Robust regression with high-dimensional data

Bachelor Thesis Econometrie en Operationele Research

Erasmus University Rotterdam
Erasmus School of Economics

Luuk van Maasakkers (414156)
Supervisor: dr. A. Alfons
Second assessor: dr. M. Zhelonkin
July 2, 2017

Abstract

In many statistical methods, complications arise when the number of dimensions d in a data set is relatively large. Due to overfitting and multicollinearity, linear regression estimates often suffer from numerical instability when the amount of predictors is large. In this research, I compare regularized regression methods that are developed to alleviate the consequences of multicollinearity and overfitting, such as ridge regression and lasso. I combine these methods with an outlier detection algorithm, developed by Rousseeuw & Van den Bossche (2016), that is capable of finding outlying cells and rows in high-dimensional data sets, taking correlations between variables into account. With this combination, high-dimensional regression estimates are found that are robust to both rowwise and cellwise outliers.

Contents

1 Introduction

A lot of data sets contain outliers. It is important to detect these outliers, because they may be undesirable errors or valuable pieces of information. The term outlier typically refers to an observation (row) in the data set. Since the 1960's, a lot of research has been done in the field of robust statistics to develop fitting methods that are less sensitive to outlying rows in the data set. However, recent research has shown that this *rowwise outlier paradigm* is no longer sufficient for modern, high-dimensional data sets. In many cases, only a few cells in a row are outlying, whereas the remaining cells are regular. Eliminating the influence of rows with a relatively small amount of outlying cells results in a waste of data. This indicates that a *cellwise outlier paradigm* would be more appropriate, which requires new methods to detect outlying data cells. In the first part of this research, I re-implement such a method, called *DetectDeviatingCells* (Rousseeuw & Van den Bossche, 2016), and apply it to four different data sets. The objective for this part is to replicate the results as presented by Rousseeuw & Van den Bossche (2016).

In the second part of the research, I focus on robust regression with high-dimensional data sets. When a dependent variable $\mathbf{y}$ is regressed on the explanatory variables (columns) in $\mathbf{X}$, standard linear regression would yield the estimated coefficient vector

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

When the number of variables in $\mathbf{X}$ is high compared to the number of observations, two major problems arise in standard linear regression:

1. The linear regression fit often has low bias but high variance. This problem is called *overfitting*: due to the excess of predictors, the regression fit tends to describe the random noise in the observed data instead of the underlying relation between variables. This results in numerical instability of the estimates: a small change in the data set can have a relatively large influence on the estimated coefficients. Another cause of numerical instability in is *multicollinearity*, which occurs when predictors are highly correlated with each other. Overfitting and multicollinearity harm the predictive performance of the regression estimates. In some cases, prediction accuracy can be improved by sacrificing a small amount of bias in order to decrease the variance.

2. The linear regression "freely" assigns coefficients to the predicting variables. However, in most applications with high-dimensional data sets, it is highly unlikely that all available predictors really influence the dependent variable. For the sake of interpretation, we sometimes want to find a smaller set of important variables. Linear regression is unable to make such a selection.

Note that when the number of dimensions d exceeds the number of observations n, the linear regression estimate in equation 1 is not even well-defined, because $\mathbf{X}'\mathbf{X}$ is not invertible. The problems mentioned above can (partially) be solved by using regularized regression methods such as *ridge regression*, *lasso* and *elastic net regression*. These methods are also called penalized regression methods, as they penalize the size of the coefficients in the objective function they minimize. Ridge regression penalizes the ℓ_2-norm (sum of squared coefficients), lasso penalizes the ℓ_1-norm (sum of absolute coefficients) and elastic net penalizes a convex combination of the ℓ_1- and ℓ_2-norm. As a result of the penalization, the regression estimates are shrunk towards zero. This decreases the variance of the estimates and thus results in more stable, 'regular' estimates. This solves problem (1). Lasso and elastic net also (partially) solve problem (2), as they usually yield sparse solutions. That is, the estimated coefficient vector contains zero values. In this research, I compare the performance of ridge regression, lasso and elastic net. Besides that, I also study the effect of applying the DDC algorithm on the quality of the estimates. I aim to find answers to the following questions:

- How does applying the DDC algorithm on the set of dependent and independent variables influence the performance of regularized regression estimates, in terms of prediction accuracy and distance to the real coefficient vector?
- Which regularized regression method performs the best, in terms of prediction accuracy and distance to the real coefficient vector, when the DDC algorithm is first applied on the set of dependent and independent variables?

In the next section, I will discuss previous studies about cellwise outliers and regularized regression. These two can be considered as different subjects, which I combine in this study. Afterwards, I will discuss the DDC algorithm in more detail and compare my results with the results of Rousseeuw & Van den Bossche (2016). Finally, I will compare different regularized regression methods by applying them on simulated data sets.

2 Literature review

2.1 Outlying cells

In the early 1960's, Tukey (1962) and Huber et al. (1964) introduced a contamination model which assumed that a small fraction ϵ of cases in a data set is affected by abnormal noise. In the past decades, this *Tukey-Huber contamination model* (THCM) has formed the foundation of most robust statistical procedures: identifying outlying *cases* and downweighting their influence. However, the model has been criticized because it assumes that the majority of the cases is free of contamination. Another criticism is that downweighting an entire case may be inconvenient when the number of cases is low compared to the number of variables. Consequently, alternative contamination models have been developed, such as the *(fully) independent contamination model* ((F)ICM) as described by Alqallaf, Van Aelst, Yohai, & Zamar (2009). This model assumes that the probability of a data cell to be contaminated equals ϵ, independent from other cells. The researchers showed how contaminated cells propagate when computations are performed on the data matrix to a point where classical robust estimators are not reliable anymore. Therefore, it is important to detect outlying cells.

Gervini & Yohai (2002) developed an univariate filter that could detect cells that where outlying compared to the other cells in the same column. The attention for cellwise outliers has grown quickly in the past few years. Danilov (2010) investigated methods to identify independently contaminated data cells, using both univariate and multivariate detecting techniques. Univariate techniques only use information within one variable (column) to detect outliers, whereas multivariate techniques also use the relationship between multiple variables. Danilov proposed to replace outliers by missing values, instead of reducing their effect with some form of Winsorization. Rowwise outliers were not considered, in contrast to the *DetectDeviatingCells* algorithm developed by Rousseeuw & Van den Bossche (2016), which I study in this research. Unlike most other existing detection algorithms, DDC can deal with data with over 50% contaminated rows. Also, the algorithm performs well on high-dimensional data.

