the computing burden. In this chapter, we provide some techniques and methods for the regression for big data.

Some of the techniques are very simple, but there is a possibility to have drawbacks. Others methods are efficient and others are computationally slow. Of course, there are and more complicated methods for regression. The most famous of these are described below.

2.1.1 Randomization

The simplest method is the selection of a subset of a data randomly (simple random sampling) and the analysis only of this. Namely, we apply regression

https://doi.org/10.26219/heal.aueb.5567

in the subdata, but the standard error of the coefficients of regression will be the subdata's standard error. Therefore, the estimation of the model error will have a high variance. It is preferable to use as much relevant information as possible. So, if we are not lucky, the estimation of the coefficients of the subdata will probably be different from the full data.

2.1.2 Divided Regression Analysis

It is a very common approach to handle big data (Jun, Lee & Ryu, 2015). Initially, we divide the data into smaller pieces (subdata), which are of the same size, for parallel analysis. There are many methods of data splitting, that will be described afterwards. Here, we choose the random split, in which each observation has the same probability to be selected from the subdata. Let k be the number of subadatasets, which we created by the full dataset. Then, we compute the regression coefficients in each of them. Let $\boldsymbol{\hat{B}_i}$ be the vector of regression coefficients of the i subdata. Thus, in order to combine the results, we use the mean. Namely, the regression coefficient of the big data will be

$$\boldsymbol{\hat{B}} = \frac{\sum_{i=1}^{k} \boldsymbol{\hat{B}}_i}{k}.$$

The mean as a function for combining is very simple and easily used. However, this is sensitive to extreme values. So if we do not have an appropriate splitting and one subdata is very different from the others, the mean can distort the results. Thus, we can also modify this approach by transforming the last step, such as the next method.

2.1.3 The Split-Analyze-Meta analyze approach (SAM)

According to Cheung and Jak (2016), this approach is similar to the previous one. The big dataset is divided into small datasets randomly and each of them is analyzed independently. We apply regression analysis and the

https://doi.org/10.26219/heal.aueb.5567

regression coefficients are returned for any subdataset. However, in order to combine our findings, we will use meta analysis.

Generally speaking, **meta analysis** is a statistical procedure for combining results of multiple scientific studies. The results usually vary from one study to the other, so we cannot base on only one study. There is a common statistical measure among the studies and it is called effect size. In meta analysis we have to make an important decision , which is whether we will use a fixed-effects or a random-effects model(Haidich, 2010). A fixed-effect model assumes that all the differences in the observed effect size are based on the sampling error. The models are homogeneous and each study expected the same effect. Contrariwise, when there are differences between the studies, we use random-effects model. It is appropriate when there is a statistical heterogeneity in the results of studies.

In the SAM approach, each of the subdataset is considered as a study and the parameter estimates are the effect sizes in the studies. It is obvious that since the big dataset was divided randomly, we know that we will use fixed-effects model because the same thing has been measured. In other words, the parameters in the studies are assumed to be equal. It is deemed that all the studies have a common effect(the vector of regression coefficients) and each of the study calculate an estimation of this, the $\hat{\boldsymbol{b}}_{\boldsymbol{i}}$. Thus, we will combine the results using the formula

$$\hat{\boldsymbol{B}} = \frac{\sum_{i=1}^{k} w_i \hat{\boldsymbol{b}}_{\boldsymbol{i}}}{\sum_{i=1}^{k} w_i},$$

where k is the number of subdatasets and w_i is the inverse of $se(\hat{\boldsymbol{b}}_{\boldsymbol{i}})^2$ (Nieminen et al., 2013). We use the weights, since each study contributes differently to the estimates of overall effect.

https://doi.org/10.26219/heal.aueb.5567

2.1.4 Using sufficient statistics

It is known that the regression coefficients are calculated using the formula $\hat{\boldsymbol{\beta}} = (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T Y}$. In large datasets, the inversion of $\boldsymbol{X^T X}$ is time consuming, thus it is a good idea to split the dataset in parts, work with each one separately and update the results. In this way, only a portion of data is on the memory.

For example, we assume that we have a dataset with n observations and $p-1$ explanatory variables and we split it into two subdatasets. The first subdataset has the first k observations and the second one has from the $k{+}1^{th}$ to n^{th} observation. We also denote the design matrix of the i subdataset as $\boldsymbol{X_{(i)}}$ and the response of the correspondent subdataset as $\boldsymbol{Y_{(i)}}$. Furthermore, we write $X_{j,k}$ for the j^{th} variable measured for the k^{th} observation. Then, we calculate the regression coefficients of the first subdata with the known formula and we store the matrices as

$$\mathbf{C}_1 = \boldsymbol{X^T_{(1)} X_{(1)}} \qquad \textit{and} \qquad \mathbf{d}_1 = \boldsymbol{X^T_{(1)} Y_{(1)}},$$

where

$$\mathbf{C}_1 = \boldsymbol{X^T_{(1)} X_{(1)}} = \begin{bmatrix} k & \sum_{i=1}^k (X_{1,i}) & \dots & \sum_{i=1}^k (X_{p,i}) \\ \sum_{i=1}^k (X_{1,i}) & \sum_{i=1}^k (X_{1,i})^2 & \dots & \sum_{i=1}^k (X_{1,i}X_{p,i}) \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^k (X_{p,i}) & \sum_{i=1}^k (X_{1,i}X_{p,i}) & \dots & \sum_{i=1}^k (X_{p,i})^2 \end{bmatrix},$$

$$\mathbf{d}_1 = \boldsymbol{X^T_{(1)} Y_{(1)}} = \begin{bmatrix} \sum_{i=1}^k (Y_i) \\ \sum_{i=1}^k (X_{1,i}Y_i) \\ \vdots \\ \sum_{i=1}^k (X_{p,i}Y_i) \end{bmatrix}.$$

Next, we handle the second subdataset and we update the $\mathbf{d}_1$ and one of $\mathbf{C}_1$ or $\mathbf{C}_1^{-1}$ with the new one. Specifically, for the two datasets together, we obtain the below results(Kosmidis, 2015):

$$\mathbf{d}_{1\cup 2} = \mathbf{d}_1 + \mathbf{d}_2 = \boldsymbol{X}_{(1)}^{T}\boldsymbol{Y}_{(1)} + \boldsymbol{X}_{(2)}^{T}\boldsymbol{Y}_{(2)}, \tag{2.1}$$

$$\mathbf{C}_{1\cup 2} = \mathbf{C}_1 + \mathbf{C}_2 = \boldsymbol{X}_{(1)}^{T}\boldsymbol{X}_{(1)} + \boldsymbol{X}_{(2)}^{T}\boldsymbol{X}_{(2)} \tag{2.2}$$

or

$$\mathbf{C}_{1\cup 2}^{-1} = \mathbf{C}_1^{-1} - \mathbf{C}_1^{-1}\boldsymbol{X}_{(2)}^{T}(\mathbf{I} + \boldsymbol{X}_{(2)}\mathbf{C}_1^{-1}\boldsymbol{X}_{(2)}^{T})^{-1}\boldsymbol{X}_{(2)}\mathbf{C}_1^{-1}, \tag{2.3}$$

where the index $1 \cup 2$ represents the union of the first and second subdataset. The last equation (2.3) arises from applying Woodbury matrix identity (Seber ,2007). As we can see, we only need to calculate the sums, which are the sufficient statistics.

It is preferable to update the inverse of $\boldsymbol{C_{1\cup 2}}$, since in this way we do not need to inverse the matrix from scratch, which speeds up the process. Finally, the estimation of regression coefficients will be

$$\hat{\mathbf{b}}_{1\cup 2} = \mathbf{C}_{1\cup 2}^{-1}\mathbf{d}_{1\cup 2}.$$

Obviously, this approach can be generalized and the equations (2.1),(2.2) and (2.3) are key to the updating process.

2.1.5 The VIF regression

VIF regression is a fast algorithm for variable selection for big datasets and provides accurate predictions. It handles problems where the number of the features in the dataset is larger than the number of the observations. In this case, it assumes that only a subset of the predictors has nonzero coefficients. In essence, this algorithm uses streamwise regression and the a-investing rule for variable selection. In streamwise regression, variables can be generated dynamically and each of them is evaluated once, when it is generated. In contrast to stepwise regression, in which in each step all the variables are evaluated, the streamwise regression can be also used when the size of the

https://doi.org/10.26219/heal.aueb.5567

variable set is unknown or infinite (Zhou J. et al.,2006).

According to Lin, Foster and Ungar (2011), the VIF approach includes two steps. In the first one (the evaluation step) we correct the marginal correlation using a pre-sampled set of data. Specifically, we use stagewise regression, which evaluates variables only using marginal correlations. In the second step (the search step) we test each variable using the a-investing rule. The VIF regression is described thoroughly below.

Assuming that the variables are centered and normalized, we have the model

$$\boldsymbol{y} = \beta_0 + \beta_1 \boldsymbol{x_1} + \beta_2 \boldsymbol{x_2} + \ldots + \beta_k \boldsymbol{x_k} + \beta_{new} \boldsymbol{x_{new}} + \boldsymbol{\epsilon}, \qquad where \qquad \boldsymbol{\epsilon} \sim N(\boldsymbol{0}, \sigma^2 \boldsymbol{I}),$$

in which exists k variables and we are searching for the k+1 variable. In stagewise regression, we consider the model

$$\boldsymbol{r} = \gamma_{new} \boldsymbol{x_{new}} + \tilde{\boldsymbol{\epsilon}}, \qquad \tilde{\boldsymbol{\epsilon}} \sim N(\boldsymbol{0}, \tilde{\sigma}^2 \boldsymbol{I}),$$

where $\boldsymbol{r}$ is the residual of projecting $\boldsymbol{y}$ on $\{\boldsymbol{1}, \boldsymbol{x_1}, \ldots, \boldsymbol{x_k}\}$.
Moreover, we denote $\boldsymbol{X}$ as the matrix $\boldsymbol{X} = [\boldsymbol{1}, \boldsymbol{x_1}, \ldots, \boldsymbol{x_k}]$. Let $\hat{\beta}_{new}$,$\hat{\gamma}_{new}$ be the least square estimate of β_{new} and γ_{new} respectively, then

$$\hat{\gamma}_{new} = \rho^2 \hat{\beta}_{new},$$

where $\rho^2 = \boldsymbol{x_{new}^T}(\boldsymbol{I} - \boldsymbol{X}(\boldsymbol{X^T X})^{-1}\boldsymbol{X^T})\boldsymbol{x_{new}}$.
We can prove that

$$\hat{\beta}_{new} \sim N(\beta_{new}, \rho^{-2}\sigma^2),$$

thus

$$\hat{\gamma}_{new} \sim N(\rho^2 \beta_{new}, \rho^2 \sigma^2).$$

One way to estimate σ is computing the root mean square error(RMSE) under the null hypothesis that $\beta_{new} = 0$ and ρ can be estimated with the relation $\rho^2 = 1 - R^2_{new|1\ldots k}$.

As a consequence, $\dfrac{\hat{\gamma}_{new}}{|\rho|\sigma} \sim\ N(0,1)$, hence we can test whether $\beta_{new} = 0$ or not with the t-ratio. The evaluation procedure, which is described below, compute this t-ratio.

In the fast evaluation procedure, our variables are centered. At each step of the regression, we define a set of explanatory variables $\mathcal{C} = \{\boldsymbol{x_1}, \ldots, \boldsymbol{x_k}\}$, which are included in the model. From the previous step, we can calculate the residuals and the RMSE as

$$\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{X_{\mathcal{C}}}(\boldsymbol{X'_{\mathcal{C}}}\boldsymbol{X_{\mathcal{C}}})^{\boldsymbol{-1}}\boldsymbol{X'_{\mathcal{C}}}\boldsymbol{y} \qquad \textit{and} \qquad \hat{\sigma} = \frac{\|\boldsymbol{r}\|}{\sqrt{n - |\boldsymbol{\mathcal{C}}| - 1}},$$

respectively. Afterwards, we sample a subset of observations, namely $\mathcal{I} \subset \{1, \ldots, n\}$ and we denote as $\boldsymbol{x^{\mathcal{I}}}$ the corresponding subsample of explanatory variables $\boldsymbol{x}$. In this way, we compute the coefficient estimate $\hat{\gamma}_{new}$ and the $R^2_{\mathcal{I}}$, fitting $\boldsymbol{r}$ on $\dfrac{\boldsymbol{x_{new}}}{\|\boldsymbol{x_{new}}\|}$ and $\boldsymbol{x^{\mathcal{I}}_{new}}$ on $\{\boldsymbol{x^{\mathcal{I}}_1}, \ldots, \boldsymbol{x^{\mathcal{I}}_k}\}$ respectively, i.e.

$$\hat{\gamma}_{new} = \frac{\langle \boldsymbol{r}, \boldsymbol{x_{new}} \rangle}{\|\boldsymbol{x_{new}}\|} \qquad \textit{and} \qquad R^2_{\mathcal{I}} = \frac{\boldsymbol{x'_{new}}\boldsymbol{X^{\mathcal{I}}_{\mathcal{C}}}((\boldsymbol{X^{\mathcal{I}}_{\mathcal{C}}})'\boldsymbol{X^{\mathcal{I}}_{\mathcal{C}}})^{-1}(\boldsymbol{X^{\mathcal{I}}_{\mathcal{C}}})'\boldsymbol{x_{new}}}{\|\boldsymbol{x_{new}}\|^2}.$$

Finally, we return the t-ratio

$$\hat{t}_{new} = \frac{\hat{\gamma}_{new}}{\hat{\sigma}\sqrt{1 - R^2_{\mathcal{I}}}}.$$

Now, we will try to explain how the a-investing rule works. This rule is a methodology for testing multiple hypotheses and it is used for variable selection (Foster & Stine, 2007). Specifically, assuming that we play a game with a series of tests. Let's suppose that we start with a initial wealth w_0. We denote as w_j the wealth that we have after j tests. Furthermore, let p_j be the p-value of the test of H_j hypothesis. At the step i, an a-investing rule sets the level a_i and

https://doi.org/10.26219/heal.aueb.5567

if $p_i \leq a_i$, we reject H_i and we earn a pay-out Δw, i.e. $w_{i+1} = w_i + \Delta w$ otherwise,
if $p_i \geq a_i$, the wealth is decreased by $\frac{a_i}{1-a_i}$, i.e. $w_{i+1} = w_i - \frac{a_i}{1-a_i}$.
We usually set $\Delta w = w_0 = 0.05$ and $a_i = \frac{w_i}{1+i-f}$, where f is the time when the last hypothesis was rejected. This is continued until $w_i \leq 0$.

In essence, the a-investing rule controls the marginal False Discovery Rate (mFDR). mFDR is the ratio of the expected number of false rejections to the expected number of rejections and FDR is the expected value of this ratio. Namely, assuming that V is the number of falsely rejected hypotheses and R is the total number of rejected hypotheses, we get

$$mFDR = \frac{E(V)}{E(R)+1} \;\; or \;\; \frac{E(V)}{E(R)} \qquad and \qquad FDR = E(\frac{V}{R}|R > 0)P(R > 0).$$

The algorithm of VIF regression is described below (Lin, Foster & Ungar, 2011). It is worth noticing that we incorporate the evaluation procedure, which is detailed previously, into a streamwise regression using a-investing rule.

Algorithm 1: VIF Regression

Input: $\boldsymbol{y}, \boldsymbol{x_1}, \ldots$ (centered)
Set: initial wealth:$w_0 = 0.05$, pay-out:$\Delta w = 0.05$, subsample size:m
Initialize: $\mathcal{C} = \{0\}, \boldsymbol{r} = \boldsymbol{y} - \bar{\boldsymbol{y}}, \hat{\sigma} = sd(\boldsymbol{y}), i = 1, w_1 = w_0, f = 0$
Sample: $\mathcal{I} \subset \{1, \ldots, n\}$
repeat
  Set threshold $a_i = \dfrac{w_i}{1 + i - f}$
  Attain $\hat{t}_i$ from the fast evaluation procedure
  if $2\Phi(\mid t_i \mid) > 1 - \alpha_i$ **then**
    $\mathcal{C} = \mathcal{C} \cup \{i\}$
    update $\boldsymbol{r} = \boldsymbol{y} - \hat{\boldsymbol{y}}_{\mathcal{C}}$, $\hat{\sigma} = RMSE_{\mathcal{C}}$
    $w_{i+1} = w_i + \Delta w$
    $f = i$
  else
    $w_{i+1} = w_i - \dfrac{a_i}{1 - a_i}$
  end
  $i = i + 1$
until *maximum CPU time or Memory is reached*

2.1.6 ℓ_1Regularized High-Dimensional Sparse Regression

In many big data problems, such as in biology, medicine etc, the number of observations can be smaller than the number of variables. In this case ($p >> n$), ordinary least square methods are not consistent. This leads us to high dimensional sparse regression. A sparse statistical model is one having a small number of nonzero coefficients. As a consequence, most of our variables are not significant.

One of the most famous methods of high dimensional sparse regression

https://doi.org/10.26219/heal.aueb.5567

is the ℓ_1-norm regularized approach, which is called LASSO(Least Absolute Shrinkage and Selection Operation). We consider a high dimensional linear model

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

where $\boldsymbol{Y} \in \mathbb{R}^n$ is the response variable, $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ is the design matrix and $\boldsymbol{\epsilon} \in \mathbb{R}^n$ is the error vector. Our aim is to estimate the regression coefficients in case of $p \geq n$. In general, the estimation of coefficients in LASSO can be written as

$$\hat{\boldsymbol{\beta}}_{LASSO} = \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}} \frac{1}{2n}\|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1,$$

where $\lambda > 0$ is the regularization parameter or penalty level. The parameter λ controls the strength of the penalty. If $\lambda = 0$, we get the linear regression estimate. If $\lambda = \infty$, then $\hat{\boldsymbol{\beta}}_{LASSO} = 0$. Due to the shape of the ℓ_1-norm ball, some coefficients are shrunken to zero exactly. Next, according to Li(2013), we will describe the family of Lasso regression (FLARE) for the high-dimensional sparse linear model i.e. Dantzig selector, LAD Lasso, SQRT Lasso, ℓ_q Lasso.

