FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

BACHELOR THESIS

Matvei Slavenko

Behavior of Total Least Squares Method for Models With Multiple Observations

Department of Numerical Mathematics

Supervisor of the bachelor thesis: RNDr. Iveta Hnětynková, Ph.D.
Study programme: General Mathematics
Study branch: Stochastics

Prague 2020

This is not a part of the electronic version of the thesis, do not scan!

I declare that I carried out this bachelor thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In .... date ....

Author's signature

First of all, my sincere gratitude belongs to my supervisor, Iveta Hnětynková. Her support, patience, attention and guidance made this work possible and greatly motivated me during the whole process of writing.

I would like to thank Jakub Klíč, who has been an ideal room-mate during these years.

I am very grateful to Nataliya M. Freyberg and Alexey S. Lakhtin, for it was them who initially inspired my interest to mathematics.

I also thank my friends and family for their support. I express my special feelings of thanks to Kristina Mozyreva and Jakub Grulich.

Title: Behavior of Total Least Squares Method for Models With Multiple Observations

Author: Matvei Slavenko

Department: Department of Numerical Mathematics

Supervisor: RNDr. Iveta Hnětynková, Ph.D., Department of Numerical Mathematics

Abstract: Linear approximation problems arise in various applications and can be solved by a large variety of methods. One of such methods is total least squares (TLS), an approach that allows to correct errors both in the linear model and available set of observations. In this work we collect and compare the main theoretical results related to TLS with multiple right-hand side. Particularly we describe the classification of TLS problems and summarise the solvability analysis that has currently been spread over various sources. The second part of the work is dedicated to an approach called core data reduction (CDR) and proof-of-concept programme demonstrating the CDR numerical behaviour.

Keywords: total least squares, multiple right-hand side, multiple observations, linear approximation problems, orthogonal regression, errors-in-variables regression, core problem, core data reduction, core transformation

Contents

Introduction

Linear approximation problems appear in many fields: in medicine (for instance, radiology and tomography), physics, economics, finances, geology, image processing etc. (see [1, Introduction] for more details). Data collected in real world applications are noisy, contain rounding, sampling, modelling and other errors. The need to compensate for them leaded to development of many data-correction techniques that are able to deliver reasonable approximate solutions.

In this work we will focus on the approximation problems with multiple right-hand side in the form:

$$AX \approx B,$$
$$A \in \mathbb{R}^{m \times n}, \qquad X \in \mathbb{R}^{n \times d}, \qquad B \in \mathbb{R}^{m \times d}.$$

Please note that in our setting the right-hand side is represented by a *matrix* instead of a vector, the columns of B correspond to d available measurements, hence the name *multiple right-hand side or observations.*

One of the most well-known methods for solving linear approximation problems is the *least squares (LS)* method. The method allows to compensate for the errors in the right-hand side: it seeks to find the smallest perturbation of B that will make the system compatible. In many applied tasks, however, the errors are present both in the observations B and in the model A. *Total least squares (TLS)* introduced already in [2] and [3] is one of techniques dealing with such type of errors. It is worth noting that in different fields the method is known under different names. For instance, in the context of statistics it is often called *orthogonal regression* or *error-in-variables regression.*

TLS method has been studied for a long time and today it could be considered one of "classic" approaches (see, e.g., [4, Section 6.3]). The monograph [1] dedicated to the TLS problem and considered the key source was published in 1991. It presents an extensive analysis of the TLS problem and a deep study of its computational aspects. The book focuses mainly on the algebraic properties of the problem, but certain statistical aspects are discussed, too. The so-called *classical TLS algorithm* is described and presented there (see [1, Section 3.6]).

The analysis reveals that TLS problems with multiple right-hand side are particularly difficult and cunning. Despite the long history of TLS, the analysis provided in [1] was not exhaustive: the introduced classification turned out to be incomplete and limited to special cases only. This gap has been successfully filled in the following 30 years. For instance, one of the articles [5] attempting to extend the analysis was published in 2000. It took 10 years more to finally mark the "Case of TLS Problem with Multiple Right Side" as "closed": the article [6] was published in 2011 presenting complete classification and exhaustive existence and uniqueness analysis for any data matrix (B, A).

In summary, the last monograph was published in 1991 and the analysis was not complete, the information acquired since then was spread and scattered among various articles, the notation and other technical details varied and were inconsistent. This lead us to the first goal pursued in this work. The first chapter of the thesis is meant to be an *overview* of the basic solvability analysis related to the TLS problems with multiple right-hand side, bringing together the results, comparing the notation etc. Taking into account the fact this work is merely a bachelor thesis, the idea was to write it the way that it could be used as an entry point for students and scholars interested in the topic and having no experience, without attempting to make the description complete.

