

UNIVERSITÉ PARIS-SACLAY

M2 - OPTIMIZATION

Master Thesis

Discipline: Applied Mathematics

QUANTILE REGRESSION IN LARGE ENERGY DATASETS

Author
Nhat-Thien PHAM
Université Paris-Sud

Supervisor
Prof. Leo LIBERTI
École Polytechnique

Host Organization
LIX, École Polytechnique
Bâtiment Alan Turing, 91120 Palaiseau

Intern Period
From 1st April 2018
To 30th September 2018

Palaiseau, September 2018

Table of contents

Acknowledgements

Thank all those who helped me during the master and the internship. Specially, I would like to sincerely thank Prof. Leo Liberti, my kind supervisor during the internship at LIX. His advices and discussions, not only about science but also about life, gave me very good experiences.

I would like to thank all tutors in the program Optimization and Prof. Filippo Santambrogio who personally helped me a lot during my master. Also, I would like to thank Manuel Ruiz and Maxime Fortin who gave me very helpful discussions and knowledge in the energy industry.

Finally, thanks my family and my friends for inspiring and supporting me. Thank you!

Abstract

In most of the prediction problems, we are often interested in knowing the uncertainty in our predictions. Knowing about the range of predictions, as opposed to only knowing the point estimates, can significantly improve decision making processes for many problems. Quantile regression helps to provides sensible prediction intervals even for datasets whose residuals have non-constant variance or non-normal distribution.

Quantile regression problems can be formulated as Linear Programming (LP) problems. However, these LPs turn out to have large size as the size of the dataset could be very large. By using random projections, they could be solved approximately but efficiently.

Keywords: Quantile Regression , Linear Programming, Random Projections, Optimization

---

Chapter 0

Introduction

Contents

The word *quantile* comes from the word *quantity*. Quantile is one of the most important measures in statistics. A very first introduction to quantile is as follow. Given a set $S = \{1, 3, 4, 9, 20, 30, 51, 99\}$, we call the number 20 the 0.5-quantile of S since 50% of the values in S fall below 20 (easy to verify). Similarly, the 0.75-quantile of S is the number 30, because 75% of the values fall below it. Quantile can also be applied to continuous case. The mathematical definition of quantile will be recalled in Section 2.1. Now, let us introduce the objective of the stage and the structure of this thesis.

0.1 Motivation and objective of the stage

Electricity Transmission Network, usually known as RTE[1], is the electricity transmission system operator of France. It is responsible for the operation, maintenance and development of the French high-voltage transmission system. With nearly 105,000 km of lines, RTE's grid is the biggest in Europe [16]. Within the framework of the European directive for the Energy and Climate package, France has committed itself to a share of renewable energies accounting for 23% of final energy consumption by the year 2020. Applied to power generation, this target results in increasing the share of renewable energies (mainly hydropower, wind power, photovoltaic power and biomass energy) to 27% [16].

[1] Réseau de Transport d'Électricité (French)

Motivated by research work at RTE (see Chapter 3 and Chapter 4) and a recent paper on using random projections (paper [18]), this thesis focuses on formulating and solving several kinds of quantile regression problems. These problems both are random generated and arise in real-life applications. We will see in Section 2.2 that quantile regression problems can be formulated as LPs. However, solving these LPs would be expensive to apply to large datasets. We shall show later (Chapter 5) how to solve them approximately but efficiently by using random projections.

Let us briefly describe the main problems we will consider in this thesis. The first problem is **constrained quantile regression**, in which some constraints will be added to a quantile regression problem. In fact, inequality constrained quantile regression has been studied by R. Koenker and Pin Ng (see [13]). In this thesis, motivated by some applications, we will consider the *parallel* and *non-crossing* constrained quantile regression.

Next, given a dataset that assumed to be large and high-dimension, the question is: could we *"learn"* the distribution of the data points in this dataset? In other words, could we randomly generate a new dataset that well captures *"the way data points distributed"* in the original one? It turns to study **a workflow to generate quantile invariant data**. This is the second problem we consider.

The more accurately we forecast the generated wind power, the more active we are in operating the electricity transmission. As a part of the process to improve the accuracy, our last problem focuses on **forecasting the quantile of the generated wind power** based on meteorological data. More precisely, there are over 14,000 weather stations in France that provide us the wind speed every 10 minutes, our goal is to build a model that receives wind speed at the weather stations as input and will return the quantile of the generated wind power as output. The model is built for each region and each lead time individually. Moreover, one of the main goals is to identify the weather stations that have most impacts on forecasting wind power for a given region.

0.2 Structure of the thesis

This thesis is organized as follows. In Chapter 1, we briefly introduce some preliminaries on convex optimization problems, probability theory as well as preliminaries on the topic of linear regression and LP. This informal introduction will make it easier for readers to follow our work.

In Chapter 2, we will introduce the concepts of quantile and quantile regression. We will see that solving a quantile regression problem is indeed the same as solving an LP. We also give several examples of solving constrained quantile regression problems.

In Chapter 3, motivated by some real-life applications at RTE, we introduce a workflow to generate quantile invariant data. This means that the new random generated data will have the same

quantiles as the original ones. These kinds of data are typically high-dimension, so we will also use Principal Component Analysis (PCA) to reduce the dimension.

In Chapter 4, we will study the problem of forecasting the quantiles of the generated wind power in France based on the weather data for some given regions and lead times. We propose a linear model applied to wind speed at the weather stations and then use the quantile regression framework to find the coefficients. The results after that will be used to reduce the dimension of the problem and improve the forecasts.

In Chapter 5, we introduce the Johnson-Lindenstrauss lemma and recall a recent result in solving LP using random projections. We will see how the random projections approximately preserve the feasibility and optimality of LP instances. At the end of this chapter, we will use random projections to solve the quantile regression problems discussed in the previous chapters and then compare the results.

Finally, Chapter 6 concludes the thesis and discusses some further works.

Chapter 1

Background

Contents

1.1 Preliminaries on convex optimization problems

Convex optimization is a subfield of optimization that studies the problem of minimizing convex functions over convex sets. Let us define the general convex optimization problem as follows. Given a set $X \in \mathbb{R}^n$ and a function $f : X \mapsto \mathbb{R}$ convex on X, we want to find $x^* \in X$ such that for all $x \in X$, there holds $f(x) \geqslant f(x^*)$. The function f is called *objective function* and the variable vector x is called *decision* variable vector. Usually, we only consider the case where X is a convex subset of $\mathbb{R}^n$ defined by constraints. Hence, a general convex optimization problem is defined by

$$\begin{cases} \min_{x \in \mathbb{R}^n} f(x) \\ c_i(x) = 0 \quad i \in E, \\ c_i(x) \leqslant 0 \quad i \in I, \end{cases}$$

where $c_i : \mathbb{R}^n \mapsto \mathbb{R}$, c_i is affine for all $i \in \mathbb{E}$, c_i convex for all $i \in I$ and E, I are two disjoint sets of integers.

There is no analytical formula for the solution of convex optimization problems in general, but there are many effective algorithms for solving them. Interior point methods work very well in practice, and in some cases can be proved to solve the problems to a specified accuracy with a

number of operations that does not exceed a polynomial of problem dimensions [3].

Optimization algorithms are usually iterative. Most of them begin with an initial guess and generated a sequence of improved estimates until they terminate, hopefully at a solution. Some algorithms accumulate information gathered at previous iterations, while the others use only local information obtained at the current point. A good algorithm should possess the following properties.

- Accuracy. It should be able to identify a solution with precision.
- Efficiency. It should not require excessive computer time and storage.
- Robustness. It should perform well on a wide variety of problems in their class, for all reasonable values of the starting point. It should not be overly sensitive to errors in the data or to the arithmetic rounding errors that occur when the algorithm in implemented on a computer.

With today's solution technology, solving convex optimization can easily be achieved using off-the-shelf solver. According to recent results on convex optimization research, if we can formulate a problem as a convex optimization problem, then we can solve it efficiently. So, the challenge in using convex optimization is simply in recognizing and formulating the problems.

1.2 Preliminaries on probability theory

In this section, we will briefly present some concepts and definitions in probability theory, which are elementary but useful in order to follow the rest of thesis.

In probability theory, a probability space consists of three parts:

1. a sample space Ω, which is the set of all possible outcomes;
2. a set of events $\mathcal{A}$, where each event is a subset of Ω containing zero or more outcomes;
3. the assignment of probability to the events, that is, a function $\mathbb{P}$ from events to probabilities.

Mathematically, if Ω is a non-empty set, $\mathcal{A}$ is a σ-algebra over Ω and $\mathbb{P}$ is a probability measure[1] on $\mathcal{A}$ then the triplet $(\Omega, \mathcal{A}, \mathbb{P})$ forms a probability space.

A function $X : \mathcal{A} \to \mathbb{R}$ is called a random variable if for all Lebesgue measurable set Y in $\mathbb{R}$, $X^{-1}(Y) \in \mathcal{A}$. In many cases, at least in this thesis, random variables are real-valued. Roughly speaking, a random variable is a variable whose possible values are outcomes of a random phenomenon. For example, the demands of electricity of certain regions are random variables, because

[1] $\mathbb{P} : \mathcal{A} \to [0, 1]$, $\mathbb{P}(\Omega) = 1$ and $\mathbb{P}$ satisfy the countable additivity property, *i.e.*, for all countable collections $\{E_i\}$ of pairwise disjoint sets, $\mathbb{P}\Big(\bigcup_{i\in I} E_i\Big) = \sum_{i\in I} \mathbb{P}(E_i)$

they could change after a certain time, depending on several causes. To understand such random variables, in statistics, we collect a dataset containing a number of observations of the random variables, then consider different quantitative measures associated to it. The most important and popularly used are: *mean, variance, standard deviation* and *correlation*.

Definition 1.2.1 (Mean). *Let $\{x_1, x_2, \ldots, x_n\}$ be a sample of a random variable X. The mean of X is defined as*

$$\mu_X = \frac{1}{n}\sum_{i=1}^{n} x_i.$$

Note that the mean here is the sample mean, *i.e.*, it refers to a central value of a discrete set of numbers. In this thesis, most of the definitions and properties are defined based on samples.

Definition 1.2.2 (Variance). *Let $\{x_1, x_2, \ldots, x_n\}$ be a sample of a random variable X. The variance of X is defined as the averaged value of the squared deviation from the mean of X:*

$$var(X) = \frac{\sum_{i=1}^{n}(x_i - \mu_X)^2}{n-1}$$

where μ_X is the sample mean of X.

Variance is a measure of the spread of data in a dataset. For example, a dataset with higher variance will have more spread in data from the mean than a the lower variance one. The *standard deviation* of a random variable is defined as the square root of its variance.

Definition 1.2.3 (Covariance). *Let $\{x_1, x_2, \ldots, x_n\}$ and $\{y_1, y_2, \ldots, y_n\}$ be samples of random variables X and Y, respectively. The covariance between X and Y is defined as*

$$cov(X, Y) = \frac{\sum_{i=1}^{n}(x_i - \mu_X)(y_i - \mu_Y)}{n-1}.$$

In other words, the covariance between two real-valued random variables X and Y is defined as the expected product of their deviations from their individual expected values. The variance can be thought of as the covariance of a random variable with itself.

Definition 1.2.4 (Correlation). *The correlation between two random variable X and Y is defined as*

$$corr(X, Y) = \frac{cov(X, Y)}{\sigma_X \sigma_Y},$$

where σ_X, σ_X are the standard deviations of X and Y.

1.3 Linear regression

Suppose that we have a set $\boldsymbol{X}$ which contains m "observations", *i.e.*, vectors, and a set $\boldsymbol{Y}$ which contains m corresponding "outputs" for each observation in $\boldsymbol{X}$. We denote the ith observation in $\boldsymbol{X}$ by $x_i \in \mathbb{R}^n$ and the corresponding output by $y_i \in \mathbb{R}^k$. One of the main aims of statistics is to predict something by means of (observation, output) pairs. We call the prediction task *regression* when we are predicting a quantitative output, and *classification* when we are predicting a qualitative output. Typically, in machine learning, we call the pair $(\boldsymbol{X}, \boldsymbol{Y})$, where $\boldsymbol{X} = (X_1, \ldots, X_n)$, a *training set*, $\boldsymbol{Y}$ the *target* and components of an observation, *i.e.*, $X_1, \ldots, X_n$, the *variables* (or *features*). Throughout this thesis, we should keep in mind that the input is always a matrix whose rows and columns are observations and variables, respectively, whereas the output can be a matrix (such as in Chapter 2) or a column vector (such as in Chapter 3).

Remark 1.3.1. *In this thesis, all the vectors are column vectors. We use the uppercase letters such as $X, Y, X_1, \ldots, X_n$ for random variables. The datasets and matrices are represented by bold symbol letters. The observed values are written in lowercase, for example, the ith observation in dataset $\boldsymbol{X}$ will be accessed by x_i^T, i.e., the ith row of matrix $\boldsymbol{X}$.*

Before introducing *quantile regression*, let us recall the concepts of *linear regression*, which is a widely used approach to model the relationship between a response (dependent variable) and one or more explanatory variables (independent variables).

Given a vector of inputs $X^\mathsf{T} = (X_1, X_2, \ldots, X_n)$, a linear regression model assumes that the *regression function* is linear in the inputs, namely,

$$f(X) = \beta_0 + \sum_{j=1}^{n} X_j \beta_j,$$

where, the β_j's are unknown *parameters* or *coefficients*. Often, it is convenient to include the constant variable 1 in X (by adding one more component to X, which will be fixed to 1), include

β_0 in the vector of parameters β and then write the linear model in vector form as an inner product

$$f(X) = X \cdot \beta.$$

Here, the output of the regression function f could be any information we want to infer from X. The most widely developed one is the *conditional mean*, in which $f(X)$ should be able to return the conditional mean of the response given some values of the explanatory variables (or predictors). In this thesis, we are interested in the *conditional quantile* of the response.

Remark 1.3.2. *Depending on which loss function we are using, $f(X)$ could be regressed toward the mean, the median or any quantile of the response variable. Some regression loss functions which are popularly used in machine learning are: mean square error (MSE), mean absolute error (MAE), Huber loss and quantile loss.*

Finally, we would note that in a linear model, X_j can come from different sources:

- quantitative inputs;
- transformations of quantitative inputs, such as log, square or square root;
- combination between variables, for example, $X_3 = X_1 X_2$.

No matter what the sources of X_j are, the model is linear in the parameters. In Chapter 4 we will study a linear model that uses squares and cubes of the inputs.

1.4 Linear programming

Linear programming (LP) is an important class of optimization problems in which the objective and the constraint functions are linear. It is known that many practical problems in operations research can be expressed as LPs, such as *network flow problems*, *transportation optimization problems, blending problems...* Besides, in theoretical aspect, many problems could also be reformulated as LPs, such as *linear membership problems, Chebyshev approximation problems,...* Quantile regression is also a class of problems that can be reformulated as LPs, we will see this in the next chapter. In this section, let us recall the formulation of an LP and introduce briefly some recent results on solving LPs.

Linear programs are usually stated in the following standard form:

$$\begin{aligned} &\text{minimize } c^\mathsf{T} x \\ &\text{subject to: } \quad a_i^\mathsf{T} x = b_i, \quad i = 1, \dots, m, \\ &\qquad\qquad\quad x \in \mathbb{R}^n_+. \end{aligned}$$

Here, the vectors $c, a_1, \ldots, a_m \in \mathbb{R}^n$ and the scalars $b_1, \ldots, b_m \in \mathbb{R}$ are the problem parameters that specifies the objective and constraint functions.

There are many effective methods for solving LPs, including Dantzig's simplex method and the more recent interior point methods. It is known that the complexity in practice is order n^2m (assuming $m \geqslant n$) at least for interior point methods [3], but the worst case complexity also depends on the storage size of the list input data [11]. We can easily solve sparse problems with thousands of variables and thousands of constraints on a small desktop computer, in a matter of second. Dense problems may be much harder to solve.

Several solvers such as CLP, CPLEX, Gurobi, MOSEK, XPRESS,... and modeling languages such as AMPL, JuMP, GAMS,... have been developed to solve the LPs efficiently. Additionally, some other software have their own integrated solver and modeling environment, such as MATLAB (with Optimization Toolbox), CMPL, FICO® Xpress [6]. LP solvers offer a choice of simplex and interior-point methods for continuous problems. They have also been extended with other algorithms of various kinds, to improve their effectiveness or range of applicability. The implementation related in this thesis were carried out in Julia using CPLEX solver and JuMP modelling language.

Chapter 2

Constrained quantile regression

Contents

2.1 Quantile regression

Quantile regression is a useful approach to modeling various aspects of conditional distributions. In the last decade, quantile regression has attracted considerable attention. There are some reasons for such popularity. The first, quantile regression estimation can be robust to non-Gaussian or heavy-tailed data. Second, it includes the commonly used least absolute deviation (LAD) method as a special case. And, the quantile regression model allows us to provide more easily interpretable regression estimates obtained via quantiles $\tau \in (0, 1)$. There is an extensive literature on estimation and inference for various quantile regression models, see Koenker [12]. In this section, let us introduce some definitions and the formulation of a standard quantile regression framework.