In real datasets, there is at least a small amount of noise. Thus, we use the Dantzig selector in order to estimate β with noisy data (Candes and Tao, 2007). We obtain

$$\hat{\boldsymbol{\beta}}_{ds} = \underset{\boldsymbol{\beta}\in\mathbb{R}^p}{\operatorname{argmin}}\|\boldsymbol{\beta}\|_1, \quad s.t. \quad \|\boldsymbol{X}^{\boldsymbol{T}}(\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta})\|_\infty \leq \lambda.$$

Also, Dantzig selector is useful because it can tolerate missing values in the design matrix and response vector. Nevertheless, Lasso and Dantzig selector assume Gaussian distribution and known variance, that are difficult to find in practice. For this reason, we use other methods of Lasso, where these assumptions are not necessary.

We define the square root Lasso method (SQRT Lasso) (Belloni A., Chernozhukov V. and Wang L.,2011). So, we can estimate the coefficients as

https://doi.org/10.26219/heal.aueb.5567

follows

$$\hat{\boldsymbol{\beta}}_{SQRT} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \frac{1}{\sqrt{n}} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_2 + \lambda \|\boldsymbol{\beta}\|_1.$$

In this method, the optimal regularization parameter selection does not depend on any unknown parameter.

In the case of heavy tail random noise and outliers, we prefer the Least Absolute Deviation (LAD) Lasso (Wang L., 2013). This estimator can be written as

$$\hat{\boldsymbol{\beta}}_{LAD} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \frac{1}{n} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_1 + \lambda \|\boldsymbol{\beta}\|_1.$$

We can generalize the two previous methods as the ℓ_q Lasso. So, we obtain

$$\hat{\boldsymbol{\beta}}_{\ell_q} = \underset{\boldsymbol{\beta} \in \mathbb{R}^p}{\operatorname{argmin}} \frac{1}{\sqrt[q]{n}} \|\boldsymbol{Y} - \boldsymbol{X}\boldsymbol{\beta}\|_q + \lambda \|\boldsymbol{\beta}\|_1,$$

where $q \geq 1$ and $q \in \mathbb{R}$. In the case of $q = 1$ we have the LAD Lasso and for $q = 2$ we get the SQRT Lasso.

2.1.7 Informative Subdata Selection for Big Data

Because of the volume of big datasets, it is essential to extract only the useful information from the full data. In order to achieve this, subsampling based methods will be used. The selection of the subsample is accomplished assigning sampling probabilities π_i for all data points ($i = 1, \ldots, n$) and taking a subsample according to these probabilities.

The easiest way is the uniform random sampling, i.e. $\pi_i = \frac{1}{n}$. But it deems all data points equally significant. For this reason, it is preferable to use nonuniform sampling probabilities so that the influential points have higher probabilities to be selected.

According to HaiYing Wang(2016), let

$$y_i = \beta_0 + \sum_{j=1}^{p} z_{ij}\beta_j + \epsilon_i \quad i = 1, \ldots, n,$$

https://doi.org/10.26219/heal.aueb.5567

be a linear regression model where y_i are the responses, $\boldsymbol{z_i} = (z_{i1}, \ldots, z_{ip})^T$ are the covariates and ϵ_i are the uncorrelated errors. We also denote $\boldsymbol{x_i} = (1, \boldsymbol{z_i}^T)^T$, $\boldsymbol{X} = (\boldsymbol{x_1}, \ldots, \boldsymbol{x_n})$ and $\boldsymbol{y} = (y_1, \ldots, y_n)$. As we mentioned previously, the subsampling based method is the following:

1. Let $\boldsymbol{\pi} = \{\pi_1, \ldots, \pi_n\}$ be the nonuniform sampling probabilities and $\sum_{i=1}^{n} \pi_i = 1$

2. Let $(\boldsymbol{x_1^\star}, y_1^\star), \ldots, (\boldsymbol{x_k^\star}, y_k^\star)$ be the data points of a random subsample of size k according to the probabilities $\boldsymbol{\pi}$ ($\boldsymbol{x_i^\star}$ is a $(p+1) \times 1$ vector and $y_i^\star$ is a scalar)

3. Then, the subsampling-based estimator is

$$\tilde{\boldsymbol{\beta}} = (\sum_{i=1}^{k} w_i^\star \boldsymbol{x_i^\star} \boldsymbol{x_i^{\star T}})^{-1} \sum_{i=1}^{k} w_i^\star \boldsymbol{x_i^\star} y_i^\star,$$

   where usually $w_i^\star = \frac{1}{\pi_i}$ and π_i's are often the statistical leverage scores.

Our aim is an optimal estimator based on a subdataset. It is known that the least squares estimator minimizes the variance of estimators, where $Var(\hat{\boldsymbol{\beta}}) = (\boldsymbol{X^T X})^{-1}\sigma^2$. But, in multiple regression, the variance of an estimator is a covariance matrix, so the problem of minimizing the variance is complicated. Consequently, we will try to maximize the inverse of the covariance matrix, which is the information matrix.

Let's assume that δ_i is an indicator, which shows whether the observation i is included in the subdataset or not, such that $\sum_{i=1}^{n} \delta_i = k$. Under this assumption, the information matrix of a subdataset of size k can be written as

$$\boldsymbol{M}(\boldsymbol{\delta}) = \frac{1}{\sigma^2} \sum_{i=1}^{n} \delta_i \boldsymbol{x_i} \boldsymbol{x_i}^T,$$

where $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_n)$. In order to maximize the information matrix, we will use the D-optimality criterion. This is a popular criterion in the design of

https://doi.org/10.26219/heal.aueb.5567

experiments theory, which minimizes $|(\boldsymbol{X^T X})^{-1}|$, or equivalently maximizes the determinant of the matrix $\boldsymbol{X^T X}$. Thus, the D-optimality criterion suggests the selection of subdataset so that

$$\boldsymbol{\delta^{opt}} = \underset{\boldsymbol{\delta}}{\operatorname{argmax}} |\boldsymbol{M}(\boldsymbol{\delta})| = \underset{\boldsymbol{\delta}}{\operatorname{argmax}} \sum_{i=1}^{n} \delta_i \boldsymbol{x_i} \boldsymbol{x_i}^T.$$

This is the Information-Based Optimal Subdata Selection(IBOSS) based on D-optimality. It is worth mentioning that it is difficult to find a closed form solution for this problem. Hence, we try to find an upper bound for the determinant of $\boldsymbol{M}(\boldsymbol{\delta})$ and then we aim to approximate this bound.

2.2 Data splitting

In some regression methods for big data, we mention that we split the data and then analyze each subset independently. The problem of data splitting can be operated as a sampling problem. Hence, we will describe some of statistical sampling methods which can be used for this reason. Each of them is different, has its own advantages and disadvantages. The choice of the splitting methods depends mainly on the character of the dataset. Thus, according to Reitermanova Z. (2010), we have the following approaches.

2.2.1 Simple Random Sampling

It is the most common and simplest technique in order to split the data. We choose observations of the dataset randomly with a uniform distribution. Each observation has an equal probability and chance of being included in the subset. Thus, randomization achieves a subset with low bias. Nevertheless, the simple random sampling can create subsets, which are different from the full dataset. As a consequence, the estimation of the model error will suffer from high variance.

https://doi.org/10.26219/heal.aueb.5567

2.2.2 Trial-and-error method

In this method, in order to overcome the problem of the previous technique, we apply simple random sampling many times and then we take the mean of the results. However, this procedure has a high computational cost.

2.2.3 Systematic Sampling

It is also a simple method of data splitting but it needs appropriate dataset. Firstly, we have to put the data in order. Then, we choose randomly an observation and add each k^{th} observation in the subset, where $k = \frac{n_{dataset}}{n_{subset}}$. The main drawbacks of this method are the difficulty to find a appropriate order for each dataset and its sensitivity to periodicities in the dataset. It is obvious that if the data are not in order, the systematic sampling is equal with the simple random sampling.

2.2.4 Convenience Sampling

This technique is like the systematic sampling, because it is used for appropriate datasets(e.g. time series) and has the same drawbacks. Applying convenience sampling, we split the dataset according to the discrete blocks(e.g. time intervals).

2.2.5 DUPLEX

The first algorithm of DUPLEX split the dataset into two subsets with the way below (Snee R. D., 1977). We find the two farthest observations of the dataset according to the Euclidean distance and we put them in one subset. The next pair of observations, which is the farthest apart, is put in the second subset. In the next step, the observation which is farthest from the two points in the first subdata is added in the first subdata. Then, the observation that is farthest from the pair of observations in the second subdata is included

https://doi.org/10.26219/heal.aueb.5567

in this subdata. We continue this algorithm with the same procedure. Of course, this method can be generalized for more subsets. Nevertheless, it has a main drawback, it has a huge computational complexity.

2.2.6 Stratified sampling

Stratified sampling technique is used when the dataset is heterogeneous. Firstly, the dataset is divided into smaller groups, known as strata. This procedure is achieved with several clustering algorithms, such as c-means. Then, observations are selected separately in each stratum in order to create a subset. This technique is appropriate for datasets, which can be split into homogeneous groups .

As we mentioned above, in order to create a subset we have to take observations from each stratum. If we divide randomly each stratum into smaller groups, we have the known **Stratified random sampling**. There are several ways in order to select the number of observations per stratum. We will describe the most common one according to allocation rules:

Equal allocation : It defines equal number observations from each strata. In other words, the number of observations which we select from the stratum i will be $n_i = \frac{n_{subset}}{k}$ where k is the number of the strata.

Proportional allocation : In this rule, the number of observations,which is selected per stratum depends on the size of each stratum. Thus, $n_i = n_{subset}\frac{N_i}{\sum_{j=1}^{k} N_j}$, where N_j is the size of the stratum j.

Optimal allocation : In this allocation, the number of observations that is selected depends not only on the size of each stratum, but also on the standard deviation. Specifically, $n_i = n_{subset}\frac{N_i\sigma_i}{\sum_{j=1}^{k} N_j\sigma_j}$, where N_j is the size of the stratum j and σ_j is the standard deviation of the stratum j.

https://doi.org/10.26219/heal.aueb.5567

2.3 Conclusion of chapter 2

In this chapter, we studied several methods and techniques for big data regression. For problems where the number of observations is huge, we can split the data into parts, analyze them separately and combine the results either with mean or meta analysis. In addition, we explored the selection of an appropriate subdataset using IBOSS, an approach that proved efficient. Of course, another method for saving time and memory is the use of sufficient statistics only instead of the actual data.

We also referred to methods, that handle problems where the number of observations is smaller than the number of variables. Such methods are the ℓ_1 regularization and the VIF regression. At the end, we reported sampling techniques for data splitting. These techniques are important for splitting the full dataset in order to analyze it.

https://doi.org/10.26219/heal.aueb.5567

Chapter 3

Regression for Big Data

3.1 Updating Regression

3.1.1 Recursive formula for $\boldsymbol{X^T X}$

In a multiple linear regression, $\boldsymbol{Y}$ represents the response variable and $\boldsymbol{X}$ is the regression matrix. So, we obtain

$$Y_i = \beta_0 + \beta_1 X_{1,i} + \ldots + \beta_p X_{p,i} + \epsilon_i, \qquad i = 1, \ldots, n,$$

where $E(\epsilon_i) = 0$ and $Var(\epsilon_i) = \sigma^2$ and we write $X_{j,i}$ for the j^{th} variable measured for the i^{th} observation. From a matricial point of view, the linear model can be written as follows:

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}, \qquad \boldsymbol{Y} \in \mathbb{R}^n, \boldsymbol{X} \in \mathcal{M}_{n,p+1}(\mathbb{R}), \boldsymbol{\beta} \in \mathbb{R}^{p+1}.$$

In order to estimate $\boldsymbol{\beta}$, we take a least squares approach:

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T Y}.$$

If we have a big dataset, the inversion of $\boldsymbol{X^T X}$ will be time consuming and if we add a new variable, we have to calculate $\boldsymbol{X^T X}$ from the beginning.

https://doi.org/10.26219/heal.aueb.5567

In this chapter, we will present an alternative way to estimate $\boldsymbol{\beta}$ using a recursive formula.

If $\boldsymbol{X^T X}$ is formed, it can be used easily to add variables in the model. We denote the coefficient of the $\boldsymbol{X_j}$ variable of the model that includes r explanatory variables by $\beta_j^{(r)}$ and the parameters vector that includes r explanatory variables by $\boldsymbol{b_r}$. Let's asume that we have only **one** explanatory variable $\boldsymbol{X_1}$ i.e.

$$Y_i = \beta_0^{(1)} + \beta_1^{(1)} X_{1,i} + \epsilon_i, \qquad i = 1, \dots, n.$$

Then,

$$(\boldsymbol{X^T X})_1 = \begin{bmatrix} 1 & 1 & \dots & 1 \\ X_{1,1} & X_{1,2} & \dots & X_{1,n} \end{bmatrix} \begin{bmatrix} 1 & X_{1,1} \\ 1 & X_{1,2} \\ \vdots & \vdots \\ 1 & X_{1,n} \end{bmatrix} = \begin{bmatrix} n & \sum_{i=1}^n (X_{1,i}) \\ \sum_{i=1}^n (X_{1,i}) & \sum_{i=1}^n (X_{1,i})^2 \end{bmatrix},$$

and

$$(\boldsymbol{X^T Y})_1 = \begin{bmatrix} 1 & 1 & \dots & 1 \\ X_{1,1} & X_{1,2} & \dots & X_{1,n} \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n (Y_i) \\ \sum_{i=1}^n (X_{1,i} Y_i) \end{bmatrix}.$$

For **two** explanatory variables $\boldsymbol{X_1}$, $\boldsymbol{X_2}$ i.e.

$$Y_i = \beta_0^{(2)} + \beta_1^{(2)} X_{1,i} + \beta_2^{(2)} X_{2,i} + \epsilon_i, \qquad i = 1, \dots, n,$$

 https://doi.org/10.26219/heal.aueb.5567

we obtain

$$(\boldsymbol{X^T X})_2 = \begin{bmatrix} 1 & 1 & \dots & 1 \\ X_{1,1} & X_{1,2} & \dots & X_{1,n} \\ X_{2,1} & X_{2,2} & \dots & X_{2,n} \end{bmatrix} \begin{bmatrix} 1 & X_{1,1} & X_{2,1} \\ 1 & X_{1,2} & X_{2,2} \\ \vdots & \vdots & \vdots \\ 1 & X_{1,n} & X_{2,n} \end{bmatrix}$$

$$= \left[\begin{array}{cc|c} n & \sum_{i=1}^n (X_{1,i}) & \sum_{i=1}^n (X_{2,i}) \\ \sum_{i=1}^n (X_{1,i}) & \sum_{i=1}^n (X_{1,i})^2 & \sum_{i=1}^n (X_{1,i} X_{2,i}) \\ \hline \sum_{i=1}^n (X_{2,i}) & \sum_{i=1}^n (X_{1,i} X_{2,i}) & \sum_{i=1}^n (X_{2,i})^2 \end{array}\right]$$

and $(\boldsymbol{X^T Y})_2 = \begin{bmatrix} 1 & 1 & \dots & 1 \\ X_{1,1} & X_{1,2} & \dots & X_{1,n} \\ X_{2,1} & X_{2,2} & \dots & X_{2,n} \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \left[\begin{array}{c} \sum_{i=1}^n (Y_i) \\ \sum_{i=1}^n (X_{1,i} Y_i) \\ \hline \sum_{i=1}^n (X_{2,i} Y_i) \end{array}\right]$.

Generally, for **r** explanatory variables $\boldsymbol{X_1}, \dots, \boldsymbol{X_r}$ we get

$$Y_i = \beta_0^{(r)} + \beta_1^{(r)} X_{1,i} + \ldots + \beta_r^{(r)} X_{r,i} + \epsilon_i, \qquad i = 1, \ldots, n,$$

with

$$(\boldsymbol{X^T X})_r = \begin{bmatrix} 1 & 1 & \dots & 1 \\ X_{1,1} & X_{1,2} & \dots & X_{1,n} \\ \vdots & \vdots & \dots & \vdots \\ X_{r,1} & X_{r,2} & \dots & X_{r,n} \end{bmatrix} \begin{bmatrix} 1 & X_{1,1} & \dots & X_{r,1} \\ 1 & X_{1,2} & \dots & X_{r,2} \\ \vdots & \vdots & \dots & \vdots \\ 1 & X_{1,n} & \dots & X_{r,n} \end{bmatrix}$$

$$= \begin{bmatrix} n & \sum_{i=1}^n (X_{1,i}) & \dots & \sum_{i=1}^n (X_{r,i}) \\ \sum_{i=1}^n (X_{1,i}) & \sum_{i=1}^n (X_{1,i})^2 & \dots & \sum_{i=1}^n (X_{1,i} X_{r,i}) \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{i=1}^n (X_{r,i}) & \sum_{i=1}^n (X_{1,i} X_{r,i}) & \dots & \sum_{i=1}^n (X_{r,i})^2 \end{bmatrix}$$

$$\text{and } (\boldsymbol{X^T Y})_r = \begin{bmatrix} 1 & 1 & \dots & 1 \\ X_{1,1} & X_{1,2} & \dots & X_{1,n} \\ \vdots & \vdots & \dots & \vdots \\ X_{r,1} & X_{r,2} & \dots & X_{r,n} \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n (Y_i) \\ \sum_{i=1}^n (X_{1,i} Y_i) \\ \vdots \\ \sum_{i=1}^n (X_{r,i} Y_i) \end{bmatrix}.$$

Also,the $(\mathbf{r+1})^{\mathbf{th}}$ step is

$$Y_i = \beta_0^{(r+1)} + \beta_1^{(r+1)} X_{1,i} + \ldots + \beta_{r+1}^{(r+1)} X_{r+1,i} + \epsilon_i, \qquad i = 1, \ldots, n,$$

with

$$(\boldsymbol{X^T X})_{r+1} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ X_{1,1} & X_{1,2} & \dots & X_{1,n} \\ \vdots & \vdots & \dots & \vdots \\ X_{r+1,1} & X_{r+1,2} & \dots & X_{r+1,n} \end{bmatrix} \begin{bmatrix} 1 & X_{1,1} & \dots & X_{r+1,1} \\ 1 & X_{1,2} & \dots & X_{r+1,2} \\ \vdots & \vdots & \dots & \vdots \\ 1 & X_{1,n} & \dots & X_{r+1,n} \end{bmatrix}$$

$$= \left[\begin{array}{cccc|c} n & \sum_{i=1}^n (X_{1,i}) & \dots & \sum_{i=1}^n (X_{r,i}) & \sum_{i=1}^n (X_{r+1,i}) \\ \sum_{i=1}^n (X_{1,i}) & \sum_{i=1}^n (X_{1,i})^2 & \dots & \sum_{i=1}^n (X_{1,i} X_{r,i}) & \sum_{i=1}^n (X_{1,i} X_{r+1,i}) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sum_{i=1}^n (X_{r,i}) & \sum_{i=1}^n (X_{1,i} X_{r,i}) & \dots & \sum_{i=1}^n (X_{r,i})^2 & \sum_{i=1}^n (X_{r,i} X_{r+1,i}) \\ \hline \sum_{i=1}^n (X_{r+1,i}) & \sum_{i=1}^n (X_{1,i} X_{r+1,i}) & \dots & \sum_{i=1}^n (X_{r,i} X_{r+1,i}) & \sum_{i=1}^n (X_{r+1,i})^2 \end{array}\right]$$

$$\text{and } (\boldsymbol{X^T Y})_{r+1} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ X_{1,1} & X_{1,2} & \dots & X_{1,n} \\ \vdots & \vdots & \dots & \vdots \\ X_{r+1,1} & X_{r+1,2} & \dots & X_{r+1,n} \end{bmatrix} \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} = \left[\begin{array}{c} \sum_{i=1}^n (Y_i) \\ \sum_{i=1}^n (X_{1,i} Y_i) \\ \vdots \\ \sum_{i=1}^n (X_{r,i} Y_i) \\ \hline \sum_{i=1}^n (X_{r+1,i} Y_i) \end{array}\right].$$

So, we observe the following recursive formula

$$(\boldsymbol{X^T X})_{r+1} = \left[\begin{array}{c|c} (\boldsymbol{X^T X})_r & \boldsymbol{W} \\ \hline \boldsymbol{W^T} & z \end{array}\right],$$

where $\boldsymbol{W} = \begin{bmatrix} \sum_{i=1}^{n}(X_{r+1,i}) \\ \sum_{i=1}^{n}(X_{1,i}X_{r+1,i}) \\ \vdots \\ \sum_{i=1}^{n}(X_{r,i}X_{r+1,i}) \end{bmatrix}$ and $z = \sum_{i=1}^{n}(X_{r+1,i})^2$,

and

$$(\boldsymbol{X^T Y})_{r+1} = \left[\begin{array}{c} (\boldsymbol{X^T Y})_r \\ \hline R \end{array}\right],$$

where $R = \sum_{i=1}^{n}(X_{r+1,i}Y_i)$.