2.2 Regularized regression

In the field of statistics, regularization was first used by Hoerl & Kennard (1970). Instead of minimizing the ordinary least squares objective function

$$S_{OLS}(\hat{\boldsymbol{\beta}}) = ||\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}||_2^2, \tag{2}$$

ridge regression adds a penalty for the ℓ_2-norm of $\hat{\boldsymbol{\beta}}$ to the objective function. This yields:

$$S_{ridge}(\hat{\boldsymbol{\beta}}) = ||\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}||_2^2 + \lambda||\hat{\boldsymbol{\beta}}||_2^2, \tag{3}$$

which reaches its minimum for

$$\hat{\boldsymbol{\beta}}^{ridge} = (\mathbf{X}'\mathbf{X} + \lambda\mathbf{I}_d)^{-1}\mathbf{X}'\mathbf{y} \tag{4}$$

Because of the penalization, the estimated coefficients are shrunk towards zero. λ is called the shrinkage or penalty parameter, as it controls the strength of the shrinkage. A higher value of λ will result in estimates closer to zero. This will increase the bias of the ridge estimate, but also decreases its variance. For $\lambda = 0$, ridge is equivalent to ordinary least squares. For $\lambda \to \infty$, ridge will yield $\hat{\boldsymbol{\beta}}^{ridge} = \mathbf{0}$. In high dimensions, it is shown that ridge performs better than OLS in terms of predictive ability. This solves the first problem that is discussed in the introduction. However, ridge is unable to make a variable selection. Even though the estimated coefficients are shrunk towards zero, they will never be exactly equal to zero.

Interpretability is one of the reasons why other types of regularized regression have been developed. Frank & Friedman (1993) proposed *bridge regression*, which uses the more general penalty $\lambda||\hat{\boldsymbol{\beta}}||_\gamma^\gamma$. Ridge is a special case of bridge (in case $\gamma = 2$). Breiman (1995) proposed a *non-negative garrote* that was able to select important variables and compete with ridge regression in terms of prediction accuracy. The garrote starts with OLS estimates and shrinks them by non-negative factors whose sum is constrained. The main drawback of this method was that it did not perform well in cases of overfit or highly correlated variables. However, Breiman's ideas formed the foundation for the more well known *lasso (least absolute shrinkage and selection operator)* (Tibshirani, 1996). The main difference between lasso and ridge is that lasso penalizes the ℓ_1-norm of the coefficient vector instead of the ℓ_2-norm. Lasso minimizes the objective function

$$S_{lasso}(\hat{\boldsymbol{\beta}}) = ||\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}||_2^2 + \lambda||\hat{\boldsymbol{\beta}}||_1 \tag{5}$$

Unlike ridge regression, there is no analytic solution for the lasso. For $\lambda = 0$, lasso is equivalent to ordinary least squares. For $\lambda \to \infty$, lasso will yield $\hat{\boldsymbol{\beta}}^{lasso} = \mathbf{0}$. For all other values of λ, the lasso estimates are different from the ridge estimates. Because of the ℓ_1-penalty, lasso shrinks the coefficients of some predictors to exactly zero. This is why we say that lasso yields *sparse* solutions. The predictors with nonzero estimated coefficients can be seen as a set of 'important' variables. The size of this set decreases for higher values of λ. Lasso also has its drawbacks. For example, when a group of variables has high pairwise correlations, lasso tends to select only one of the variables, which can be inappropriate in some situations. Besides that, ridge performs better in terms of prediction accuracy than lasso when $n > d$ and the correlations between variables are high.

Because of the limitations of lasso, Zou & Hastie (2005) proposed a new type of regularization, the *elastic net*. The elastic net uses a mixture of the ℓ_1- and ℓ_2-norm as penalty. The researchers discovered that a so-called 'naïve' elastic net approach, with objective function

$$S_{naive\ elnet}(\hat{\boldsymbol{\beta}}) = ||\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}||_2^2 + \lambda_1||\hat{\boldsymbol{\beta}}||_1 + \lambda_2||\hat{\boldsymbol{\beta}}||_2^2, \tag{6}$$

did not perform well because it incurred a double amount of shrinkage, leading to unnecessary extra bias. They corrected for this by setting $\hat{\boldsymbol{\beta}}^{elnet} = (1+\lambda_2)\hat{\boldsymbol{\beta}}^{naive\ elnet}$. Friedman, Hastie, & Tibshirani (2010) used the following notation for the elastic net objective function:

$$S_{elnet}(\hat{\boldsymbol{\beta}}) = \frac{1}{2n}||\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}||_2^2 + \lambda(\alpha||\hat{\boldsymbol{\beta}}||_1 + \frac{1-\alpha}{2}||\hat{\boldsymbol{\beta}}||_2^2) \tag{7}$$

In this notation, α is the mixture parameter, that indicates how the ℓ_1-norm and ℓ_2-norm are mixed in the elastic net penalty. For $\alpha = 0$, the elastic net penalty equals the ridge penalty, whereas for $\alpha = 1$, the elastic net penalty equals the lasso penalty. As α increases form 0 to 1, the sparsity of the elastic net estimator (the arg min of the objective function in (7)) monotonically increases from zero to the sparsity of the lasso estimator. It has been shown by Zou & Hastie (2005) that elastic net regression, in contrast to the lasso, is able to select a group of highly correlated variables in its solution.

3 Replication of the DDC algorithm

3.1 Outlier detection algorithm

In this section, I describe the *DetectDeviatingCells* algorithm, developed by Rousseeuw & Van den Bossche (2016). The algorithm consists of eight main steps, which will not all be described in full detail here. For all details, I refer to the original paper. The underlying model of the algorithm is that the data are generated from a multivariate gaussian distribution but afterwards some cells were corrupted or omitted. Therefore, the variables in the data set should be numerical and take on more than a few different values. Computations are only performed on columns that satisfy these conditions. These columns should be approximately gaussian as well, apart from their outlying values. Skewed data can be transformed beforehand to satisfy this condition.

In the first step, every column of the data matrix is separately standardized, using robust estimators for the location (*robLoc*) and scale parameter(*robScale*). Details of the *robLoc* and *robScale* function can be found in the Appendix of (Rousseeuw & Van den Bossche, 2016). The resulting standardized matrix is called $\mathbf{Z}$. In the second step, the standardized values that are higher (in absolute value) than the 99%-quantile of the chi-squared distribution with one degree of freedom, called c from now on, are removed and set to NA. In short, the researchers removed the values that were anomalously high or low, relative to the other values in their respective columns. The adjusted standardized data matrix is called $\mathbf{U}$.