In the last two decades, a new approach to the TLS problems has been developed: the so-called *core data reduction (CDR)* leading to the *core problem* (see [7], [8]). The idea of CDR is to extract the information necessary and sufficient for solving the TLS problem by applying orthogonal transformations to the data (B, A). For the multiple right-hand side, the approach is described in [8], and the second chapter of this work is dedicated to it. We describe the CDR step by step in terms of manipulations with the data matrix and also discuss its properties.

It must be said that so far the core problem concept has been studied only in the context of exact arithmetic, its numerical properties are still to be scrutinised. So, despite the fact that this work is mainly focused on theoretical aspects of TLS, for the sake of completeness and as a *proof-of-concept*, the third chapter is dedicated to numerical experiments demonstrating certain computational properties of CDR.

To sum up, this work is organised as follows:

- *Chapter 1* describes the TLS problems with multiple right-hand side, their properties, and analytical results related to their solvability.
- *Chapter 2* is devoted to the description of the core data reduction procedure.
- *Chapter 3* focuses on a proof-of-concept implementation of the CDR algorithm and includes several numerical experiments examining the core reduction behaviour.

1. Total Least Squares Problem

1.1 Preliminaries

In this section we are going to summarise the known definitions, theorems, and facts that will be necessary in the course of the work.

Theorem 1 (Singular value decomposition—SVD). *Let* $A \in \mathbb{R}^{m \times n}$, *rank*$(A) = r$. *Then there exist orthonormal matrices* U', V', *and a matrix* Σ' *such that:*

$$\begin{aligned}
U' &= (u'_1, \ldots, u'_m) \in \mathbb{R}^{m \times m} \\
V' &= (v'_1, \ldots, v'_n) \in \mathbb{R}^{n \times n} \\
\Sigma' &= \begin{pmatrix} \Sigma'_r & 0 \\ 0 & 0 \end{pmatrix} \in \mathbb{R}^{m \times n} \\
\Sigma'_r &= diag(\sigma'_1, \ldots, \sigma'_r) \\
\sigma'_1 &\geq \ldots \geq \sigma'_r > 0 \\
A &= U' \Sigma' V'^{\intercal}
\end{aligned} \tag{1.1}$$

Proof. For the proof, see [4, Theorem 2.4.1] or [9, p.150-151, Theorem 2.6.3] □

Definition 1 (Singular values, vectors, and triplets)**.** The representation (1.1) is called the *singular value decomposition* of the matrix A.
The numbers σ'_i are called *singular values* of A.
The vectors u'_i (v'_i) are called the *ith left (right) singular vectors*.
The triplet (u'_i, σ'_i, v'_i) is called the *ith singular triplet*.

Remark 1. Through the course of the work we will stick to the convention that dashed entities (U', Σ', V', σ'_i etc.) refer to the matrix A (unless another matrix is explicitly mentioned), while the equivalents with no dashes (U, Σ, V, σ_i etc.) are reserved for the extended matrix (B, A). The convention might initially seem slightly confusing, but we believe it is the best option.

Definition 2 (Dyadic decomposition)**.** Consider the setting as in Theorem 1. The following representation of the matrix A as a finite series is called the *dyadic decomposition*:

$$A = \sum_{i=1}^{r} \sigma'_i u'_i v_i'^{\intercal}. \tag{1.2}$$

Remark 2. The dyadic decomposition represents the matrix A of rank r as a sum of r matrices of rank one.

Theorem 2 (Eckart-Young-Mirsky matrix approximation theorem)**.** *Consider the matrix* A *from Theorem 1 and its dyadic decomposition* (1.2).
Let $k \in \mathbb{N}, k < r$. *Then the matrix*

$$A_k := \sum_{i=1}^{k} \sigma'_i u'_i v_i'^{\intercal} \tag{1.3}$$

is the best rank k approximation of the matrix A in the following sense:

$$\underset{rank(C)=k}{minimize} \|A - C\|_2 = \|A - A_k\|_2 = {\sigma'}^{k+1}, \tag{1.4}$$

$$\underset{rank(C)=k}{minimize} \|A - C\|_F = \|A - A_k\|_F = \sqrt{\sum_{i=k+1}^{r} {\sigma'}^2_i}. \tag{1.5}$$