From the above analysis, we conclude that each $\boldsymbol{X^T X}$ matrix is a block matrix, which can be written as $(\boldsymbol{X^T X})_r = \left[\begin{array}{c|c} \boldsymbol{A} & \boldsymbol{B} \\ \hline \boldsymbol{B^T} & d \end{array}\right]$.
It's necessary to inverse this matrix in order to estimate b.

3.1.2 Estimation regression coefficients

Theorem 1. *Let* $\boldsymbol{R} = \left[\begin{array}{c|c} \boldsymbol{A} & \boldsymbol{B} \\ \hline \boldsymbol{C} & \boldsymbol{D} \end{array}\right]$ *. Assume* $\boldsymbol{A}$ *is nonsingular; then the matrix* $\boldsymbol{R}$ *is invertible if and only if the Schur complement* $\boldsymbol{D} - \boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B}$ *of* $\boldsymbol{A}$ *is invertible, and* $\boldsymbol{R}^{-1} = \begin{bmatrix} \boldsymbol{A}^{-1} + \boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{D} - \boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\boldsymbol{C}\boldsymbol{A}^{-1} & -\boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{D} - \boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B})^{-1} \\ -(\boldsymbol{D} - \boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\boldsymbol{C}\boldsymbol{A}^{-1} & (\boldsymbol{D} - \boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B})^{-1} \end{bmatrix}$.

Applying Theorem 1(Lu & Shiou, 2002) to our previous result, we get

$$(\boldsymbol{X^T X})_r^{-1} = \left[\begin{array}{c|c} \boldsymbol{A} & \boldsymbol{B} \\ \hline \boldsymbol{B^T} & d \end{array}\right]^{-1} = \begin{bmatrix} \boldsymbol{A}^{-1} + \frac{1}{k}\boldsymbol{A}^{-1}\boldsymbol{B}\boldsymbol{B^T}\boldsymbol{A}^{-1} & -\frac{1}{k}\boldsymbol{A}^{-1}\boldsymbol{B} \\ -\frac{1}{k}\boldsymbol{B^T}\boldsymbol{A}^{-1} & \frac{1}{k} \end{bmatrix},$$

https://doi.org/10.26219/heal.aueb.5567

where $k = d - \boldsymbol{B^T A^{-1} B}$. So, we only need the inverse of $\boldsymbol{A}$ i.e. the inverse of the previous $\boldsymbol{X^T X}$.

We define $(\boldsymbol{X^T X})_r^{-1} = \left[\begin{array}{c|c} \boldsymbol{A} & \boldsymbol{B} \\ \hline \boldsymbol{B^T} & d \end{array}\right]^{-1} = \left[\begin{array}{c|c} \boldsymbol{K} & \boldsymbol{L} \\ \hline \boldsymbol{M} & N \end{array}\right]$ and $(\boldsymbol{X^T X})_{r+1}^{-1} = \left[\begin{array}{c|c} \boldsymbol{S} & \boldsymbol{T} \\ \hline \boldsymbol{U} & V \end{array}\right]$.

So, in order to estimate $\boldsymbol{\beta}$, we have

$$\hat{\boldsymbol{b}}_{\boldsymbol{r}} = (\boldsymbol{X^T X})_r^{-1} (\boldsymbol{X^T Y})_r = \left[\begin{array}{c|c} \boldsymbol{K} & \boldsymbol{L} \\ \hline \boldsymbol{M} & N \end{array}\right] (\boldsymbol{X^T Y})_r. \tag{3.1}$$

Remember that we have defined

$$(\boldsymbol{X^T X})_{r+1} = \left[\begin{array}{c|c} (\boldsymbol{X^T X})_r & \boldsymbol{W} \\ \hline \boldsymbol{W^T} & z \end{array}\right] \quad \textit{and} \quad (\boldsymbol{X^T Y})_{r+1} = \left[\begin{array}{c} (\boldsymbol{X^T Y})_r \\ \hline R \end{array}\right].$$

Then

$$\begin{aligned}
\hat{\mathbf{b}}_{\mathbf{r+1}} &= \left[\begin{array}{c|c} \boldsymbol{S} & \boldsymbol{T} \\ \hline \boldsymbol{U} & V \end{array}\right] \left[\begin{array}{c} (\boldsymbol{X^T Y})_r \\ \hline R \end{array}\right] = \begin{bmatrix} \boldsymbol{S}(\boldsymbol{X^T Y})_r + \boldsymbol{T}R \\ \boldsymbol{U}(\boldsymbol{X^T Y})_r + VR \end{bmatrix} \overset{(Theorem 1)}{=} \\
&= \begin{bmatrix} (\boldsymbol{X^T X})_r^{-1}(\boldsymbol{X^T Y})_r + \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W W^T}(\boldsymbol{X^T X})_r^{-1}(\boldsymbol{X^T Y})_r - \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}R \\ -\frac{1}{c}\boldsymbol{W^T}(\boldsymbol{X^T X})_r^{-1}(\boldsymbol{X^T Y})_r + \frac{1}{c}R \end{bmatrix} = \\
&\overset{(3.1)}{=} \begin{bmatrix} \hat{\mathbf{b}}_{\mathbf{r}} + \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W W^T}\hat{\mathbf{b}}_{\mathbf{r}} - \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}R \\ -\frac{1}{c}\boldsymbol{W^T}\hat{\mathbf{b}}_{\mathbf{r}} + \frac{1}{c}R \end{bmatrix} \\
&= \begin{bmatrix} \hat{\mathbf{b}}_{\mathbf{r}} + (\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}(\frac{1}{c}\boldsymbol{W^T}\hat{\mathbf{b}}_{\mathbf{r}} - \frac{1}{c}R) \\ -(\frac{1}{c}\boldsymbol{W^T}\hat{\mathbf{b}}_{\mathbf{r}} - \frac{1}{c}R) \end{bmatrix}.
\end{aligned}$$

In conclusion,

$$\Rightarrow \hat{\mathbf{b}}_{\mathbf{r+1}} = \begin{bmatrix} \hat{\mathbf{b}}_{\mathbf{r}} + (\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}(\frac{1}{c}\boldsymbol{W^T}\hat{\mathbf{b}}_{\mathbf{r}} - \frac{1}{c}R) \\ -(\frac{1}{c}\boldsymbol{W^T}\hat{\mathbf{b}}_{\mathbf{r}} - \frac{1}{c}R) \end{bmatrix}, \tag{3.2}$$

 https://doi.org/10.26219/heal.aueb.5567

where $c = z - \boldsymbol{W^T}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}$
We assume that the response and the covariates are standardized i.e.

$$\sum_{i=1}^{n}(Y_i) = 0, \qquad \sum_{i=1}^{n}(X_{j,i}) = 0, \qquad S^2_{\boldsymbol{X_j}} = \frac{\sum_{i=1}^{n}(X_{j,i})^2}{n-1} = 1,$$

$$S_{\boldsymbol{X_k},\boldsymbol{X_l}} = \frac{\sum_{i=1}^{n}(X_{k,i}, X_{l,i})}{n-1},$$

$$with \qquad j = 1, \ldots, p, \qquad k, l = 1, \ldots, p \qquad and \qquad k \neq l.$$

So, if we add a new explanatory variable, we don't need all the data set in order to estimate $\boldsymbol{b_{r+1}}$. We have already calculated $\hat{\boldsymbol{b_r}}$ and $(\boldsymbol{X^T X})_r^{-1}$, and then we need only the covariances between the new variable and the existing variables , i.e. $S_{\boldsymbol{X_{r+1}},\boldsymbol{X_j}}$, $j = 1, \ldots, r$ and $S_{\boldsymbol{X_{r+1}},\boldsymbol{Y}}$.
Especially, for the $\hat{\boldsymbol{b}}_{\boldsymbol{r+1}}$, we need:

- $\hat{\boldsymbol{b}}_r$ and $(\boldsymbol{X^T X})_r^{-1}$. These are already calculated.
- $\boldsymbol{W} = \begin{bmatrix} \sum_{i=1}^{n}(X_{r+1,i}) \\ \sum_{i=1}^{n}(X_{1,i}X_{r+1,i}) \\ \vdots \\ \sum_{i=1}^{n}(X_{r,i}X_{r+1,i}) \end{bmatrix} = \begin{bmatrix} 0 \\ (n-1)S_{\boldsymbol{X_1},\boldsymbol{X_{r+1}}} \\ \vdots \\ (n-1)S_{\boldsymbol{X_r},\boldsymbol{X_{r+1}}} \end{bmatrix}$
- $R = \sum_{i=1}^{n}(X_{r+1,i}Y_i) = (n-1)S_{\boldsymbol{X_{r+1}},\boldsymbol{Y}}$
- $c = z - \boldsymbol{W^T}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}$ (c is an number),
  where $z = \sum_{i=1}^{n}(X_{r+1,i})^2 = (n-1)S^2_{\boldsymbol{X_{r+1}}} = n-1$

3.2 Special cases

In the previous section, we proved a recursive formula to estimate the regression coefficients. Now, assuming that the covariates are standardized, we shall focus on two special cases:

1) If the explanatory variable $\boldsymbol{X_{r+1}}$ is **uncorrelated** with all other variables, i.e. $S_{\boldsymbol{X_i},\boldsymbol{X_{r+1}}} = 0 \qquad i = 1, \dots r$ then

$$\boldsymbol{W} = \begin{bmatrix} 0 \\ (n-1)S_{\boldsymbol{X_1},\boldsymbol{X_{r+1}}} \\ \vdots \\ (n-1)S_{\boldsymbol{X_r},\boldsymbol{X_{r+1}}} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \textit{and} \quad c = z - \boldsymbol{W^T}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W} = z.$$

Hence, the formula (1.2) will be

$$\hat{\boldsymbol{b_{r+1}}} = \begin{bmatrix} \hat{\boldsymbol{b_r}} \\ S_{\boldsymbol{X_{r+1}},\boldsymbol{Y}} \end{bmatrix}.$$

In this case, we notice that the new variable $\boldsymbol{X_{r+1}}$ doesn't affect the previous coefficients $\beta_0, \dots, \beta_r$, as expected.

2) If the explanatory variable $\boldsymbol{X_{r+1}}$ is **highly correlated** with one of the other variables, e.g. with $\boldsymbol{X_1}$, and uncorrelated with the remaining, i.e. $S_{\boldsymbol{X_1},\boldsymbol{X_{r+1}}} \simeq 1$ and $S_{\boldsymbol{X_i},\boldsymbol{X_{r+1}}} = 0, \forall i \in \{2, \dots, r\}$,
we will obtain

$$\boldsymbol{W} = \begin{bmatrix} 0 \\ (n-1)S_{\boldsymbol{X_1},\boldsymbol{X_{r+1}}} \\ (n-1)S_{\boldsymbol{X_2},\boldsymbol{X_{r+1}}} \\ \vdots \\ (n-1)S_{\boldsymbol{X_r},\boldsymbol{X_{r+1}}} \end{bmatrix} \simeq \begin{bmatrix} 0 \\ n-1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

and

$$\boldsymbol{W^T}\hat{\boldsymbol{b_r}} \simeq \begin{bmatrix} 0 & n-1 & 0 & \dots & 0 \end{bmatrix} \begin{bmatrix} \beta_0^{(r)} \\ \beta_1^{(r)} \\ \vdots \\ \beta_r^{(r)} \end{bmatrix} \simeq (n-1)\beta_1^{(r)}.$$

https://doi.org/10.26219/heal.aueb.5567

Therefore, the coefficient of $\boldsymbol{X_{r+1}}$ will be

$$\beta_{r+1}^{(r+1)} \simeq -(\frac{1}{c}\boldsymbol{W^T}\hat{\boldsymbol{b_r}} - \frac{1}{c}R) \simeq \frac{R-(n-1)\beta_1^{(r)}}{c}$$

Hence,

$$\hat{\boldsymbol{b}}_{\boldsymbol{r+1}} \simeq \begin{bmatrix} \hat{\boldsymbol{b}}_{\boldsymbol{r}} - (\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}\dfrac{R-(n-1)\beta_1^{(r)}}{c} \\ \dfrac{R-(n-1)\beta_1^{(r)}}{c} \end{bmatrix}. \tag{3.3}$$

Let's see a realistic example. We assume that we have the model

$$Y_i = \beta_0^{(2)} + \beta_1^{(2)}X_{1,i} + \beta_2^{(2)}X_{2,i} + \epsilon_i, \qquad i = 1, \ldots, n,$$

and we want to add a new variable, $\boldsymbol{X_3}$, which will be highly correlated with $\boldsymbol{X_1}$ and uncorrelated with $\boldsymbol{X_2}$, namely $S_{\boldsymbol{X_1},\boldsymbol{X_3}} = \rho_1$ and $S_{\boldsymbol{X_2},\boldsymbol{X_3}} = \rho_2$, where ρ_1 tends to one and ρ_2 tends to zero. The new model will be

$$Y_i = \beta_0^{(3)} + \beta_1^{(3)}X_{1,i} + \beta_2^{(3)}X_{2,i} + \beta_3^{(3)} + \epsilon_i, \qquad i = 1, \ldots, n.$$

So, from the relation (3.3) we will get

$$\begin{bmatrix} \beta_0^{(3)} \\ \beta_1^{(3)} \\ \beta_2^{(3)} \end{bmatrix} = \begin{bmatrix} \beta_0^{(2)}(\equiv 0 \;\; \textit{due to standardization}) \\ \beta_1^{(2)} - \dfrac{[\rho_1 - \rho_2 S_{\boldsymbol{X_1},\boldsymbol{X_2}}][S_{\boldsymbol{X_3},\boldsymbol{Y}} - \beta_1^{(2)}]}{1 - S^2_{\boldsymbol{X_1},\boldsymbol{X_2}} + \rho_2^2 - \rho_1^2} \\ \beta_2^{(2)} - \dfrac{[\rho_2 - \rho_1 S_{\boldsymbol{X_1},\boldsymbol{X_2}}][S_{\boldsymbol{X_3},\boldsymbol{Y}} - \beta_1^{(2)}]}{1 - S^2_{\boldsymbol{X_1},\boldsymbol{X_2}} + \rho_2^2 - \rho_1^2} \end{bmatrix}$$

and

$$\beta_3^{(3)} = \frac{R-(n-1)\beta_1^{(2)}}{c} = \frac{[S_{X_3,Y} - \beta_1^{(2)}][1 - S^2_{\boldsymbol{X_1},\boldsymbol{X_2}}]}{1 - S^2_{\boldsymbol{X_1},\boldsymbol{X_2}} + \rho_2^2 - \rho_1^2}.$$

Obviously, the coefficient of the added variable depends solely on the coefficient which is highly correlated with the new variable. In addition, the

https://doi.org/10.26219/heal.aueb.5567

example described above with the three covariates, can be generalized. It indicates the estimation of regression coefficients, for any relation between the new and the previous covariates (i.e. for any ρ_1, ρ_2).

3.3 Recursive formula using hat matrix

In multiple linear regression, it is more efficient to use matrices due to the potentially large number of explanatory variables. There are several matrix formulations for the linear regression model. One of the matrices that appear in many formulations is the projection matrix. Therefore, this section contains further discussion of the relation (3.2) using this matrix.

