Insert here your thesis' task.

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Bachelor's thesis

Probabilistic algorithms for computing the LTS estimate

Martin Jenč

Department of Applied Mathematics
Supervisor: Ing. Karel Klouda, Ph.D.

March 8, 2019

THANKS to everybody

Acknowledgements

THANKS to everybody

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as school work under the provisions of Article 60(1) of the Act.

In Prague on March 8, 2019 . . . .

Citation of this thesis

Jenč, Martin. *Probabilistic algorithms for computing the LTS estimate*. Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2019.

Abstrakt

V několika větách shrňte obsah a přínos této práce v českém jazyce.

Klíčová slova LTS odhad, lineÃąrnÃŋ regrese, optimalizace, nejmenÅąÃŋ usekanÃľ ÄŊtvrece, metoda nejmenÅąÃŋch ÄŊtvercÅŕ, outliers

Abstract

The least trimmed squares (LTS) method is a robust version of the classical method of least squares used to find an estimate of coefficients in the linear regression model. Computing the LTS estimate is known to be NP-hard, and hence suboptimal probabilistic algorithms are used in practice.

Keywords LTS, linear regressin, robust estimator, least trimmed squares, ordinary least squares, outliers, outliers detection

Contents

List of Figures

Introduction

CHAPTER 1

Linear Regression

1.1 Description

1.2 Computation

1.3 Downfalls

Chapter 2

The Least trimmed squares

2.0.1 Objective function

2.0.1.1 Problems

Chapter 3

Algorithms

3.1 FAST-LTS

In this section we will introduce FAST-LTS algorithm[1]. It is, as well as in other cases, iterative algorithm. We will discuss all main components of the algorithm starting with its core idea called concentration step which authors simply call C-step.

3.1.1 C-step

We will show that from existing LTS estimate $\boldsymbol{\hat{w}_{old}}$ we can construct new LTS estimate $\boldsymbol{\hat{w}_{new}}$ which objective function is less or equal to the old one. Based on this property we will be able to create sequence of LTS estimates which will lead to better results.

Theorem 1. Consider dataset consisting of $\boldsymbol{x_1}, \boldsymbol{x_2} \ldots, \boldsymbol{x_n}$ explanatory variables where $\boldsymbol{x_i} \in \mathbb{R}^p, \forall \boldsymbol{x_i} = (x_1^i, x_2^i, \ldots, x_p^i)$ where $x_1^i = 1$ and its corresponding $y_1, y_2, \ldots, y_n$ response variables. Let's also have $\boldsymbol{\hat{w}_0} \in \mathbb{R}^p$ any p-dimensional vector and $H_0 = \{h_i; h_i \in \mathbb{Z}, 1 \leq h_i \leq n\}, |H_0| = h$. Let's now mark $RSS(\boldsymbol{\hat{w}_0}) = \sum_{i \in H_0}(r_0(i))^2$ where $r_0(i) = y_i - (w_1^0 x_1^i + w_2^0 x_2^i + \ldots + w_p^0 x_p^i)$. Let's take $\hat{n} = \{1, 2, \ldots, n\}$ and mark $\pi : \hat{n} \to \hat{n}$ permutation of $\hat{n}$ such that $|r_0(\pi(1))| \leq |r_0(\pi(2))| \leq \ldots \leq |r_0(\pi(n))|$ and mark $H_1 = \{\pi(1), \pi(2), \ldots \pi(h)\}$ set of h indexes corresponding to h smallest absolute residuals $r_0(i)$. Finally take $\boldsymbol{\hat{w}_1^{OLS(H_1)}}$ ordinary least squares fit on H_1 subset of observations and its corresponding $RSS(\boldsymbol{\hat{w}_1}) = \sum_{i \in H_1}(r_1(i))^2$ sum of least squares. Then

$$RSS(\boldsymbol{\hat{w}_1}) \leq RSS(\boldsymbol{\hat{w}_0}) \tag{3.1}$$

Proof. Because we take h observations with smallest absolute residuals r_0, then for sure $\sum_{i \in H_1}(r_0(i))^2 \leq \sum_{i \in H_0}(r_0(i))^2 = RSS(\boldsymbol{\hat{w}_0})$. When we take into account that Ordinary least squares fit OLS_{H_1} minimize objective function of H_1 subset of observations, then for sure $RSS(\boldsymbol{\hat{w}_1}) = \sum_{i \in H_1}(r_1(i))^2 \leq$