In the third step, robust estimates for the correlation between each pair of variables (j, h) are computed. The function *robCorr* that is used in this step is also explained in the Appendix of (Rousseeuw & Van den Bossche, 2016). A pair (j, h) is considered to be *connected* when the estimated correlation $\hat{\rho}_{jh}$ between the two variables is higher than 0.5 in absolute value. For every connected pair, the algorithm also estimates the slope b_{jh} of a robust regression line without intercept that predicts variable j from variable h. In step 4, these slope estimates are used to compute a predicted value for all cells. The prediction for cell (i, j) becomes:

$$\hat{z}_{ij} = \frac{\sum\limits_{h \in H_j} |\hat{\rho}_{jh}| b_{jh} u_{ih}}{\sum\limits_{h \in H_j} |\hat{\rho}_{jh}|}, \tag{8}$$

where H_j is the set of variables that are connected with variable j, including j itself. Basically, this prediction takes the weighted mean over all $b_{jh}u_{ih}$, where corresponding absolute correlations $|\hat{\rho}_{jh}|$ function as weights. Because the predictions tend to shrink the scale of the entries, all predictions $\hat{z}_{ij}$ are replaced by $a_j \hat{z}_{ij}$ in step 5, where a_j is an estimate of the slope of a robust regression line without intercept that predicts the observed $z_{.j}$ from the predicted $\hat{z}_{.j}$.

Based on the predictions from the previous steps, the algorithm computes standardized cell residuals in step 6 as follows:

$$r_{ij} = \frac{z_{ij} - \hat{z}_{ij}}{robScale_{i'}(\hat{z}_{i'j} - z_{i'j})} \tag{9}$$

Here, *robScale* is the same function as in step 1. All cells for which $|r_{ij}| \geq c^1$ are flagged as cellwise outliers. Afterwards, all flagged cellwise outliers and all NAs are replaced by their predicted value $\hat{z}_{ij}$, which results in an imputed standardized data matrix $\mathbf{Z_{imp}}$. In step 7, the algorithm flags rowwise outliers. For every row, the following criterion is computed:

$$T_i = \frac{1}{d} \sum_{j=1}^{d} F(r_{ij}^2), \tag{10}$$

where F is the χ_1^2 cumulative distribution function. After standardizing T_i using *robLoc* and *robScale*, the algorithm flags all rows as outlying for which $|T_i|$ exceeds c. Finally, in step 8, the imputed matrix $\mathbf{Z_{imp}}$ is unstandardized to obtain $\mathbf{X_{imp}}$ and all outlying cells and rows are reported.

3.2 Data

Before combining it with different sorts of regularized regression, I applied the outlier detection algorithm on its own. I used four different data sets for this. The same data sets were used in the

[1] 99%-quantile of the chi-squared distribution with one degree of freedom

original paper by Rousseeuw & Van den Bossche (2016):

- **Cars**: The first dataset was scraped from the Top Gear website by Alfons (2016) and contains information about 297 different cars. The original data set contains 32 variables, but only 11 satisfy the conditions mentioned in 3.1. Five of these variables (*Price*, *Displacement*, *BHP*, *Torque* and *TopSpeed*) were logarithmically transformed as they were rather skewed. Finally, row 70 and 96 were taken out because they contained over 50% NAs.

- **TVs**: The second dataset is from Philips and contains 9 characteristics about 677 TV parts from a new production line. This data set was first used by Rousseeuw & Driessen (1999).

- **Mortality**: The third dataset is from the Human Mortality Database and contains mortality rates by age among males in France, from 1816 to 2010. An earlier version was analyzed by Hyndman & Shang (2010).

- **Glass**: The final dataset consists of spectra with 750 wavelengths collected on 180 archaeological glass samples (Lemberge, De Raedt, Janssens, Wei, & Van Espen, 2000). In this case, the number of dimensions exceeds the number of observations, but the DDC algorithm still works in such cases. The first 13 columns were taken out, because over half of their cells contained the same value.

4 Results replication

In this section, I present my results of applying the DDC algorithm on the four different data sets mentioned in section 3.2. I also compare them with the results in (Rousseeuw & Van den Bossche, 2016). The presented cellmaps show which cells are flagged as outlying by the algorithm. To avoid confusion, the flagged outlying rows are not displayed in the figures.

4.1 TopGear data set

Figure 1 shows the results of applying the outlier detection algorithm on the TopGear data set. Only a selection of the rows is displayed. When a cell is colored red, it means that the real value of the cell is much larger than the predicted value. For blue cells, the opposite applies. The figure shows, for example, that the price of the BMW i3 is high compared to its other attributes. The Mercedes-Benz G-class is both a relatively heavy and high car, but its width is low compared to the other features. The weight of the Peugeot 107, 210 kg, is clearly an error. Also, the acceleration time of 0.0 seconds for the Ssangyong Rodius seems very low compared to other attributes. Figure 1 shows small differences between the original implementation and my own implementation. For the Ssangyong Rodius, the authors' implementation found two extra outliers in comparison with my implementation. However, the online available code for the algorithm yields the same results as my implementation. Possibly, the authors made some small changes in the code or did extra input checks which are not described in full detail. This could explain the difference. For the other three data sets, the results of my implementation are exactly the same as the results in the original paper, to the extent that it is observable. Therefore, I only show my own results for the other data sets.