Let's start from the formula

$$\hat{\boldsymbol{b}}_{\boldsymbol{r+1}} = \begin{bmatrix} \hat{\boldsymbol{b}_{\boldsymbol{r}}} + (\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}(\frac{1}{c}\boldsymbol{W^T}\hat{\boldsymbol{b}_{\boldsymbol{r}}} - \frac{1}{c}R) \\ -(\frac{1}{c}\boldsymbol{W^T}\hat{\boldsymbol{b}_{\boldsymbol{r}}} - \frac{1}{c}R) \end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{b}_{\boldsymbol{r}}} + \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}\boldsymbol{W^T}\hat{\boldsymbol{b}_{\boldsymbol{r}}} - \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}R \\ -(\frac{1}{c}\boldsymbol{W^T}\hat{\boldsymbol{b}_{\boldsymbol{r}}} - \frac{1}{c}R) \end{bmatrix},$$

where $c = z - \boldsymbol{W^T}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}$ and $z = \sum_{i=1}^n (X_{r+1,i})^2$,

$$\boldsymbol{W} = \begin{bmatrix} \sum_{i=1}^n (X_{r+1,i}) \\ \sum_{i=1}^n (X_{1,i}X_{r+1,i}) \\ \vdots \\ \sum_{i=1}^n (X_{r,i}X_{r+1,i}) \end{bmatrix} \text{ and } R = \sum_{i=1}^n (X_{r+1,i}Y_i).$$

We have to express the above sums $(z, \boldsymbol{W}, R)$ in matricial form. Especially,

$$\boldsymbol{X_{r+1}^T X_{r+1}} = \begin{bmatrix} X_{r+1,1} & \dots & X_{r+1,n} \end{bmatrix} \begin{bmatrix} X_{r+1,1} \\ \vdots \\ X_{r+1,n} \end{bmatrix} = \sum_{i=1}^n (X_{r+1,i})^2 = z,$$

$$\boldsymbol{X^T X_{r+1}} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ X_{1,1} & X_{1,2} & \dots & X_{1,n} \\ \vdots & \vdots & \dots & \vdots \\ X_{r,1} & X_{r,2} & \dots & X_{r,n} \end{bmatrix} \begin{bmatrix} X_{r+1,1} \\ X_{r+1,2} \\ \vdots \\ X_{r+1,n} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^n (X_{r+1,i}) \\ \sum_{i=1}^n (X_{1,i}X_{r+1,i}) \\ \vdots \\ \sum_{i=1}^n (X_{r,i}X_{r+1,i}) \end{bmatrix} = \boldsymbol{W},$$

https://doi.org/10.26219/heal.aueb.5567

$$\boldsymbol{X_{r+1}^T Y} = \begin{bmatrix} X_{r+1,1} & \dots & X_{r+1,n} \end{bmatrix} \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix} = \sum_{i=1}^{n} (X_{r+1,i} Y_i) = R.$$

In summary, we get

- $z = \boldsymbol{X_{r+1}^T X_{r+1}}$
- $\boldsymbol{W} = \boldsymbol{X^T X_{r+1}}$
- $\boldsymbol{W^T} = \boldsymbol{X_{r+1}^T X}$
- $R = \boldsymbol{X_{r+1}^T Y}$

Apart from these, as we can see, $\hat{\boldsymbol{Y}} = \boldsymbol{X}(\boldsymbol{X^T X})^{-1}\boldsymbol{X^T Y}$, which means that the fitted values are expressed in terms of $\boldsymbol{X}$ and $\boldsymbol{Y}$ matrices. Therefore, we define the matrix $\boldsymbol{P} = \boldsymbol{X}(\boldsymbol{X^T X})^{-1}\boldsymbol{X^T}$.

$\boldsymbol{P}$ is the projection matrix and is also named "hat matrix". Because of $\boldsymbol{PY} = \hat{\boldsymbol{Y}}$, it expresses the influence each response value has on each fitted value. $\boldsymbol{P}$ has several useful properties. First of all, $\boldsymbol{P}$ is symmetric, i.e. $\boldsymbol{P} = \boldsymbol{P^T}$. Secondly, it is idempotent, i.e. $\boldsymbol{P^2} = \boldsymbol{PP} = \boldsymbol{P}$. Thus, we also have $\boldsymbol{P(1-P)} = \boldsymbol{(1-P)P} = \boldsymbol{0}$. It is worth mentioning that the design matrix $\boldsymbol{X}$ is invariant under $\boldsymbol{P}$, i.e. $\boldsymbol{PX} = \boldsymbol{X}$, hence $\boldsymbol{(1-P)X} = \boldsymbol{0}$.

As for the residuals, we can express them as linear combinations of the Y_i, i.e. $\boldsymbol{e} = \boldsymbol{Y} - \hat{\boldsymbol{Y}} = \boldsymbol{Y} - \boldsymbol{PY} = (\boldsymbol{1-P})\boldsymbol{Y}$. Now, let:

- $\boldsymbol{P} = \boldsymbol{X}(\boldsymbol{X^T X})^{-1}\boldsymbol{X^T}$
- $\boldsymbol{M} = \boldsymbol{I_n} - \boldsymbol{P}$
- $\boldsymbol{u} = (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T X_{r+1}}$

by the formula (1.2), the vector of regression coefficients will be

 https://doi.org/10.26219/heal.aueb.5567

$$
\begin{aligned}
\hat{\boldsymbol{b}}_{\boldsymbol{r+1}} &= \begin{bmatrix} \hat{\boldsymbol{b}_{\boldsymbol{r}}} + \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}\boldsymbol{W^T}\hat{\boldsymbol{b}_{\boldsymbol{r}}} - \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W}R \\ -(\frac{1}{c}\boldsymbol{W^T}\hat{\boldsymbol{b}_{\boldsymbol{r}}} - \frac{1}{c}R) \end{bmatrix} \\
&= \begin{bmatrix} \hat{\boldsymbol{b}_{\boldsymbol{r}}} + \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T X_{r+1} X_{r+1}^T X}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T Y} - \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T X_{r+1} X_{r+1}^T Y} \\ -\frac{1}{c}\boldsymbol{X_{r+1}^T X}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T Y} + \frac{1}{c}\boldsymbol{X_{r+1}^T Y} \end{bmatrix} \\
&= \begin{bmatrix} \hat{\boldsymbol{b}_{\boldsymbol{r}}} - \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T X_{r+1} X_{r+1}^T}(\boldsymbol{Y} - \boldsymbol{X}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T Y}) \\ \frac{1}{c}\boldsymbol{X_{r+1}^T}(\boldsymbol{I_n} - \boldsymbol{X}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T})\boldsymbol{Y} \end{bmatrix} \\
&= \begin{bmatrix} \hat{\boldsymbol{b}_{\boldsymbol{r}}} - \frac{1}{c}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T X_{r+1} X_{r+1}^T}(\boldsymbol{I_n} - \boldsymbol{X}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T})\boldsymbol{Y} \\ \frac{1}{c}\boldsymbol{X_{r+1}^T}(\boldsymbol{I_n} - \boldsymbol{P})\boldsymbol{Y} \end{bmatrix}.
\end{aligned}
$$

Finally (Seber and Lee, 2003),

$$
\Rightarrow \hat{\mathbf{b}}_{\mathbf{r+1}} = \begin{bmatrix} \hat{\mathbf{b}_{\mathbf{r}}} - \frac{1}{c}\boldsymbol{u X_{r+1}^T M Y} \\ \frac{1}{c}\boldsymbol{X_{r+1}^T M Y} \end{bmatrix}, \tag{3.4}
$$

where

$$
\begin{aligned}
c = z - \boldsymbol{W^T}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{W} &= \boldsymbol{X_{r+1}^T X_{r+1}} - \boldsymbol{X_{r+1}^T X}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T X_{r+1}} \\
&= \boldsymbol{X_{r+1}^T}(\boldsymbol{I_n} - \boldsymbol{X}(\boldsymbol{X^T X})_r^{-1}\boldsymbol{X^T})\boldsymbol{X_{r+1}} \\
&= \boldsymbol{X_{r+1}^T M X_{r+1}}.
\end{aligned}
$$

Obviously, the formula (3.4) is equivalent to the formula (3.2). This is very useful, because if we set $\beta_{r+1}^{(r+1)}$ the coefficients of the new variable, i.e.

https://doi.org/10.26219/heal.aueb.5567

$\beta_{r+1}^{(r+1)} = \frac{1}{c}\boldsymbol{X_{r+1}^T M Y}$, we will get

$$\hat{\boldsymbol{b}}_{\boldsymbol{r+1}} = \begin{bmatrix} \hat{\boldsymbol{b}_r} - \boldsymbol{u}\beta_{r+1}^{(r+1)} \\ \beta_{r+1}^{(r+1)} \end{bmatrix}.$$

Hence, we can clearly see the relation between the new and the previous estimation of coefficients.

3.4 Regression by orthogonalization- Gram Schmidt procedure

Linear regression has a nice geometrical interpretation. It is based on projections and it uses the projection matrix we saw in the previous section. Using these, we will describe regression by Orthogonalization, which is another way to obtain the least square estimates. In particular, this algorithm ,which is known as the Gram Schmidt procedure, orthogonalizes the covariates and then uses simple projections.

We need some notations and clarifications in order to continue. We set $\boldsymbol{a} = (a_1, \ldots, a_n)^T$, $\boldsymbol{b} = (b_1, \ldots, b_n)^T$ and we define the inner product between the vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ as $\langle \boldsymbol{a}, \boldsymbol{b} \rangle = \sum_{i=1}^n a_i b_i = \boldsymbol{a^T b}$. Also, two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ are said to be orthogonal if $\langle \boldsymbol{a}, \boldsymbol{b} \rangle = 0$. Finally, when we say regress $\boldsymbol{b}$ on $\boldsymbol{a}$, we mean that we have a simple univariate regression of $\boldsymbol{b}$ on $\boldsymbol{a}$, without intercept, and we produce a coefficient $\hat{\gamma} = \frac{\langle \boldsymbol{a}, \boldsymbol{b} \rangle}{\langle \boldsymbol{a}, \boldsymbol{a} \rangle}$ and a residual vector $\boldsymbol{z} = \boldsymbol{b} - \hat{\gamma}\boldsymbol{a}$.

Let's begin with a simple model $\boldsymbol{y} = \beta_0 \boldsymbol{1} + \beta_1 \boldsymbol{x} + \boldsymbol{\epsilon}$. We can estimate the coefficient β_1 with the following two steps:

1. Regress $\boldsymbol{x}$ on $\boldsymbol{1}$, producing the coefficient $\frac{\langle \boldsymbol{1}, \boldsymbol{x} \rangle}{\langle \boldsymbol{1}, \boldsymbol{1} \rangle} = \frac{\sum_{i=1}^n x_i}{n} = \bar{x}$ and the residual $\boldsymbol{z} = \boldsymbol{x} - \bar{x}\boldsymbol{1}$.

2. Regress $\boldsymbol{y}$ on $\boldsymbol{z}$, producing the coefficient $\hat{\beta}_1 = \dfrac{\langle \boldsymbol{z}, \boldsymbol{y} \rangle}{\langle \boldsymbol{z}, \boldsymbol{z} \rangle} = \dfrac{\langle \boldsymbol{x} - \bar{x}\boldsymbol{1}, \boldsymbol{y} \rangle}{\langle \boldsymbol{x} - \bar{x}\boldsymbol{1}, \boldsymbol{x} - \bar{x}\boldsymbol{1} \rangle}$.

This procedure can be generalized for multiple linear regression. It is described by the following algorithm(Friedman, Hastie and Tibshirani, 2008).

Algorithm 2: Regression by Orthogonalization

Input: $\boldsymbol{Y}, \boldsymbol{X_1}, \ldots, \boldsymbol{X_{r+1}}$
Output: $\hat{\beta}_{r+1}$
Initialize $\boldsymbol{z_0} = \boldsymbol{X_0} = \boldsymbol{1}$
for $j = 1, \ldots, r+1$ **do**
  Regress $\boldsymbol{X_j}$ on $\boldsymbol{z_0}, \ldots, \boldsymbol{z_{j-1}}$ to yield coefficients
  $\hat{\gamma}_{i,j} = \dfrac{\langle \boldsymbol{z_i}, \boldsymbol{X_j} \rangle}{\langle \boldsymbol{z_i}, \boldsymbol{z_i} \rangle}, i = 0, \ldots, j-1$ and residual vector
  $\boldsymbol{z_j} = \boldsymbol{X_j} - \sum_{k=0}^{j-1} \hat{\gamma}_{k,j} \boldsymbol{z_k}$
end
Regress $\boldsymbol{Y}$ on the residual $\boldsymbol{z_{r+1}}$ to estimate $\hat{\beta}_{r+1}$

This algorithm assumes that the model has $r+1$ covariates and the output, namely the coefficient of $\boldsymbol{X_{r+1}}$, will be $\hat{\beta}_{r+1} = \dfrac{\langle \boldsymbol{z_{r+1}}, \boldsymbol{Y} \rangle}{\langle \boldsymbol{z_{r+1}}, \boldsymbol{z_{r+1}} \rangle}$. Since the residual vector $\boldsymbol{z_j}$ is produced by regressing $\boldsymbol{X_j}$ onto $\boldsymbol{z_1}, \ldots, \boldsymbol{z_{j-1}}$, $\boldsymbol{z_j}$ is orthogonal to $\boldsymbol{z_1}, \ldots, \boldsymbol{z_{j-1}}$, so $\boldsymbol{z_1}, \ldots, \boldsymbol{z_{r+1}}$ are orthogonal.

It is important to highlight that if our covariates $\boldsymbol{X_1}, \ldots, \boldsymbol{X_{r+1}}$ are orthogonal, namely $\langle \boldsymbol{X_l}, \boldsymbol{X_k} \rangle = 0$ for all $l \neq k$, then the multiple least squares estimates $\hat{\beta}_j$ are equal to $\hat{\beta}_j = \dfrac{\langle \boldsymbol{X_j}, \boldsymbol{Y} \rangle}{\langle \boldsymbol{X_j}, \boldsymbol{X_j} \rangle}$. This means that there is no effect on the estimation of the parameters. If we suppose that $\boldsymbol{X_{r+1}}$ is highly correlated with some of the other covariates, then the coefficient $\hat{\beta}_{r+1}$ will be unstable, since $\boldsymbol{z_{r+1}}$ will tend to zero.

It is interesting to note that the output $\hat{\beta}_{r+1}$ is the coefficient of the explanatory variable $\boldsymbol{X_{r+1}}$ in the multiple regression of $\boldsymbol{Y}$ on $\boldsymbol{X_1}, \ldots, \boldsymbol{X_{r+1}}$. According to lecture notes of Tibshirani (2013), the proof is simple. We know that $\boldsymbol{z_1}, \ldots, \boldsymbol{z_{r+1}}$ are orthogonal. Since $\boldsymbol{z_j} = \boldsymbol{X_j} - \sum_{k=0}^{j-1} \hat{\gamma}_{k,j} \boldsymbol{z_k}$ for any

https://doi.org/10.26219/heal.aueb.5567

$j = 1, \ldots, r+1$, each of $\boldsymbol{z_j}$ is a linear combination of $\boldsymbol{X_1}, \ldots, \boldsymbol{X_j}$, so

$$span\{\boldsymbol{z_1}, \ldots, \boldsymbol{z_j}\} \subseteq span\{\boldsymbol{X_1}, \ldots, \boldsymbol{X_j}\}.$$

But $\boldsymbol{X_j} = \boldsymbol{z_j} + \sum_{k=0}^{j-1} \hat{\gamma}_{k,j} \boldsymbol{z_k}$, thus each of $\boldsymbol{X_j}$ is a linear combination of $\boldsymbol{z_1}, \ldots, \boldsymbol{z_j}$, so

$$span\{\boldsymbol{X_1}, \ldots, \boldsymbol{X_j}\} \subseteq span\{\boldsymbol{z_1}, \ldots, \boldsymbol{z_j}\}$$

Hence

$$span\{\boldsymbol{X_1}, \ldots, \boldsymbol{X_j}\} = span\{\boldsymbol{z_1}, \ldots, \boldsymbol{z_j}\}.$$

As a consequence, the regression of $\boldsymbol{Y}$ on $\boldsymbol{X_1}, \ldots, \boldsymbol{X_{r+1}}$ is the same as the regression of $\boldsymbol{Y}$ on $\boldsymbol{z_1}, \ldots, \boldsymbol{z_{r+1}}$. We call this $\hat{\boldsymbol{Y}}$ and for some coefficients $\alpha_1, \ldots, \alpha_{r+1}$ we obtain

$$\hat{\boldsymbol{Y}} = \alpha_1 \boldsymbol{z_1} + \ldots + \alpha_{r+1} \boldsymbol{z_{r+1}}$$

Since $\boldsymbol{z_1}, \ldots, \boldsymbol{z_{r+1}}$ are orthogonal, the coefficients $\alpha_1, \ldots, \alpha_{r+1}$ can be calculated by univariate linear regression, that is

$$\alpha_{r+1} = \frac{\langle \boldsymbol{z_{r+1}}, \boldsymbol{Y} \rangle}{\langle \boldsymbol{z_{r+1}}, \boldsymbol{z_{r+1}} \rangle} = \hat{\beta}_{r+1}.$$

Thus,

$$\hat{\boldsymbol{Y}} = \alpha_1 \boldsymbol{z_1} + \ldots + \alpha_r \boldsymbol{z_r} + \hat{\beta}_{r+1} \boldsymbol{z_{r+1}}.$$

From the relation $\boldsymbol{X_{r+1}} = \boldsymbol{z_{r+1}} + \sum_{k=0}^{r} \hat{\gamma}_{k,j} \boldsymbol{z_k}$, the explanatory variable $\boldsymbol{X_{r+1}}$ exists only through the $\boldsymbol{z_{r+1}}$ and its coefficient is 1, therefore for some $c_1, \ldots, c_r$ we can write

$$\hat{\boldsymbol{Y}} = c_1 \boldsymbol{X_1} + \ldots + c_r \boldsymbol{X_r} + \hat{\beta}_{r+1} \boldsymbol{X_{r+1}}.$$

Hence, we prove our claim.