$\sum_{i \in H_1}(r_0(i))^2$. Together we get

$$RSS(\boldsymbol{\hat{w}_1}) = \sum_{i \in H_1}(r_1(i))^2 \leq \sum_{i \in H_1}(r_0(i))^2 \leq \sum_{i \in H_0}(r_0(i))^2 = RSS(\boldsymbol{\hat{w}_0})$$

□

Corollary 2. Based on previous theorem, using some $\boldsymbol{\hat{w}^{OLS(H_{old})}}$ on H_{old} subset of observations we can construct H_new subset with corresponding $\boldsymbol{\hat{w}^{OLS(H_{new})}}$ such that $RSS(\boldsymbol{\hat{w}^{OLS(H_{new})}}) \leq RSS(\boldsymbol{\hat{w}^{OLS(H_{old})}})$. With this we can apply above theorem again on $\boldsymbol{\hat{w}^{OLS(H_{new})}}$ with H_{new}. This will lead to the iterative sequence of $RSS(\boldsymbol{\hat{w}_{old}}) \leq RSS(\boldsymbol{\hat{w}_{new}}) \leq \ldots$. One step of this process is described by following pseudocode. Note that for C-step we actually need only $\boldsymbol{\hat{w}}$ without need of passing H.

Algorithm 1: C-step

Input: dataset consiting of $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ and $\boldsymbol{y} \in \mathbb{R}^{n \times 1}$, $\boldsymbol{\hat{w}_{old}} \in \mathbb{R}^{p \times 1}$

Output: $\boldsymbol{\hat{w}_{new}}$, H_{new}

$R \leftarrow \emptyset$;
for $i \leftarrow 1$ ***to*** n **do**
  $R \leftarrow R \cup \{|y_i - \boldsymbol{\hat{w}_{old}} \boldsymbol{x_i}^T|\}$;
end
$H_{new} \leftarrow$ select set of h smallest absolute residuals from R;
$\boldsymbol{\hat{w}_{new}} \leftarrow OLS(H_{new})$;
return $\boldsymbol{\hat{w}_{new}}$, H_{new};

Observation 3. Time complexity of algorithm C-step 1 is the same as time complexity as OLS. Thus $O(p^2n)$ ***TODO***

Lemma 4. Time complexity of OLS on $\boldsymbol{X}^{n \times p}$ and $\boldsymbol{Y}^{n \times 1}$ is $O(p^2n)$.

Proof. Normal equation of OLS is $\boldsymbol{\hat{w}} = (\boldsymbol{X^T X})^{-1}\boldsymbol{X^T Y}$. Time complexity of matrix multiplication $\boldsymbol{A}^{m \times n}$ and $\boldsymbol{B}^{n \times p}$ is $\sim \mathcal{O}(mnp)$. Time complexity of matrix $\boldsymbol{C}^{m \times m}$ is $\sim \mathcal{O}(m^3)$ So we need to compute $\boldsymbol{A} = \boldsymbol{X^T X} \sim \mathcal{O}(p^2 n)$ and $\boldsymbol{B} = \boldsymbol{X^T Y} \sim \mathcal{O}(pn)$ and $\boldsymbol{C} = \boldsymbol{A}^{-1} \sim \mathcal{O}(p^3)$ and finally $\boldsymbol{CB} \sim \mathcal{O}(p^2)$. That gives us $\mathcal{O}(p^2 n + pn + p^3 + p^2)$. Because $\mathcal{O}(p^2 n)$ and $\mathcal{O}(p^3)$ asymptotically dominates over $\mathcal{O}(p^2)$ and $\mathcal{O}(pn)$ we can write $\mathcal{O}(p^2 n + p^3)$.