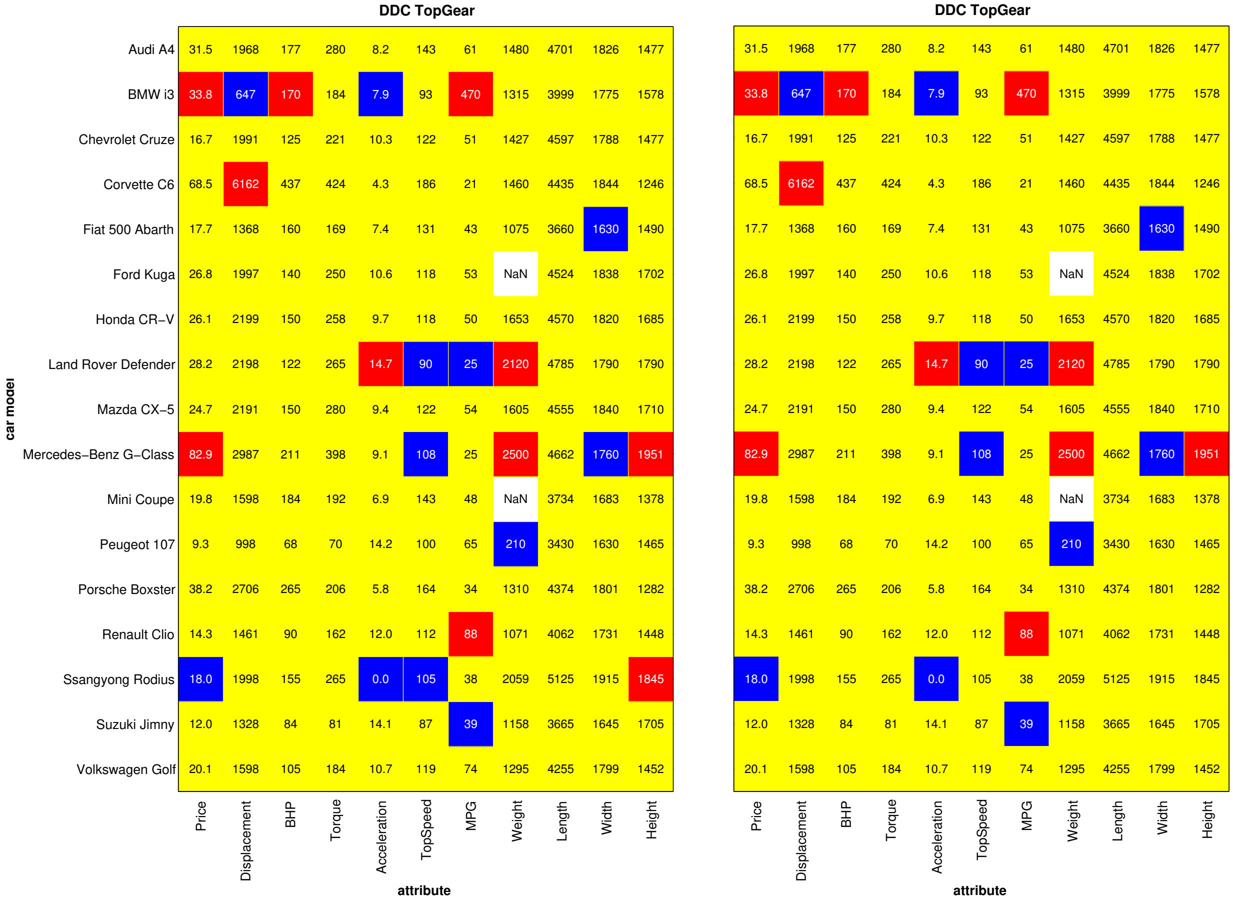

Figure 1: Cell map for a selection of cars from the TopGear data set: (left) for the original paper (Rousseeuw & Van den Bossche, 2016) and (right) for my own implementation. Red and blue cells correspond to respectively anomalously high and anomalously low values.

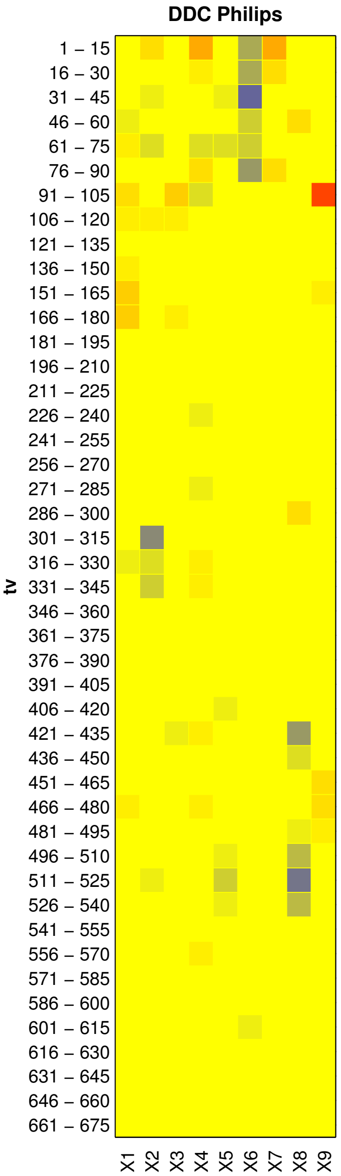

Figure 2: Cell map for a Philips production line. For visibility, cells are grouped in blocks of 15 by 1 and their 'average' color is displayed.

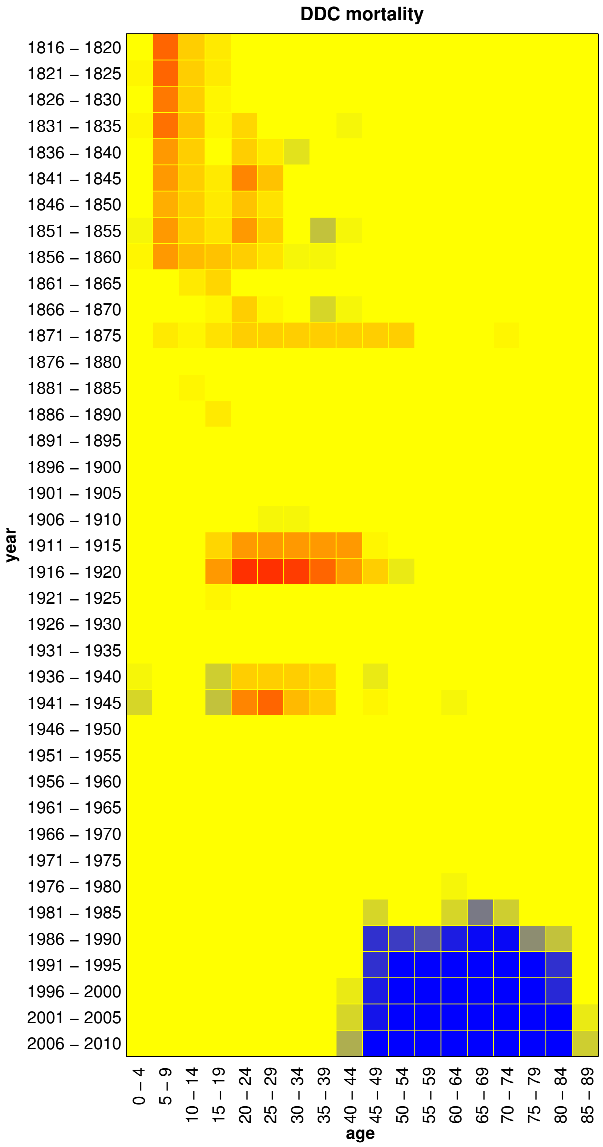

Figure 3: Cell map for male mortality rates in France between 1816 and 2010. For visibility, cells are grouped in blocks of 5 by 5 and their 'average' color is displayed.

4.2 Philips data set

The results for the Philips data set are displayed in figure 2. For visibility, rows are grouped in blocks of 15. The resulting cells attain the 'average' color of the five merged cells. The last two rows are not displayed, as they could not be fit in a block of 15. At the start of the production line, a lot of outliers are detected. Especially, the values for variable X6 are anomalously low. At a later point in the production line, variable X8 attains some anomalously low values.

4.3 Mortality data set

In figure 3, the cell map for male mortality rates in France is displayed. This time, the cells are blocked in groups of 5 by 5. The cellmap clearly shows a relatively high mortality rate among children between 1816 and 1860. In three later periods, the mortality rate among people between the age of 15 and 40 is relatively high, which could be explained by the Prussian War and both World Wars. In recent years, the mortality rates among elderly people is anomalously low. This might be a result of medical advances.