Another way to show this is proving that the coefficient of the variable

https://doi.org/10.26219/heal.aueb.5567

$\boldsymbol{X_{r+1}}$ from the formula (3.4) is the same as the output of the previous algorithm. Specifically, we proved that

$$\hat{\mathbf{b}}_{\mathbf{r+1}} = \begin{bmatrix} \hat{\mathbf{b}}_{\mathbf{r}} - \frac{1}{c}\boldsymbol{u}\boldsymbol{X}_{\boldsymbol{r+1}}^{\boldsymbol{T}}\boldsymbol{M}\boldsymbol{Y} \\ \frac{1}{c}\boldsymbol{X}_{\boldsymbol{r+1}}^{\boldsymbol{T}}\boldsymbol{M}\boldsymbol{Y} \end{bmatrix},$$

where $c = \boldsymbol{X}_{\boldsymbol{r+1}}^{\boldsymbol{T}}\boldsymbol{M}\boldsymbol{X}_{\boldsymbol{r+1}}$ and $\boldsymbol{M} = \boldsymbol{I_n} - \boldsymbol{P}$.
Thus, the coefficient of $\boldsymbol{X_{r+1}}$ in the multiple regression is

$$\begin{aligned}\hat{\beta}_{r+1}^{(r+1)} &= \frac{\boldsymbol{X}_{\boldsymbol{r+1}}^{\boldsymbol{T}}\boldsymbol{M}\boldsymbol{Y}}{\boldsymbol{X}_{\boldsymbol{r+1}}^{\boldsymbol{T}}\boldsymbol{M}\boldsymbol{X}_{\boldsymbol{r+1}}} \overset{(\boldsymbol{M}^2=\boldsymbol{M})}{=} \frac{\boldsymbol{X}_{\boldsymbol{r+1}}^{\boldsymbol{T}}\boldsymbol{M}\boldsymbol{Y}}{\boldsymbol{X}_{\boldsymbol{r+1}}^{\boldsymbol{T}}\boldsymbol{M}\boldsymbol{M}\boldsymbol{X}_{\boldsymbol{r+1}}} \overset{(\boldsymbol{M}^{\boldsymbol{T}}=\boldsymbol{M})}{=} \frac{(\boldsymbol{M}\boldsymbol{X}_{\boldsymbol{r+1}})^{\boldsymbol{T}}\boldsymbol{Y}}{(\boldsymbol{M}\boldsymbol{X}_{\boldsymbol{r+1}})^{\boldsymbol{T}}\boldsymbol{M}\boldsymbol{X}_{\boldsymbol{r+1}}} \\ &= \frac{\langle \boldsymbol{M}\boldsymbol{X}_{\boldsymbol{r+1}}, \boldsymbol{Y}\rangle}{\langle \boldsymbol{M}\boldsymbol{X}_{\boldsymbol{r+1}}, \boldsymbol{M}\boldsymbol{X}_{\boldsymbol{r+1}}\rangle},\end{aligned}$$

which is the same as the output of the algorithm $\hat{\beta}_{r+1} = \frac{\langle \boldsymbol{z_{r+1}}, \boldsymbol{Y}\rangle}{\langle \boldsymbol{z_{r+1}}, \boldsymbol{z_{r+1}}\rangle}$ because $\boldsymbol{z_{r+1}}$ is the residual of the orthogonal projection of $\boldsymbol{X_{r+1}}$ onto $\boldsymbol{z_0}, \ldots, \boldsymbol{z_r}$, i.e. the residual from regressing $\boldsymbol{X_{r+1}}$ onto $\boldsymbol{1}, \boldsymbol{X_1}, \ldots, \boldsymbol{X_r}$. Algorithm 2 computes only the coefficients of the last variable. We will describe the procedure, which is applied for the estimation of the other coefficients (Zhang, 2016). We assume that we have only three explanatory variables in the model i.e.

$$\boldsymbol{Y} = \beta_0\boldsymbol{1} + \beta_1\boldsymbol{X_1} + \beta_2\boldsymbol{X_2} + \beta_3\boldsymbol{X_3} + \boldsymbol{\epsilon}.$$

From the algorithm, we compute the residual

$$\begin{aligned}\boldsymbol{z_0} &= \boldsymbol{1}, \\ \boldsymbol{z_1} &= \boldsymbol{X_1} - \hat{\gamma}_{0,1}\boldsymbol{z_0}, \\ \boldsymbol{z_2} &= \boldsymbol{X_2} - \hat{\gamma}_{0,2}\boldsymbol{z_0} - \hat{\gamma}_{1,2}\boldsymbol{z_1}, \\ \boldsymbol{z_3} &= \boldsymbol{X_3} - \hat{\gamma}_{0,3}\boldsymbol{z_0} - \hat{\gamma}_{1,3}\boldsymbol{z_1} - \hat{\gamma}_{2,3}\boldsymbol{z_2}.\end{aligned}$$

 https://doi.org/10.26219/heal.aueb.5567

As detailed below, $span\{\mathbf{1}, \mathbf{X_1}, \mathbf{X_2}, \mathbf{X_3}\} = span\{\mathbf{1}, \mathbf{z_1}, \mathbf{z_2}, \mathbf{z_3}\}$,because

$$\begin{aligned}
\beta_0\mathbf{1} + \beta_1\mathbf{X_1} + \beta_2\mathbf{X_2} + \beta_3\mathbf{X_3} &= \beta_0\mathbf{1} + \beta_1(\mathbf{z_1} + \hat{\gamma}_{0,1}\mathbf{z_0}) + \beta_2(\mathbf{z_2} + \hat{\gamma}_{0,2}\mathbf{z_0} + \hat{\gamma}_{1,2}\mathbf{z_1}) + \\
&\beta_3(\mathbf{z_3} + \hat{\gamma}_{0,3}\mathbf{z_0} + \hat{\gamma}_{1,3}\mathbf{z_1} + \hat{\gamma}_{2,3}\mathbf{z_2}) \\
&= \mathbf{z_0}(\beta_0 + \hat{\gamma}_{0,1}\beta_1 + \hat{\gamma}_{0,2}\beta_2 + \hat{\gamma}_{0,3}\beta_3) + \\
&\mathbf{z_1}(\beta_1 + \hat{\gamma}_{1,2}\beta_2 + \hat{\gamma}_{1,3}\beta_3) + \mathbf{z_2}(\beta_2 + \hat{\gamma}_{2,3}\beta_3) + \mathbf{z_3}\beta_3 \\
&= \eta_0\mathbf{z_0} + \eta_1\mathbf{z_1} + \eta_2\mathbf{z_2} + \eta_3\mathbf{z_3}.
\end{aligned}$$

So we can regress $\mathbf{Y}$ on $\{\mathbf{1}, \mathbf{z_1}, \mathbf{z_2}, \mathbf{z_3}\}$ one by one and calculate the coefficients, namely

- Regress $\mathbf{Y}$ on $\mathbf{z_3}$ and compute the coefficient

$$\hat{\beta}_3 = \frac{\langle \mathbf{z_3}, \mathbf{Y} \rangle}{\langle \mathbf{z_3}, \mathbf{z_3} \rangle}.$$

- Regress $\mathbf{Y}$ on $\mathbf{z_2}$, producing $\hat{\eta}_2$ and estimate the coefficient

$$\hat{\beta}_2 = \hat{\eta}_2 - \hat{\beta}_3\hat{\gamma}_{2,3}.$$

- Regress $\mathbf{Y}$ on $\mathbf{z_1}$, producing $\hat{\eta}_1$ and estimate the coefficient

$$\hat{\beta}_1 = \hat{\eta}_1 - \hat{\beta}_3\hat{\gamma}_{1,3} - \hat{\beta}_2\hat{\gamma}_{1,2}.$$

- Regress $\mathbf{Y}$ on $\mathbf{z_0} \equiv \mathbf{1}$, producing $\hat{\eta}_0$ and estimate the coefficient

$$\hat{\beta}_0 = \hat{\eta}_0 - \hat{\beta}_3\hat{\gamma}_{0,3} - \hat{\beta}_2\hat{\gamma}_{0,2} - \hat{\beta}_1\hat{\gamma}_{0,1}.$$

This way can be generalized for any number of explanatory variables. If we incorporate this procedure into the algorithm, we can estimate all the coefficients of the model. Our aim is to compare this algorithm with the recursive formula, which is proved in a previous section. It is obvious that

https://doi.org/10.26219/heal.aueb.5567

the results of these two procedures are the same. However, Algorithm 2 makes extra computations in order to estimate β_j for $j = r+1, r, \ldots, 0$ (i.e. in inverse order). Thus, this procedure is slower than the recursive formula we proved.

3.5 QR decomposition

3.5.1 Regression and QR decomposition

The matricial expression of multiple regression is

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{b} + \boldsymbol{\epsilon},$$

where $\boldsymbol{X}$ is the $n \times p$ regression matrix, n is the number of observations, $p-1$ is the number of variables and we assume $n \geq p$. We will try to make the least squares problem simpler using QR decomposition.

Generally speaking, QR decomposition is a factorization of the matrix $\boldsymbol{X}$ into a product of matrices, namely $\boldsymbol{X} = \boldsymbol{Q}\boldsymbol{R}$, where $\boldsymbol{Q}$ is an $n \times n$ orthogonal matrix and $\boldsymbol{R}$ is an $n \times p$ upper triangular matrix that is zero below the main diagonal. For our regression matrix $\boldsymbol{X}$ (Bates 2010), the QR decomposition is written

$$\boldsymbol{X} = \boldsymbol{Q}\begin{bmatrix}\boldsymbol{R_1}\\ \boldsymbol{0}\end{bmatrix} = \begin{bmatrix}\boldsymbol{Q_1} & \boldsymbol{Q_2}\end{bmatrix}\begin{bmatrix}\boldsymbol{R_1}\\ \boldsymbol{0}\end{bmatrix} = \boldsymbol{Q_1}\boldsymbol{R_1},$$

where $\boldsymbol{R_1}$ is an $p \times p$ upper triangular matrix, $\boldsymbol{Q_1}$ is the first p columns of $\boldsymbol{Q}$ and $\boldsymbol{Q_2}$ is the last $n-p$ columns of $\boldsymbol{Q}$.

Therefore, substituting $\boldsymbol{X} = \boldsymbol{Q_1 R_1}$,the least squares estimates of $\boldsymbol{\beta}$ is

$$\begin{aligned}\hat{\boldsymbol{\beta}} &= (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T Y} \\ &= ((\boldsymbol{Q_1 R_1})^T \boldsymbol{Q_1 R_1})^{-1}(\boldsymbol{Q_1 R_1})^T \boldsymbol{Y} \\ &= (\boldsymbol{R_1^T Q_1^T Q_1 R_1})^{-1}\boldsymbol{R_1^T Q_1^T Y} \\ &= \boldsymbol{R_1^{-1}}(\boldsymbol{R_1^T})^{-1}\boldsymbol{R_1^T Q_1^T Y} \\ &= \boldsymbol{R_1^{-1} Q_1^T Y} \\ &= \boldsymbol{R_1^{-1} d},\end{aligned}$$

where $\boldsymbol{d} = \boldsymbol{Q_1^T Y}$. Thus, we have an upper triangular system of equations $\boldsymbol{R_1 \beta} = \boldsymbol{d}$ which may be solved quickly by back substitution.

There are several methods for computing the QR decomposition. We will describe two of them.

3.5.2 Methods for QR decomposition

Householder Reflections

An efficient approach for QR factorization is to use Householder matrices which introduce zeros in the subdiagonal elements of a column. Let $\boldsymbol{A}$ be a $n \times p$ matrix and $\boldsymbol{x}$ be the first column of the matrix,

$$\boldsymbol{A} = \begin{bmatrix} \alpha_{11} & \alpha_{12} & \dots & \alpha_{1p} \\ \alpha_{21} & \alpha_{22} & \dots & \alpha_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_{n1} & \alpha_{n2} & \dots & \alpha_{np} \end{bmatrix} \qquad \textit{and} \qquad \boldsymbol{x} = \begin{bmatrix} \alpha_{11} \\ \alpha_{21} \\ \vdots \\ \alpha_{n1} \end{bmatrix}.$$

For any vector $\boldsymbol{u} \in \mathbb{R}^n$, $\boldsymbol{u} \neq 0$, we define the Householder matrix as

$$\boldsymbol{H} = \boldsymbol{I} - \frac{2\boldsymbol{u u^T}}{\boldsymbol{u^T u}}.$$

We have to calculate a vector $\boldsymbol{u}$, such that

$$\boldsymbol{H_1}\begin{bmatrix}\alpha_{11}\\ \alpha_{21}\\ \vdots\\ \alpha_{n1}\end{bmatrix}=\begin{bmatrix}\dot{\alpha}_{11}\\ 0\\ \vdots\\ 0\end{bmatrix}.$$

This is $\boldsymbol{u}=\boldsymbol{x}+sign(\alpha_{11})\|\boldsymbol{x}\|_2\boldsymbol{e_1}$, so

$$\boldsymbol{A^{(1)}}=\boldsymbol{H_1A}=\begin{bmatrix}\dot{\alpha}_{11} & \dot{\alpha}_{12} & \dots & \dot{\alpha}_{1p}\\ 0 & \dot{\alpha}_{22} & \dots & \dot{\alpha}_{2p}\\ \vdots & \vdots & \ddots & \vdots\\ 0 & \dot{\alpha}_{n2} & \dots & \dot{\alpha}_{np}\end{bmatrix}.$$

Then, we calculate a new vector $\boldsymbol{u}$, such that

$$\boldsymbol{\hat{H}_2}\begin{bmatrix}\dot{\alpha}_{22}\\ \vdots\\ \dot{\alpha}_{n2}\end{bmatrix}=\begin{bmatrix}*\\ 0\\ \vdots\\ 0\end{bmatrix}\quad,\qquad \boldsymbol{H_2}=\begin{bmatrix}1 & 0\\ 0 & \boldsymbol{\hat{H}_2}\end{bmatrix}.$$

Thus, $\boldsymbol{A^{(2)}}=\boldsymbol{H_2A^{(1)}}$. Continuing this process, after κ iterations, where $\kappa=min(n-1,p)$, we obtain

$$\boldsymbol{A^{(\kappa)}}=\boldsymbol{H_\kappa}\cdots\boldsymbol{H_2H_1A}=\boldsymbol{R},$$

where $\boldsymbol{R}$ is an upper triangular matrix. For $j=1,\dots,\kappa$

$$\boldsymbol{H_j}=\begin{bmatrix}\boldsymbol{I_{j-1}} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{\hat{H}_j}\end{bmatrix}.$$

Hence, $\boldsymbol{R}=\boldsymbol{Q^TA}$ with $\boldsymbol{Q^T}=\boldsymbol{H_\kappa}\cdots\boldsymbol{H_2H_1}$ i.e. $\boldsymbol{Q}=\boldsymbol{H_1H_2}\cdots\boldsymbol{H_\kappa}$.
So, using this procedure we have factored the matrix $\boldsymbol{A}$ as $\boldsymbol{A}=\boldsymbol{QR}$.

https://doi.org/10.26219/heal.aueb.5567

Givens Rotations

Another method for QR decomposition is using Givens matrices in order to progressively introduce zeros below the diagonal to one element at a time (Hammarling S. and Lucas C., 2008). Let $\boldsymbol{Q}$ be the rotation matrix, which rotates $\boldsymbol{x} \in \mathbb{R}^2$ by θ

$$\boldsymbol{Q} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}.$$

So, we get

$$\boldsymbol{Q^T x} = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \begin{bmatrix} x_i \\ x_j \end{bmatrix} = \begin{bmatrix} x_i cos(\theta) + x_j sin(\theta) \\ -x_i sin(\theta) + x_j cos(\theta) \end{bmatrix}.$$

If we write $\boldsymbol{x} = \sqrt{x_i^2 + x_j^2} \begin{bmatrix} \frac{x_i}{\sqrt{x_i^2 + x_j^2}} \\ \frac{x_j}{\sqrt{x_i^2 + x_j^2}} \end{bmatrix}$ and set

$$sin(\theta) = \frac{x_j}{\sqrt{x_i^2 + x_j^2}} \qquad and \qquad cos(\theta) = \frac{x_i}{\sqrt{x_i^2 + x_j^2}},$$

we will obtain

$$\boldsymbol{Q^T x} = \sqrt{x_i^2 + x_j^2} \begin{bmatrix} \frac{x_i}{\sqrt{x_i^2 + x_j^2}} cos(\theta) + \frac{x_j}{\sqrt{x_i^2 + x_j^2}} sin(\theta) \\ -\frac{x_i}{\sqrt{x_i^2 + x_j^2}} sin(\theta) + \frac{x_j}{\sqrt{x_i^2 + x_j^2}} cos(\theta) \end{bmatrix} = \begin{bmatrix} \sqrt{x_i^2 + x_j^2} \\ 0 \end{bmatrix}.$$

As we can see, we set an element to zero. Generalizing, a Givens matrix

https://doi.org/10.26219/heal.aueb.5567

$\boldsymbol{G(i,j)} \in \mathbb{R}^{n\times n}$ has the form

$$\boldsymbol{G(i,j)} = \begin{bmatrix} \boldsymbol{I} & & & & \\ & c & & s & \\ & & \boldsymbol{I} & & \\ & -s & & c & \\ & & & & \boldsymbol{I} \end{bmatrix},$$

where $c = cos(\theta)$ and $s = sin(\theta)$. Again, setting

$$s = -\frac{x_j}{\sqrt{x_i^2 + x_j^2}} \qquad and \qquad c = \frac{x_i}{\sqrt{x_i^2 + x_j^2}}.$$

For $\boldsymbol{x} \in \mathbb{R}^n$, we get $\boldsymbol{G(i,j)^T x = y}$ where

$$y_k = \begin{cases} cx_i - sx_j, & k = i \\ 0, & k = j \\ x_k, & k \neq i, j \end{cases}.$$

There is a significant advantage here, that is only the ith and jth rows are affected. Our aim is to transform $\boldsymbol{A}$ to an upper triangular matrix. So, we will use Givens matrices to introduce one zero at a time in the subdiagonal elements of $\boldsymbol{A}$, namely

$$\underbrace{\boldsymbol{G(n-1,n)^T}}_{\text{p th column}} \cdots \underbrace{\boldsymbol{G(2,3)^T \cdots G(n-1,n)^T}}_{\text{2nd column}} \underbrace{\boldsymbol{G(1,2)^T \cdots G(n-2,n-1)^T G(n-1,n)^T}}_{\text{1st column}} \boldsymbol{A}$$
$$= \boldsymbol{Q^T A = R}.$$

where $\boldsymbol{Q = G(n-1,n)^T G(n-2,n-1)^T \cdots G(1,2)^T \cdots G(n-1,n)^T}$ This process is called QR decomposition with Givens Rotation. The following algorithm shows how to calculate c and s for Givens matrices. We search for c and s,

https://doi.org/10.26219/heal.aueb.5567

such that $s^2 + c^2 = 1$ and $\begin{bmatrix} c & s \\ -s & c \end{bmatrix}^T \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} d \\ 0 \end{bmatrix}$.