Proof. For the proof, see [10] and [11]. □

Theorem 3 (Rouché-Capelli (Frobenius) theorem)**.** *Let*

$$A \in \mathbb{R}^{m \times n}, \qquad X \in \mathbb{R}^{n \times d}, \qquad B \in \mathbb{R}^{m \times d}. \tag{1.6}$$

Consider the system of equations $AX = B$. Then the following statements are equivalent:

1. *The system $AX = B$ has a solution,*
2. *$rank(B, A) = rank(A)$.*

Proof. For the proof, see [12, p.56 & Theorem 2.38]. □

1.2 Total Least Squares Problem

In this section we are going to define the concept of the total least squares problem. We will also define the TLS solutions and the TLS solutions minimal in norm. A cheat sheet summarising the key aspects of this section is available in Appendix A.

Definition 3 (Multidimensional total least squares problem)**.** Let

$$\begin{aligned} &A \in \mathbb{R}^{m \times n}, && X \in \mathbb{R}^{n \times d}, && B \in \mathbb{R}^{m \times d}; \\ &rank(B) = d, && m \geq n + d, && A^{\intercal} B \neq 0. \end{aligned} \tag{1.7}$$

Consider (an overdetermined) set of linear equations $AX \approx B$ in variables X or, equivalently,

$$(B, A) \begin{pmatrix} -I_d \\ X \end{pmatrix} \approx 0. \tag{1.8}$$

We call the matrices A and B the *system* and *observation* matrices respectively. The matrix (B, A) is called the *extended* or *data* matrix.

Let $\mathcal{R}(A) \nsubseteq \mathcal{R}(B)$. The *multidimensional total least squares (TLS)* seeks to

$$\underset{A', B'}{minimize} \ \|(B', A') - (B, A)\|_F \tag{1.9}$$

$$\text{subject to} \qquad \mathcal{R}(B') \subseteq \mathcal{R}(A').$$

The minimizing matrices (B', A') are called *corrected matrices.* We then call any matrix X' satisfying the condition

$$A'X' = B' \tag{1.10}$$

a *(generic) TLS solution*, and $(\Delta B, \Delta A) := (B', A') - (B, A)$ is then the corresponding *TLS correction.*

Remark 3. Note that in the case $\mathcal{R}(A) \subseteq \mathcal{R}(B)$, the problem is reduced to solving a compatible set of linear equations. Hence, the condition $\mathcal{R}(A) \nsubseteq \mathcal{R}(B)$ simply stresses the fact that we are solving an approximation problem.

Remark 4. The situation $A^\intercal B = 0$ means that the right and left sides are uncorrelated/orthogonal and the only meaningful solution is $X := 0$. The condition $A^\intercal B \neq 0$ therefore excludes this trivial case.

Remark 5. The conditions $m \geq n + d$ and $rank(B) = d$ are introduced for the sake of keeping the calculations more neat and unambiguous, and they are not too restrictive. The matrix B could always be preprocessed to meet the latter condition. As for the former constraint, zero rows could be added to the matrices A and B so that it is fulfilled.

Remark 6. The reason why we have to stress the solution is generic is explained in the Section 1.5. See [1, Section 3.4] and [6, Section 5] for more details.

Various sources describe the TLS problem in slightly different ways. Some other possible definitions could be found in [4, p.320, Section 6.3] and [6, Formulae (1.1)–(1.3)]. We defined the TLS problem for the Frobenius norm. Sometimes it is worth considering the problem for other norms. The most common case would be the 2-norm, but other options are possible as well (see, for instance, [13]). Now we define a solution minimal in the selected norm:

Definition 4 (TLS solution minimal in norm)**.** Consider the setting from Definition 3. We say X' is the *TLS solution minimal in Frobenius norm* if it satisfies

$$X' = \underset{X,\ A'X=B'}{argmin} \|X\|_F. \tag{1.11}$$

Solutions minimal in other norms (e.g. in 2-norm) are defined in the same manner.

1.3 SVD of Data Matrix (B, A)

In this section we will describe notation that will make our exposition easier and as transparent as possible. The paper [6] and the book [1] were the main sources used while writing this part. A cheat sheet summarising this section is available in Appendix A.