4.4 Glass data set

Finally, the cell map for archaelogical glass samples is displayed in figure 4. The figure clearly shows which wave lengths are anomalous for which glass samples. This provides information about the responsible chemical elements.

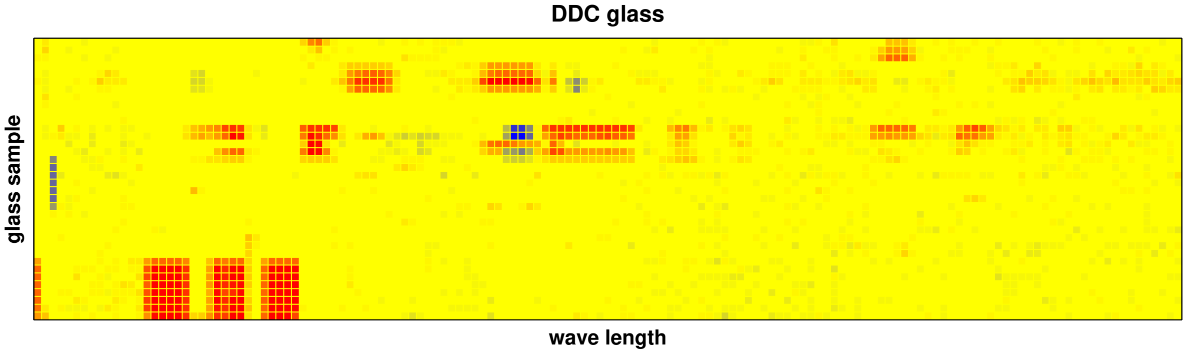

Figure 4: Cell map for spectra of wavelengths for different archaelogical glass samples. For visibility, cells are grouped in blocks of 5 by 5.

5 Regularized regression comparison

In the second part of the research, I extend the study of the DDC algorithm by combining it with regularized regression methods. I generate simulated contaminated data to compare the performance of ridge regression, lasso and elastic net, both before and after applying DDC to clean the simulated data. In this section, I describe my methodology for this simulation study.

5.1 Considered regularization methods

I considered three regularization methods: ridge, lasso and elastic net with mixing parameter $\alpha = 0.5$. For each of the three methods, I estimated the regression coefficients both before and after applying the DDC algorithm on the used data. The algorithm imputed the flagged outlying cells and removed flagged outlying rows. Consequently, I ended up with six different regression estimates for each simulated data set. The estimations were done in MATLAB, using the built-in functions for ridge, lasso and elastic net. The ridge function computes the ridge estimator as formulated in equation 4, whereas the lasso function minimizes the objective function in equation 7 for $\alpha = 1$ (lasso) and $\alpha = 0.5$ (elastic net). Both MATLAB functions standardize the predictors first, to guarantee that all coefficients are penalized on the same scale. After transforming the estimated coefficients back to the scale of the original data, both functions return a coefficient vector $\hat{\boldsymbol{\beta}}$ and an intercept $\hat{\alpha}_0$.

To determine the penalty parameter λ for the regularization methods, I used 10-fold cross-validation. For each estimation, the observations were randomly separated in 10 equally sized groups. Then, each group was once used as a test set, and the concerned estimator was computed based on the sample consisting of the other nine groups, for a given value of λ. In each iteration of the cross-validation, the DDC algorithm was only applied on the sample of nine groups. As a result, I obtained an estimated coefficient vector $\hat{\boldsymbol{\beta}}_k^{\lambda}$ and an estimated intercept $\hat{\alpha}_{0,k}^{\lambda}$ in each iteration k. For this estimate, I computed the median squared prediction error for the test set as follows:

$$MedSPE = \underset{i \in T}{\text{median}}\{(Y_i - \hat{\alpha}_{0,k}^{\lambda} - \mathbf{X_i}\hat{\boldsymbol{\beta}}_k^{\lambda})^2\}, \tag{11}$$

where T is the test set containing 10% of the observations. I chose the λ that yielded the estimates with the lowest average median squared prediction error for the test set over the 10 iterations. I used the median squared prediction error instead of the mean squared prediction error, because the test set was not 'cleaned' by the DDC algorithm and the median squared prediction error gives a robust performance measure for the prediction accuracy of the obtained estimate. For computational reasons, the optimal λ was determined with a precision of only one significant digit. Finally, I used the optimal value of λ to compute the concerned estimator based on the entire sample, yielding estimate $\hat{\boldsymbol{\beta}}$. For this final estimate, the DDC algorithm was applied on the entire data set.

5.2 Data simulation

To analyze the performance of the robust regularized regression estimators, I first generated clean multivariate gaussian data with mean vector $\boldsymbol{\mu} = \mathbf{0}$ and two types of covariance matrices $\boldsymbol{\Sigma}$ with unit diagonal. Firstly, I implemented the algorithm described by Agostinelli et al. (2015) to generate random correlation matrices. These 'ALYZ' random correlation matrices yield relatively low correlations between variables. Secondly, I used the A09 correlation matrix, where the correlation between variable h and j is given by $\rho_{jh} = (-0.9)^{|h-j|}$. This correlation matrix yields both high and low correlations between the variables. After taking n draws from this distribution, I ended up with an n by d data matrix, which is called $\mathbf{X}$. I also generated a d by 1 coefficient vector $\boldsymbol{\beta}$ by randomly setting 10 elements[2] of $\boldsymbol{\beta}$ equal to a random draw from the standard normal distribution and all $d-10$ other elements equal to zero. Finally, I generated a dependent variable $\mathbf{Y}$ as follows:

$$Y_i = \mathbf{X_i}\boldsymbol{\beta} + \epsilon_i \tag{12}$$

for $i = 1, .., n$, where $\mathbf{X_i}$ is the i^{th} row of $\mathbf{X}$ and ϵ is an error term drawn from a normal distribution with mean zero and variance σ^2. σ was chosen such that the signal-to-noise ratio equalled three, i.e. $\sqrt{\boldsymbol{\beta'\Sigma\beta}}/\sigma = 3$. A much higher sigma-to-noise ratio would make it too easy for the regression models to retrieve the real $\boldsymbol{\beta}$, whereas a much lower ratio would make it too hard.