Algorithm 3: Calculate c and s for Givens Rotation

Input: a, b
Result: c, s
if $b = 0$ **then**
  $c = 1, s = 0$
else
  if $\mid b \mid \geq \mid a \mid$ **then**
    $t = -\dfrac{a}{b}, s = \dfrac{1}{\sqrt{1+t^2}}, c = st$
  else
    $t = -\dfrac{b}{a}, c = \dfrac{1}{\sqrt{1+t^2}}, s = ct$
  end
end

Comparing the two methods, the Givens Rotations method can be preferable for three reasons (Lambers, 2010). First of all, it can apply sets of rotations in parallel, because each rotation modifies only two rows of the matrix $\boldsymbol{A}$. Secondly, it is easier to update the QR decomposition using the Givens Rotations, for example adding a column to $\boldsymbol{A}$, as we can see in next subsection. And finally, it is more efficient when $\boldsymbol{A}$ is sparse.

3.5.3 Updating the QR decomposition

In previous section, we proved a recursive formula to estimate the coefficients of a multiple regression model. In order to achieve this, every time we added a new variable in the model, we tried to find a recursive relation with the previous model. Now, we will engage in the same problem using QR decomposition (Hammarling S. and Lucas C., 2008). At the beginning, we will describe the framework for a general least squares problem and then specifically for regression.

Let an $n \times p$ matrix $\boldsymbol{X}$ and the factorization of it $\boldsymbol{X} = \boldsymbol{Q}\boldsymbol{R}$, where $\boldsymbol{Q}$ is an $n \times n$ orthogonal matrix and $\boldsymbol{R}$ is an $n \times p$ upper triangular matrix. Let us consider the linear system

$$\begin{aligned} \boldsymbol{Y} &= \boldsymbol{X}\boldsymbol{b} \\ \boldsymbol{Y} &= \boldsymbol{Q}\boldsymbol{R}\boldsymbol{b} \\ \boldsymbol{Q}^{\boldsymbol{T}}\boldsymbol{Y} &= \boldsymbol{R}\boldsymbol{b} \\ \boldsymbol{d} &= \boldsymbol{R}\boldsymbol{b}, \end{aligned}$$

where $\boldsymbol{R}\boldsymbol{b} = \boldsymbol{d}$ is an upper triangular system, which may be solved by back substitution. We want to add a new column in $\boldsymbol{X}$ and then we update $\boldsymbol{X}$ appropriately, without recomputing the factorization from scratch.

So, let $\boldsymbol{u} \in \mathbb{R}^n$ be a new variable in the $p+1$th column of $\boldsymbol{X}$. Thus,

$$\tilde{\boldsymbol{X}} = [\ \boldsymbol{X} \quad \boldsymbol{u}\ ],$$

then

$$\boldsymbol{Q}^{\boldsymbol{T}}\tilde{\boldsymbol{X}} = [\ \boldsymbol{R} \quad \boldsymbol{v}\ ], \qquad \textit{where} \qquad \boldsymbol{v} = \boldsymbol{Q}^{\boldsymbol{T}}\boldsymbol{u}.$$

We will give an example of the procedure in order to understand it easily. We assume that we have $n = 7$ and $p = 3$ and we add a new variable. Therefore,

$$[\ \boldsymbol{R} \quad \boldsymbol{v}\ ] = \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \\ 0 & 0 & 0 & * \\ 0 & 0 & 0 & * \\ 0 & 0 & 0 & * \end{bmatrix}.$$

https://doi.org/10.26219/heal.aueb.5567

As we can see, the three last elements of the forth column have to be zero, so the matrix $\boldsymbol{R}$ will be upper triangular. For this reason, we will define $(n-p-1) = 3$ Givens matrices.

Generalizing, we will update the matrix $\boldsymbol{R}$ and the vector $\boldsymbol{d}$. We define $(n-p-1)$ Givens matrices, $\boldsymbol{G(i,j)} \in \mathbb{R}^{(n\times n)}$, so that the elements of vector $\boldsymbol{v}$ from the $(p+1)$th to the nth position will be zero. Then,

$$\tilde{\boldsymbol{R}} = \tilde{\boldsymbol{Q}}^{\boldsymbol{T}}\tilde{\boldsymbol{X}} = (\boldsymbol{G(p+1,p+2)^T} \cdots \boldsymbol{G(n-1,n)^T Q^T})\tilde{\boldsymbol{X}}$$

and

$$\tilde{\boldsymbol{d}} = \tilde{\boldsymbol{Q}}^{\boldsymbol{T}}\tilde{\boldsymbol{Y}} = (\boldsymbol{G(p+1,p+2)^T} \cdots \boldsymbol{G(n-1,n)^T Q^T})\boldsymbol{Y},$$

where $\tilde{\boldsymbol{R}}$ is an $n\times(p+1)$ upper triangular matrix and
$\tilde{\boldsymbol{Q}} = \boldsymbol{Q}(\boldsymbol{G(n-1,n)}\cdots\boldsymbol{G(p+1,p+2)})$ is the updated $n\times n$ orthogonal matrix.

In multiple regression, we proved that the estimation of the coefficients is given by

$$\boldsymbol{\beta} = \boldsymbol{R_1}^{-1}\boldsymbol{d} \qquad where \qquad \boldsymbol{d} = \boldsymbol{Q_1^T Y}.$$

However, $\boldsymbol{R_1}$ is an $p\times p$ matrix and $\boldsymbol{Q_1}$ is the first p columns of $\boldsymbol{Q}$. Because of

$$\boldsymbol{X} = \boldsymbol{Q}\begin{bmatrix}\boldsymbol{R_1}\\ \boldsymbol{0}\end{bmatrix} = \begin{bmatrix}\boldsymbol{Q_1} & \boldsymbol{Q_2}\end{bmatrix}\begin{bmatrix}\boldsymbol{R_1}\\ \boldsymbol{0}\end{bmatrix} = \boldsymbol{Q_1R_1},$$

we follow the same updating procedure explained before, using solely the first p columns of $\boldsymbol{Q}$ and the square form of $\boldsymbol{R}$ (i.e. without the rows, which are zero), so that $\boldsymbol{R_1}$ can be invertible. With this technique, a new variable is added in the regression model, we need not calculate the QR decomposition of $\boldsymbol{X}$ from scratch in order to estimate the regression coefficients.

 https://doi.org/10.26219/heal.aueb.5567

3.6 Comparison of the methods

The aim of this thesis is to estimate the regression coefficients faster. Therefore, we will compare how time consuming is the formula, which we proved, relative to the other methods, i.e. the ordinary least squares estimation, the Gram Schmidt procedure and the QR decomposition. All of these methods and algorithms, which we described in this chapter, have been transformed into codes for the programming language R.

We have to report that the updating QR decomposition for regression will not be included in the comparison that follows. The reason is that the code for this method is very slow for big data relative to the others. This occurs because at the beginning of this approach we need the entire matrix $\boldsymbol{Q}$. Remember that

$$\boldsymbol{Q^T}\tilde{\boldsymbol{X}} = [\ \boldsymbol{R} \quad \boldsymbol{v}\ ], \qquad where \qquad \boldsymbol{v} = \boldsymbol{Q^T}\boldsymbol{u},$$

i.e. the new matrix $\boldsymbol{R}$ will be upper triangular, if some elements of the new variable are zero. In order to achieve this, we need the complete $n \times n$ matrix $\boldsymbol{Q}$. Thus, if n is very large, $\boldsymbol{Q}$ is huge. As a consequence, the algorithm is computationally slow. In addition, R language (version 3.2.4) cannot allocate such a matrix (for $n \geq 100000$).

Next, we will study the required time of every method to compare their efficiency. At first, we simulate data from normal distribution for some values of n and p and then we measure the time needed to execute every method. In addition, we repeat every procedure 100 times in order to avoid possible extreme time values. There is a table below, which includes the results with running times measured in seconds and some plots, which compare OLS with updating regression.

In the first and second columns of the table, the number of observations(n) and the number of the explanatory variables(p) can be seen respectively. In the third column, the ratio $\frac{n}{p}$ is listed. In the fourth column, we have the

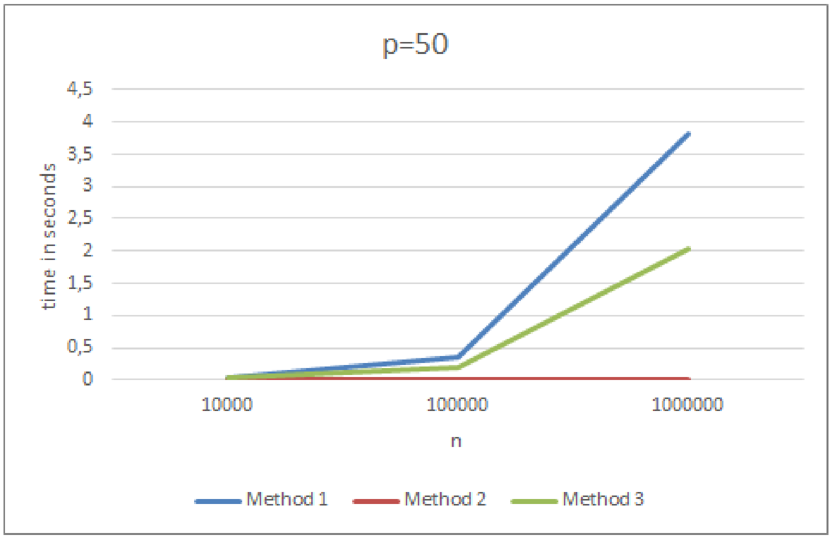
https://doi.org/10.26219/heal.aueb.5567

running times, which arise from the least squares estimation of regression coefficients, i.e. $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X})^{-1}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{Y}$. In the same way, the fifth column describes the times of the recursive formula (3.2) given the covariance matrix and the covariance between the response and the explanatory variables. The sixth column is similar to the fifth, except that the times include the calculation of the covariance matrix and the covariance between the response and the explanatory variables. Finally, we get the times for regression by orthogonalization and the regression with QR decomposition in the seventh and eighth columns respectively.

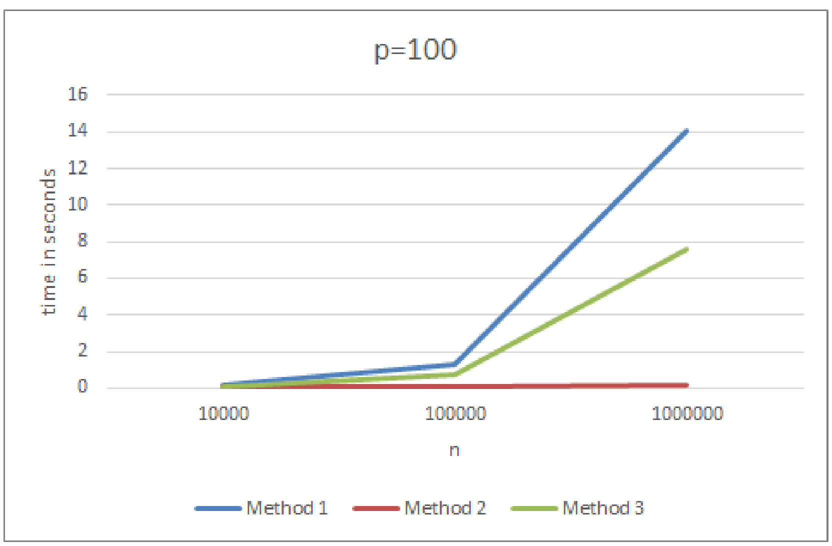
Table 3.1: Comparison of the methods

n	p	n/p	Method 1	Method 2	Method 3	Method 4	Method 5
10000	50	200	0.032	0.008	0.026	0.477	0.076
100000	50	2000	0.364	0.007	0.203	6.584	0.781
1000000	50	20000	3.828	0.015	2.042	75.678	10.025
10000	100	100	0.124	0.038	0.107	1.772	0.231
100000	100	1000	1.301	0.036	0.744	17.543	2.456
1000000	100	10000	14.035	0.122	7.554	317.532	44.825
10000	500	20	2.821	8.709	10.337	43.738	5.313
100000	500	200	27.512	9.024	24.755	667.963	55.552
10000	200	50	0.526	0.325	0.585	7.067	0.884
10000	1000	10	12.465	140.167	145.790	165.746	19.709

We compare the average running time of each method (measured in seconds).
Method 1: Ordinary Least Squares estimation
Method 2: The recursive formula that we proved
Method 3: Method 2 including the calculation of the covariances
Method 4: Regression by orthogonalization
Method 5: Regression using QR decomposition



(a) p=50



(b) p=100

Figure 3.1: Plots for methods comparison

https://doi.org/10.26219/heal.aueb.5567

From the above, the following results are derived:

1. Our updating regression estimates with huge speed the regression coefficients in comparison with others methods, especially when n is very big. This method performs the best when we already know the covariances (Method 2).

2. Even if we integrate the time to compute the covariance matrices into the procedure (Method 3), it is still faster than the ordinary estimation (Method 1).

3. Our updating regression is the fastest, as the ratio n/p increases. Conversely, when the ratio n/p is small (i.e. 10, 20), the classic formula $\hat{\beta} = (X^T X)^{-1} X^T Y$ takes the least time to run. As a consequence, our method does not work fast for small ratios n/p.

4. Method 4 and Method 5 are always time consuming in comparison with the other methods.

5. Despite the fact that regression with QR decomposition is slow, it is faster than our updating regression when the ratio n/p is small.

6. Regression by orthogonalization is the most time consuming. This occurs because it makes extra computations in order to estimate β_j for $j = r+1, r, \ldots, 0$, as we mentioned in previous section.

https://doi.org/10.26219/heal.aueb.5567

https://doi.org/10.26219/heal.aueb.5567

Chapter 4

Generalized Linear Models extension

4.1 Introduction of Generalized Linear Models

We have previously worked with regression models where the response variable is quantitative and normally distributed. In real world, there are many different occasions where the normal distribution is not appropriate, as in the case of yes/no responses (binary data). In addition, if the data's mean is restricted to a range of values or the assumption of constant variance is not realistic, linear models are not suitable. Therefore, the need for Generalized Linear Models (GLMs), which are extensions of traditional regression models, is evident. In GLMs, the response variable is assumed to follow any member of the exponential family distributions.

There are three components for any Generalized Linear Model (Sahu S. K.,2002), which have to be defined:

1. **Random Component**: It specifies the probability distribution of the response variable.

https://doi.org/10.26219/heal.aueb.5567

2. **Systematic Component**: It defines a model for the mean of the data. It is achieved through linear predictors.

3. **Link Function**: It determines the link between random and systematic components.

As we mentioned above, the probability distribution of the response variable $\boldsymbol{Y}$ belongs to exponential family distribution and takes the form

$$f_{\boldsymbol{Y}}(y;\theta,\phi) = exp\Big\{\frac{y\theta - b(\theta)}{\alpha(\phi)} + c(y,\phi)\Big\},$$

where $b(\theta)$, $\alpha(\phi)$ and $c(y,\phi)$ are known functions and θ and ϕ are parameters. We note that θ is the canonical parameter and ϕ is the dispersion parameter (when ϕ is unknown, we call this an exponential dispersion family). It can be shown that if $\boldsymbol{Y}$ has a distribution that belongs in the exponential family, then it has mean and variance

$$E(Y) = \mu = b'(\theta),$$

$$Var(Y) = \alpha(\phi)b''(\theta) = \alpha(\phi)Var(\mu).$$

A Linear predictor is denoted by $\boldsymbol{\eta}$ and is expressed as a linear combination of the parameters of the model. In other words, it is the quantity which incorporates the information about the independent variables into the model. The linear predictor can be expressed as

$$\boldsymbol{\eta} = \boldsymbol{X}\boldsymbol{\beta}.$$

The link between the distribution of $\boldsymbol{Y}$ and the linear predictor $\boldsymbol{\eta}$ is provided by the link function g, as in

$$\boldsymbol{\eta} = g(\boldsymbol{\mu}),$$

https://doi.org/10.26219/heal.aueb.5567

where $\boldsymbol{\mu} = E(\boldsymbol{Y})$. There are many commonly used link functions, such as identity, inverse, log, logit, probit etc.

It is important to mention some notation for GLMs. First of all, they assume linear relation between the transformed response in terms of the link function and the explanatory variables. The homogeneity of the variance is not necessarily satisfied. Moreover, it uses maximum likelihood estimation rather than ordinary least squares to estimate the parameters. The latter will be analyzed in a following section.

4.2 Sparsity

A sparse statistical model is a model that among the coefficients that describe it, only a small number of them are non-zero. Therefore, sparsity is a desirable property, because it uses fewer memory and storage space and it is much easier to estimate and interpret with it than it is with a dense model. In high dimensional data, where the number of features are very large, the model can be prone to overfitting. One appropriate method that can reduce overfitting is regularization(Liu J., Chen J. and Ye J., 2009).

The most famous types of regularization are the ℓ_1 and ℓ_2 norm regularization. The ℓ_2-norm regularization leads to a smooth (differentiable) unconstrained convex optimization problem. However, we will deal with ℓ_1-norm regularization, where the regularization term is non-differentiable. This is preferable, since some coefficients are shrunken exactly to zero. Thus, this yields sparse models that are more easily interpreted and it is also appropriate for variable selection.

In general, regularization tries to minimize a loss function with a penalty (Schmidt M. ,Fung G. and Rosales R.,2007). Specifically, the ℓ_1-norm regularization is the problem

$$\min_{\boldsymbol{x}} L(\boldsymbol{x}) + \lambda \|\boldsymbol{x}\|_1, \tag{4.1}$$

https://doi.org/10.26219/heal.aueb.5567

where $L(\boldsymbol{x})$ is a loss function, $\|\cdot\|_1$ is the ℓ_1-norm and λ is the regularization parameter.