Definition 5 (p, q and e numbers for singular values)**.** Consider the matrices

$$A \in \mathbb{R}^{m\times n}, m \geq n + d; \qquad B \in \mathbb{R}^{m\times d}. \tag{1.12}$$

Let

$$(B, A) = U\Sigma V^\intercal \tag{1.13}$$

be the corresponding singular value decomposition. We define the numbers p, q, and e as follows:

$$\sigma_{n-q} > \underbrace{\sigma_{n-q+1} = \ldots = \sigma_n}_{q} = \underbrace{\sigma_{n+1} = \ldots = \sigma_{n+e}}_{e} > \sigma_{n+e+1}, \tag{1.14}$$

$$p := n - q, \tag{1.15}$$

$$\sigma_p := \sigma_{n-q}. \tag{1.16}$$

If $\sigma_n > \sigma_{n+1}$, we define $q := 0 =: e$.

Remark 7. Note that the definition is consistent in the following sense: $\sigma_p := \sigma_{n-q}$ is undefined if and only if $q \geq n$. Similarly, σ_{n+e+1} is undefined if and only if $e \geq d$.

Remark 8. The singular value σ_{n+1} plays a key role in the following discussions. In order to handle the (possible) multiplicity of σ_{n+1}, notation (1.14) was suggested in the paper [6, Formula (2.5)]. Provided that $\sigma_n = \sigma_{n+1}$, the numbers q and e represent *left-hand multiplicity* of σ_n and, correspondingly, *right-hand multiplicity* of σ_{n+1}.

Definition 6 (e-q matrix partitioning)**.** Consider the setting from Definition 5. We introduce so called *e-q partitioning* as follows:

$$V = \begin{array}{c} \overbrace{\hphantom{V_{11}\qquad V_{12}\qquad V_{13}}}^{\mathtt{d+n}} \\ \left(\begin{array}{ccc} V_{11} & V_{12} & V_{13} \\ \underbrace{V_{21}}_{\mathtt{n-q}} & \underbrace{V_{22}}_{\mathtt{e+q}} & \underbrace{V_{23}}_{\mathtt{d-e}} \end{array}\right) \end{array} \begin{array}{l} \}\,\mathtt{d} \\ \}\,\mathtt{n} \end{array} \tag{1.17}$$

For the sake of convenience, we also introduce the following notation:

$$\mathbf{V}_1 := (V_{12}, V_{13}) \in \mathbb{R}^{d\times(d+q)}, \tag{1.18}$$

$$\mathbf{V}_2 := (V_{22}, V_{23}) \in \mathbb{R}^{n\times(d+q)}. \tag{1.19}$$

Remark 9. The partitioning in Definition 6 brings together two consequent partitionings described in [6, Formulae (2.6), (4.5)].

Recall that we denote the singular values of the matrix A as σ'_i (see Theorem 1 and Remark 1).

Theorem 4 (e-q partitioning properties)**.** *Consider the setting as in Definition 5 and the e-q partitioning of the matrix V given by* (1.17)–(1.19)*.*
If $\sigma'_p := \sigma'_{n-q} > \sigma_{n-q+1} =: \sigma_{p+1}$, then

1. $\sigma_{n-q} > \sigma_{n-q+1}$;
2. $\mathbf{V}_1$ *is of full row rank, rank*$(\mathbf{V}_1) = d$*;*
3. V_{21} *is of full column rank, rank*$(V_{21}) = n - q$*.*

Proof. For the proof, see [1, p.32, Theorem 2.4; pp.64–65, Lemma 3.1] and [6, pp.750–751, Theorem 2.1]. □

Remark 10. Theorem 4 is a special case of one presented in the paper [6, pp.750–751, Theorem 2.1].

1.4 The TLS Problems Classification

In this section, we will summarise the TLS problems classification introduced in the paper [6]. We will also list the key properties of each problem class. The proofs could be found in [6] or, partially, in [1, Chapter 3]. Some proofs and ideas are also presented in Section 1.6 of the present work.

Definition 7 (The first and second class problems)**.** Consider the setting from Definition 3. Let (1.13) be the SVD of the extended matrix (B, A) and let (1.17) and (1.18) be the corresponding e-q partitioning of the matrix V from SVD.

1. We say the TLS problem belongs to *the first class* if and only if the matrix $\mathbf{V}_1$ is of full row rank (i.e. $rank(\mathbf{V}_1) = d$). The set of all first class problems is denoted as $\mathcal{F}$.
2. We say the TLS problem is of *the second class* if and only if the matrix $\mathbf{V}_1$ is rank deficient (i.e. $rank(\mathbf{V}_1) < d$). We denote the set of all second class problems as $\mathcal{S}$.