Then, I contaminated the clean data. To generate outlying cells, I replaced a random subset of all $n*(d+1)$ cells[3] by a value γ which was varied to see its effect. Outlying rows were generated in the direction of the eigenvector $\mathbf{v}$ corresponding to the smallest eigenvalue of the true covariance matrix $\boldsymbol{\Sigma}$, as the placement of outliers in this direction is the least favourable for the proposed estimator (Agostinelli et al., 2015). I rescaled $\mathbf{v}$ such that $\mathbf{v'\Sigma^{-1}v} = d$ and then replaced a random subset of the rows in $\mathbf{X}$ by $\gamma\mathbf{v}$. To contaminate the corresponding elements of $\mathbf{Y}$ for these observations as well, I set $\boldsymbol{\beta^*} = -\boldsymbol{\beta}$ and computed the contaminated values of $\mathbf{Y}$ by multiplying the corresponding contaminated rows of $\mathbf{X}$ by $\boldsymbol{\beta^*}$. I repeated the simulations for d ranging from 20 to 200 and n from 10 to 100, each time with 25 replications. Note that the used values of d are relatively large compared to the used values of n, as I am especially interested in high-dimensional data.

5.3 Evaluation of the regularization methods

I evaluated the performance of the methods in two different ways. Firstly, I computed the Euclidean distance between the estimated coefficient vector $\hat{\boldsymbol{\beta}}$ and the real coefficient vector $\boldsymbol{\beta}$:

$$ED(\hat{\alpha_0}, \hat{\boldsymbol{\beta}}) = \sqrt{\hat{\alpha_0}^2 + \sum_{i=1}^{d}(\beta_i - \hat{\beta}_i)^2} \tag{13}$$

The intercept is squared as there was no intercept included in the real coefficient vector. Secondly, I computed the mean squared prediction error (MSPE) of the regression estimate on a newly generated clean data set $\mathbf{X'}$ from the same distribution as $\mathbf{X}$ and with half the amount of observations. $\mathbf{Y'}$ was generated in the same way as $\mathbf{Y}$ (see equation 12). The MSPE was computed as follows:

$$MSPE(\hat{\alpha_0}, \hat{\boldsymbol{\beta}}) = \frac{1}{n}\sum_{i=1}^{n}(Y_i' - \hat{\alpha_0} - \mathbf{X_i'}\hat{\boldsymbol{\beta}})^2 \tag{14}$$

[2] this was possible as I used d $>$ 10 in all simulations

[3] $d+1$ as $\mathbf{Y}$ is also contaminated

6 Results of regularized regression comparison

First, I show the results of applying the proposed regularized regressors on the *cellwise* contaminated simulated data sets, for both ALYZ and A09 correlation matrices. Afterwards, I show the results of the *rowwise* contaminated simulated data sets, again for both types of correlation matrices. I do not display the results of all simulation runs, but, unless specifically mentioned, the comparisons yield qualitatively the same conclusions under different settings.

6.1 Results cellwise contamination

In figure 5, the results of one simulation with a cellwise contaminated data set is displayed, for the two types of correlation matrices. The data set used for the top panels is generated with an ALYZ random correlation matrix, whereas the data set for the bottom panels is generated with the A09 correlation matrix. The top left panel shows the Euclidean distance between the estimated and real coefficient vector for different values of contamination parameter γ. For the same simulation run, the mean squared prediction errors (MSPEs) for the clean data set are shown in the top right panel. The solid lines show the results of the estimations done before applying DDC, whereas the dashed lines show the results of the estimations done after applying DDC.

For $\gamma < 3$, the estimations with DDC do not perform better than the estimations without DDC. This is not strange, as these values of γ are not really outstanding compared to the other values in the data. For higher values of γ, the estimations with DDC clearly outperform their respective counterparts, as they yield lower MSPEs and lower distances to the real coefficients. As γ increases, the MSPEs and distances of the estimations without DDC keep increasing, whereas the MSPEs and distances of the estimations with DDC stay the same or even decrease. In the vast majority of the simulations with the ALYZ model and cellwise contamination, combining DDC with lasso (red) yields the lowest MSPEs and distances, followed by elastic net (blue) and ridge (black). This is also the case for the displayed simulation run. A factor that might have had influence here is the sparse nature of the used coefficient vectors in the data generating process. As ridge regression does not yield sparse estimates, in contrast to lasso and elastic net, this might have been a disadvantage for the ridge estimates.

For the A09 model, the same conclusions can be drawn about applying DDC: it yields lower MSPEs and lower distances for $\gamma \geq 3$. However, over all simulation runs, this time none of the regularization methods uniformly dominated the other two. The reason why lasso performs relatively worse compared to ridge in this case, is probably the use of the A09 correlation matrix, which yields higher correlations than the ALYZ random correlation matrix. When a group of variables are highly correlated, lasso only selects one due to the sparse nature of its estimates. Elastic net and ridge do not have this problem, which might explain why the three methods are very close in terms of their predictive performance and their ability to retrieve the original coefficient vector.

6.2 Results rowwise contamination

In figure 6, the results of one simulation with a rowwise contaminated data set is displayed, again for both types of correlation matrices. For the ALYZ model, roughly the same conclusions can be