Now, we will focus on a logistic regression model, where the dependent variable is binary. We have the data points $(\boldsymbol{x_i}, y_i)$ with $i = 1, \ldots, n$, where $y_i \in \{0, 1\}$ and $\boldsymbol{x_i} \in \mathbb{R}^p$ is the p-dimensional feature vector. In the logistic model, the probability distribution has the form

$$P(y = 1|\boldsymbol{x}; \boldsymbol{\beta}) = \frac{1}{1 + \exp\{-\boldsymbol{\beta^T x}\}},$$

where $\boldsymbol{\beta} \in \mathbb{R}^p$ is the parameter vector. The likelihood function of the logistic regression is $\prod_{i=1}^{n} P(y_i|\boldsymbol{x_i}; \boldsymbol{\beta})$. The empirical logistic loss is defined as the negative of the log-likelihood function and the average logistic loss as

$$\begin{aligned} f(\boldsymbol{\beta}) &= -\frac{1}{n} log \prod_{i=1}^{n} P(y_i|\boldsymbol{x_i}; \boldsymbol{\beta}) = -\frac{1}{n} \sum_{i=1}^{n} log P(y_i|\boldsymbol{x_i}; \boldsymbol{\beta}) \\ &= \frac{1}{n} \sum_{i=1}^{n} log(1 + \exp\{-\boldsymbol{\beta^T x}\}), \end{aligned}$$

which is a smooth and convex function. Generally, we minimize the average logistic loss ($\min\limits_{\boldsymbol{\beta}} f(\boldsymbol{\beta})$) in order to determine $\boldsymbol{\beta}$ leading to a smooth convex optimization problem. In high dimensional data, where the number of features is larger than the number of observations, logistic regression can lead to overfitting, rendering regularization a necessity. The ℓ_1-norm regularized logistic regression problem is given by

$$\min_{\boldsymbol{\beta}} f(\boldsymbol{\beta}) + \lambda \|\boldsymbol{\beta}\|_1. \tag{4.2}$$

Notice that the regularization term $\|\boldsymbol{\beta}\|_1$ is non-differentiable. The previous problem can be written in the equivalent form of an ℓ_1-ball constrained

https://doi.org/10.26219/heal.aueb.5567

optimization problem with a smooth objective function

$$\min_{\boldsymbol{\beta}} f(\boldsymbol{\beta}) \quad \textit{subject to} \quad \|\boldsymbol{\beta}\|_1 \leq c, \tag{4.3}$$

where c is positive and it can be chosen manually or via cross-validation.

According to Jun Liu, Jianhui Chen and Jieping Ye (2009), there are several strategies for solving ℓ_1-norm regularization problems that are also applicable to the ℓ_1-norm regularized logistic regression.

1. First strategy: It uses algorithms, based on subgradients. It treats our problem as a nonsmooth optimization problem. The most famous algorithms are Gauss-Seidel(Shevade and Keerthi, 2003), grafting(Perkins S., Lacker K., and Theiler J., 2003), shooting(Fu W., 1998), BBR(Eyheramendy S., Genkin A., Ju W., Lewis D., and Madigan D., 2003) etc.

2. Second strategy: It tries to approximate the problem (4.1) with a smooth function in order to solve it with smooth optimization methods, applying smooth approximation to the ℓ_1-norm. It is achieved by the smoothl1 method(Schmidt M. ,Fung G. and Rosales R., 2007), where the loss function is now $g(\boldsymbol{x}) = L(\boldsymbol{x}) + \lambda \sum_{i=1}^{n} \mid \boldsymbol{x} \mid_\alpha$, and $\mid \boldsymbol{x} \mid_\alpha$ converges to $\mid \boldsymbol{x} \mid$ as α approaches ∞. $\mid \boldsymbol{x} \mid_\alpha$ is twice differentiable.

3. Third strategy: It converts the problem (4.1) or (4.2) to a smooth optimization problem with smooth constraint functions, introducing additional variables. Some examples are l1-logreg(Koh K., Kim S. and Boyd S., 2007) and ProjectionL1(Schmidt M. ,Fung G. and Rosales R., 2007).

4. Fourth strategy: It converts a ℓ_1-norm regularized problem to the equivalent ℓ_1-ball constrained optimization problem with a smooth objective function, just as we converted the problem (4.2) to the problem (4.3). IRLS-LARS(Lee S., Lee H., Abbeel P. and Ng A. Y., 2006) and

https://doi.org/10.26219/heal.aueb.5567

Lassplore algorithm(Liu J., Chen J. and Ye J., 2009) belong to this strategy.

4.3 Optimal Subsampling for Big Datasets

In chapter 2, we mentioned methods for regression for Big Data and techniques for subdata selection. Similar methods exist for GLMs. Now, we will describe one of them, which is known as OSMAC (Optimal Subsampling Method under the A-optimality Criterion).

Let's assume that we have n data points and the response variable $y \in \{0, 1\}$. The logistic regression model is

$$P(y = 1|\boldsymbol{x}) = p(\boldsymbol{x}; \boldsymbol{\beta}) = \frac{e^{\boldsymbol{x^T\beta}}}{1 + e^{\boldsymbol{x^T\beta}}}$$

where $\boldsymbol{x} \in \mathbb{R}^d$ are the covariates and $\boldsymbol{\beta}$ is the $d \times 1$ parameters vector. The vector $\boldsymbol{\beta}$ can be estimated by maximum likelihood estimation (MLE). Since it is not easy to find the maximum likelihood estimator, we will try to approximate it.

The general subsampling approach for Big Data is to assign sampling probabilities π_i $(i = 1, \dots, n)$ for all data points, take a random subsample of size r according to π_i and then maximize the weighted log-likelihood to get an estimate $\tilde{\boldsymbol{\beta}}$, of $\hat{\boldsymbol{\beta}}_{\boldsymbol{MLE}}$. But, how can we determine π_i? There are several ways. The OSMAC method aim to derive the asymptotic distribution of $\tilde{\boldsymbol{\beta}}$, and then derive π_i's, which minimize the asymptotic MSE of $\tilde{\boldsymbol{\beta}}$. We describe this below, using some theorems (Wang, Zhu, and Ma, 2017).

Regarding the asymptotic behaviour of $\tilde{\boldsymbol{\beta}}$, there exists the following theorem

Theorem 2. *Under some regularity conditions, as $n \to \infty$ and $r \to \infty$,*

 https://doi.org/10.26219/heal.aueb.5567

conditional on $\mathcal{F}_n$ in probability,

$$\boldsymbol{V}^{-1/2}(\tilde{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_{\boldsymbol{MLE}}) \longrightarrow N(0, \boldsymbol{I})$$

in distribution, where

$$\boldsymbol{V} = \boldsymbol{M_x}^{-1}\boldsymbol{V_c}\boldsymbol{M_x}^{-1},$$

$$\boldsymbol{V_c} = \frac{1}{rn^2}\sum_{i=1}^{n}\frac{\{y_i - p_i(\hat{\boldsymbol{\beta}}_{\boldsymbol{MLE}})\}^2\boldsymbol{x_i}\boldsymbol{x_i^T}}{\pi_i},$$

$$\boldsymbol{M_x} = \frac{1}{n}\sum_{i=1}^{n} p_i(\hat{\boldsymbol{\beta}}_{\boldsymbol{MLE}})\{1 - p_i(\hat{\boldsymbol{\beta}}_{\boldsymbol{MLE}})\}\boldsymbol{x_i}\boldsymbol{x_i^T}.$$

Thus, we derived the asymptotic distribution of $\tilde{\boldsymbol{\beta}}$ and now we have to minimize the asymptotic MSE (AMSE) of $\tilde{\boldsymbol{\beta}}$. In order to achieve this, we will use the A-optimality criterion. According to this criterion, we minimize the trace of $\boldsymbol{V}$ instead of the AMSE of $\tilde{\boldsymbol{\beta}}$. This occurs since

$$(\tilde{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_{\boldsymbol{MLE}})|\mathcal{F}_n \overset{a}{\sim} N(0, \boldsymbol{V}).$$

In addition,

$$\boldsymbol{M_x}(\tilde{\boldsymbol{\beta}} - \hat{\boldsymbol{\beta}}_{\boldsymbol{MLE}})|\mathcal{F}_n \overset{a}{\sim} N(0, \boldsymbol{V_c}).$$

Therefore, the AMSE of $\boldsymbol{M_x}\tilde{\boldsymbol{\beta}}$ is the trace of $\boldsymbol{V_c}$. Thus, we can minimize either the trace of $\boldsymbol{V}$, or the trace of $\boldsymbol{V_c}$.

The two following theorems calculate the subsampling probabilities π_i's, which are essential in order to select the subsample and estimate the coefficients of the logistic regression model.

Theorem 3. *If the subsampling probability is chosen such that*

$$\pi_i^{mMSE} = \frac{|\, y_i - p_i(\hat{\boldsymbol{\beta}}_{\boldsymbol{MLE}}) \,|\, \|\boldsymbol{M_x}^{-1}\boldsymbol{x_i}\|}{\sum_{j=1}^{n} |\, y_j - p_j(\hat{\boldsymbol{\beta}}_{\boldsymbol{MLE}}) \,|\, \|\boldsymbol{M_x}^{-1}\boldsymbol{x_j}\|} \qquad i = 1, \ldots, n,$$

then the asymptotic MSE of $\tilde{\boldsymbol{\beta}}$, tr($\boldsymbol{V}$), attains its minimum.

https://doi.org/10.26219/heal.aueb.5567

Theorem 4. *If the subsampling probabilities are chosen such that*

$$\pi^{mVc} = \frac{\mid y_i - p(\boldsymbol{x_i}; \boldsymbol{\hat{\beta}_{MLE}}) \mid \|\boldsymbol{x_i}\|}{\sum_{j=1}^{n} \mid y_j - p(\boldsymbol{x_j}; \boldsymbol{\hat{\beta}_{MLE}}) \mid \|\boldsymbol{x_j}\|} \qquad i = 1, \ldots, n,$$

then the asymptotic MSE of $\boldsymbol{M_x}\boldsymbol{\tilde{\beta}}$, tr($\mathbf{V_c}$), attains its minimum.

The subsampling probabilities π^{mVc} can be calculated faster, because it is not required the calculation of $\|\boldsymbol{M_x}^{-1}\boldsymbol{x_i}\|$ for $i = 1, \ldots, n$.

Nevertheless, the subsampling probabilities depend on the $\boldsymbol{\hat{\beta}_{MLE}}$. So, the OSMAC cannot apply directly. Therefore, there is the following two step algorithm, which approximate the OSMAC.

1. Take a subsample of size r_0, using e.g. uniform subsampling, in order to obtain the estimator $\boldsymbol{\tilde{\beta}_0}$. Replace $\boldsymbol{\hat{\beta}_{MLE}}$ with $\boldsymbol{\tilde{\beta}_0}$ in order to take an approximately optimal subsampling probabilities π_i's.

2. Take a subsample of size r using replacement according to the previous optimal subsampling probabilities and estimate the coefficients for the total subsample of size $r_0 + r$.

4.4 Iteratively Reweighted Least Squares algorithm

In Generalized Linear Models, the estimation of the parameters is accomplished using Maximum Likelihood Estimation (MLE). In practice, the traditional way of MLE has no analytic solution. Therefore, we rely on numerical methods to solve this, from which the most famous is the Iteratively Reweighted Least Squares algorithm (IRLS). In this section, we will study this algorithm in detail(Léo, 2013).

Let g be the link function and $\boldsymbol{Y}$ be the response variable. We define the

https://doi.org/10.26219/heal.aueb.5567

random variable $g(\boldsymbol{Y})$. Using the Taylor approximation , we obtain

$$g(\boldsymbol{Y}) \simeq g(\boldsymbol{\mu}) + (\boldsymbol{Y} - \boldsymbol{\mu})g'(\boldsymbol{\mu}).$$

Then, let

$$\boldsymbol{Z} = \boldsymbol{\eta} + (\boldsymbol{Y} - \boldsymbol{\mu})g'(\boldsymbol{\mu}) = \boldsymbol{X\beta} + (\boldsymbol{Y} - \boldsymbol{\mu})g'(\boldsymbol{\mu}),$$

with

$$E(\boldsymbol{Z}) = \boldsymbol{\eta} = \boldsymbol{X\beta},$$

$$Var(\boldsymbol{Z}) = Var(\boldsymbol{Y})(g'(\boldsymbol{\mu}))^2 = \frac{\phi Var(\boldsymbol{\mu})(g'(\boldsymbol{\mu}))^2}{w}.$$

Hence, we have the model

$$\boldsymbol{Z} = \boldsymbol{X\beta} + \boldsymbol{\epsilon},$$

where $Var(\boldsymbol{\epsilon}) = Var(\boldsymbol{Z}) = \boldsymbol{u}$. Finally, we obtain an improved estimate of $\boldsymbol{\beta}$ regressing the dependent variable $\boldsymbol{Z}$ on the explanatory variables $\boldsymbol{X_1}, \ldots, \boldsymbol{X_p}$ using weights , i.e. we calculate the weighted least squares estimate

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X^T W X})^{-1} \boldsymbol{X^T W Z},$$

where $\boldsymbol{X}$ is the model matrix, $\boldsymbol{Z}$ is the response vector, which is assumed to be known and $\boldsymbol{W}$ is the following diagonal matrix

$$\boldsymbol{W} = diag(\frac{1}{u_1}, \ldots, \frac{1}{u_n}).$$

Thus, an iterative procedure can be created, known as Iteratively Reweighted Least Squares. We describe this algorithm below (Kane M. and Lewis B. W., 2014).

https://doi.org/10.26219/heal.aueb.5567

Algorithm 4: Iteratively Reweighted Least Squares

Input: $\boldsymbol{Y}$, matrix $\boldsymbol{X}$, $rmax$, tol
Initialize: $\beta^{(0)} = 0$
for $r = 1, \ldots, rmax$ **do**
  Compute
  $\boldsymbol{\eta^{(r)}} = \boldsymbol{X\beta^{(r-1)}}$
  $\boldsymbol{\mu^{(r)}} = g^{-1}(\boldsymbol{\eta^{(r)}})$
  $\boldsymbol{Z^{(r)}} = \boldsymbol{\eta^{(r)}} + (\boldsymbol{Y} - \boldsymbol{\mu^{(r)}})g'(\boldsymbol{\mu^{(r)}})$
  $\boldsymbol{W^{(r)}} = diag(v_1, \ldots, v_n) \quad where \quad v_i = \dfrac{w_i}{\phi Var(\boldsymbol{\mu^{(r)}})(g'(\boldsymbol{\mu^{(r)}}))^2}$
  $\boldsymbol{\beta^{(r)}} = (\boldsymbol{X^T W^{(r)} X})^{-1} \boldsymbol{X^T W^{(r)} Z^{(r)}}$
  Stop if $\|\boldsymbol{\beta^{(r)}} - \boldsymbol{\beta^{(r-1)}}\| < tol$
end

In Algorithm 4, we repeat the procedure until $\|\boldsymbol{\beta^{(r)}} - \boldsymbol{\beta^{(r-1)}}\|$ is sufficiently small. So, tol is a termination tolerance. In addition, we give a specific maximum number of iterations $rmax$, since we do not know if the algorithm converges.

4.5 Recursive formula for GLMs

In previous chapter, we proved a recursive relation in order to estimate the coefficients of a linear regression model faster. Because of the importance of big data, it is essential to improve the estimation of the parameters of a generalized linear model in order not to be so time consuming. For this reason, we have to find a recursive formula for

$$\hat{\boldsymbol{\beta}} = (\boldsymbol{X^T W X})^{-1} \boldsymbol{X^T W Y},$$

and then insert this in the IRLS algorithm.

Using the same procedure as in the linear regression case, we can prove that

$$(\boldsymbol{X^TWX})_{r+1} = \left[\begin{array}{c|c} (\boldsymbol{X^TWX})_r & \boldsymbol{U} \\ \hline \boldsymbol{U^T} & z \end{array}\right] \qquad and \qquad (\boldsymbol{X^TWY})_{r+1} = \left[\begin{array}{c} (\boldsymbol{X^TWY})_r \\ \hline V \end{array}\right],$$

where

$$\boldsymbol{U} = \begin{bmatrix} \sum_{i=1}^n (w_i X_{r+1,i}) \\ \sum_{i=1}^n (w_i X_{1,i} X_{r+1,i}) \\ \vdots \\ \sum_{i=1}^n (w_i X_{r,i} X_{r+1,i}) \end{bmatrix} = \boldsymbol{X^TWX_{r+1}}$$

$$z = \sum_{i=1}^n (w_i X_{r+1,i}^2) = \boldsymbol{X_{r+1}^T W X_{r+1}}$$

$$V = \sum_{i=1}^n (w_i X_{r+1,i} Y_i) = \boldsymbol{X_{r+1}^T W Y}.$$

Thus, applying Theorem 1 we obtain

$$\hat{\mathbf{b}}_{\mathbf{r+1}} = \begin{bmatrix} \hat{\mathbf{b}_{\mathbf{r}}} + (\boldsymbol{X^TWX})_r^{-1} \boldsymbol{U} (\frac{1}{k} \boldsymbol{U^T} \hat{\mathbf{b}_{\mathbf{r}}} - \frac{1}{k} V) \\ -(\frac{1}{k} \boldsymbol{U^T} \hat{\mathbf{b}_{\mathbf{r}}} - \frac{1}{k} V) \end{bmatrix}, \tag{4.4}$$

where $k = z - \boldsymbol{U^T} (\boldsymbol{X^TWX})_r^{-1} \boldsymbol{U}$.
Set $\boldsymbol{P} = \boldsymbol{X}(\boldsymbol{X^TWX})^{-1}\boldsymbol{X^TW}$, $\boldsymbol{H} = \boldsymbol{I_n} - \boldsymbol{P}$, $\boldsymbol{l} = (\boldsymbol{X^TWX})^{-1}\boldsymbol{X^TWX_{r+1}}$, then

$$\hat{\mathbf{b}}_{\mathbf{r+1}} = \begin{bmatrix} \hat{\mathbf{b}_{\mathbf{r}}} - \dfrac{1}{k} \boldsymbol{l} (\boldsymbol{WX_{r+1}})^T \boldsymbol{HY} \\ \dfrac{1}{k} (\boldsymbol{WX_{r+1}})^T \boldsymbol{HY} \end{bmatrix}, \tag{4.5}$$

where $k = \boldsymbol{X_{r+1}^T W H X_{r+1}}$.

Just like in the linear regression case, this recursive formula is more efficient than performing regression from the beginning in each step, since the latter requires the inversion of a $r+2 \times r+2$ matrix $(\boldsymbol{X^TWX})_{r+1}$ in the $r+1$th step. In our method, the inverse of $(\boldsymbol{X^TWX})_r$, which is a $r+1 \times r+1$ matrix, is already available, and the only inversion required is

https://doi.org/10.26219/heal.aueb.5567

that of the number k.

4.6 Updating Generalized Linear Models

Our aim is to combine the IRLS algorithm with the previous recursive formula. The question is how will this formula be integrated to the algorithm. We will describe a method below and we will justify our choice.