2.1.1 Definitions

Definition 2.1.1 (Quantile). *Let Y be a real valued random variable with cumulative distribution function $\mathcal{F}_Y(y) = \mathbb{P}(Y \leqslant y)$. For all $0 \leqslant \tau \leqslant 1$, the τ-quantile of Y is given by*

$$\mathcal{Q}_Y(\tau) = \mathcal{F}_Y^{-1}(y) = \inf\{y : \mathcal{F}_Y(y) \geqslant \tau\}.$$

From the definition, τ-quantile of a continuous random variable Y is the point at which the area under the PDF[1] curve, from the left to that point, is equal to τ, see Figure 2.1 for examples.

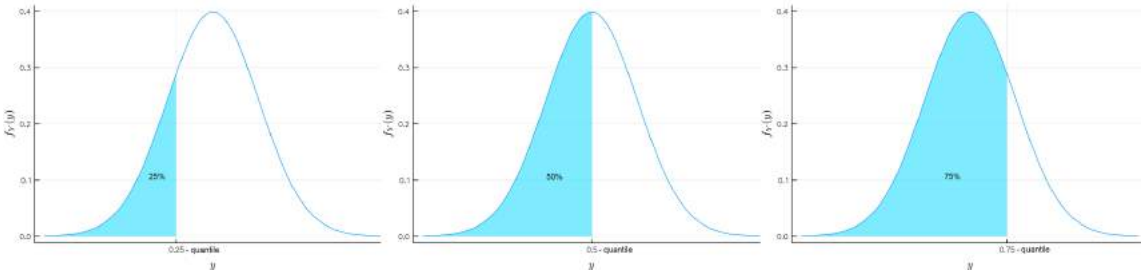


Figure 2.1: Examples of the quantiles of a Normal distribution

We known that, in linear regression, the most commonly used loss function is the mean squared error (MSE) function. Similar to the linear regression framework, in order to penalize and infer the parameters, we need a loss function for quantile regression:

Definition 2.1.2 (Quantile loss function). *Given $0 \leqslant \tau \leqslant 1$, the quantile loss function is defined as*

$$\ell_\tau(u) = u(\tau - \mathbb{I}_{\{u<0\}}), \quad \forall u \in \mathbb{R},$$

where $\mathbb{I}$ is the indicator function.

Note that the quantile loss function could also be rewritten as

$$\ell_\tau(u) = \begin{cases} \tau u & \text{if } u \geqslant 0 \\ (\tau - 1)u & \text{if } u < 0 \end{cases}.$$

Now, given $\tau \in (0, 1)$, let Y be a real-valued random variable with cumulative distribution function $\mathcal{F}_Y(y) = \mathbb{P}(Y \leqslant y)$, the problem under consideration is the minimization of a convex stochastic

[1] probability density function

function:

$$\underset{q_\tau \in \mathbb{R}}{\text{minimize}}\ \mathbb{E}\big[\ell_\tau(Y - q_\tau)\big]. \tag{2.1}$$

Here, q_τ can be understood as an estimation of the τ-quantile of Y. If the quantile is supposed to be a *conditional quantile*, *i.e.*, q_τ depends on a given set of observations $\boldsymbol{X}$, then inferring the relationship between $\boldsymbol{X}$ and the quantile of Y is called **quantile regression**. In other words, quantile regression aims at estimating the quantiles of the response variables given certain values of the predictor variables. In general, we seek a function f such that

$$q_\tau = f(X),$$

where $X = (X_1, X_2, \ldots, X_n)$ is the vector of stochastic predictor variables with observations observed in $\boldsymbol{X}$. We have the following proposition:

Proposition 2.1.1. *If q_τ^* is the optimal solution of problem (2.1) then q_τ^* is the τ-quantile of Y, namely, if*

$$q_\tau^* = \underset{q_\tau \in \mathbb{R}}{\textit{argmin}}\ \mathbb{E}\big[\ell_\tau(Y - q_\tau)\big]$$

then $\mathcal{F}_Y(q_\tau^*) = \tau$, where $\mathcal{F}_Y(y)$ is the cumulative distribution function of Y.

Proof. Indeed, the problem (2.1) can be written as

$$\underset{q_\tau \in \mathbb{R}}{\text{minimize}}\ \left\{(\tau - 1)\int_{-\infty}^{q_\tau}(y - q_\tau)d\mathcal{F}_Y(y) + \tau\int_{q_\tau}^{\infty}(y - q_\tau)d\mathcal{F}_Y(y)\right\}. \tag{2.2}$$

If we denote the objective function by $H(q_\tau)$ and assume that q_τ^* is the optimal solution of (2.2) then we must have

$$\nabla H(q_\tau^*) = 0,$$

where $\nabla H(\cdot)$ is the derivative of $H(\cdot)$ with respect to q_τ. It means

$$-(\tau - 1)\int_{-\infty}^{q_\tau^*} d\mathcal{F}_Y(y) - \tau\int_{q_\tau^*}^{\infty} d\mathcal{F}_Y(y) = 0.$$

By noting that $\int_{-\infty}^{\infty} d\mathcal{F}_Y(y) = 1$, the above equation is reduced to

$$\int_{-\infty}^{q_\tau^*} d\mathcal{F}_Y(y) - \tau = 0,$$

which is equivalent to $\mathcal{F}_Y(q_\tau^*) = \tau$. □

Note that to be more general, in the proof of Proposition 2.1.1 we used the notations in the continuous case, *i.e.*, Y is assumed to be continuous. However, in machine learning, the target Y is always a discrete random variable conditioned on the set of observations $\boldsymbol{X}$. So, the considered problem (2.1) becomes

$$\underset{q_\tau \in \mathbb{R}^n}{\text{minimize}} \sum_{i=1}^{m} \ell_\tau(y_i - q_\tau), \tag{2.3}$$

where $y_1, y_2, \ldots, y_m$ are observed values of Y. Here, we neglected the constant $\frac{1}{m}$ in the expression of expectation. To finish this subsection, we would emphasize that q_τ is the estimation of the τ-quantile of Y, while $\mathcal{Q}_Y(\tau)$ denotes the exact τ-quantile of Y. In other words, $\mathcal{Q}_Y(y)$ is estimated by q_τ.

2.1.2 Quantile regression using a linear model

A simple but very efficient way to estimate $\mathcal{Q}_Y(\tau)$ is using **linear model**, in which τ-quantile of the output depends linearly on the features of the inputs. Namely, in linear model, $\mathcal{Q}_Y(\tau)$ is estimated by $X^\mathsf{T}\beta_\tau$ where $\beta_\tau \in \mathbb{R}^n$ is the unknown parameter (sometimes called *coefficient* or *weight*). Therefore, the problem (2.3) becomes

$$\underset{\beta_\tau \in \mathbb{R}^n}{\text{minimize}} \sum_{i=1}^{m} \ell_\tau(y_i - x_i^\mathsf{T}\beta_\tau). \tag{2.4}$$

Here, recall that, we used x_i to refer to an observation of the stochastic variable X, that is the ith row of the matrix $\boldsymbol{X}$. So, if β_τ^* is the solution of (2.4) then the $\tau-$quantile of the response variable Y at an *unseen* input $x \in \mathbb{R}^n$ can be simply estimate by

$$\mathcal{Q}_Y(\tau) = x^\mathsf{T}\beta_\tau^*.$$

From now on, by quantile regression we mean quantile regression using a linear model.

Remark 2.1.1. *In the input-output vector space $\mathbb{R}^n \times \mathbb{R}$, $\mathcal{Q}_Y(\tau)$ is a hyperplane with coefficients defined by β_τ^*. In case $n = 2$, the more τ close to one, the more training points lies under the quantile hyperplane.*

2.2 From quantile regression to linear programming

In this section we will show that the problem (2.4) is essentially equivalent to a standard LP. By rewriting a quantile regression problem as an LP, we simultaneously get rid of the ambiguous in

definition of quantile loss function and are able to solve the problem efficiently using simplex or interior point methods.

Firstly, by definition of quantile loss function, we have

$$\begin{aligned}\sum_{i=1}^{m} \ell_\tau(y_i - x_i^\mathsf{T}\beta_\tau) &= \sum_{i=1}^{m} (y_i - x_i^\mathsf{T}\beta_\tau)\Big(\tau - \mathbb{I}_{\{y_i - x_i^\mathsf{T}\beta_\tau < 0\}}\Big) \\ &= \sum_{i=1}^{m} u_i\big(\tau - \mathbb{I}_{\{u_i<0\}}\big),\end{aligned}$$

where $u_i := y_i - x_i^\mathsf{T}\beta_\tau$ for all $i = 1, 2, \ldots, n$. Therefore, the problem (2.4) can be rewritten as

$$\begin{cases} \text{minimize } \sum_{i=1}^{m}\Big(\tau u_i^+ + (1-\tau)u_i^-\Big) \\ \text{subject to: } \quad y_i - x_i^\mathsf{T}\beta_\tau = u_i^+ - u_i^-, \\ \qquad\qquad\quad \beta_\tau \in \mathbb{R}^n, \\ \qquad\qquad\quad u_i^+, u_i^- \in \mathbb{R}, \quad u_i^+, u_i^- \geqslant 0 \quad \forall i = 1, 2, \ldots, n, \end{cases}$$

here, we expressed u_i as the difference of two non-negative variables

$$u_i^+ = \max\{u_i, 0\} \text{ and } u_i^- = -\min\{u_i, 0\}.$$

To formulate this problem to a standard LP, we again split β_τ into two non-negative variables as we did with u_i's:

$$\beta_\tau = \beta_\tau^+ - \beta_\tau^-, \quad \text{with } \beta_\tau^+ = \max\{\beta_\tau, 0\} \text{ and } \beta_\tau^- = -\min\{\beta_\tau, 0\}.$$

Hence, the standard LP for a quantile regression problem is

$$\begin{cases} \text{minimize } \sum_{i=1}^{m}\Big(\tau u_i^+ + (1-\tau)u_i^-\Big) \\ \text{subject to: } \quad x_i^\mathsf{T}\beta_\tau^+ - x_i^\mathsf{T}\beta_\tau^- + u_i^+ - u_i^- = y_i, \\ \qquad\qquad\quad \beta_\tau^+, \beta_\tau^- \in \mathbb{R}^n, \quad \beta_\tau^+, \beta_\tau^- \geqslant 0, \\ \qquad\qquad\quad u_i^+, u_i^- \in \mathbb{R}, \quad u_i^+, u_i^- \geqslant 0 \quad \forall i = 1, 2, \ldots, n. \end{cases}$$

This LP can be written in matrix form as

$$(\boldsymbol{QR}) \qquad \begin{cases} \text{minimize} \quad \tau e^\mathsf{T} u^+ + (1-\tau)e^\mathsf{T} u^- \\ \text{subject to:} \quad \boldsymbol{X}\beta_\tau^+ - \boldsymbol{X}\beta_\tau^- + u^+ - u^- = Y, \\ \qquad\qquad\quad \beta_\tau^+, \beta_\tau^- \in \mathbb{R}^n \quad \beta_\tau^+, \beta_\tau^- \geqslant 0, \\ \qquad\qquad\quad u^+, u^- \in \mathbb{R}^m, \quad u^+, u^- \geqslant 0, \end{cases}$$

where $e = (1, \ldots, 1)^\mathsf{T} \in \mathbb{R}^m$. In this thesis, sometimes we also use "($\boldsymbol{QR}$)" to refer to the non-

standard form:

$$(\boldsymbol{QR}) \qquad \begin{cases} \text{minimize} & \tau e^\mathsf{T} u^+ + (1-\tau) e^\mathsf{T} u^- \\ \text{subject to:} & \boldsymbol{X}\beta_\tau + u^+ - u^- = Y, \\ & \beta_\tau \in \mathbb{R}^n, \\ & u^+, u^- \in \mathbb{R}^m, \quad u^+, u^- \geqslant 0, \end{cases}$$

in which, we keep the unknown parameters unsigned variables. As mentioned in Section 1.3, in the next chapters, we will consider multiple kinds of the training sets, and depending on the circumstances, the notations in (***QR***) could be changed, we will note clearly in those cases.

2.3 Solving multiple quantiles

Because of the fact that for each quantile the loss function is different, to find multiple quantiles of a given training set, we have to solve multiple quantile regression problems: each problem for each quantile. However, as shown above, solving a quantile regression problem is equivalent to solving an LP. It suggests that we can combine multiple quantile problems into a block structured LP. Actually, they are equivalent, provided there is no constraints on the quantiles, *i.e.*, the blocks are completely separated. That could be useful if the solver prefers solving a big block structured LP to doing a loop over the set of quantiles and solving the problems one by one, especially when the modeling languages take a considerable amount of time to create the models for each problem.

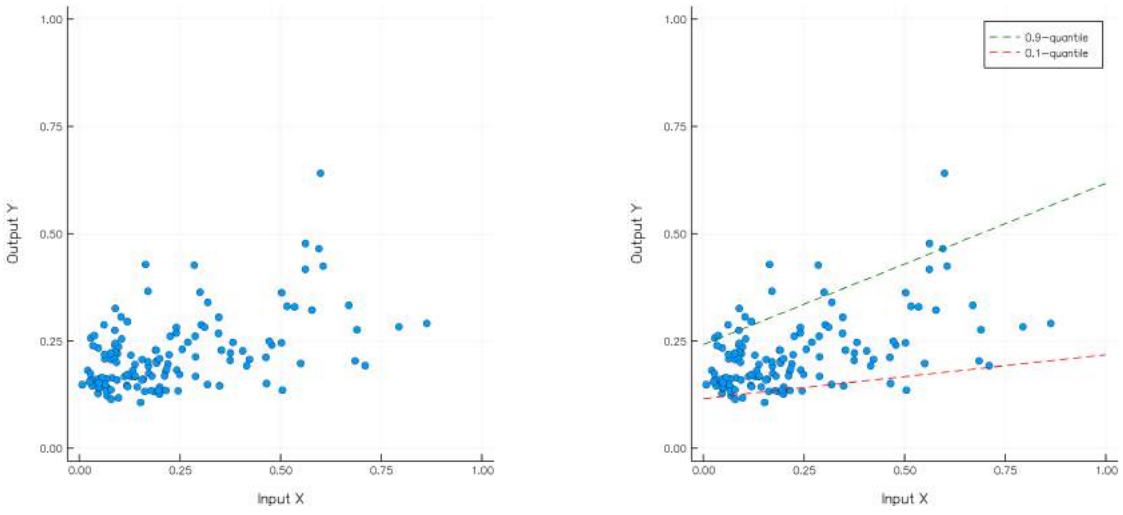
Let $X = (x_1, x_2, \ldots, x_m) \in \mathbb{R}^m$ be the vector of inputs and $Y = (y_1, y_2, \ldots, y_m) \in \mathbb{R}^m$ be the corresponding outputs vector. Each pair (x_i, y_i) forms an individual point in input-output space, see Figure 2.2a. Suppose the τ-quantile of Y is linear in X, *i.e.*,

$$\mathcal{Q}_Y(\tau) = a_\tau X + b_\tau,$$

with $a, b \in \mathbb{R}$ are the unknown slope and intercept, respectively, of the τ-quantile line. Note that in the context of Section 1.3 and Section 2.1, the input matrix $\boldsymbol{X}$ is $(X, 1) \in \mathbb{R}^{m\times 2}$ and the coefficient vector β_τ is $(a_\tau, b_\tau) \in \mathbb{R}^2$. Substituting to (***QR***), we obtain the following LP:

$$\begin{cases} \text{minimize} & \tau e^\mathsf{T} u_\tau^+ + (1-\tau) e^\mathsf{T} u_\tau^- \\ \text{subject to:} & a_\tau X + b_\tau e + u_\tau^+ - u_\tau^- = Y, \\ & a_\tau, b_\tau \in \mathbb{R}, \\ & u_\tau^+, u_\tau^- \in \mathbb{R}^m, \quad u_\tau^+, u_\tau^- \geqslant 0, \end{cases}$$

where, $e = (1, \ldots, 1)^\mathsf{T} \in \mathbb{R}^m$. Here, we indexed τ to the variables u's to emphasize that those variables are associated with the quantile τ. Finally, implementing this LP for each τ in $\{0.1, 0.9\}$ on a training set with $m = 100$ we obtain the result as shown in Figure 2.2b.



(a) (b)

Figure 2.2: Example of solving a quantile regression problem.