TODO CO zo toho je vic? Neni casove narocnejsi vynasobeni $\boldsymbol{X^T X}$ nez inverze, kdyz bereme v uvahu $n >> p$??? □

Proof. In C-step we must compute n absolute residuals. Computation of one absolute residual consists of matrix multiplication of shapes $1 \times p$ and $p \times 1$ that gives us $\mathcal{O}(p)$. Rest is in constant time. So time of computation n residuals is $\mathcal{O}(np)$. Next we must select set of h smallest residuals which can be done in $\mathcal{O}(n)$ using modification of algorithm QuickSelect. reference: ***TODO*** Finally we must compute $\hat{w}$ OLS estimate on h subset of data. Because h is linearly dependent on n, we can say that it is $\mathcal{O}(p^2n + p^3)$ which is asymptotically dominant against previous steps which are $\mathcal{O}(np + n)$. □

As we stated above, repeating algorithm C-step will lead to sequence of $\boldsymbol{\hat{w}_1}, \boldsymbol{\hat{w}_2} \dots$ on subsets $H_1, H_2 \dots$ with corresponding residual sum of squares $RSS(\boldsymbol{\hat{w}_1}) \geq RSS(\boldsymbol{\hat{w}_2}) \geq \dots$. One could ask if this sequence will converge, so that $RSS(\boldsymbol{\hat{w}_i}) == RSS(\boldsymbol{\hat{w}_{i+1}})$. Answer to this question will be presented by the following theorem.

Theorem 5. Sequence of C-step will converge to $\boldsymbol{\hat{w}_m}$ after maximum of $m = \binom{n}{h}$ so that $RSS(\boldsymbol{\hat{w}_m}) == RSS(\boldsymbol{\hat{w}_n}), \forall n \geq m$ where n is number of data samples and h is size of subset H_i.

Proof. Since $RSS(\boldsymbol{\hat{w}_i})$ is non-negative and $RSS(\boldsymbol{\hat{w}_i}) \leq RSS(\boldsymbol{\hat{w}_{i+i}})$ the sequence will converge. $\boldsymbol{\hat{w}_i}$ is computed out of subset $H_i \subset \{1, 2, \dots, n\}$. When there is finite number of subsets of size h out of n samples, namely $\binom{n}{h}$, the sequence will converge at the latest after this number of steps. □

Above theorem gives us clue to create algorithm described by following pseudocode.

Algorithm 2: Repeat-C-step

Input: dataset consiting of $\boldsymbol{X} \in \mathbb{R}^{n \times p}$ and $\boldsymbol{y} \in \mathbb{R}^{n \times 1}$, $\boldsymbol{\hat{w}_{old}} \in \mathbb{R}^{p \times 1}$, H_0
Output: $\boldsymbol{\hat{w}_{final}}$, H_{final}

$\boldsymbol{\hat{w}_{new}} \leftarrow \emptyset$;
$H_{new} \leftarrow \emptyset$;
$RSS_{new} \leftarrow \infty$;
while $True$ **do**
  $RSS_{old} \leftarrow RSS(\boldsymbol{\hat{w}_{old}})$;
  $\boldsymbol{\hat{w}_{new}}$, $H_{new} \leftarrow \boldsymbol{X}$, $\boldsymbol{y}$, $\boldsymbol{\hat{w}_{old}}$;
  $RSS_{new} \leftarrow RSS(\boldsymbol{\hat{w}_{new}})$;
  if $RSS_{old} == RSS_{new}$ **then**
    break
  end
  $\boldsymbol{\hat{w}_{old}} \leftarrow \boldsymbol{\hat{w}_{new}}$
end
return $\boldsymbol{\hat{w}_{new}}$, H_{new};

It is important to note, that although maximum number of steps of this algorithm is $\binom{n}{h}$ in practice it is very low, most often under 20 steps. ***TODO*** nejaky hezky grafik ktery to ukazuje.... That is not enough for the algorithm $Repeat-C-step$ to converge to global minimum, but it is necessary condition. That gives us an idea how to create the final algorithm. [1]

Choose a lot of initial subsets H_1 and on each of them apply algorithm Repeat-C-step. From all converged subsets with corresponding $\hat{w}$ estimates choose that which has lowest $RSS(\hat{w})$.