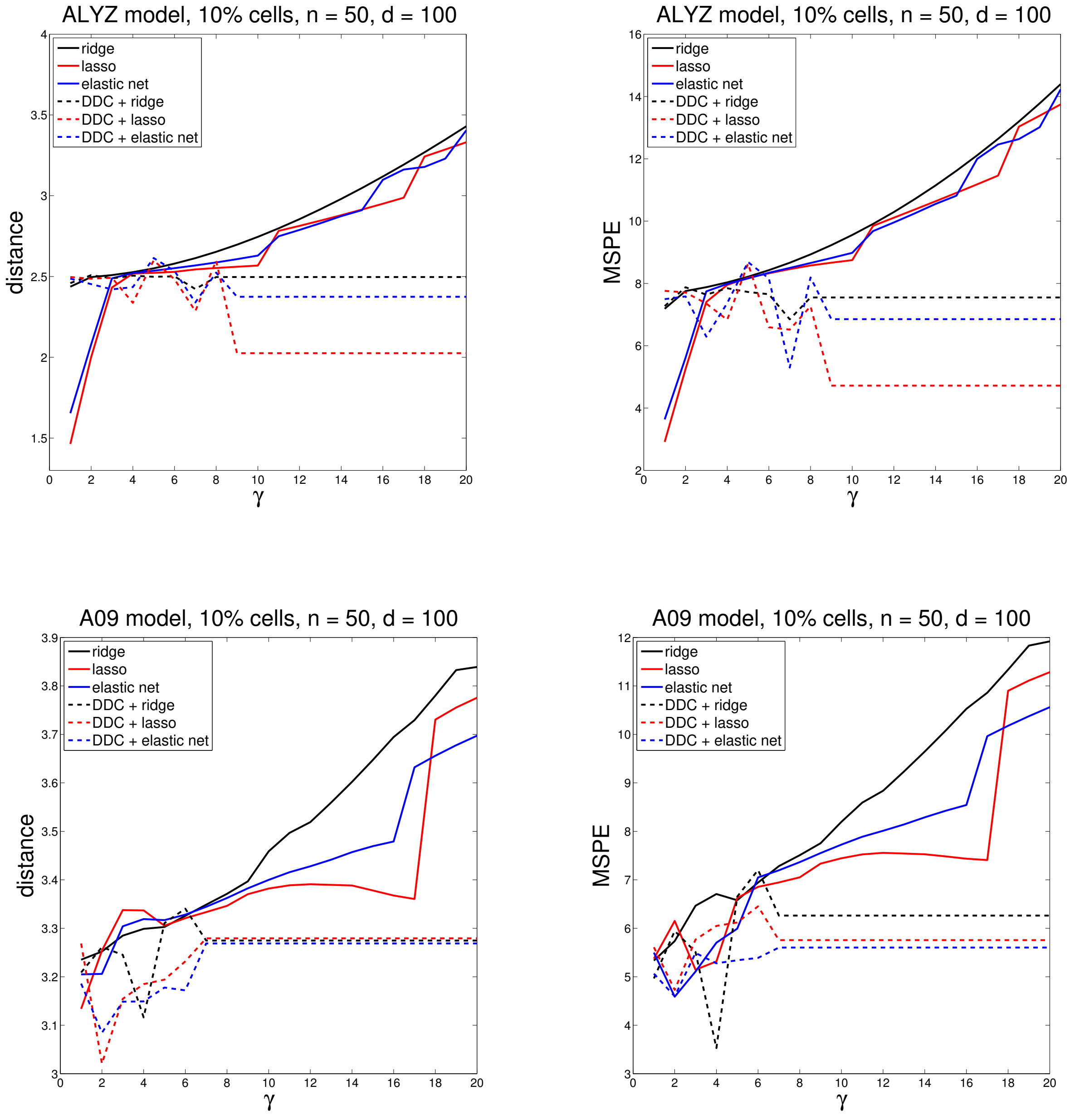

Figure 5: Comparison of different estimators in case of cellwise contamination for different values of contamination parameter γ. Left panels show the Euclidean distance between estimated and real regression coefficient vectors, right panels show the mean squared prediction errors with estimated coefficients for a clean data set: (top) with ALYZ random correlation matrix, (bottom) with A09 correlation matrix. For both types of correlation matrices, the left and right graph correspond to the same simulation.

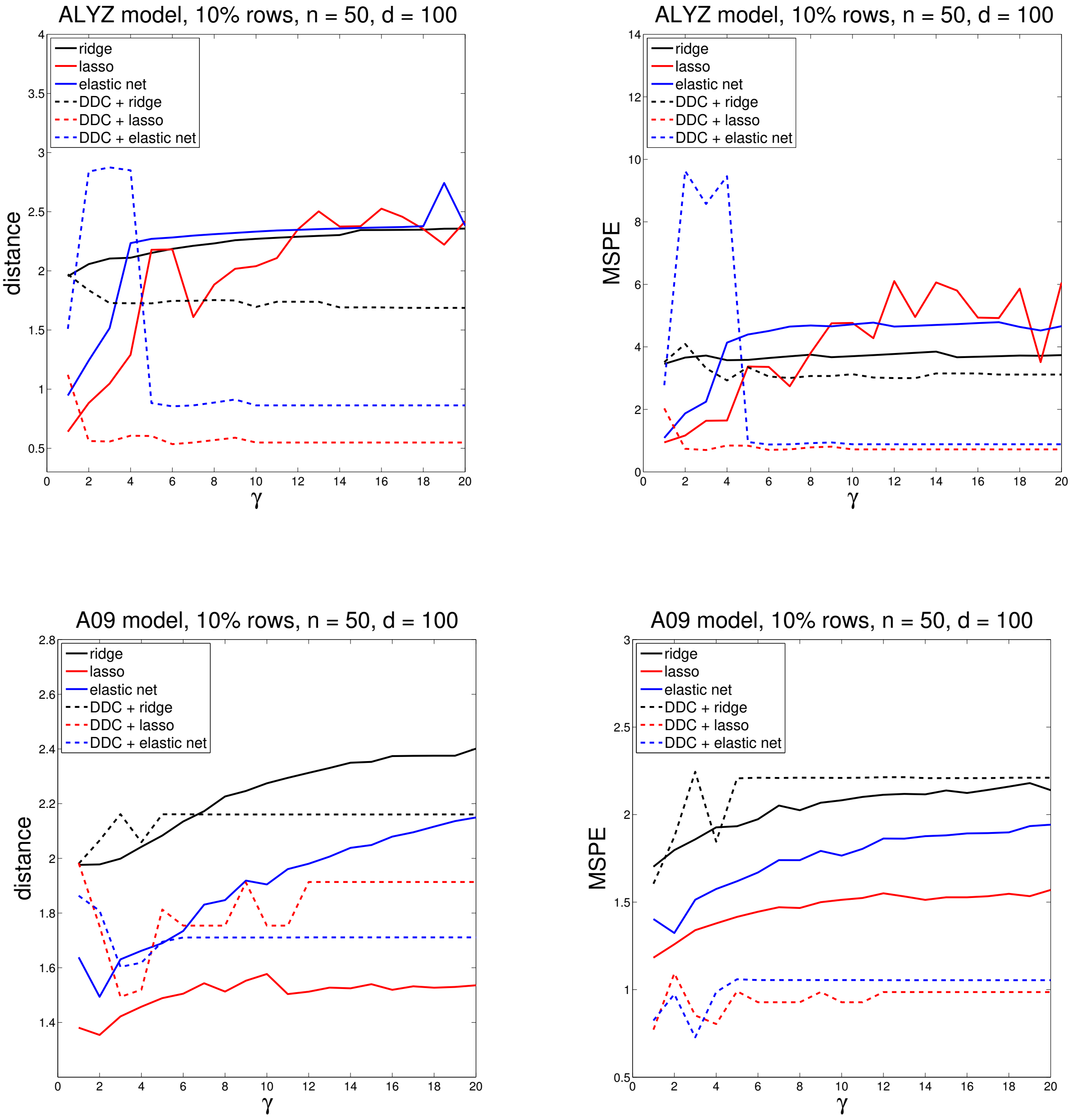

Figure 6: Comparison of different estimators in case of rowwise contamination for different values of contamination parameter γ. Left panels show the Euclidean distance between estimated and real regression coefficient vectors, right panels show the mean squared prediction errors with estimated coefficients for a clean data set: (top) with ALYZ random correlation matrix, (bottom) with A09 correlation matrix. For both types of correlation matrices, the left and right graph correspond to the same simulation.

drawn as for the cellwise contamination: applying DDC yields lower MSPEs and lower distances for high values of γ. Combining DDC with lasso yields the lowest MSPEs and distances in most of the simulations, followed by elastic net and ridge.