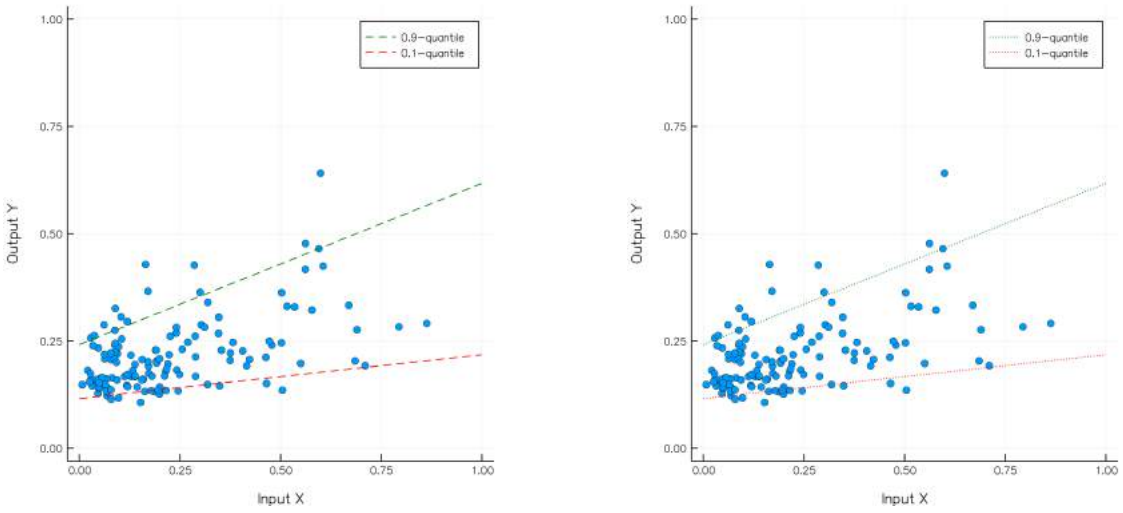
Now, on the same training set, we will try to solve the 0.1-quantile and the 0.9-quantile at the same time. To do that, we simply put together two problems into one LP as:

$$
\left\{
\begin{array}{ll}
\text{minimize} & 0.1e^{\mathsf{T}}u_{0.1}^{+} + (1-0.1)e^{\mathsf{T}}u_{0.1}^{-} + 0.9e^{\mathsf{T}}u_{0.9}^{+} + (1-0.9)e^{\mathsf{T}}u_{0.9}^{-} \\
\text{subject to:} & a_{0.1}X + b_{0.1}e + u_{0.1}^{+} - u_{0.1}^{-} = Y, \\
& a_{0.1}, b_{0.1} \in \mathbb{R}, \\
& u_{0.1}^{+}, u_{0.1}^{-} \in \mathbb{R}^m, \quad u_{0.1}^{+}, u_{0.1}^{-} \geqslant 0, \\
& a_{0.9}X + b_{0.9}e + u_{0.9}^{+} - u_{0.9}^{-} = Y, \\
& a_{0.9}, b_{0.9} \in \mathbb{R}, \\
& u_{0.9}^{+}, u_{0.9}^{-} \in \mathbb{R}^m, \quad u_{0.9}^{+}, u_{0.9}^{-} \geqslant 0.
\end{array}
\right. \tag{2.5}
$$

Clearly, this LP has a blocks structure: each block for each quantile. As we can see, the obtained quantile lines are the same with the ones solved separately, see Figure 2.3.

2.4 Constrained quantile regression

As discussed in the previous section, if the problem is just finding the quantiles on a same training set then it is equivalent to solve the quantiles separately or simultaneously. However, if we add some conditions involving the quantiles, *e.g.*, parallel condition, then it is necessary to combine the quantile problems and solve them at the same time with additional constraints on the parameters, *i.e.*, the coefficients of the quantile hyperplane. Follows are two examples of constrained quantile regression problems that could be faced in applications.



(a) Quantile lines solved separately.

(b) Quantile lines solved simultaneously.

Figure 2.3: Example of solving multiple quantiles at the same time.

2.4.1 Example 1: parallel constraint

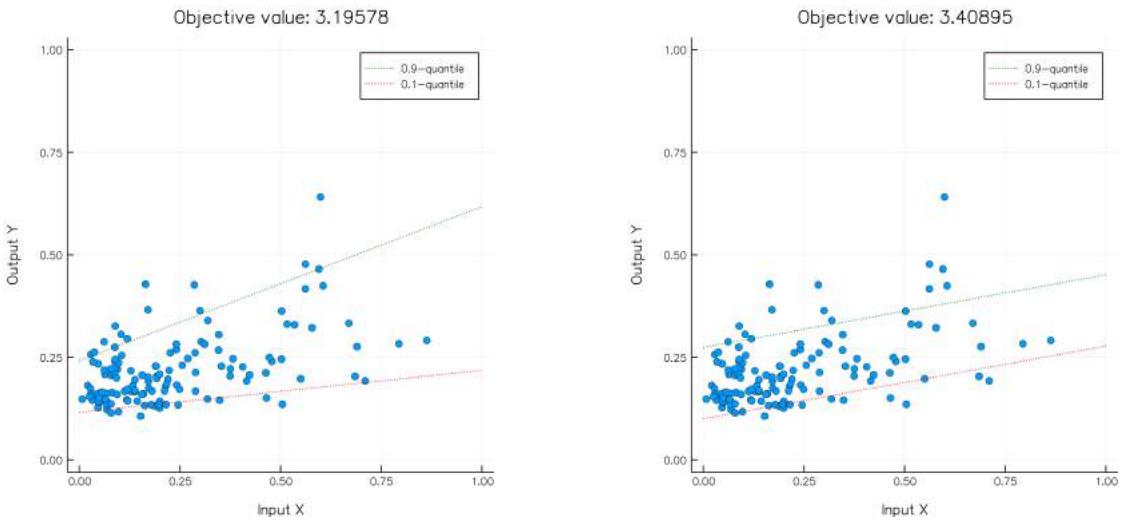
Sometimes, as that we will see in Chapter 3, it could useful to impose the quantile hyperplanes to be parallel. In this example, we will try to find the 0.1-quantile and 0.9-quantile on the same data as Section 2.3 but with constraint that the 0.1-quantile has to be parallel with the 0.9-quantile. It means we will reuse the LP (2.5) and add to it a "parallel constraint". This could be done by impose the hyperplanes to have the same normal vector, in this case, it is

$$a_{0.1} = a_{0.9}.$$

Thus, our LP will be:

$$\text{(2.6)} \quad \begin{cases} \text{minimize} & 0.1e^\mathsf{T}u_{0.1}^+ + (1-0.1)e^\mathsf{T}u_{0.1}^- + 0.9e^\mathsf{T}u_{0.9}^+ + (1-0.9)e^\mathsf{T}u_{0.9}^- \\ \text{subject to:} & a_{0.1}X + b_{0.1} + u_{0.1}^+ - u_{0.1}^- = Y, \\ & a_{0.1}, b_{0.1} \in \mathbb{R}, \\ & u_{0.1}^+, u_{0.1}^- \in \mathbb{R}^m, \quad u_{0.1}^+, u_{0.1}^- \geqslant 0, \\ & a_{0.9}X + b_{0.9} + u_{0.9}^+ - u_{0.9}^- = Y, \\ & a_{0.9}, b_{0.9} \in \mathbb{R}, \\ & u_{0.9}^+, u_{0.9}^- \in \mathbb{R}^m, \quad u_{0.9}^+, u_{0.9}^- \geqslant 0, \\ & a_{0.1} = a_{0.9}. \end{cases}$$

The Figure 2.4.1 shows us the result of solving (2.6) comparing with (2.5). As we expect, the parallel constraint leads to the increase in the objective value.



(a) Quantile lines obtained by solving (2.5).

(b) Quantile lines obtained by solving (2.6)

Figure 2.4: Example of solving block QR imposing parallel constraint

2.4.2 Example 2: non-crossing constraint

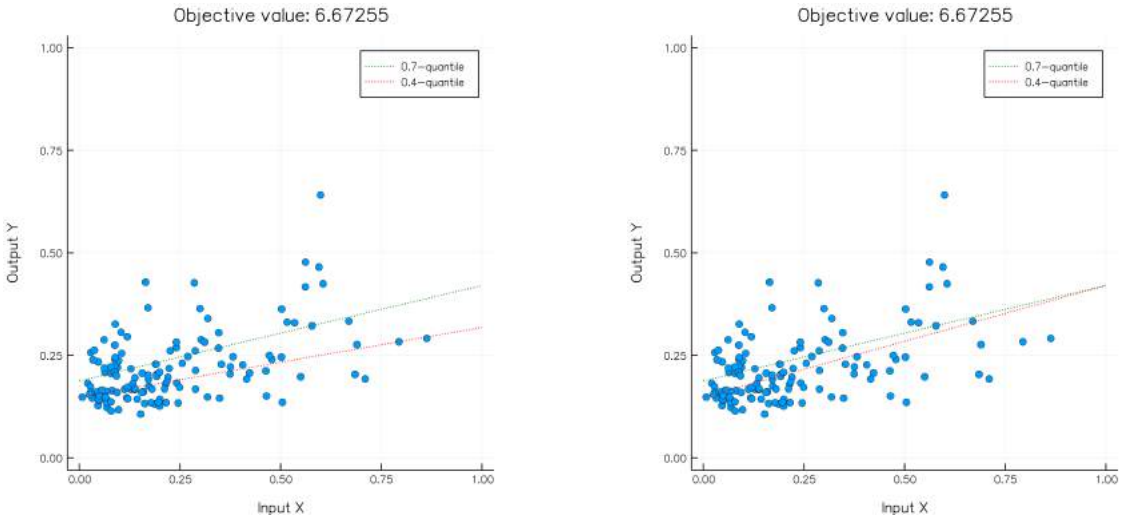
We known that, all the real-valued sample must satisfy the non crossing condition: the smaller quantiles must be "under" the bigger ones.

In this example, we try to solve at the same time the 0.4-quantile and the 0.7-quantile imposing they do not cross each other. This condition can be straightforwardly formulated by imposing

$$a_{0.4}\min(X) + b_{0.4} \leqslant a_{0.7}\min(X) + b_{0.7}, \tag{2.7}$$

$$a_{0.4}\max(X) + b_{0.4} \leqslant a_{0.7}\max(X) + b_{0.7}. \tag{2.8}$$

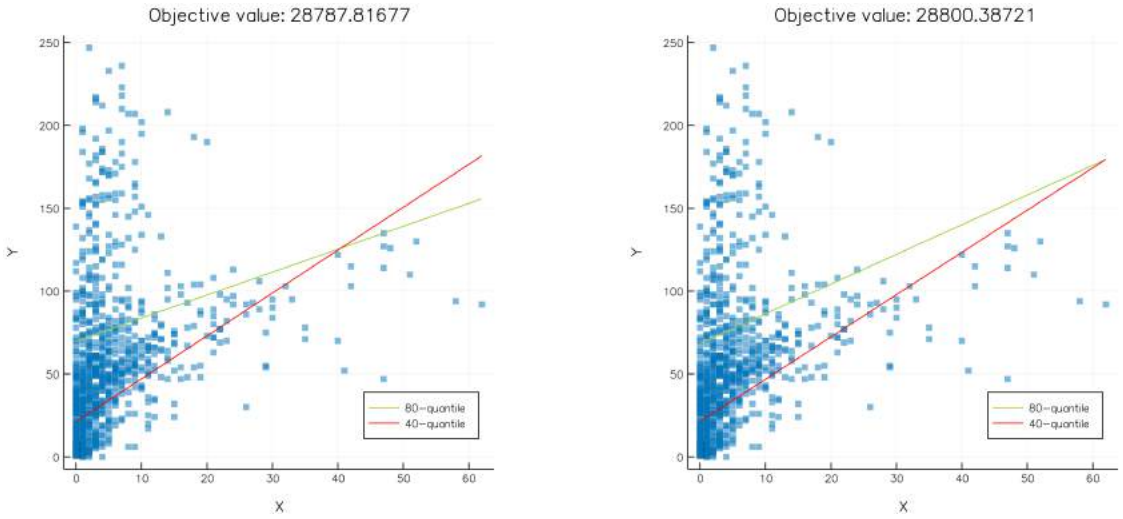
Adding (2.7) and (2.8) to (2.5), we obtain a new LP. Implementing this LP we got the result as in Figure 2.5 and 2.6



(a) No non-crossing constraint.

(b) Non-crossing constraint.

Figure 2.5: Example of solving block QR imposing non-crossing constraint.



(a) No non-crossing constraint.

(b) Non-crossing constraint.

Figure 2.6: Another example of solving block QR imposing non-crossing constraint with the last two columns of bike sharing dataset (see Chapter 3).

We see that, imposing non crossing constraint indeed potentially returns a different optimal solution. In the second example, although the objective value is increasing, imposing non-crossing constraint give a more reasonable result.

Chapter 3

Generating quantile invariant data

Contents

3.1 Motivation

In statistics, correlation mostly used when we are interested in how close two variables are to having a linear relationship with each other. Correlations are useful because they can indicate a predictive relationship that can be exploited in practice. For example, an electricity demand may produce less power on a mild day based on the correlation between electricity demand and weather. In this example, there is a causal relationship, because extreme weather causes people to use more electricity for heating or cooling. In recent years, quantile regression has been used in a spatial context, as it allows the effects of variables on the response to vary across quantiles as well as spatially.

Given a dataset, *i.e.*, a matrix, in many applications, it could be useful to capture the spatial correlation, *i.e.*, the correlations between the columns of this dataset. Here, *capture* means that we should be able to generate new dataset which relatively has the same spatial correlation with

the original data. As a part of this study, in this chapter, we focuses on formulating a workflow using quantile regression that helps to generate *quantile invariant* data. This means our workflow will approximately preserve the quantiles from the original dataset to the new generated one.

3.2 A workflow for generating quantile invariant data

3.2.1 Formulation

As mentioned earlier, in this chapter, the target $\boldsymbol{Y}$ is a matrix, so the coefficients in this model are also matrices. For convenience, we will write, for example, $n \in N$ instead of $n \in \{1, \dots, N\}$. Let $\boldsymbol{Y} \in \mathbb{R}^{H \times N}$ be a matrix containing H historical observations of N stochastic variables $Y_1, Y_2, \dots, Y_N$. Let $\boldsymbol{X} \in \mathbb{R}^{P \times N}$ be the set of *explicating data* and $\boldsymbol{A} \in \mathbb{R}^{H \times P}, b \in \mathbb{R}^H$ be the unknown parameters. We consider the following model

$$\boldsymbol{Y} = \boldsymbol{A} \cdot \boldsymbol{X} + b \cdot e^{\mathsf{T}}, \tag{3.1}$$

in which, $e = (1, \dots, 1)^{\mathsf{T}} \in \mathbb{R}^N$. As discussed above, the question is how to find $\boldsymbol{A}$ and b such that the new data generated by model (3.1) in some senses have the same distribution as the original data. The idea here is to characterize the coordinate of each point in $\boldsymbol{Y}$ via *quantile coefficients*. More precisely, these coefficients (obtained by solving a related quantile regression problem) are able to generate new data from the explicating data, in which the new points will surround the original point.

First of all, let us perform PCA on the original dataset $\boldsymbol{Y}$ as follows. We compute the covariance matrix of $\boldsymbol{Y}$ as

$$M = \begin{bmatrix} \operatorname{var}(Y_1) & \operatorname{cov}(Y_1, Y_2) & \dots & \operatorname{cov}(Y_1, Y_N) \\ \operatorname{cov}(Y_2, Y_1) & \operatorname{var}(Y_2) & \dots & \operatorname{cov}(Y_2, Y_N) \\ \vdots & \vdots & \ddots & \vdots \\ \operatorname{cov}(Y_N, Y_1) & \operatorname{cov}(Y_N, Y_2) & \dots & \operatorname{var}(Y_N) \end{bmatrix}.$$

We denote by $\lambda_1, \lambda_2, \dots, \lambda_N$ the eigenvalues and by $V_1, V_2, \dots, V_N$ the eigenvectors of M. Additionally, we assume that $\lambda_1 \geqslant \lambda_2 \geqslant \dots \geqslant \lambda_N$. The eigenvectors form an orthonormal basis for the space of the observed data: in this basis, the largest eigenvalues correspond to the principal components that are associated with the most of the covariability among a number of observed data. Next, we choose P ($P \ll N$) eigenvectors which are corresponding to the P largest eigenvalues and project $\boldsymbol{Y}$ onto them by performing

$$\hat{\boldsymbol{X}} = \boldsymbol{Y}\boldsymbol{V},$$

here, $\boldsymbol{V} = (V_1, V_2, \ldots, V_P) \in \mathbb{R}^{N\times P}$ is the basic of the PCs space and $\hat{\boldsymbol{X}} \in \mathbb{R}^{H\times P}$ denotes the projection.

Next, for each column of $\hat{\boldsymbol{X}}$, we find the minimum and maximum values and then obtain a *hyper-rectangle*, which is

$$\mathbb{X} := [\min \hat{X}_1, \max \hat{X}_1] \times \ldots \times [\min \hat{X}_P, \max \hat{X}_P]. \tag{3.2}$$

Because of the idea described above, we will have in total H sub-models: each one for each (original) data points. In order to infer the coefficients, we need training data which at the same time well characterize the original data and are easy to sampled. Hence, we propose to sample the explicating data $\boldsymbol{X}$ as follows. For each $p \in P$, let sample randomly T point $X_{p,t} \in \mathbb{R}^N, t \in T$ from the hyper-cube

$$[\min \hat{X}_p, \max \hat{X}_p] \underset{N \text{ times}}{\times} \ldots \times [\min \hat{X}_p, \max \hat{X}_p].$$

In our experiment, T in range $4, \ldots, 10$ is the good choices, *i.e.*, we will have $4, \ldots, 10$ training points for each (A_i, b_i).