Before we can construct final algorithm we must decide how to choose initial subset H_1 and how many of them mean "*a lot of*". First let's focus on how to choose initial subset H_1.

3.1.2 Choosing initial H_1subset

It is important to note, that when we choose H_1 subset such that it contains outliers, then iteration of C-steps usually won't converge to good results, so we should focus on methods with non zero probability of selecting H_1 such that it won't contain outliers. There are a lot of possibilities how to create initial hH_1 subset. Lets start with most trivial one.

3.1.2.1 Random selection

Most basic way of creating H_1 subset is simply to choose random $H_1 \subset \{1, 2, \dots, n\}$. Following observation will show that it not the best way.

Observation 6. With increasing number of data samples, thus with increasing n, the probability of choosing among m random selections of $H_{1_1}, \dots, H_{1_m}$ the probability of selecting at least one H_{1_i} such that its corresponding data samples does not contains outliers, goes to 0.

Proof. Consider dataset of n containing $\epsilon > 0$ relative amount of outliers. Let $h = (n + p + 1)/2$ and m is number of selections random $|H| = h$ subsets. Then

$$\begin{aligned}
P(one\ random\ data\ sample\ not\ outliers) &= (1 - \epsilon) \\
P(one\ subset\ without\ outliers) &= (1 - \epsilon)^h \\
P(one\ subset\ with\ at\ least\ one\ outlier) &= 1 - (1 - \epsilon)^h \\
P(m\ subsets\ with\ at\ least\ one\ outlier\ in\ each) &= (1 - (1 - \epsilon)^h)^m \\
P(m\ subsets\ with\ at\ least\ one\ subset\ without\ outliers) &= 1 - (1 - (1 - \epsilon)^h)^m
\end{aligned}$$

Because $n \to \infty \Rightarrow (1 - \epsilon)^h \to 0 \Rightarrow 1 - (1 - \epsilon)^h \to 1 \Rightarrow (1 - (1 - \epsilon)^h)^m \to 1 \Rightarrow 1 - (1 - (1 - \epsilon)^h)^m \to 0$ □

That means that we should consider other options of selecting H_1 subset. Actually if we would like to continue with selecting some random subsets, previous observation gives us clue, that we should choose it independent of n. Authors of algorithm came with such solution and it goes as follows.

3.1.2.2 P-subset selection

Let's choose subset $J \subset \{1, 2, \dots, n\}, |J| = p$. Next compute rank of matrix $\boldsymbol{X}_{J:}$. If $rank(\boldsymbol{X}_{J:}) < p$ add randomly selected rows to $\boldsymbol{X}_{J:}$ without repetition until $rank(\boldsymbol{X}_{J:}) = p$. Let's from now on suppose that $rank(\boldsymbol{X}_{J:}) = p$.

Next let us mark $\hat{\boldsymbol{w}}_{\mathbf{0}} = OLS(J)$ and corresponding $(r_0(1)), (r_0(2)), \ldots, (r_0(n))$ residuals. Now mark $\hat{n} = \{1, 2, \ldots, n\}$ and let $\pi : \hat{n} \to \hat{n}$ be permutation of $\hat{n}$ such that $|r(\pi(1))| \leq |r(\pi(2))| \leq \ldots \leq |r(\pi(n))|$. Finally put $H_1 = \{\pi(1), \pi(2), \ldots\pi(h)\}$ set of h indexes corresponding to h smallest absolute residuals $r_0(i)$.