Let us have r explanatory variables, namely $\boldsymbol{X^{(r)}} = (\boldsymbol{X_1}, \ldots, \boldsymbol{X_r})$. Run the IRLS algorithm with e.g 10 iterations. Then, add a new variable $\boldsymbol{X_{r+1}}$, to obtain $\boldsymbol{X^{(r+1)}} = (\boldsymbol{X_1}, \ldots, \boldsymbol{X_{r+1}})$. Now, update $\boldsymbol{\beta}$ from the recursive formula (4.4). In this formula, we have to use the matrix $\boldsymbol{W}$ from the last step of the IRLS algorithm. Finally, calculate again the coefficients $\boldsymbol{\beta}$ from the IRLS using the matrix $\boldsymbol{X^{(r+1)}}$.

Algorithm 5: Updating Generalized Linear Models

```
Input: Y, matrix X = (X_1, ..., X_p), p
Initialize: β^(0) = 0
for r = 1, ..., p do
    Let X^(r) = (X_1, ..., X_r)
    Run IRLS and estimate β with 10 iterations (fixed X^(r))
    Add a new variable: X^(r+1) = (X_1, ..., X_{r+1})
    Update β from our recursive formula (fixed W)
end
```

This is an iterative procedure for all covariates. The number of iterations for the IRLS is an example. It depends on how many iterations the algorithm approximately needs in order to converge. The weight matrix $\boldsymbol{W}$ from the last step of the IRLS is useful. Starting from the optimal matrix $\boldsymbol{W}$ of the step r will lead to fewer iterations in the step r+1. This means faster convergence.

Now, we will give an example, so that the previous procedure will be easy to understand. Telecommunications companies want to identify customers

https://doi.org/10.26219/heal.aueb.5567

that have a tendency to unsubscribe in order to take measures to retain such customers. Thus, the dataset contains 5000 observations(subscribers), of which we have 707 (14.14%) customers who churned, and our goal is to predict if they have a high probability of unsubscribing now or in the near future. Therefore, the variable churn is a binary dependent variable, which takes the values $\{0, 1\}$. In addition, we will use four of the explanatory variables for our example ($\boldsymbol{X_1}$: number of voicemail messages, $\boldsymbol{X_2}$: number of total minutes during the day, $\boldsymbol{X_3}$: number of total minutes in the evening, $\boldsymbol{X_4}$: number of total minutes during the night). In the table below, some summary statistics of the data are presented.

Table 4.1: Summary statistics of the data

variables	mean	sd	median	min	max
number of voicemail messages	7.76	13.55	0	0	52.0
number of total minutes during the day	180.29	53.89	180.1	0	351.5
number of total minutes in the evening	200.64	50.55	201.0	0	363.7
number of total minutes during the night	200.39	50.53	200.4	0	395.0

Subsequently, we will standardize our variables. Hence, in the following diagram, we can see the steps of the procedure, which estimates the coefficients of the logistic model

$$logit(p_i) = \beta_0 + \beta_1 X_{1,i} + \beta_2 X_{2,i} + \beta_3 X_{3,i} + \beta_4 X_{4,i},$$

$$p_i = P(Y_i = 1 | \boldsymbol{X_1}, \boldsymbol{X_2}, \boldsymbol{X_3}, \boldsymbol{X_4}), \qquad i = 1, \ldots, 5000.$$

 https://doi.org/10.26219/heal.aueb.5567

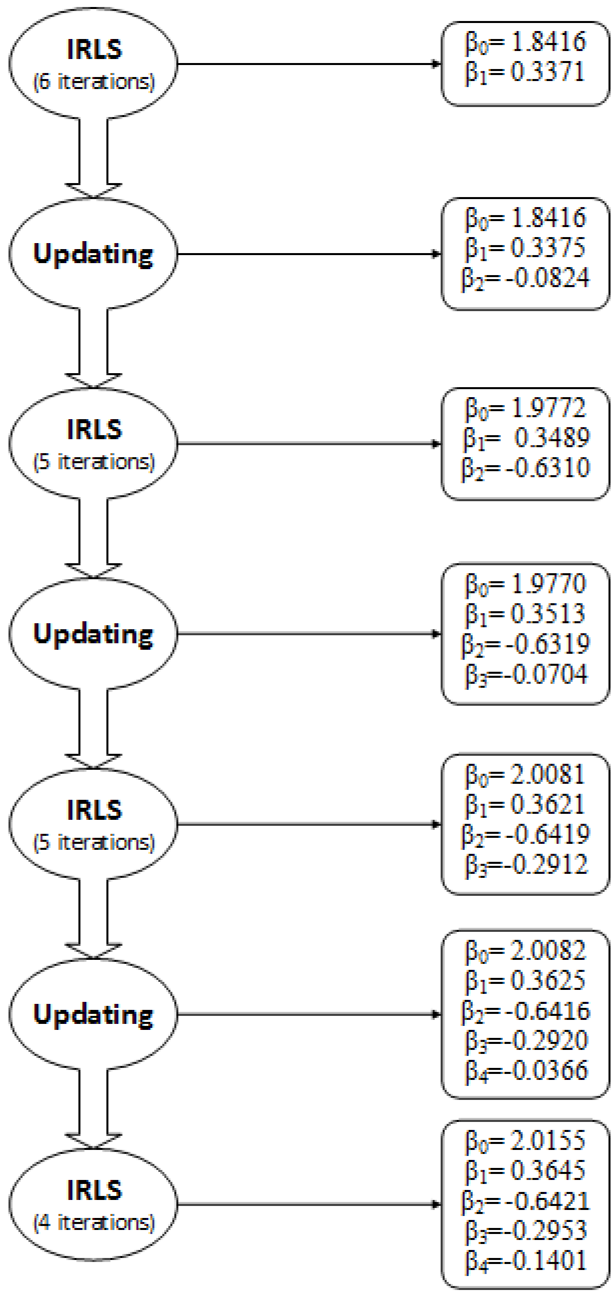


Figure 4.1: Procedure of Updating GLM
In IRLS: rmax=10 and tol=10^{-8}

https://doi.org/10.26219/heal.aueb.5567

It is worth noticing that in our example the estimation of the coefficients from the recursive formula is close to the estimation given by the IRLS algorithm. Moreover, the IRLS algorithms did not require 10 iterations in order to converge. Specifically, they need 6,5,5 and 4 iterations respectively. Therefore, it is very interesting to investigate the final results of our method for several maximum number of iterations. The next figure presents the results for maximum number of iterations 10,3,2 and 1 (namely, rmax=10,3,2,1).

(a) rmax=10	(b) rmax=3	(c) rmax=2	(d) rmax=1
β_0= 2.0155262	β_0= 2.0155262	β_0= 2.0155197	β_0= 2.0098196
β_1= 0.3644900	β_1= 0.3644900	β_1= 0.3644881	β_1= 0.3632885
β_2= -0.6421414	β_2= -0.6421414	β_2= -0.6421385	β_2= -0.6405660
β_3=-0.2952714	β_3=-0.2952714	β_3=-0.2952699	β_3=-0.2939653
β_4=-0.1400982	β_4=-0.1400982	β_4=-0.1400973	β_4=-0.1373266

Figure 4.2: Estimation coefficients for several maximum number of iterations of IRLS

As we can see, if the IRLS algorithms make at least 3 iterations, the estimation of coefficients is exactly the same. In other words, every time we added a new variable, we needed 6,5,5 and 4 iterations respectively and we obtain the same results with maximum three iterations in each iterative procedure[1]. Furthermore, the estimation of coefficients for two iterations of the IRLS algorithm has a minimal difference of order 10^{-5}. In the case of a sole iteration, there is sensible difference, which can be essential depending on the accuracy of estimation that we want to achieve. Consequently, the choice of rmax has to be appropriate so that the procedure will be fast and the estimation of coefficients will be accurate.

Generally, regarding our method, it is necessary to highlight that our R code requires to standardize our variables, otherwise the estimation of coefficients is unstable. Apart from this, we have realized that our method

[1]Of course, minor differences exist, but we have defined tolerance=10^{-8}

is time-consuming. This is a consequence of the inclusion of the weighted least squares estimate $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{Z}$ to the IRLS. Thus, the inverse of $\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{X}$ is calculated many times, until the algorithm converges. It practically means that we do not avoid the inversion of $\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{W}\boldsymbol{X}$! Although we can choose a small value for maximum number of iterations in IRLS in order to improve computationally the method, it does not suffice to speed up the procedure enough. In addition, we calculate the $\boldsymbol{\beta}$ coefficients in every step, i.e. every time we add a new variable, slowing down the procedure. On the other hand, the latter is not necessarily a drawback, since we provide the estimation of coefficients for all the intermediate models.

4.7 Conclusion of chapter 4

In this chapter, we expanded our approach to the Generalized Linear Model case. At the beginning, we studied a general term, sparsity, which is very important for big datasets. Thus, we reported the ℓ_1-norm regularization, which lead to sparse models, and strategies for solving ℓ_1-norm regularization problems. Furthermore, we referred to an optimal subsampling method under the A-optimality criterion for the logistic regression model.

The aim of this chapter was to find a similar algorithm like the one used in linear regression in order to speed up the estimation of the coefficients of a GLM. We also studied the IRLS algorithm and we attempted to combine it with the recursive formula that we mentioned earlier.

https://doi.org/10.26219/heal.aueb.5567

Chapter 5

Conclusions-Further Research

5.1 Conclusions

In the current thesis we studied regression for big datasets. We accomplished to construct an algorithm for the estimation of regression coefficients using a recursive formula, which is faster than the Ordinary Least Squares estimation. This occurs because we do not need to invert the matrix $\boldsymbol{X^T X}$ from scratch. In addition, if our variables are standardized, it is sufficient to know the covariance matrix of the explanatory variables and the covariance between the response and the explanatory variables. The full dataset is not required.

Apart from these, we studied two additional methods for estimating coefficients, which are the regression by orthogonalization and the factorisation of matrix $\boldsymbol{X}$ using QR decomposition. Comparing all the methods and via various examples we noted that if the ratio n/p has large value, the method with the recursive formula is the fastest computationally. Contrariwise, the regression by orthogonalization is the most time-consuming due to back substitution.

In the end, we expanded on generalized linear models. Due to data's volume, we attempted to combine the IRLS algorithm, which estimates the

https://doi.org/10.26219/heal.aueb.5567

coefficients of the GLM, with a similar formula like the one used in linear regression in order to speed up the procedure. Unfortunately, we cannot accomplish it since we do not avoid the inversion of $\boldsymbol{X^T W X}$ inside the IRLS algorithm. However, this procedure gives us the estimation of coefficients for all the intermediate models, since we were estimating a model and then we were adding a new variable.

5.2 Discussion-Further Research

One of the first aspects of Big Data that comes to mind is computation time. Thus, an important part of this thesis was the comparison methods. To derive results and notes, we relied on the running time of each method. Generally, a programmer will have to consider three basic factors, when comparing methods for regression calculations. The first one is the memory that is required. The second one is the efficiency, which includes the time it takes to execute an algorithm. This is measured by counting the number of flops. The last one is the accuracy, namely how close are the computed regression quantities to the true quantities. Hence, it would be preferable to examine all these criteria.

Two other issues that could be further explored are the case of QR decomposition when we add a new variable and the combination of IRLS algorithm with the updating technique. Regarding the first, and given that the theoretical idea of our method is interesting, it is worth mentioning that the updating code of the QR factorization could potentially be refined. Further hardware or software constraints (RAM size / the programming language) and the implications of them to our method could also be researched. Moreover, a possible review and optimization of the updating GLMs method could be useful. Specifically, it would be ideal to find a formula for the inversion of the matrix $\boldsymbol{X^T W X}$. In addition, in this dissertation, we primarily handle logistic regression with logit link function. It is a good idea to examine

https://doi.org/10.26219/heal.aueb.5567

several generalised linear models with various link functions.

Big Data is characterized by massive sample size and high-dimensionality. We predominantly studied the case of massive sample size. However, it would be interesting to study the other case, where the number of features is large and bigger than the sample size. Examples of this are plentiful in biology, climatology, geology, health science, economics etc. Dimension reduction and variable selection play pivotal roles in analyzing such high dimensional problems. The ℓ_1-norm penalized least squares problem was mentioned in the dissertation, because it can solve the problem of high dimensionality. Apart from that, there are many techniques which can handle the big number of features, such as several forms of regularization, PCA and others and it is worth being examined.

https://doi.org/10.26219/heal.aueb.5567

References

Bates D. (2010). *Matrix decompositions for regression analysis.* Lecture notes, STAT 849, University of Wisconsin-Madison, 2010-09-07 Tue

Belloni A., Chernozhukov V. and Wang L.(2011). Square-root lasso: pivotal recovery of sparse signals via conic programming, *Biometrika*, 98(4):791–806.

Candes E. and Tao T. (2007). The dantzig selector: Statistical estimation when p is much larger than n, *The Annals of Statistics*, 35(6):2313–2351.

Cheung M. W.-L. and Jak S. (2016). Analyzing Big Data in Psychology: A Split/Analyze/Meta-Analyze Approach, *journal Frontiers in Psychology*, 7: 738..

Cukier K.(2010). Data, data everywhere: A special report on managing information, *The Economist.*

Eyheramendy S., Genkin A., Ju W., Lewis D., and Madigan D. (2003). Sparse bayesian classifiers for text categorization, *Technical report, Department of Statistics, Rutgers University.*

Foster D. P. and Stine R. A. (2007). Alpha-investing: a procedure for sequential control of expected false discoveries, *Journal of the Royal Statistical Society Series B (Statistical Methodology)*, 70(2):429-444.

Fu W. (1998). Penalized regressions: the bridge versus the lasso, *Journal of Computational and Graphical Statistics*, 7:397–416.

https://doi.org/10.26219/heal.aueb.5567

Gandomi A. , Haider M.(2015). Beyond the hype: Big data concepts, methods, and analytics, *International Journal of Information Management*, 35, 137–144.

Haidich A. B. (2010). Meta-analysis in medical research, *Hippokratia*, 14(Suppl 1): 29–37.

Hammarling S. and Lucas C. (2008). Updating the QR Factorization and the Least Squares Problem, *MIMS EPrints.*

Hastie T., Tibshirani R. and Friedman J. (2008). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Second Edition, Springer Series in Statistics

Jun S., Lee S.-J. and Ryu J.-B. (2015). A Divided Regression Analysis for Big Data, *International Journal of Software Engineering and Its Applications*, Vol. 9, No. 5 (2015), pp. 21-32.

Kane M. and Lewis B. W. (2014). Generalized linear models, abridged

Koh K., Kim S., and Boyd S. (2007). An interior-point method for large-scale ℓ_1-regularized logistic regression, *Journal of Machine Learning Research*, 8:1519–1555.

Kosmidis I. (2015). *Regression modelling for massive data sets.* [PowerPoint slides]

Lambers J. (2010). *The QR Factorization.* Lecture notes, MAT 610, University of Southern Mississippi, Summer Session 2009-10, Lecture 9

Lee S., Lee H., Abbeel P. and Ng A. Y.(2006). Efficient ℓ_1 regularized logistic regression, *In The Twenty-first National Conference on Artificial Intelligence.*

Léo R.-B. (2013). *Generalized Linear Models.* Course notes, MA 523

Li X. (2013). ℓ_1 *Regularized High-Dimensional Sparse Regression.* (Master's thesis, University of Minnesota Duluth)

https://doi.org/10.26219/heal.aueb.5567

Lin D., Foster D. P. and Ungar L. H.(2011). VIF Regression: A Fast Regression Algorithm for Large Data, *Journal of the American Statistical Association*, 106(493):232-247 .

Liu J., Chen J. and Ye J. (2009). Large-scale sparse logistic regression, *ACM SIGKDD.*

Lu T.-T. and Shiou S.-H., (2002). Inverses of 2×2 Block Matrices , *Computers and Mathematics with Applications*, 43, 119-129.

Nieminen P., Lehtiniemi H., Vähäkangas K., Huusko A. and Rautio A. (2013) . Standardised regression coefficient as an effect size index in summarising findings in epidemiological studies, *Epidemiology Biostatistics and Public Health*, Volume 10, Number 4.

Perkins S., Lacker K., and Theiler J.(2003). Grafting: fast, incremental feature selection by gradient descent in function space, *Journal of Machine Learning Research*, 3:1333–1356.

Reitermanova Z. (2010). Data Splitting, *WDS'10 Proceedings of Contributed Papers*, Part I, 31–36.

Sahu S. K. (2002). *Statistical Methods 2.* Lecture notes, MA 325 , University of Southampton Highfield, Academic year 2002-2003

Schmidt M. ,Fung G. and Rosales R. (2007). Fast Optimization Methods for L1 Regularization: A Comparative Study and Two New Approaches, *The Eighteenth European Conference on Machine Learning*, 286–297.

Seber G. A. F. (2007). *A Matrix Handbook for Statisticians.* Wiley Series in Probability and Statisticspp. 309

Seber G. A. F. and Lee A. J.(2003). *Linear Regression Analysis 2nd Edition.* Wiley Series in Probability and Statistics, Chapters 3,11

Shevade S. K. and Keerthi S. S. (2003). A simple and efficient algorithm for gene selection using sparse logistic regression, *Bioinformatics*, 19(17):2246–2253.

https://doi.org/10.26219/heal.aueb.5567

Snee R. D. (1977). Validation of Regression Models: Methods and Examples, *Technometrics*, Vol. 19, No. 4. (Nov., 1977), pp. 415-428.

TechAmerica Foundation's Federal Big Data Commission. (2012). Demystifying big data: A practical guide to transforming the business of Government, pp. 10.

Tibshirani R. (2013). *Regression 1: Different perspectives.* [PowerPoint slides]

Wang H. (2016). *Informative Subdata Selection for Big Data.* [PowerPoint slides]

Wang H., Zhu R. and Ma P. (2017). Optimal Subsampling for Large Sample Logistic Regression, February 7, 2017.

Wang L. (2013). The L1 penalized LAD estimator for high dimensional linear regression, *Journal of Multivariate Analysis*, 120:135–151.

Zhang H. H.(2016). *Regression Methods I (Linear Regression).* [PowerPoint slides]

Zhou J., Foster D. P., Stine R. A. and Ungar L. H. (2006). Streamwise Feature Selection, *Journal of Machine Learning Research*, 7 (2006) 1861-1885.