Now, given a set of quantiles $\mathcal{T}$, *e.g.*, $\mathcal{T} = \{0.1, 0.9\}$, for each $h \in H$ the problem under consideration is:

$$\begin{cases} \text{minimize} & \sum_{\tau\in\mathcal{T}} \sum_{n\in N} \sum_{t\in T} \left(\tau\varepsilon^+_{h,t,n,\tau} + (1-\tau)\varepsilon^-_{h,t,n,\tau}\right) \\ \text{subject to:} & a_h^\mathsf{T} X_{t,n} + b_h + \varepsilon^+_{h,t,n,\tau} - \varepsilon^-_{h,t,n,\tau} = Y_{h,n}, \\ & \varepsilon^+_{h,t,n,\tau}, \varepsilon^-_{h,t,n,\tau} \in \mathbb{R}_+, \quad \forall t \in T, \quad \forall n \in N, \quad \forall \tau \in \mathcal{T} \\ & a_h \in \mathbb{R}^P, b_h \in \mathbb{R}, \end{cases}$$

in which $X_{t,n} \in \mathbb{R}^P$ and the variables $a_h \in \mathbb{R}^P, b_h \in \mathbb{R}$ are the seeking coefficients for the data point y_h^T. As we saw in Chapter 2, solving these LPs for each $h \in H$ is indeed the same as solving the "big" LP:

$$\begin{cases} \text{minimize} & \sum_{h\in H} \sum_{\tau\in\mathcal{T}} \sum_{n\in N} \sum_{t\in T} \left(\tau\varepsilon^+_{h,t,n,\tau} + (1-\tau)\varepsilon^-_{h,t,n,\tau}\right) \\ \text{subject to:} & a_h^\mathsf{T} X_{t,n} + b_h + \varepsilon^+_{h,t,n,\tau} - \varepsilon^-_{h,t,n,\tau} = Y_{h,n}, \\ & \varepsilon^+_{h,t,n,\tau}, \varepsilon^-_{h,t,n,\tau} \in \mathbb{R}_+, \quad \forall t \in T, \quad \forall n \in N, \quad \forall \tau \in \mathcal{T}, \quad \forall h \in H \end{cases}$$

which can be rewrite in matrix form as

$$\begin{cases} \text{minimize} & \sum_{\tau\in\mathcal{T}} \sum_{t\in T} \left(\tau\boldsymbol{\varepsilon}^+_{t,\tau} + (1-\tau)\boldsymbol{\varepsilon}^-_{t,\tau}\right) \\ \text{subject to:} & \boldsymbol{A} \cdot \boldsymbol{X}_t + b \cdot e^\mathsf{T} + \boldsymbol{\varepsilon}^+_{t,\tau} - \boldsymbol{\varepsilon}^-_{t,\tau} = \boldsymbol{Y}, \\ & \boldsymbol{\varepsilon}^+_{t,\tau}, \boldsymbol{\varepsilon}^-_{t,\tau} \in \mathbb{R}_+^{H\times N}, \quad \forall t \in T, \quad \forall \tau \in \mathcal{T}, \end{cases} \tag{3.3}$$

where $\boldsymbol{X}_t^\mathsf{T} \in \mathbb{X}^N$ with $\mathbb{X}$ defined in (3.4). Recall that, in the above LP, $\boldsymbol{A}, b, \boldsymbol{\varepsilon}^+$ and $\boldsymbol{\varepsilon}^-$ are the decision variables matrices.

3.2.2 Solving the LP, generating new data and validating the model

The next step of the workflow is to solve the LP (3.3). Let us have a look at the number of variables and constraints. Firstly, we have HP (H times P) variables in $\boldsymbol{A}$ and H variables in b. For each $\tau \in \mathcal{T}$, we also have HNT variables in ε_τ^+ and HNT variables in ε_τ^-. About the number of constraints, for each $\tau \in \mathcal{T}, t \in T$, we have HP equality constraints and $2HN$ inequality constraints. So, in general, this LP is supposed to be very large.

Assume that $(\boldsymbol{A}_*, b_*)$ is the optimal solution of $(\boldsymbol{QR_T})$, then the next step is to generate new data using that $(\boldsymbol{A}_*, b_*)$. To do that, we simply sample new explicating data, say $\boldsymbol{Z}$, as the same way as we sampled the matrix $\boldsymbol{X}_t$'s, then the new data will be generated by

$$\boldsymbol{Y}_* = \boldsymbol{A}_* \cdot \boldsymbol{Z} + b_* \cdot e^{\mathsf{T}}.$$

The last step is to validate the model by comparing the quantiles of the columns of $\boldsymbol{Y}_*$ versus the quantiles of the columns of $\boldsymbol{Y}$.

3.3 Numerical experiments

3.3.1 An energy dataset

We first test our workflow on a dataset $\boldsymbol{Y}$ of size 5000×30 which comes from energy industry. See Table 3.1 and 3.2 for the comparisons between the quantiles of the original dataset (oriQ's) and the quantiles of the generated dataset (newQ's). In the tables, Q1, Q2, Q3, Q4, Q5 are, respectively, the 0.05-quantile, 0.25-quantile, 0.75-quantile, 0.9-quantile and 0.95-quantile. The dark areas mean that the biases between the oriQ's and the newQ's are less than 10% in comparing with the 90% volume of the original data (for each column). More precisely, we marked the pairs (oriQx,newQx) dark if

$$\frac{|\mathsf{newQx} - \mathsf{oriQx}|}{|\mathsf{oriQ5} - \mathsf{oriQ1}|} \leqslant 10\%.$$

Note that each table is obtained as an average over 5 instances of explicating data with the same P and T. In addition, at each time we obtained the optimum $(\boldsymbol{A}_*, b_*)$, the quantiles are the average over 5 explicating data $\boldsymbol{Z}$'s.

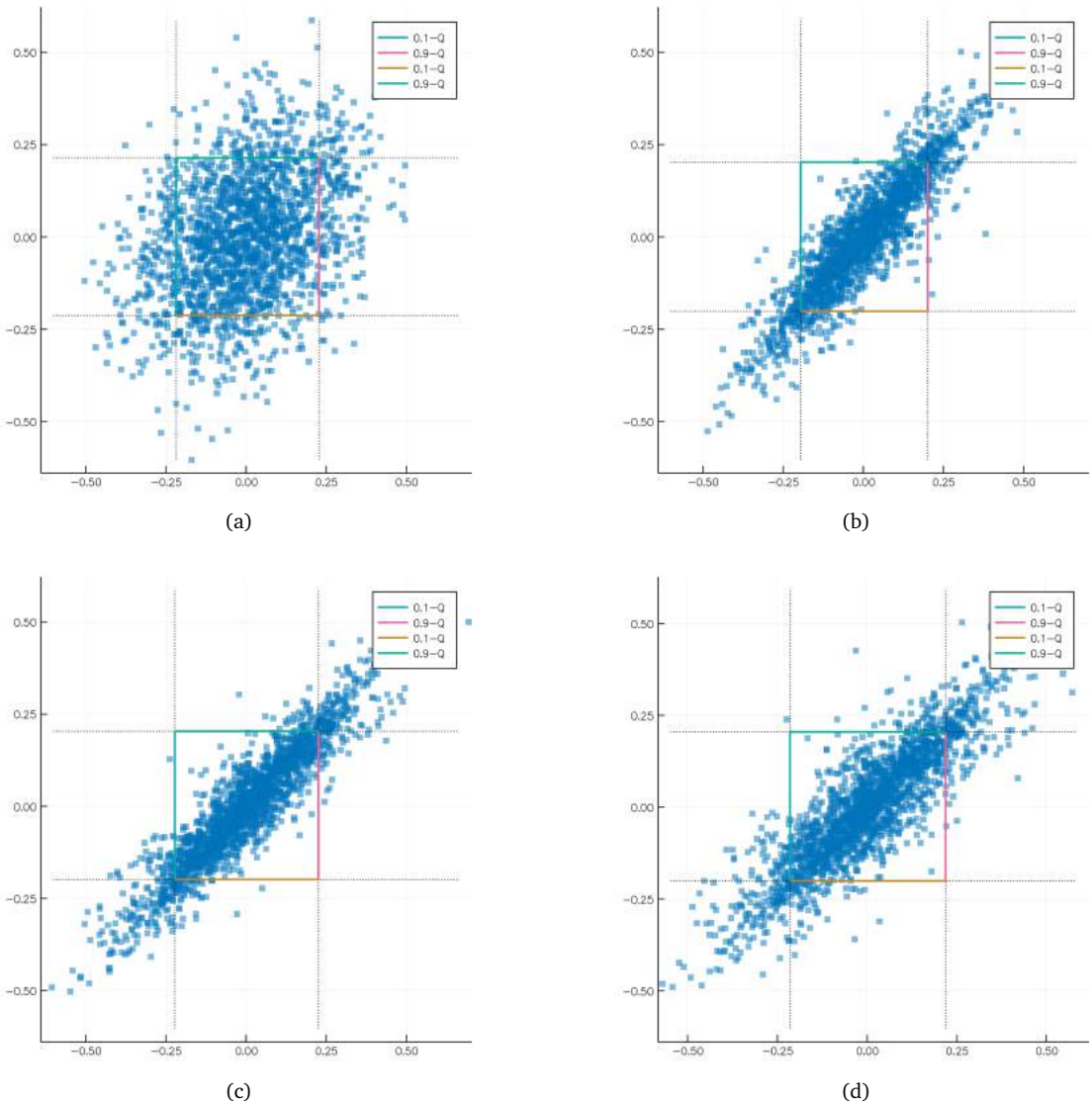
n	oriQ1	newQ1	oriQ2	newQ2	oriQ3	newQ3	oriQ4	newQ4	oriQ5	newQ5	oriQ6	newQ6	oriQ7	newQ7
1	-0.118	-0.083	-0.027	-0.015	0.102	0.091	0.263	0.205	0.414	0.325	0.547	0.428	0.631	0.49
2	-0.138	-0.079	-0.062	-0.012	0.068	0.094	0.2	0.206	0.344	0.327	0.465	0.43	0.539	0.49
3	-0.142	-0.091	-0.07	-0.021	0.015	0.088	0.118	0.203	0.209	0.325	0.3	0.432	0.35	0.49
4	-0.201	-0.092	-0.103	-0.022	0.047	0.087	0.219	0.205	0.387	0.327	0.525	0.435	0.637	0.50
5	-0.212	-0.092	-0.099	-0.024	0.075	0.089	0.273	0.207	0.488	0.33	0.664	0.438	0.773	0.50
6	-0.162	-0.094	-0.065	-0.025	0.087	0.087	0.283	0.206	0.455	0.331	0.623	0.439	0.714	0.50
7	-0.139	-0.087	-0.052	-0.018	0.089	0.09	0.241	0.207	0.407	0.329	0.551	0.436	0.638	0.50
8	-0.186	-0.09	-0.089	-0.022	0.063	0.088	0.22	0.206	0.376	0.328	0.522	0.435	0.612	0.50
9	-0.154	-0.094	-0.072	-0.024	0.032	0.087	0.153	0.205	0.278	0.329	0.386	0.438	0.458	0.50
10	-0.2	-0.097	-0.105	-0.026	0.051	0.085	0.223	0.203	0.414	0.327	0.576	0.435	0.676	0.50
11	-0.164	-0.097	-0.081	-0.027	0.046	0.086	0.208	0.205	0.362	0.329	0.492	0.437	0.584	0.50
12	-0.067	-0.09	0.016	-0.022	0.138	0.087	0.286	0.204	0.441	0.325	0.575	0.432	0.65	0.49
13	-0.157	-0.09	-0.091	-0.021	0.017	0.088	0.152	0.205	0.275	0.326	0.384	0.434	0.445	0.49
14	-0.025	-0.095	0.031	-0.026	0.134	0.085	0.249	0.204	0.355	0.327	0.477	0.436	0.533	0.50
15	-0.131	-0.093	-0.056	-0.023	0.052	0.088	0.172	0.206	0.297	0.329	0.402	0.438	0.469	0.50
16	-0.142	-0.095	-0.062	-0.026	0.079	0.086	0.222	0.204	0.374	0.328	0.51	0.438	0.602	0.50
17	-0.149	-0.087	-0.058	-0.019	0.075	0.091	0.232	0.207	0.374	0.329	0.505	0.436	0.603	0.50
18	-0.126	-0.092	-0.056	-0.021	0.069	0.09	0.19	0.206	0.316	0.33	0.437	0.438	0.5	0.50
19	-0.173	-0.089	-0.098	-0.021	0.017	0.089	0.145	0.204	0.275	0.326	0.378	0.434	0.458	0.
20	-0.134	-0.093	-0.054	-0.023	0.068	0.086	0.211	0.203	0.342	0.325	0.468	0.431	0.533	0.49
21	-0.15	-0.094	-0.072	-0.023	0.061	0.087	0.211	0.207	0.357	0.332	0.492	0.442	0.56	0.51
22	-0.15	-0.097	-0.078	-0.025	0.044	0.087	0.175	0.206	0.298	0.331	0.414	0.441	0.474	0.50
23	-0.174	-0.093	-0.075	-0.022	0.07	0.087	0.242	0.205	0.416	0.328	0.55	0.434	0.64	0.50
24	-0.15	-0.093	-0.064	-0.023	0.073	0.089	0.231	0.206	0.382	0.33	0.526	0.438	0.611	0.50
25	-0.16	-0.098	-0.076	-0.026	0.073	0.087	0.251	0.206	0.429	0.33	0.574	0.441	0.661	0.50
26	-0.092	-0.088	-0.036	-0.019	0.061	0.091	0.174	0.206	0.294	0.328	0.39	0.435	0.451	0.50
27	-0.18	-0.088	-0.101	-0.019	0.055	0.09	0.22	0.206	0.387	0.329	0.524	0.435	0.617	0.50
28	-0.233	-0.096	-0.122	-0.025	0.037	0.087	0.214	0.206	0.394	0.331	0.555	0.44	0.652	0.50
29	-0.144	-0.094	-0.072	-0.024	0.04	0.087	0.17	0.205	0.29	0.328	0.399	0.436	0.464	0.50
30	-0.214	-0.094	-0.118	-0.023	0.072	0.086	0.269	0.204	0.464	0.327	0.63	0.436	0.744	0.50

Table 3.1: Energy dataset. Implementing the workflow with $P = 3, T = 5$. Comparing with 10% of the distance from oriQ1 to oriQ5.

n	oriQ1	newQ1	oriQ2	newQ2	oriQ3	newQ3	oriQ4	newQ4	oriQ5	newQ5	oriQ6	newQ6	oriQ7	newQ7
1	-0.118	-0.1	-0.027	-0.031	0.102	0.08	0.263	0.204	0.414	0.33	0.547	0.439	0.631	0.50
2	-0.138	-0.103	-0.062	-0.031	0.068	0.081	0.2	0.204	0.344	0.331	0.465	0.441	0.539	0.5
3	-0.142	-0.099	-0.07	-0.028	0.015	0.086	0.118	0.209	0.209	0.336	0.3	0.448	0.35	0.51
4	-0.201	-0.095	-0.103	-0.025	0.047	0.084	0.219	0.206	0.387	0.329	0.525	0.438	0.637	0.50
5	-0.212	-0.099	-0.099	-0.03	0.075	0.083	0.273	0.204	0.488	0.33	0.664	0.442	0.773	0.50
6	-0.162	-0.098	-0.065	-0.028	0.087	0.081	0.283	0.204	0.455	0.328	0.623	0.436	0.714	0.50
7	-0.139	-0.101	-0.052	-0.032	0.089	0.08	0.241	0.201	0.407	0.326	0.551	0.435	0.638	0.50
8	-0.186	-0.093	-0.089	-0.026	0.063	0.082	0.22	0.202	0.376	0.324	0.522	0.431	0.612	0.49
9	-0.154	-0.097	-0.072	-0.028	0.032	0.082	0.153	0.204	0.278	0.328	0.386	0.435	0.458	0.50
10	-0.2	-0.096	-0.105	-0.028	0.051	0.082	0.223	0.202	0.414	0.326	0.576	0.435	0.676	0.50
11	-0.164	-0.098	-0.081	-0.027	0.046	0.086	0.208	0.21	0.362	0.336	0.492	0.447	0.584	0.51
12	-0.067	-0.098	0.016	-0.029	0.138	0.08	0.286	0.202	0.441	0.327	0.575	0.436	0.65	0.50
13	-0.157	-0.104	-0.091	-0.034	0.017	0.079	0.152	0.202	0.275	0.329	0.384	0.44	0.445	0.5
14	-0.025	-0.093	0.031	-0.023	0.134	0.085	0.249	0.205	0.355	0.329	0.477	0.436	0.533	0.50
15	-0.131	-0.1	-0.056	-0.028	0.052	0.081	0.172	0.205	0.297	0.33	0.402	0.44	0.469	0.50
16	-0.142	-0.106	-0.062	-0.035	0.079	0.076	0.222	0.199	0.374	0.326	0.51	0.436	0.602	0.50
17	-0.149	-0.098	-0.058	-0.027	0.075	0.083	0.232	0.204	0.374	0.328	0.505	0.436	0.603	0.50
18	-0.126	-0.096	-0.056	-0.027	0.069	0.083	0.19	0.204	0.316	0.329	0.437	0.438	0.5	0.50
19	-0.173	-0.093	-0.098	-0.024	0.017	0.085	0.145	0.203	0.275	0.327	0.378	0.434	0.458	0.
20	-0.134	-0.096	-0.054	-0.026	0.068	0.084	0.211	0.204	0.342	0.328	0.468	0.437	0.533	0.50
21	-0.15	-0.097	-0.072	-0.028	0.061	0.082	0.211	0.203	0.357	0.325	0.492	0.434	0.56	0.50
22	-0.15	-0.094	-0.078	-0.025	0.044	0.085	0.175	0.207	0.298	0.332	0.414	0.443	0.474	0.50
23	-0.174	-0.097	-0.075	-0.028	0.07	0.084	0.242	0.205	0.416	0.329	0.55	0.438	0.64	0.50
24	-0.15	-0.101	-0.064	-0.031	0.073	0.08	0.231	0.2	0.382	0.324	0.526	0.432	0.611	0.49
25	-0.16	-0.103	-0.076	-0.032	0.073	0.076	0.251	0.199	0.429	0.323	0.574	0.432	0.661	0.49
26	-0.092	-0.098	-0.036	-0.027	0.061	0.085	0.174	0.206	0.294	0.332	0.39	0.443	0.451	0.51
27	-0.18	-0.099	-0.101	-0.029	0.055	0.081	0.22	0.202	0.387	0.326	0.524	0.434	0.617	0.50
28	-0.233	-0.089	-0.122	-0.022	0.037	0.086	0.214	0.204	0.394	0.327	0.555	0.434	0.652	0.49
29	-0.144	-0.101	-0.072	-0.029	0.04	0.082	0.17	0.207	0.29	0.333	0.399	0.444	0.464	0.51
30	-0.214	-0.101	-0.118	-0.03	0.072	0.08	0.269	0.201	0.464	0.325	0.63	0.433	0.744	0.50

Table 3.2: Energy dataset. Implementing the workflow with $P = 5, T = 5$. Comparing with 10% of the distance from oriQ1 to oriQ5.