Remark 11. The classification and notation were suggested in paper [6, Definition 2.2].

The next definition is based on the matrices V_{12} and V_{13} from Definition 6. Hence it is worth recalling their dimensions: $V_{12} \in \mathbb{R}^{d\times(e+q)}$, $V_{13} \in \mathbb{R}^{d\times(d-e)}$.

Definition 8 (Partitioning of the first class problems)**.** Consider the setting from Definition 7 and let the problem be of the first class. We denote the set of all problems for which:

1. $rank(V_{12}) = e$ and $rank(V_{13}) = d - e$ (V_{13} is of full column rank) as $\mathcal{F}_1$,
2. $rank(V_{12}) > e$ and $rank(V_{13}) = d - e$ (V_{13} is of full column rank) as $\mathcal{F}_2$,
3. $rank(V_{12}) > e$ and $rank(V_{13}) < d - e$ (V_{13} is rank deficient) as $\mathcal{F}_3$.

Remark 12. $\mathcal{F}_1$, $\mathcal{F}_2$, and $\mathcal{F}_3$ are mutually disjoint, $\mathcal{F}_1 \cup \mathcal{F}_2 \cup \mathcal{F}_3 = \mathcal{F}$.

The singular values of the extended matrix reveal a lot of information about the possible solutions. The following definition introduces three possible scenarios that we differ.

Definition 9 (Singular values distribution scenarios)**.** Consider the setting from Definition 3. Let (1.13) be the SVD of the extended matrix (B, A) and let e-q numbers be defined as in Definition 5. We then say that the singular values σ_i:

1. have a *pretty pivot* if $\sigma_n > \sigma_{n+1}$;
2. have a *long tail* if $\sigma_p > \sigma_{p+1} = \ldots = \sigma_{n+1} = \ldots = \sigma_{d+n}$;
3. have a *general distribution* in all other cases, i.e. if $n \geq q > 0$ (i.e. $0 \leq p < n$), $e < d$, and

$$\sigma_1 \geq \ldots \geq \sigma_p > \sigma_{p+1} = \ldots = \sigma_{n+1} = \ldots = \sigma_{n+e} > \sigma_{n+e+1} \geq \ldots \geq \sigma_{n+d}.$$

We call the pretty pivot and long tail properties the *special properties.*

Remark 13. When we say “the singular values’ distribution of the problem”, we refer to the distribution of singular values of the extended matrix.

The following Theorems 5–8 summarise the results presented in the paper [6]. We begin with a theorem explaining the importance of the special properties.

Theorem 5 (Relation between $\mathcal{F}$ and the special properties)**.** *Consider a TLS problem. Let the problem be of the first class. If the singular values' distribution has:*

1. *a* pretty pivot*, the problem belongs to* $\mathcal{F}_1$ *and has a unique solution;*
2. *a* long tail*, the problem belongs to* $\mathcal{F}_1$*.*

The next three theorems describe the properties of each set of the TLS problems.

Theorem 6 ($\mathcal{F}_1$ properties)**.** *Let the problem be from the set* $\mathcal{F}_1$*. Then the TLS problem:*

1. *has a solution;*
2. *may have infinitely many solutions;*
3. *has a unique solution minimal in 2- and Frobenius norms given by* $X' = -\mathbf{V}_2\mathbf{V}_1^\dagger$*.*

Theorem 7 ($\mathcal{F}_2$ properties)**.** *Consider a TLS problem from the set* $\mathcal{F}_2$*. Then the TLS problem:*

1. *has a solution;*
2. *may have infinitely many solutions;*
3. *has a solution minimal in 2-norm;*
4. *has a solution minimal in Frobenius norm;*
5. *the solutions minimal in 2- and Frobenius norm may not coincide (moreover, the sets of solutions minimal in the norms may be disjoint);*
6. *may have infinitely many solutions minimal in 2- or Frobenius norm.*

Remark 14. The points 5 and 6 of the theorem are rather counter intuitive. See [6, Section 4.2] for a more detailed explanation.

Theorem 8 ($\mathcal{F}_3$ and $\mathcal{S}$ properties)**.** *Consider a TLS problem. Let the problem belong to the set* $\mathcal{F}_3$ *or* $\mathcal{S}$*. Then the problem has no TLS solution.*

Remark 15. It could be shown that for the problems from the sets $\mathcal{F}_3$ and $\mathcal{S}$ it is possible to find a correction with an arbitrary small norm that will still make the system compatible. Therefore, the minimum for the correction norm cannot be reached. See [6, Sections 4.3, 5]. The paper [3] (besides introducing the TLS problem) demonstrates an example of a one-dimensional ($d = 1$) TLS problem that has no minimal correction.