Observation 7. With increasing number of data samples, thus with increasing n, the probability of choosing among m random selections of $J_{1_1}, \ldots, J_{1_m}$ the probability of selecting at least one J_{1_i} such that its corresponding data samples does not contains outliers, goes toho

$$1 - (1 - (1 - \epsilon)^h)^m > 0$$

Proof. Similarly as in previous observation. □

Note that there are other possibilities of choosing H_1 subset other than these presented in [1]. We'll properly discuss them in chapter ***TODO***.

Last missing piece of the algorithm is determining number of m initial H_1 subsets, which will maximize probability to at least one of them will converge to good solution. Simply put, the more the better. So before we will answer this question properly, let's discuss some key observations about algorithm.

3.1.3 Speed-up of the algorithm

In this section we will describe important observations which will help us to formulate final algorithm. In two subsections we'll briefly describe how to optimize current algorithm.

3.1.3.1 Selective iteration

Input: A finite set $A = \{a_1, a_2, \ldots, a_n\}$ of integers
Output: The largest element in the set

```
max ← a_1
for i ← 2 to n do
    if a_i > max then
        max ← a_i
    end
end
return max
```

Algorithm 8 is a greedy change-making algorithm (Slide 19 in Class Slides).

Algorithm 14 and Algorithm 13 will find the first duplicate element in a sequence of integers.

```
Input: A set C = {c_1, c_2, ..., c_r} of denominations of coins, where
       c_i > c_2 > ... > c_r and a positive number n
Output: A list of coins d_1, d_2, ..., d_k, such that sum_{i=1}^{k} d_i = n and k is
        minimized
C ← ∅
for i ← 1 to r do
    while n ≥ c_i do
        C ← C ∪ {c_i}
        n ← n − c_i
    end
end
return C
```

```
Input: A sequence of integers ⟨a_1, a_2, ..., a_n⟩
Output: The index of first location witht he same value as in a
        previous location in the sequence
location ← 0
i ← 2
while i ≤ n and location = 0 do
    j ← 1
    while j < i and location = 0 do
        if a_i = a_j then
            location ← i
        else
            j ← j + 1
        end
    end
    i ← i + 1
end
return location
```

```
Input: A sequence of integers ⟨a_1, a_2, ..., a_n⟩
Output: The index of first location witht he same value as in a
        previous location in the sequence
location ← 0
i ← 2
while i ≤ n ∧ location = 0 do
    j ← 1
    while j < i ∧ location = 0 do
        if a_i = a_j then  location ← i

        else  j ← j + 1

    end
    i ← i + 1
end
return location
```

In this section we will describe FAST-LTS algorithm and its main properties. The main idea of this algorithm is based on the fact that from one approximation of the algorithm we can compute another which can have lower objective function. TA DAAAAAAAAAAAAAAAAA Theorem 1: [2] Let w0 ... wp be the LTS estimate. for each data sample we can compute —y-wx—

3.2 Exact algorithm

3.3 Feasible solution

3.4 MMEA

3.5 Branch and bound

3.6 Adding row

Chapter 4

Experiments

4.1 Data

4.2 Results

4.3 Outlier detection

Conclusion

Bibliography

[1] Rousseeuw, P. J.; Driessen, K. V. An Algorithm for Positive-Breakdown Regression Based on Concentration Steps. In *Data Analysis: Scientific Modeling and Practical Application*, edited by M. S. W. Gaul, O. Opitz, Springer-Verlag Berlin Heidelberg, 2000, pp. 335–346.

[2] Rybicka, J. *LaTeX pro začátečníky*. Brno: Konvoj, third edition, ISBN 80-7302-049-1.

Appendix A

Datasets

GUI Graphical user interface

XML Extensible markup language

Appendix **B**

Contents of enclosed CD

```
readme.txt ........................ the file with CD contents description
exe ................................. the directory with executables
src ................................. the directory of source codes
  wbdcm ............................. implementation sources
  thesis ............. the directory of LaTeX source codes of the thesis
text ................................ the thesis text directory
  thesis.pdf .................. the thesis text in PDF format
  thesis.ps .................... the thesis text in PS format
```