We can see from the above tables, the quantiles of the columns, *e.g.*, $n = 20$ and $n = 21$ were quite well preserved. This could be verified by looking at the visualization in Figure 3.1 where we scattered the original data and three randomly generated data[1]. As we can see, the *shapes*[2] of the clusters could vary but the quantiles are almost the same.



(a) (b)

(c) (d)

Figure 3.1: Visualizing the 0.1-quantile and 0.9-quantile of the generated data. The upper-left is the original data, the others are the generated ones.

[1] by randomly generating three $\boldsymbol{Z}$'s and then performing $Y_* = \boldsymbol{A}_* \cdot \boldsymbol{Z} + b_* \cdot e^{\mathsf{T}}$

[2] *i.e.*, the correlations

3.3.2 Bike sharing dataset

Let us try implementing our workflow on another dataset called Bike Sharing[3]. This dataset consists of 17389 rows and 16 columns, contains the hourly and daily count of rental bikes between years 2011 and 2012 in Capital bike share system with the corresponding weather and seasonal information.

In our experiments, we were only interested in the *non-time* columns temporarily, *i.e.*, the last 7 columns. More precisely, we extracted the columns 10,11,12,13 to form a new dataset $\boldsymbol{Y}_1$ and extracted the columns 14,15,16 to form a new dataset $\boldsymbol{Y}_2$, then treated these ones independently. The results are shown in Table 3.3, Table 3.4 (for $\boldsymbol{Y}_1$) and Table 3.5, Table 3.6 (for $\boldsymbol{Y}_2$). Here, a bit different from the comparison for energy dataset, we marked dark if the error is less than or equal to 5%.

[3]available at https://archive.ics.uci.edu/ml/datasets/bike+sharing+dataset

n	oriQ1	newQ1	oriQ2	newQ2	oriQ3	newQ3	oriQ4	newQ4	oriQ5	newQ5	oriQ6	newQ6	oriQ7	newQ7
10	-0.297	-0.415	-0.257	-0.33	-0.157	-0.18	0.003	0.003	0.163	0.173	0.243	0.327	0.303	0.41
11	-0.264	-0.417	-0.233	-0.332	-0.142	-0.182	0.009	0.002	0.145	0.175	0.221	0.336	0.267	0.42
12	-0.317	-0.452	-0.257	-0.356	-0.147	-0.19	0.003	0.002	0.153	0.184	0.253	0.356	0.303	0.45
13	-0.19	-0.454	-0.19	-0.356	-0.086	-0.189	0.004	0.002	0.064	0.185	0.168	0.354	0.228	0.45

Table 3.3: Bike sharing dataset. Approximately preserving the quantiles of low-variant columns with $P = 1, T = 5$.

n	oriQ1	newQ1	oriQ2	newQ2	oriQ3	newQ3	oriQ4	newQ4	oriQ5	newQ5	oriQ6	newQ6	oriQ7	newQ7
10	-0.297	-0.245	-0.257	-0.197	-0.157	-0.109	0.003	0.001	0.163	0.098	0.243	0.172	0.303	0.21
11	-0.264	-0.253	-0.233	-0.203	-0.142	-0.116	0.009	0.001	0.145	0.1	0.221	0.175	0.267	0.2
12	-0.317	-0.277	-0.257	-0.216	-0.147	-0.116	0.003	0.001	0.153	0.097	0.253	0.183	0.303	0.23
13	-0.19	-0.262	-0.19	-0.206	-0.086	-0.113	0.004	-0.001	0.064	0.093	0.168	0.177	0.228	0.22

Table 3.4: Bike sharing dataset. Approximately preserving the quantiles of low-variant columns with $P = 2, T = 5$.

n	oriQ1	newQ1	oriQ2	newQ2	oriQ3	newQ3	oriQ4	newQ4	oriQ5	newQ5	oriQ6	newQ6	oriQ7	newQ7
14	-35.676	-144.273	-34.676	-139.785	-31.676	-112.548	-18.676	-32.876	12.324	71.531	56.324	213.008	102.424	324.83
15	-149.787	-144.334	-146.787	-139.273	-119.787	-111.188	-38.787	-31.658	66.213	72.523	200.213	215.092	311.213	332.6
16	-184.463	-163.862	-180.463	-159.464	-149.463	-129.031	-47.463	-38.723	91.537	79.36	261.737	236.344	373.637	366.97

Table 3.5: Bike sharing dataset. Approximately preserving the quantiles of high-variant columns with $P = 2, T = 5$.

n	oriQ1	newQ1	oriQ2	newQ2	oriQ3	newQ3	oriQ4	newQ4	oriQ5	newQ5	oriQ6	newQ6	oriQ7	newQ7
14	-35.676	-148.277	-34.676	-145.255	-31.676	-119.045	-18.676	-35.348	12.324	71.638	56.324	203.577	102.424	306.10
15	-149.787	-154.033	-146.787	-150.897	-119.787	-123.655	-38.787	-36.688	66.213	74.749	200.213	211.471	311.213	317.85
16	-184.463	-156.093	-180.463	-152.926	-149.463	-125.319	-47.463	-37.325	91.537	75.534	261.737	214.559	373.637	322.36

Table 3.6: Here, $P = 1$ and $T = 4$. We can see, the quantiles were preserved very well even when we took only 1 pricinpal vector. It could be understood by looking at the eigenvalues of the covariance matrix: 56024.7 $\gg$ 2216.75 $\gg$ 6.33578e-12.

3.4 Another idea

Before closing this chapter, in this informal section we would present another potential model which we are partly studying to generate quantile invariant data.

Let $\boldsymbol{Y} \in \mathbb{R}^{H\times N}$ be a matrix containing H historical observations of N stochastic variables $Y_1, Y_2, \ldots, Y_N$. Let $\boldsymbol{X} \in \mathbb{R}^{H\times P}$ be the set of *explicating data* and $\boldsymbol{A} \in \mathbb{R}^{P\times N}, b \in \mathbb{R}^N$ be the unknown parameters. The quantile regression problem associated with the quantile τ is given by

$$(\boldsymbol{QR_\tau}) \qquad \begin{cases} \text{minimize} & f_\tau(\boldsymbol{\varepsilon}_\tau^+, \boldsymbol{\varepsilon}_\tau^-) \\ \text{subject to:} & \boldsymbol{Y} = \boldsymbol{X}_\tau \cdot \boldsymbol{A}_\tau + e \cdot b_\tau^\mathsf{T} + \boldsymbol{\varepsilon}_\tau^+ - \boldsymbol{\varepsilon}_\tau^-, \\ & \boldsymbol{\varepsilon}_\tau^+, \boldsymbol{\varepsilon}_\tau^- \geqslant 0, \quad \boldsymbol{\varepsilon}_\tau^+, \boldsymbol{\varepsilon}_\tau^- \in \mathbb{R}^{H\times N}, \end{cases}$$

in which, $e = (1, \ldots, 1)^\mathsf{T} \in \mathbb{R}^H$ and

$$f_\tau(\boldsymbol{\varepsilon}_\tau^+, \boldsymbol{\varepsilon}_\tau^-) = \tau \sum_{h=1}^{H}\sum_{n=1}^{N} {\boldsymbol{\varepsilon}_\tau^+}_{hn} + (1-\tau)\sum_{h=1}^{H}\sum_{n=1}^{N} {\boldsymbol{\varepsilon}_\tau^-}_{hn}.$$

It is known that the solution of $(\boldsymbol{QR_\tau})$, *i.e.*, $(\boldsymbol{A}_\tau, b_\tau)$, allows us to compute the $\tau-$quantile of each column of $\boldsymbol{Y}$. For example, the nth column of $\boldsymbol{A}_\tau$ together with the nth component of b_τ will provide a model to predict the τ-quantile of Y_n. The aim of the study is to have the same $(\boldsymbol{A}, b)$ for a set of $\tau \in T$. To do that, we consider the problem

$$(\boldsymbol{QR_T}) \qquad \begin{cases} \text{minimize} & \sum_{\tau\in T} f_\tau(\boldsymbol{\varepsilon}_\tau^+, \boldsymbol{\varepsilon}_\tau^-) \\ \text{subject to:} & \boldsymbol{Y} = \boldsymbol{X}_\tau \cdot \boldsymbol{A} + e \cdot b^\mathsf{T} + \boldsymbol{\varepsilon}_\tau^+ - \boldsymbol{\varepsilon}_\tau^-, \\ & \boldsymbol{\varepsilon}_\tau^+, \boldsymbol{\varepsilon}_\tau^- \geqslant 0, \quad \boldsymbol{\varepsilon}_\tau^+, \boldsymbol{\varepsilon}_\tau^- \in \mathbb{R}^{H\times N}, \quad \forall \tau \in T. \end{cases}$$

Recall that, in case T is singleton, *e.g.*, $T = \{0.1\}$, then $(\boldsymbol{QR_T})$ is exactly $(\boldsymbol{QR_\tau})$.

Sampling explicating data: We do the same workflow in Section 3.2.1 till obtain the hyper-rectangle

$$\mathbb{X} := [\min \hat{X}_1, \max \hat{X}_1] \times \ldots \times [\min \hat{X}_P, \max \hat{X}_P]. \tag{3.4}$$

Next, as opposed to the previous idea, we form $\boldsymbol{X}$ by sampling each row of $\boldsymbol{X}$ uniformly from $\mathbb{X}$. In other words, the explicating data will be uniformly distributed inside the boundaries of the projection of the original data onto the PCs space.

Solving the LP and generating new data: This LP has PN (P times N) variables in $\boldsymbol{A}$ and N variables in b. For each $\tau \in T$, we also have HN variables in $\boldsymbol{\varepsilon}_\tau^+$ and HN variables in $\boldsymbol{\varepsilon}_\tau^-$. About the number of constraints, for each $\tau \in T$, we have HN equality constraints and $2HN$ inequality constraints.

Assume that $(\boldsymbol{A}_*, b_*)$ is the optimal solution of **($\boldsymbol{QR_T}$)**, then the next step is to generate new data using that $(\boldsymbol{A}_*, b_*)$. Similarly to the previous workflow, we simply sample new explicating data, say $\boldsymbol{Z}$, as the same way as we sampled the matrix $\hat{\boldsymbol{X}}$, then the new data will be generated by

$$\boldsymbol{Y}_* = \boldsymbol{Z} \cdot \boldsymbol{A}_* + e \cdot b_*^{\mathsf{T}}.$$

This model is still being studied.

Chapter 4

Meteorological based wind power forecast

Contents

4.1 Introduction

4.1.1 Motivation

Wind power is the use of wind to provide the mechanical power to turn electric generators. A wind farm consists of many wind turbines, which are connected to the electric transmission network. In 2017 France reached a total of 13,759 MW installed wind power capacity[1]. Wind power gives variable power, which is very consistent from year to year but has significant variation over shorter time scales. This variation directly impacts on the transmission network and the energy market, as

[1]The Statistics Portal, *Total installed wind power capacity in France from 2002 to 2017*, https://www.statista.com/statistics/421684/wind-power-capacity-in-france/

energy is sold before it is actually produced. Hence forecasting wind power over short time scales becomes more important nowadays, specially when the installed wind power capacity of France is going to be double by 2023 [2].

The current forecast system of France is organized as follows. Every day, the forecasted wind power generation is available since 6:00 PM for 72 hours ahead. On the next day, the energy company will provides the forecast updates per time unit. The forecasting method is based on:

- the wind energy generated on the grid in the nearing hours,
- wind scenarios provided by Meteo France (French weather forecasting service),
- technical specifications of the connected wind farms,
- wind farms localization.

Intuitively, one can make a forecast for a region by summing the forecasts of all the wind farms within that region. However, by this method, we also sum up the errors which arise when we forecast for the wind farms. Moreover, forecasting for a region may be very different from forecasting for a wind farm due to the correlation between the data provided by the weather stations. Therefore, in this chapter, we will find a new model that directly uses the meteorological and installed capacity data to estimate the value of wind power production of the regions in France. Another reason to study the meteorological based forecasting model is that, by looking at model, we can measure the impact of the weather stations on the wind power generation in each regions, hence, reduce the dimension of the problem in the future forecasts.

4.1.2 Goal description

The main goal of this topic is to build a model which is able to forecast the wind power generation for each region in France, based on the meteorological data and the installed capacity data of these regions. More specifically, the problem is described as follows.

- France is divided into 7 regions coded by: SEE, SENE, SENP, SEO, SERAA, SESE and SESO. Each of them has a number of wind farms inside. These farms are scattered over the regions, some of them are offshore while the others are onshore.
- In France, there are 14541 weather stations. Each station has a fixed location, characterized by a *latitude* and a *longitude*. These stations provide us, every 3 hours, the speed and the direction of wind at the areas where they locate.
- Additionally, we also know the daily installed capacity of these 7 regions, for example, the installed capacity of SEE on 1^{st} Jan 2015 was 1747.32 MW and it increased to 2522.22 MW on 31^{th} Dec 2017.

- Every day, based on the meteorological and installed capacity data, we forecast the quantiles of the wind power could be generated in the next 72 hours, 3-hourly (so, we must forecast totally 25 lead times: $0, 3, 6, \ldots, 72$) for all the regions.

So, for a given $\alpha \in \{0.1, 0.2, \ldots, 0.9\}$, the model should be able to forecast the α-quantile of the generated power of regions. Thus, for example, if we are interested in these nine quantiles, we must study a total of $7 \times 25 \times 9 = 1575$ different models.

4.1.3 The input data

It is important to have a look at what datasets we have. As partly described above, the data we have to build the model are as follows.

(a) *10-minute* wind power data of 7 regions, over 3 years, from 2015-01-10 00:00:00 to 2017-12-31 22:50:00. So totally, we have 1104482 records.

(b) *Daily* installed capacities of 7 regions over 3 years. Note that typically, these values do not change daily, they only change when new wind turbines are constructed or damaged.

(c) *Latitudes* and *longitudes* of 14541 weather stations in France. They are arranged into a grid of size 111×131.

(d) *3-hourly* wind forecasts (speed and direction) over the grid of weather stations, over 3 years.

4.2 Formulation

4.2.1 Notations and the general model

Without loss of generality, we assume that the weather stations are arranged into a grid of $M \times N$ locations, and from now on we will refer to them by $W_{ij}, 1 \leqslant i \leqslant M, 1 \leqslant j \leqslant N$. We denote by $\widehat{X}_{\mu,i,j,T|t}$ the value of the weather variable μ at station W_{ij} provided at time t for the lead time T, $C_{r,T}$ the installed wind power capacity of region r observed at time T and $\widehat{Y}_{\alpha,r,T|t}$ the forecasted α-quantile of wind power of region r provided at time t for the lead time T. Then the general model will be written as

$$\widehat{Y}_{r,\alpha,T|t} = f(\widehat{X}_{\mu,i,j,T|t}, C_{r,T}), \tag{4.1}$$

where f is the regression function. Let us explain more clearly the variables in model (4.1). Here, the weather variable μ refers to *speed* and/or *direction* of the wind, the lead time T is the time at

which the forecasts are validated[2], while t is the time at which we make the forecasts.

Next, we are going to apply the quantile regression framework to solve our problem.