For the A09 model, the results are slightly different. Under these settings, applying DDC does not always yield lower MSPEs and/or distances, even though the algorithm was always able to exactly identify and remove the artificially contaminated rows for $\gamma > 2$. In the displayed simulation, applying DDC yields higher distances for lasso and higher MSPEs for ridge. Possibly, the DDC algorithm flags some cells that are not really outlying, which harms the quality of the estimates. Over all simulations, lasso and elastic net did not dominate the other one uniformly. In most simulations, ridge yielded higher MSPEs and distances than the other two methods when they were combined with DDC. Again, the high correlations in A09 in combination with the lack of ability for lasso to select a group of highly correlated variables might be a reason why lasso estimates do not outperform elastic net estimates, whereas they do for the ALYZ random correlation matrix.

7 Conclusion

The objective of the first part of this research was to replicate the results of Rousseeuw & Van den Bossche (2016). My results for the TopGear data differed slightly from their results, possibly due to a small difference in the used algorithm. For the other three data sets, the results are exactly the same. For the second part of the research, I aimed to find answers to two questions:

- How does applying the DDC algorithm on the set of dependent and independent variables influence the performance of regularized regression estimates, in terms of prediction accuracy and distance to the real coefficient vector?
- Which regularized regression method performs the best, in terms of prediction accuracy and distance to the real coefficient vector, when the DDC algorithm is first applied on the set of dependent and independent variables?

As regards the first question, applying the DDC algorithm yielded estimates with lower MSPEs and lower distances to the real coefficient vector in the vast majority of the simulations for $\gamma > 3$. Only for the A09 model with contaminated rows, the DDC algorithm did not always improve the performance of the estimates compared to the estimates of the same regularization method but without applying DDC. These results support the idea that detecting outliers and reducing their influence is important to find good regression estimates. As regards the second question, lasso dominated the other two regularization methods in terms of MSPE and distance in the ALYZ model, when combined with DDC. For the A09 model, none of the methods uniformly dominated the other two, although ridge performed slightly worse in case of rowwise contamination. All in all, it can be concluded that applying the DDC algorithm on the dependent and independent variables in most cases yields estimates that are robust to cellwise and rowwise contamination.

8 Discussion

8.1 Limitations

Although this research gives some insights in the relation between different regularization methods and the importance of detecting outliers, it also has its limiations. For example, I fixed the mixture parameter α in the elastic net regression. It would be better to determine the optimal value of α in a similar way as the optimal value of λ. Because this is computationally expensive, this was not done in this research. If α would have been chosen optimally, the elastic net estimates would probably have yielded other MSPEs and distances. Of course, it would also have been better to determine the optimal value of λ with more precision. For computational reasons, I computed λ with a precision of one significant digit.

Secondly, the sparse design of the coefficient vector in the data generating process might have favored the performance of lasso (and, to a lesser extent, elastic net). The majority of the coefficients in the 'real' coefficient vector $\boldsymbol{\beta}$ are set to zero, as explained in section 5.2. Because lasso usually yields sparse solutions (in contrary to ridge), the data generating process might have given lasso an advantage.

The used contamination models are relatively simple: in each simulation, a subset of the data cells (rows) is replaced by a fixed value (row). In reality, the way and shape in which outliers appear in data sets is probably a lot more complex. Some outliers are caused by human mistakes, others are just special observations. It is highly unlikely that all outliers will deviate to the same extent, as in the contamination model I used. It is also unlikely that outliers are always completely randomly distributed over the data set. For example, when a variable is not measured correctly, its column will contain a relatively large amount of outliers compared to other columns. In short, outliers in reality do not always behave in the way outliers are generated in this research. This harms the external validity of the conclusions.

8.2 Future research

As regards the DDC algorithm, the authors of the original paper already mentioned that the algorithm could be extended to non-numerical variables, such as nominal variables (e.g. car brands in the Top Gear data sets) and binary variables (e.g. gender for a mortality data set). To achieve this, correlations would need to be replaced by measures of bivariate association. Also, linear regression would need to be replaced by logistic regression or another technique that is capable of dealing with binary variables.

As regards the comparison of regularized estimators, some points for future research follow directly from the limitations of my research. To fully investigate the performance of elastic net, the optimal value of α should be determined in a similar way as λ (e.g. with cross-validation). Also, it would be interesting to see what happens to the relative performance of the regularization methods when coefficient vector $\boldsymbol{\beta}$ is designed in another, less sparse way. Another point for future research is comparing the estimators after applying other types of outlier detection (e.g. the Gervini-Yohai filter). This would provide more insight into the added value of applying the DDC algorithm instead of other outlier detection techniques.

References

Agostinelli, C., Leung, A., Yohai, V. J., & Zamar, R. H. (2015). Robust estimation of multivariate location and scatter in the presence of cellwise and casewise contamination. *Test*, *24*(3), 441–461.

Alfons, A. (2016). *Package robusthd, r-package version 0.5. 1.*

Alqallaf, F., Van Aelst, S., Yohai, V. J., & Zamar, R. H. (2009). Propagation of outliers in multivariate data. *The Annals of Statistics*, 311–331.

Breiman, L. (1995). Better subset regression using the nonnegative garrote. *Technometrics*, *37*(4), 373–384.

Danilov, M. (2010). *Robust estimation of multivariate scatter in non-affine equivariant scenarios* (Unpublished doctoral dissertation). University of British Columbia.

Frank, L. E., & Friedman, J. H. (1993). A statistical view of some chemometrics regression tools. *Technometrics*, *35*(2), 109–135.

Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of statistical software*, *33*(1), 1.

Gervini, D., & Yohai, V. J. (2002). A class of robust and fully efficient regression estimators. *Annals of Statistics*, 583–616.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, *12*(1), 55–67.

Huber, P. J., et al. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, *35*(1), 73–101.

Hyndman, R. J., & Shang, H. L. (2010). Rainbow plots, bagplots, and boxplots for functional data. *Journal of Computational and Graphical Statistics*, *19*(1), 29–45.

Lemberge, P., De Raedt, I., Janssens, K. H., Wei, F., & Van Espen, P. J. (2000). Quantitative analysis of 16–17th century archaeological glass vessels using pls regression of epxma and μ-xrf data. *Journal of Chemometrics*, *14*(5-6), 751–763.

Rousseeuw, P. J., & Driessen, K. V. (1999). A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, *41*(3), 212–223.

Rousseeuw, P. J., & Van den Bossche, W. (2016). Detecting deviating data cells. *arXiv preprint arXiv:1601.07251*.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267–288.

Tukey, J. W. (1962). The future of data analysis. *The annals of mathematical statistics*, *33*(1), 1–67.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(2), 301–320.