4.2.2 A linear model

Let us fix a region denoted by r^* among the seven regions listed in Subsection 4.1.2. Recall that, beside the wind speed at weather stations, the historical data provide us the 10-minute observed wind power and the daily installed capacity of the whole region r^*. Now, we will propose a possible model for forecasting the α-quantile of the generated power at r^* for a lead time T.

We know that almost all of the turbines nowadays are able to be pointed into the wind whatever its direction is. So, in our model, we will neglect the effect of the wind direction on the speed of turbines and, therefore, on the generated wind power. It is known that, for a turbine with effective area[3] a, the possible generated power can be calculated by

$$P = \frac{1}{2}\rho a E s^3,$$

where P is the generated power, ρ is the air density[4], E is the wind power coefficient[5] and s is the wind speed. The coefficient E is shown to be maximal at $\frac{16}{27}$ [4]. Therefore, approximately, the generated power depends linearly on cube of the wind speed. However, in order to obtain a more general and flexible model, we will assume additionally that the generated power is a polynomial of degree three of wind speed, *i.e.,*

$$P = As^3 + Bs^2 + Cs + D, \tag{4.2}$$

where $A, B, C, D \in \mathbb{R}$ are coefficients of the polynomial.

Now, for every i, j, T and t, let us refer to the wind speed at station W_{ij} provided at time t for time T by $s_{i,j,T|t}$. This means that in the notation of the general model (4.1),

$$\widehat{\mathbf{X}}_{\mu,i,j,T|t} = s_{i,j,T|t} \quad \text{for all } i, j, T \text{ and } t.$$

Also, in a similar notation to model (4.1), we denote by $C_{r^*,T}$ the installed wind power capacity of the region r^* at time T. However, instead of studying the general model (4.1), we will rewrite

[2] $T \in \{0, 3, 6, \ldots, 72\}$
[3] the area of wing surfaces which generate rotational motion
[4] approximately 1.225 kgm^3 in typical condition
[5] also known as Betz's coefficient

it in the following form:

$$\frac{\widehat{Y}_{r^*,\alpha,T|t}}{C_{r^*,T}} = g(\widehat{X}_{\mu,i,j,T|t}) \tag{4.3}$$

and seek a possible expression of g. The reason for doing this is that: g will be more consistent than f in capturing the relationship between wind speed at the weather stations and the generated power, which we are interested in. In other words, we want this relationship to be independent from the increase of the installed capacity along time.

Hence, as discussed above, we will use the assumption that g is a polynomial of degree three of wind speed and consider the model:

$$\frac{\widehat{Y}_{r^*,\alpha,T|t}}{C_{r^*,T}} = \sum_{i=1}^{M}\sum_{j=1}^{N}\big(A_{r^*,\alpha,i,j,T}\ s^3_{i,j,T|t} + B_{r^*,\alpha,i,j,T}\ s^2_{i,j,T|t} + C_{r^*,\alpha,i,j,T}\ s_{i,j,T|t} + D_{r^*,\alpha,i,j,T}\big),$$

for all α and lead time T. This model is linear in $A_{r^*,\alpha,i,j,T}$, $B_{r^*,\alpha,i,j,T}$, $C_{r^*,\alpha,i,j,T}$ and $D_{r^*,\alpha,i,j,T}$ which are the parameters will be inferred from the historical data. So, it could be useful to denote the right hand side as a function of the parameters and rewrite this model as:

$$\frac{\widehat{Y}_{r^*,\alpha,T|t}}{C_{r^*,T}} = h(\beta_{r^*,\alpha,T}), \tag{4.4}$$

where $\beta_{r^*,\alpha,T} := (A_{r^*,\alpha,i,j,T}, B_{r^*,\alpha,i,j,T}, C_{r^*,\alpha,i,j,T}, D_{r^*,\alpha,i,j,T})_{1\leqslant i\leqslant m, 1\leqslant j\leqslant n}$. Here, the parameters are indexed by the region r, the quantile α and the lead time T, as mentioned at the end of Subsection 4.1.2, for each r, α, T, we will have a different model.

In quantile regression framework point of view, $h(\beta_{r^*,\alpha,i,j,T})$ is a linear estimation of the α-quantile of $\frac{Y_{r^*,T}}{C_{r^*,T}}$. Therefore, $\beta_{r^*,\alpha,T}$ can be computed as the solution of the following optimization problem

$$\underset{\beta_{r^*,\alpha,T}}{\text{minimize}} \sum_{T|t} \ell_\alpha\Big(\frac{Y_{r^*,T}}{C_{r^*,T}} - h(\beta_{r^*,\alpha,T})\Big), \tag{4.5}$$

in which, $\ell_\alpha(\cdot)$ is the quantile loss function. Here, the notation $\sum_{T|t}$ means that we sum up quantile losses over the historical data.

Thus, in summary, given a region r^*, a lead time T and $\alpha \in (0,1)$, if $\widehat{\beta}_{r^*,\alpha,T}$ is a solution of (4.5) then $h(\widehat{\beta}_{r^*,\alpha,T})$ will be an approximation of the α-quantile of $\frac{Y_{r^*,T}}{C_{r^*,T}}$. The optimal solution $\widehat{\beta}_{r^*,\alpha,i,j,T}$ could give us the information of how do the weather stations W_{ij} impact on the wind power of the region r^* at each lead time T.

4.3 Implementation

As mentioned earlier, for each region, each lead time and each quantile, we have a different model. Throughout this section, we will implement for the 0.8-quantile of regions SEE, SEO, SERAA and SESO (see Figure 4.1) for the lead time $T = 1$, *i.e.*, 3:00 AM. All problem were solved using CPLEX 12.6 on an Intel® i7-8550U CPU @ 1.8GHz x 8 with 16GB RAM. The computational experiment were carried out in JuMP.

Figure 4.1: The wind-power regions in France. *Source: RTE.*

Firstly, let us have a look at the averaged over the whole 3 years[6] of wind speed at the weather stations along the given latitude and longtitude (Figure 4.2).

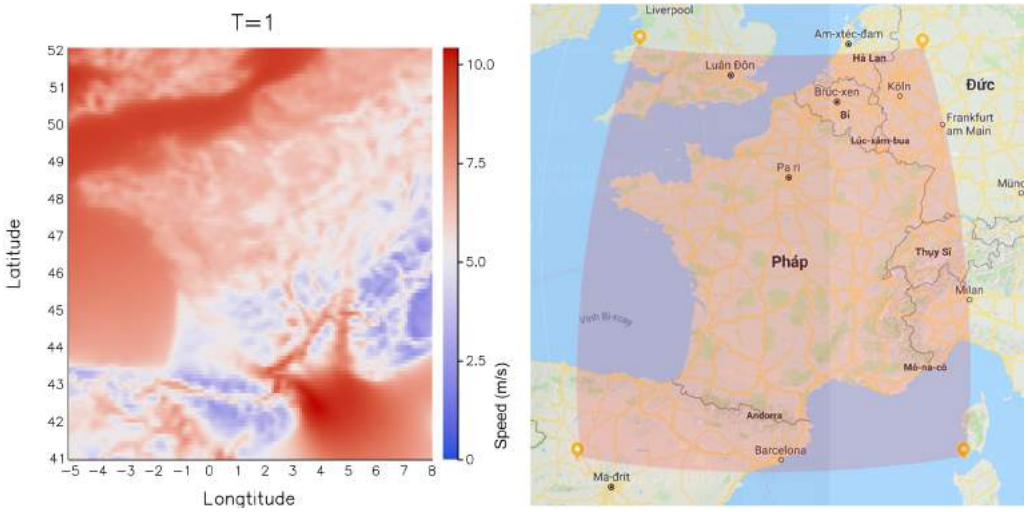


Figure 4.2: The averaged wind speed at the weather stations along the latitude and longtitude.

[6] 2015, 2016 and 2017

The Figure 4.3 describes the coefficients A, B, C, D in (4.2) obtained by solving the LP (4.5). As we can see, for all the considered regions, the coefficients $D_{i,j}$'s are almost zeros everywhere; whereas the coefficients $C_{i,j}$'s are well adapted with the wind speed heat-map. It is interesting that the coefficients $A_{i,j}$ and $B_{i,j}$ remarkably partition the map of the corresponding regions. This suggests us two ideas that may reduce the dimension and improve the forecasts: a) try the model that uses only the cubes of wind speeds; b) identify the weather stations which have the most impacts on forecasting and then build a new model that uses only the data from those such stations.

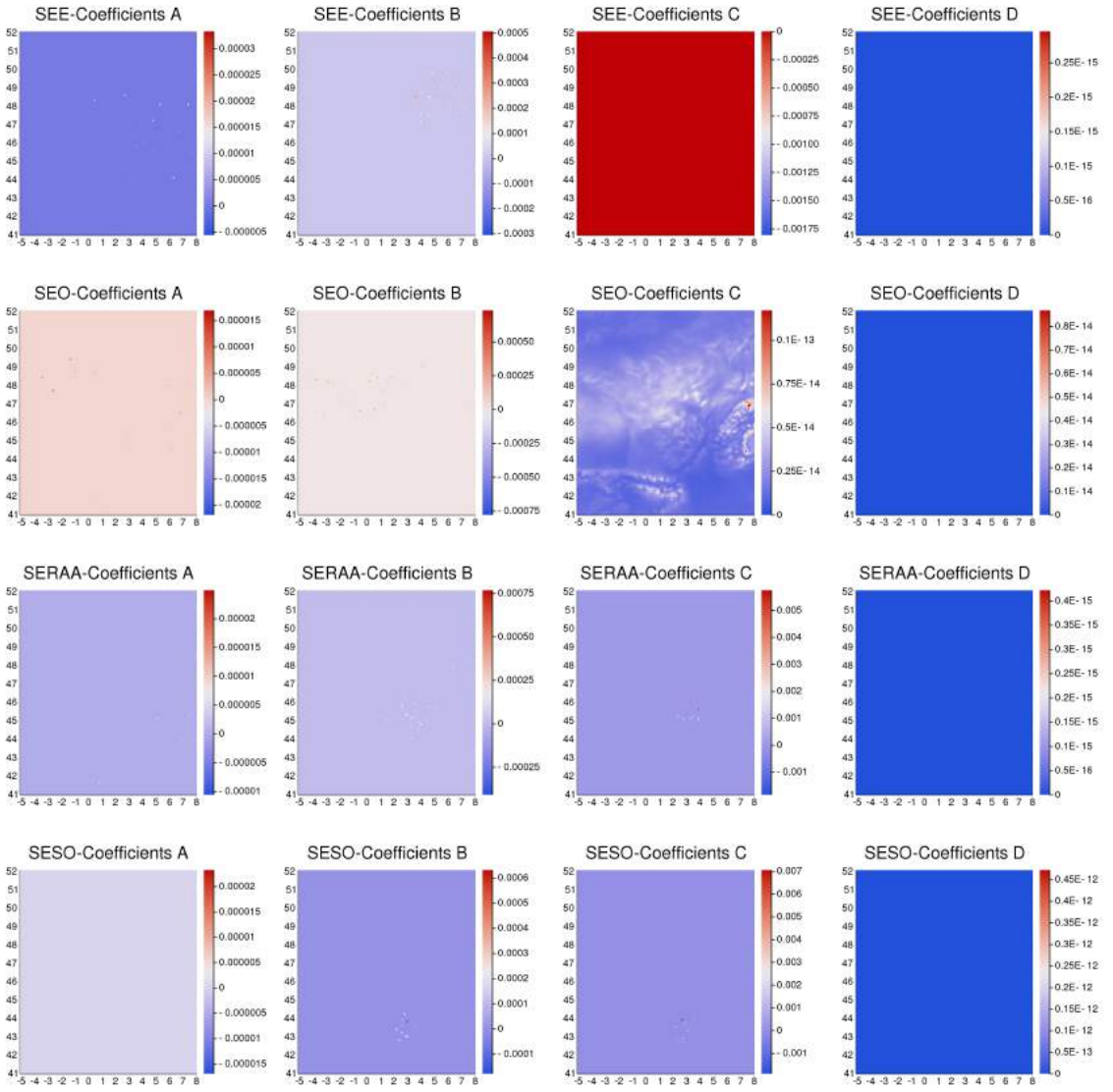


Figure 4.3: The coefficients $\beta_{r^*,\alpha,T} := (A_{r^*,\alpha,i,j,T}, B_{r^*,\alpha,i,j,T}, C_{r^*,\alpha,i,j,T}, D_{r^*,\alpha,i,j,T})_{1\leqslant i\leqslant m, 1\leqslant j\leqslant n}$, in which, $\alpha = 0.8, T = 1$ and $r^* \in \{\text{SEE, SEO, SERAA, SESO}\}$.

Table 4.1 is the showcase of solving the LP instances we solved above. These LP were solved by barrier method with optimality tolerances of 10^{-3}.

Region	m	n	Barrier Iter	CPU Time	Primal Obj
SEE	3288	64740	51	5132.47 sec	4.7671e+01
SEO	3288	64740	52	7969.83 sec	3.6834e+01
SERAA	3288	64740	45	4200.40 sec	5.2123e+01
SESO	3288	64740	43	4508.10 sec	5.3940e+01

Table 4.1: Solving the LP instances for each region using CPLEX solver with tolerances optimality of 1e-03.

In the next Chapter, we will see that these LPs could also be solved efficiently using random projections.

Chapter 5

Solving large quantile regression using random projections

Contents

5.1 Random projections

5.1.1 Motivation and definition of random projections

Random projection is a technique used to reduce the dimensionality of a set of points which lie in Euclidean space. Random projection method is a powerful method known for its simplicity and less erroneous output compared with other methods. In random projection method, the dimensions and distribution of random projection matrices[1], *i.e.*, random linear maps, are controlled so as to approximately preserve the pairwise distances between any two samples of the dataset. Firstly,

[1] simply called random projections

we would introduce the main motivation for the development of random projections, the so-called The Johnson-Lindenstrauss lemma (JLL), which is proposed by William B. Johnson and Joram Lindenstrauss in their seminar paper in 1984 [10]. JLL is stated as follows:

Lemma 5.1.1 (The Johnson-Lindenstrauss lemma [10]). *Given $\varepsilon \in (0,1)$ and an $m \times n$ matrix A, there exists an $k \times m$ matrix T such that*

$$\forall 1 \leqslant i < j \leqslant n \quad (1-\varepsilon)\|A_i - A_j\| \leqslant \|TA_i - TA_j\| \leqslant (1+\varepsilon)\|A_i - A_j\|, \tag{5.1}$$

where k is $O(\varepsilon^{-2} \ln n)$.

An elementary proof of this surprising result could be found in the paper [5].

According to JLL, we can compress n (for instance, one billion) points in $\mathbb{R}^m$ by projecting them into $\mathbb{R}^k$, with $k = O(\varepsilon^{-2} \ln n)$, such that no distance is distorted by more than $1 + 2\varepsilon$. With a reasonable choice of the error ε, the projected dimension k could be much smaller than the original dimension m. Therefore, the JJL is very meaningful for the big data cases, *i.e.*, when m and n are huge.

The existence of such the map T in JLL is shown by probabilistic methods. More precisely, T is drawn from some well structured classes of random maps such that (5.1) holds with some positive probability. This can be done if T satisfies for all $x \in \mathbb{R}^m$:

$$\mathbb{P}\Big[(1-\varepsilon)\|x\|^2 \leqslant \|Tx\|^2 \leqslant (1+\varepsilon)\|x\|^2\Big] > 1 - \frac{2}{n(n-1)}.$$

Indeed, if this is the case, then for all $1 \leqslant i < j \leqslant n$ we have

$$\mathbb{P}\Big[(1-\varepsilon)\|A_i - A_j\| \leqslant \|TA_i - TA_j\| \leqslant (1+\varepsilon)\|A_i - A_j\|\Big] > 1 - \frac{2}{n(n-1)}.$$

Hence, by applying the union bound for $\frac{n(n-1)}{2}$ pairs of points (A_i, A_j), we obtain a positive probability for the event (5.1).

Now, let us introduce a mathematical definition for a random projection.

Definition 5.1.1 (Random projection). *A random linear map $T : \mathbb{R}^m \to \mathbb{R}^k$ is called a random projection if for all $\varepsilon \in (0,1)$ and all vectors $x \in \mathbb{R}^m$, we have*

$$\mathbb{P}\Big[(1-\varepsilon)\|x\|^2 \leqslant \|Tx\|^2 \leqslant (1+\varepsilon)\|x\|^2\Big] \geqslant 1 - 2e^{-\mathcal{C}\varepsilon^2 k},$$

for some universal constant $\mathcal{C} > 0$ (independent of m, n, ε).

As shown above, one can see that if we choose k such that $1 - 2e^{-\mathcal{C}\varepsilon^2 k} > 1 - \frac{2}{n(n-1)}$ then we can obtain (5.1) with a positive probability. Actually, this positive probability could be very high. For example, if we want (5.1) holds with a probability at least 99.9%, we just simply choose k such that

$$1 - n(n-1)e^{-\mathcal{C}\varepsilon^2 k} > 1 - \frac{1}{1000},$$

which can be reduced to $k = \lceil \frac{\ln(1000) + 2\ln(n)}{\mathcal{C}\varepsilon^2} \rceil$.

5.1.2 Introducing some random projections

Follows are some choices of random projections:

- orthogonal projections on a random k-dimensional linear subspace of $\mathbb{R}^m$ (by Johnson and Lindenstrauss [10]);
- random $k \times m$ matrices with i.i.d entries drawn from $\mathcal{N}(0, 1)$ (by Indyk and Motwani [9]);
- random $k \times m$ matrices with i.i.d entries taking values $+1$ and -1, each with probability $\frac{1}{2}$ (by Achlioptas [1]);
- random $k \times m$ matrices with i.i.d entries taking vales $+1, 0$ and -1 with probability $\frac{1}{6}, \frac{2}{3}, \frac{1}{6}$ respectively (by Achlioptas [1]).

Note that all the random projectors we consider have zero mean. Additionally, the distribution used in the last three projectors are actually the special cases of a general class of the so-called *sub-Gaussian* distributions [14].

5.2 Random projections for linear programming

In this section, we represent a recent discovery, in the paper [18], that random projections can be used to approximately solve very large LPs efficiently. In particular, if we denote

$$P \equiv \min\{c^\mathsf{T} x \mid Ax = b \wedge x \in \mathbb{R}^n_+\} \quad \text{and } P_T \equiv \min\{c^\mathsf{T} x \mid TAx = Tb \wedge x \in \mathbb{R}^n_+\},$$

where T is a random projection, then in [18] the authors shown that: with a high probability (w.a.h.p.) solving P_T will be faster than P while the feasibility and optimality are preserved. Moreover, the solution could also be retrieved successfully w.a.h.p. by a proven algorithm.

We denote by $v(P)$ and $\mathcal{F}(P)$ (*resp.*, $v(P_T)$ and $\mathcal{F}(P_T)$) the optimal objective value and the feasible set of the problem P (*resp.*, problem P_T). Briefly, there are some goals to be achieved:

1. *Feasibility preservation*: If $\mathcal{F}(P)$ is feasible then $\mathcal{F}(P_T)$ must be feasible w.a.h.p and vice versa, *i.e.*, if $\mathcal{F}(P)$ is infeasible then $\mathcal{F}(P_T)$ is also infeasible w.a.h.p. Note that the former proposition is straightforward by linearity, the issue is proving the later.
2. *Optimality preservation*: If there exists an optimal solution to P then for a given $\delta > 0$, there is a random projection T such that $v(P) - \delta \leqslant v(P_T) \leqslant v(P)$ w.a.h.p.
3. *Solution retrieval*: An algorithm to retrieve an approximation $\tilde{x}$ of the optimal solution x^* of problem P.

5.2.1 Preserving feasibility

In this subsection, we represent briefly how to prove that $\mathcal{F}(P) \neq \emptyset$ if and only if $\mathcal{F}(P_T) \neq \emptyset$ w.a.h.p. As mentioned above, since the linearlity of T, we only need to prove the infeasibility of $\mathcal{F}(P)$ leading to the infeasibility of $\mathcal{F}(P_T)$ w.a.h.p. This can be done by showing that, w.a.h.p., if b is not in the cone of A then Tb is not in the cone of TA.

Firstly, we have the following propositions:

Proposition 5.2.1 (Corollary 1, [18]). *Let $T : \mathbb{R}^m \to \mathbb{R}^k$ be a random proposition as in the Definition 5.1.1 and let $x \in \mathbb{R}^m$ be a non-zero vector. Then we have*

$$\mathbb{P}\big[Tx \neq 0\big] \geqslant 1 - 2e^{-\mathcal{C}k}$$

for some constant $\mathcal{C} > 0$ (independent of m, k).

Proposition 5.2.2. *Let $T : \mathbb{R}^m \to \mathbb{R}^k$ be a random proposition as in the Definition 5.1.1 and let $b, A_1, \ldots, A_n \in \mathbb{R}^m$. Then for a given vector $x \in \mathbb{R}^n$ such that $b \neq \sum_{j=1}^n x_j A_j$, we have*

$$\mathbb{P}\Big[Tb \neq \sum_{j=1}^{n} x_j TA_j\Big] \geqslant 1 - 2e^{-\mathcal{C}k}$$

for some constant $\mathcal{C} > 0$ (independent of n,k).

The latter proposition is a direct corollary of Proposition 5.2.1 by applying to $Ax - b$. It is a certificate for the claim that: if b is not in the cone of A then Tb is not in the cone of TA w.a.h.p. However, in the paper [18], the authors even obtained a quantitative result for "w.a.h.p." as follows.

In order to state the theorem, we need some notations as follows. Given $x \in \text{cone}(A)$, the A-norm

of x is defined as

$$\|x\|_A = \min\Big\{ \sum_{j=1}^{n} \lambda_j \mid \lambda \geqslant 0 \wedge x = \sum_{j=1}^{n} \lambda_j A_j \Big\}.$$

For each $x \in \text{cone}(A)$, we say that $\lambda \in \mathbb{R}^n_+$ yields a minimal A-representation of x if and only if $\sum_{j=1}^{n} \lambda_j = \|x\|_A$. We define $\mu_A = \max\{\|x\|_A \mid x \in \text{cone}(A) \wedge \|x\| \leqslant 1\}$. In addition, assume that we are given an estimate of lower bound Δ to $d = \min_{x \in C} \|b - x\|$, where $C = \text{conv}(\{A_1, \ldots, A_n\})$, and also (without loss of generality) that b and the column vectors of A have unit Euclidean norm. Then the theorem is stated as:

Theorem 5.2.1 (Theorem 3, **[18]**). *Given an $m \times n$ matrix A and $b \in \mathbb{R}^m$ such that $b \notin \text{cone}A$. Then for any $0 < \varepsilon < \frac{\Delta^2}{\mu_A^2 + 2\mu_A\sqrt{1-\Delta^2}+1}$ and any $k \times m$ random projector T, we have*

$$\mathbb{P}\Big[Tb \notin \text{cone}(TA)\Big] \geqslant 1 - 2(n+1)(n+2)e^{-\mathcal{C}(\varepsilon^2 - \varepsilon^3)k}$$

for some constant $\mathcal{C}$ (independent of m, n, k, Δ).

Hence, by this theorem, with a suitable choice of k we can obtain a positive probability for the event $Tb \notin \text{cone}(TA)$.

5.2.2 Preserving optimality

In the following, we will show that $v(P) \approx v(P_T)$ w.a.h.p. From the previous subsection, we know that P is feasible (*resp.*, infeasible) if and only if P_T is feasible (*resp.*, infeasible) w.a.h.p. Note that it also holds for an $(k+h) \times m$ random projector of the form

$$\begin{bmatrix} I_h & 0 \\ 0 & T \end{bmatrix},$$

where T is an $k \times m$ random projection. This allows us to claim the feasibility equivalence (w.a.h.p.) even when we only project a subset of rows of A. We will use this remark in the later of this subsection.

Firstly, we assume that $\mathcal{F}(P)$ is non-empty and bounded, *i.e.*, there is an optimal solution x^* and a constant $\theta > 0$ such that

$$\sum_{j=1}^{n} x_j^* < \theta. \tag{5.2}$$

Without loss of generality, we assume further that $\theta \geqslant 1$. This assumption is used to control the excessive flatness of the involved cones, which is required in the projected separation argument.

Next, we introduce a transformation that can transform a cone to another cone so that the membership problem regarding to the new one becomes easier to solve by random projection. Namely, given a polyhedral cone

$$\mathcal{K} = \left\{ \sum_{j=1}^{n} x_j A_j \;\middle|\; x \in \mathbb{R}^n_+ \right\},$$

in other words $\mathcal{K} = \text{cone}(A)$. For any $u \in \mathbb{R}^m$, we consider the following transformation $\Phi_{u,\theta}$ defined by:

$$\Phi_{u,\theta}(\mathcal{K}) := \left\{ \sum_{j=1}^{n} x_j \left(A_j - \frac{1}{\theta} u \right) \;\middle|\; x \in \mathbb{R}^n_+ \right\}.$$

For θ defined in (5.2), we also consider the following set

$$\mathcal{K}_\theta = \left\{ \sum_{j=1}^{n} x_j A_j \;\middle|\; x \in \mathbb{R}^n_+ \wedge \sum_{j=1}^{n} x_j < \theta \right\}.$$

Then we have the following interesting lemma:

Lemma 5.2.1 (Lemma 3, [18]). *For any $u \in \mathbb{R}^m$, we have $u \in \mathcal{K}_\theta$ if and only if $u \in \Phi_{u,\theta}(\mathcal{K})$.*

This lemma has a nice proof in [18]. Now, we will present the main theorem of optimality preservation, which directly uses this lemma in proof.

Consider the following problem:

$$P_{T,\theta} \equiv \min\{c^\mathsf{T} x \mid TAx = Tb \wedge \sum_{j=1}^{n} x_j \leqslant \theta \wedge x \in \mathbb{R}^n_+\}.$$

This is the projected problem added to a constraint that $\sum_{j=1}^{n} x_j \leqslant \theta$. Follow is the main theorem, which asserts that the optimal objective value of P can be well approximated by that of $P_{T,\theta}$.

Theorem 5.2.2 (Theorem 4, [18]). *Assume $\mathcal{F}(P)$ is bounded and non-empty. Let y^* be an optimal dual solution of P of minimal Euclidean norm. Given $0 < \delta \leqslant |v(P)|$, we have*

$$v(P) - \delta \leqslant v(P_{T,\theta}) \leqslant v(P) \tag{5.3}$$

with probability at least $p = 1 - 4ne^{-\mathcal{C}(\varepsilon^2 - \varepsilon^3)k}$, where $\varepsilon = O(\frac{\delta}{\theta^2 \|y^*\|})$.

The following is the idea of the proof. Firstly, we observe that for any $\delta > 0$, the problem

$$Ax = b \ \wedge \ x \geqslant 0 \ \wedge \ c^{\mathsf{T}}x \leqslant v(P) - \delta$$

is infeasible. Next, by rewriting this problem in the standard form as

$$\begin{bmatrix} c^{\mathsf{T}} & 1 \\ A & 0 \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} = \begin{bmatrix} v(P) - \delta \\ b \end{bmatrix}, \quad \text{where} \quad \begin{bmatrix} x \\ s \end{bmatrix} \geqslant 0,$$

and applying a random projection of the form

$$\begin{bmatrix} 1 & 0 \\ 0 & T \end{bmatrix}, \quad \text{where } T \text{ is an } k \times m \text{ random projector,}$$

we obtain the following problem

$$\begin{cases} c^{\mathsf{T}}x + s &= v(P) - \delta \\ TAx &= Tb \\ x &\geqslant 0 \\ s &\geqslant 0 \end{cases},$$

which is supposed to be infeasible too w.a.h.p. Now, we use the prior information about the optimal solution x^*, *i.e.* $\sum_{j=1}^{n} x_j \leqslant \theta$, by adding it into the above projected problem. As mentioned at the beginning, this does not change its feasibility, but later can be used to transform the corresponding cone into the one which is easier to deal with. Therefore, w.a.h.p., the problem

$$\begin{cases} c^{\mathsf{T}}x &\leqslant v(P) - \delta \\ TAx &= Tb \\ \sum_{j=1}^{n} x_j &\leqslant \theta \\ x &\geqslant 0 \end{cases}$$

is infeasible. Hence, we deduce that $c^{\mathsf{T}}x \geqslant v(P) - \delta$ holds w.a.h.p. for any feasible solution x of the problem $P_{T,\theta}$, and this proves the left hand side of (5.3). For the right hand side, the proof is trivial since P_T is a relaxation of P with the same objection function.

5.2.3 Retrieving solution

In this subsection, we present the idea of how to retrieve an approximation $\tilde{x}$ of the optimal solution x^* of problem P.

Let $\delta > 0$ and assume that $x' \in \mathbb{R}^n_+$ is an optimal solution of problem $P_{T,\theta}$. Then w.a.h.p. we have $TAx' = Tb$ and $v(P) - \delta \leqslant cx' \leqslant v(P)$, by Theorem 5.2.2. However, the following proposition

asserts that, almost surely, x' is not a feasible solution of problem P, which means x' only gives us an approximation for the optimal objective value, but not the optimal solution itself. More precisely, let $0 \leqslant \nu \leqslant \delta$ such that $v(P_T) = v(P) - \nu$ and let

$$\widetilde{A} = \begin{bmatrix} c \\ A \end{bmatrix}, \quad \tilde{b} = \begin{bmatrix} v(P) - \nu \\ b \end{bmatrix}, \quad \widetilde{T} = \begin{bmatrix} 1 \\ T \end{bmatrix}.$$

We denote $F' = \{x \in \mathbb{R}^n_+ \mid \widetilde{T}\widetilde{A}x = \widetilde{T}\tilde{b}\}$. Then the proposition is stated as follows:

Proposition 5.2.3 (Proposition 3, [18]). *Assume that cone(A) is full dimensional in $\mathbb{R}^m$ and that any optimal solution of P has at least m non-zero components. Let x' be uniformly chosen in F'. Then, almost surely, $\widetilde{A}x' = \tilde{b}$ does not hold.*

Nevertheless, we still have a hope to retrieve an approximation of the optimal solution of P, by looking at the *dual optimal solution*. Indeed, consider the dual problem and its projection:

$$D \equiv \max\{b^\mathsf{T} y \mid y^\mathsf{T} A \leqslant c \ \wedge \ y \in \mathbb{R}^m\}$$
$$D_T \equiv \max\{(Tb)^\mathsf{T} y \mid y^\mathsf{T} TA \leqslant c \ \wedge \ y \in \mathbb{R}^k\}.$$

We denote by y^* and y_T the optimal solution of D and D_T, respectively. Let define $y_{\text{prox}} = T^\mathsf{T} y_T$ (it is easy to see that y_{prox} is also a feasible solution of the dual problem D). The following lemma shows that y_{prox} is approximately close to y^* w.a.h.p.

We need some assumptions and notations as follows. Firstly, we assume that $b \in \mathbb{R}^m$ belongs to the relative interior of the normal cone at some vertex of the dual polyhedron. This assumption guarantees that the dual problem has unique optimal solution, see Figure 5.1.

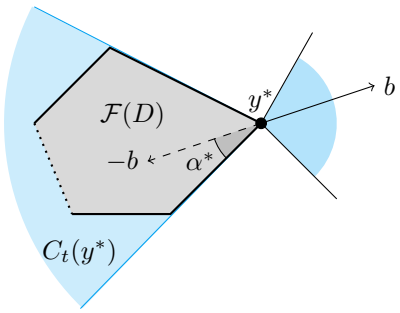


Figure 5.1: Illustrating the polyhedron $\mathcal{F}(D)$ and the cones in 2D.

Next, we denote by $C_t(y^*)$ the tangent cone of the dual polyhedron $\mathcal{F}(D) = \{y \in \mathbb{R}^m \mid y^\mathsf{T} A \leqslant c\}$ at y^*. It is the convex cone generated by a set of vectors $v^i = y^i - y^*$ where y^i are the neighboring vertices of y^* for $i \leqslant p$. For each $1 \leqslant i \leqslant p$, let α_i be the angle between the vector $-b$ and v^i.

Finally, let denote

$$\alpha^* \in \underset{\alpha_1,\ldots,\alpha_p}{\operatorname{argmin}} \cos(\alpha_i).$$

The lemma is stated as follows:

Lemma 5.2.2 (Lemma 4, [18]). *For any $\varepsilon > 0$, there is a constant $\mathcal{C}$ such that*

$$\|y^* - y_{prox}\|_2 \leqslant \frac{\mathcal{C}\theta^2\varepsilon}{\cos(\alpha^*)\|b\|_2}\|y^*\|_2$$

with probability at least $1 - 4ne^{-\mathcal{C}(\varepsilon^2-\varepsilon^3)k}$.

In [18], this lemma was used to prove that the Algorithm 1 indeed returns an optimal basic solution.

Algorithm 1 ([18]) Retrieving an approximate solution of P

INPUT: $y_T \in \mathbb{R}^k$, the associated basic dual solution of the projected dual problem D_T.
OUTPUT: $x \in \mathbb{R}^n$, an approximate solution of P.

for all $1 \leqslant j \leqslant n$ **do**
$z_j := \frac{c_j - A_j^\mathsf{T} y_{\text{prox}}}{\|A_j\|_2}$
end for
Let $\mathcal{B}$ be the set of indices j corresponding to the m smallest values of z_j.
$x_\mathcal{B} := A_\mathcal{B}^{-1} b$.
return $x \in \mathbb{R}^n$, where the non-basic indices are filled by zeros.

Before stating the main proposition for optimality preservation, let us denote by $\mathcal{B}^*$ the optimal basic obtained from the Algorithm 1, and denote

$$d^* = \min_{j \notin \mathcal{B}^*} \frac{c_j - A_j^\mathsf{T} y^*}{\|A_j\|_2}.$$

Note that d^* is the shortest distance from y^* to any facet $A_j^\mathsf{T} y = c_j$ for $j \notin \mathcal{B}^*$. The following is the main proposition:

Proposition 5.2.4 (Proposition 4, [18]). *Assume that the LP problem P satisfies the following two assumption:*
(a) there is no degenerated vertex in the dual polyhedron;
(b) the vector $b \in \mathbb{R}^m$ belongs to the relative interior of the normal cone at some vertex of the dual polyhedron.
If

$$\frac{\mathcal{C}\theta^2\varepsilon}{\cos(\alpha^*)\|b\|_2}\|y^*\|_2 < \frac{d^*}{2}, \tag{5.4}$$

where $\mathcal{C}$ is the universal constant in Lemma 5.2.2, then with probability at least $p = 1 - 4ne^{-\mathcal{C}(\varepsilon^2-\varepsilon^3)k}$, the Algorithm 1 returns an optimal basic solution.

5.3 Application in solving large quantile regression problems

In this section, we will use random projections to solve the quantile regression problems in the previous chapters. First of all, let us formalize the framework of applying random projection method as follows:

1. sampling a random projection matrix T;
2. perform the multiplication TA and Tb;
3. solve the projected problem $P_T \equiv \min\left\{c^\mathsf{T}x \mid TAx = Tb \wedge x \in \mathbb{R}^n_+\right\}$;
4. retrieve an approximate solution for the original problem P.

Applying this framework to the LP instances in Section 4.3, we obtain the coefficients as described in the following figures. Here, we compare the coefficients for each region: the *original* coefficients are the ones in Figure 4.3, whereas the *retrieved* coefficients are the ones obtained by solving the projected LPs. As we can see, by solving the projected LPs, we could also approximately identify the most significant weather stations for each region.

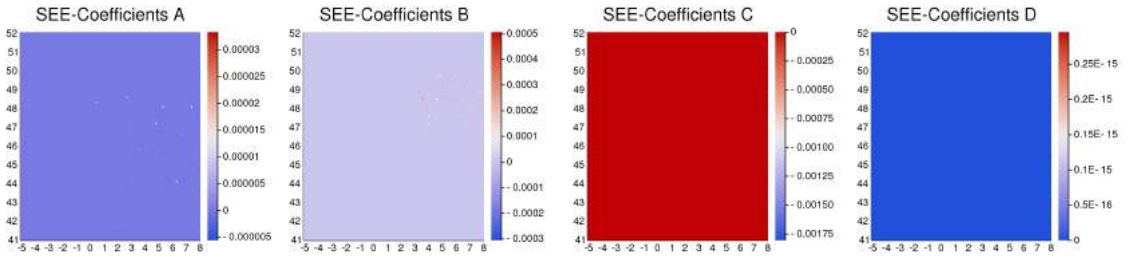


Figure 5.2: The original coefficients at SEE ($\alpha = 0.8, T = 1$).

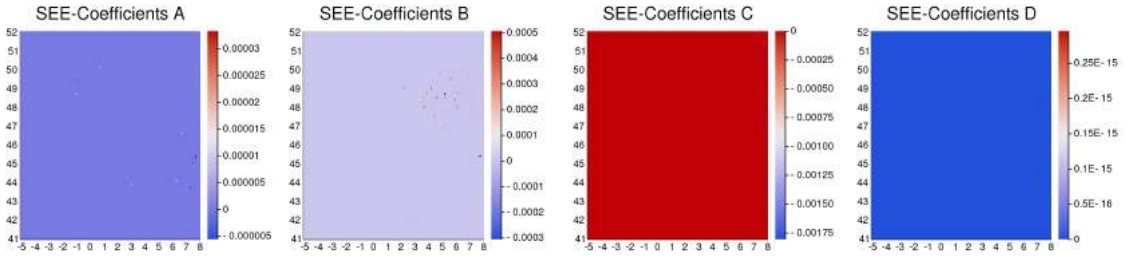


Figure 5.3: The retrieved coefficients at SEE ($\alpha = 0.8, T = 1$).

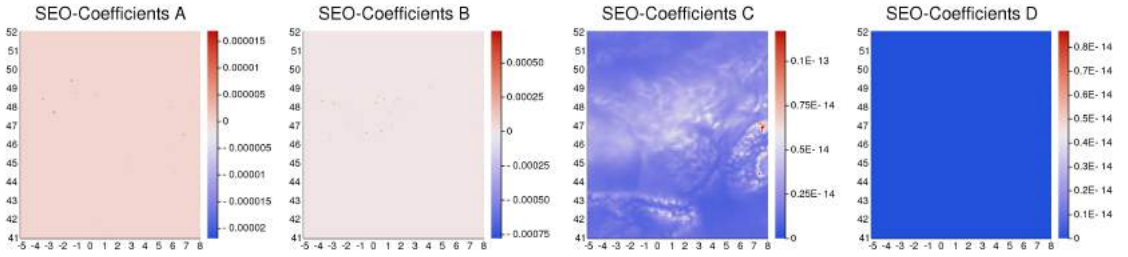


Figure 5.4: The original coefficients at SEO ($\alpha = 0.8, T = 1$).

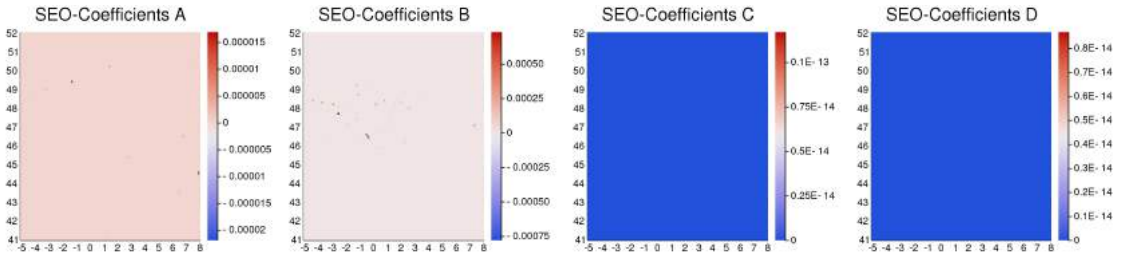


Figure 5.5: The retrieved coefficients at SEO ($\alpha = 0.8, T = 1$).

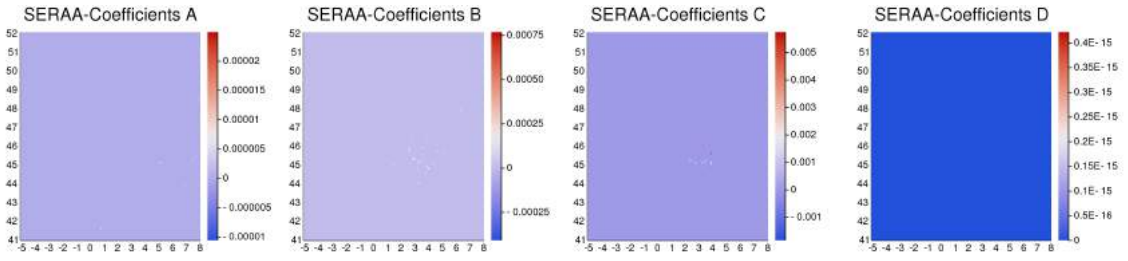


Figure 5.6: The original coefficients at SERAA ($\alpha = 0.8, T = 1$).

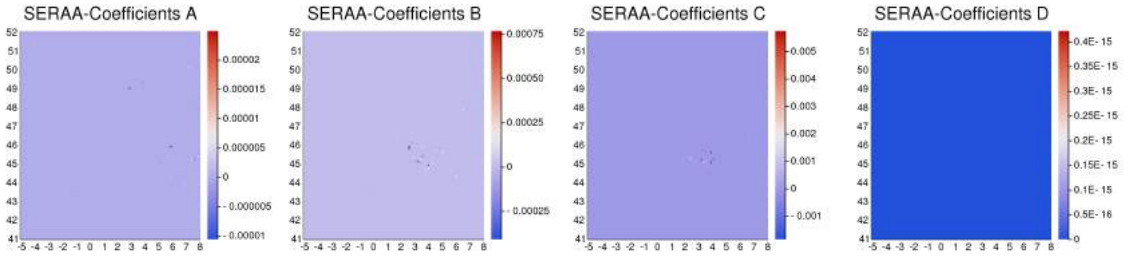


Figure 5.7: The retrieved coefficients at SERAA ($\alpha = 0.8, T = 1$).

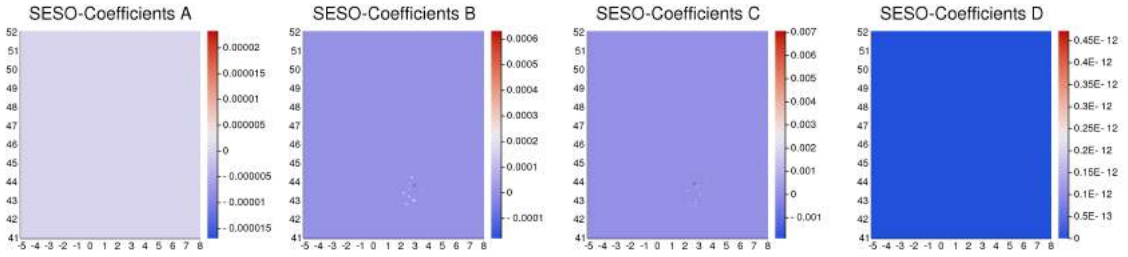


Figure 5.8: The original coefficients at SESO ($\alpha = 0.8, T = 1$).

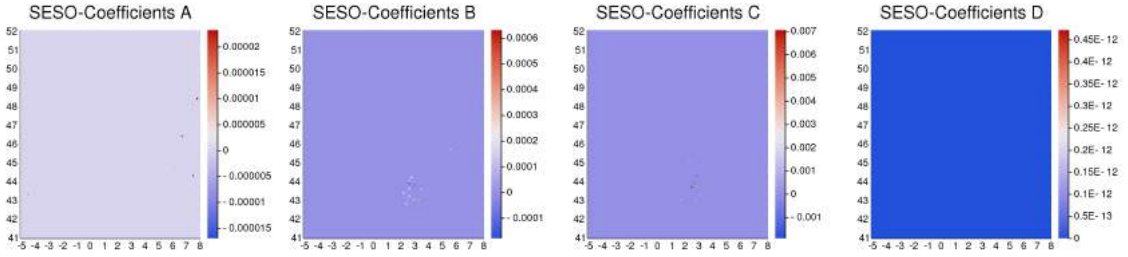


Figure 5.9: The retrieved coefficients at SESO ($\alpha = 0.8, T = 1$).

Table 5.2 show the comparison between solving the original LPs and solving the projected LPs.

Region	k	m	n	oriCPU	prjCPU	oriObj	retObj	ratio
SEE	2771	3288	64740	5132.47	2759.04	4.7671e+01	4.0080e+01	84.08%
SEO	2771	3288	64740	7969.83	4531.27	3.6834e+01	3.0660e+01	83.24%
SERAA	2771	3288	64740	4200.40	2062.60	5.2123e+01	4.3871e+01	84.17%
SESO	2771	3288	64740	4508.10	2548.27	5.3940e+01	4.5471e+01	84.30%

Table 5.1: Solving the projected LP instances with $\varepsilon = 0.2$ for each region. Here, the dimension of the projection space is $k = \lceil 10\varepsilon^{-2} \ln n \rceil + 1$; ratio is the proportional between prjCPU and oriObj.

Finally, let us compare the expected quantile-loss when we use the *original* optimal solution and the *retrieved* optimal solution. Recall that our dataset has 4384 observations ans is partitioned into training set and testing set with 3288 and 1096 records, respectively. Therefore the expected quantile-loss on the training set and testing set are computed as follows

$$\mathsf{TrainLoss} = \frac{1}{3288} \sum_{i=1}^{3288} \ell_\tau(y_i - x_i^\mathsf{T} \beta_\tau) \quad \text{and} \quad \mathsf{TestLoss} = \frac{1}{1096} \sum_{i=3289}^{4384} \ell_\tau(y_i - x_i^\mathsf{T} \beta_\tau).$$

The columns with prefix ori is the result obtained by original solution, whereas the column with prefix ret is the result obtained by retrieved solution.

Region	oriTrainLoss	retTrainLoss	oriTestLoss	retTestLoss
SEE	0.0144984	0.0150665	0.0135265	0.0144282
SEO	0.0112025	0.0115962	0.0113749	0.0119286
SERAA	0.0158524	0.0165415	0.0146666	0.0146811
SESO	0.0164052	0.0168556	0.0153827	0.0157807

Table 5.2: Comparing the expected quantile-loss of the original solution and the retrieved solution on the training set and testing set.

Chapter 6

Conclusion

Contents

6.1 Summary

This thesis focused on formulating and applying some kind of quantile regression problems. In contrast to minimizing the MSE loss function, minimizing the quantile loss function could be *translated* to solving a linear programming. This takes advantages of the algorithms which have been widely developed for solving linear programming efficiently.

Given a high-dimension dataset, apart from using PCA to extract its main features, we introduced in Chapter 3 the workflow to generate new dataset which is quantile invariant in comparing with the original one. Our approach uses quantile regression framework which eventually leads to solving a large linear programming problem.

Motivated by the importance of maintaining the balance between production and consumption in energy markets, we proposed a model that aims at forecasting the quantiles of wind power of the regions in France based on the meteorological data. This model is not only used to provide prediction intervals, but also used to determine the weather stations those have the most impacts on forecasting wind power for a given region (see Figure 4.3). This means that we could significantly reduce the dimension of problems and improve the predictions by building new model that only use data from the involved weather stations.

As one of the main results, we recalled the the recent results in using random projections solving

LPs and used this technique to solve the quantile regression problems. Applied to the problem of forecasting the quantiles of wind power generation, using random projection technique gave us promising results as we see at the end of Chapter 5. We would note that random projections could also be used to solve other important optimization problems, including integer programming, convex optimization with linear constraints, membership and approximate nearest neighbor and trust-region sub-problems.

6.2 Further works

Firstly, regarding to generating quantile invariant data, we would complete the idea described in Section 3.4 to capture quantiles of a given high-dimension dataset. In particular, in the highest level, the matrix $\boldsymbol{A}$ and the uniform vectors used to build $\boldsymbol{A}$, *i.e.*, the columns of $\boldsymbol{X}$, should be variables. Moreover, as introduced, our workflow only approximately preserved the quantiles, the next step is therefore to formulate a workflow to approximately preserve also the correlation. In an ideal case, given a high-dimension dataset, we could infer the coefficients so that by sampling some points in the PCs space, one can generate new dataset which has the same spatial correlation and quantiles as the original one w.a.h.p. .

Next, for wind power forecast, we would try the model where the power depends only in the cubes and squares of wind speed due to the fact that the obtained coefficients $C_{i,j}$'s and $D_{i,j}$'s are almost zeros everywhere. In addition, we would use an appropriate benchmark for evaluating the obtained models, *i.e.*, the benchmark allows us to compare our models to the current-used one.

Bibliography

[1] D. Achlioptas, *Database-friendly random projections: Johnson-Lindenstrauss with binary coins*, Journal of Computer and System Sciences, 66(4):671-687, 2003.

[2] Ademe, *Analysis of the french wind power sector: overview, prospective analysis and strategy*, 2017.

[3] S. Boyd, L. Vandenberghe, *Convex optimization*, Cambridge University Press, Cambridge, 2004.

[4] T. Burton, *Wind Energy Handbook*, John Wiley and Sons Press, 2001, page 45.

[5] S. Dasgupta, A. Gupta, *An elementary proof of a theorem of Johnson and Lindenstrauss*, Random structures and algorithms, volume 22, issue 1, page 60-65, 2003.

[6] R. Fourer, *Linear Programming Software Survey 2017*, INFORMS, volume 44.

[7] T. Hastie, R. Tibshirani, J. Friedman, *The elements of statistical learning*, Second edition, Springer Press, New York, 2013.

[8] IBM, *IBM ILOG CPLEX Optimization Studio CPLEX User's Manual*, IBM 2015.

[9] P. Indyk, R. Motwani, *Approximate nearest neighbors: Towards removing the curse of dimensionality*, Proceedings of the Thirtieth Annual ACM Synposium on Theory of Computing, pages 604-613, New York, 1998.

[10] W. Johnson, J. Lindenstrauss, *Extensions of Lipschitz mapping into a Hilbert space*, Conference in modern analysis and probability, volume 26 of Contemporary Mathematics, page 189-206.

[11] N. Karmarkar, *A new polynomial-time algorithm for linear programming II*, Combinatorica. 4. page 373-395.

[12] R. Koenker, *Quantile regression*, Cambridge University Press, Cambridge, 2005.

[13] R. Koenker, Pin Ng, *Inequality constrained quantile regression*, Sankhyā: The Indian Journal of Statistics (2003-2007) Volume 67, No 2, Quantile Regression and Related Methods (May, 2005), page 418-440

[14] J. Matoušek, *On variants of the Johnson-Lindenstrauss lemma*, Random Structures and Algorithms, volume 33, issue 2, page 142-156, 2008.

[15] J. Nocedal, S.J. Wright, *Numerical optimization*, Second edition, Springer Press, New York, 2006.

[16] RTE, *Europe's biggest transmission system*, RTE in figures, 2017.

[17] K.K. Vu, *Random projection for high-dimension optimization*, PhD thesis, University Paris-Saclay, 2016.

[18] K.K. Vu, P.L. Poirion, L. Liberti, *Random projections for linear programming*, Mathematics of Operations Research, accepted.