1.5 Comparison with Classic Approach

In this section, we will briefly compare the classic approach to analysing the TLS problem presented in [1, Chapter 3] with the modern results presented in [6] and which this work is based on.

Please note the difference in **notation**: Sabine Van Huffel's and Joos Vandewalle's analysis uses the extended matrix (A, B). Besides, the authors split the matrix V from the SVD to only 4 sub-matrices (we denote the corresponding matrices as V_{11}, V_{21}, $\mathbf{V}_1$ and $\mathbf{V}_2$, see Table B.2). A cheat sheet comparing the notation in the used resources is presented in Attachment B.

The book [1] provides an important statistical **perspective** on the TLS problem. Indeed, consider a matrix (B, A) of m observations of an exact but unobservable linear relation

$$A_0 X_0 = B_0. \tag{1.20}$$

Let the observations and the real data be in the following relation:

$$(B, A) = (B_0, A_0) + (B_e, A_e), \tag{1.21}$$

where (B_e, A_e) is an error matrix with independent identically distributed rows with zero mean and the covariance matrix $\sigma^2 I_{d+n}$. Then we could perceive the TLS problem as constructing an estimator for the matrix X_0 (errors-in-variables regression). The paper [6] focuses on the algebraic side of the TLS problem only.

The book [1] uses a slightly different TLS **problems classification**. In essence, it is a straight-forward generalisation of the one-dimensional TLS problems classification. The book describes *generic* and *non-generic* problems. The paper [6] introduced correspondingly the names *first* ($\mathcal{F}$) and *second* ($\mathcal{S}$) class problems instead. Besides that, the paper presented an additional classification for the problems of the first class, namely the sets $\mathcal{F}_1$–$\mathcal{F}_3$.

The definitions *generic* and *non-generic* **solutions** are also introduced in [1, Section 3.4]:

The generic solution is a TLS solution as it is described in Definition 3. Generic solution reduces the extended matrix (B, A) rank by d (for the correction matrix, correspondingly, $rank(\Delta B, \Delta A) = d$).

The non-generic solution *does not* solve the TLS problem as it is presented in Definition 3. The concept is one way to deal with the TLS problems that have no (generic) solution and is a straightforward generalisation of non-generic solutions for the one-dimensional TLS problems. For the corresponding correction matrix it holds that $rank(\Delta B, \Delta A) > d$, the (B, A) rank is reduced by more than d. The detailed description could be found in [1, Section 3.4] or in [6, Section 5].

The book [1] fully covers solving the $\mathcal{F}$-problems with **special properties**, namely a pretty pivot or a long tale property, and the $\mathcal{S}$-problems. Note that the authors claim that any $\mathcal{F}$-problem (or, using their terminology, any generic problem) has a generic solution. The paper [6] demonstrates that this is not the case (see also Theorem 8). For this reason, we opt for the $\mathcal{F}$-$\mathcal{S}$ terminology when classifying the TLS problems.

As for the $\mathcal{F}$-problems with the **general distribution** of singular values, the authors of [1] provide no linear algebraic analysis for them. However, they take into account the statistical angle and present interesting ways of dealing with such problems. Two scenarios are possible:

1. If the singular values $\sigma_{p+1}, \ldots, \sigma_{d+n}$ are *close* to each other, we could say that these are in fact the same singular value σ_{d+n} of the matrix (B_0, A_0) but perturbed with noise. Then the problem could be perceived as having a long tale property and we focus on minimizing the norm of the solution $\|X'\|$ instead of enforcing the minimality of the value $\|(\Delta B, \Delta A)\|$.
2. If the last singular values differ *significantly*, the system $AX \approx B$ is highly incompatible. Hence there are two ways to deal with it:
    (a) We could refuse solving the problem, saying it is not suitable for linear modelling.
    (b) We could solve the TLS problem anyway (hence opting to minimize the value $\|(\Delta B, \Delta A)\|$). The authors, however, provided no algebraic analysis for this case, claiming that: “Procedures... have not yet been fully analysed.” see [1, p. 66].

The paper [6] fully described the key properties of the problems with the general singular distribution. Nevertheless, it does not provide an algorithm for solving the problems from the subset $\mathcal{F}_2$.

To sum up:

$\mathcal{F}_1$, $\mathcal{S}$: fully analysed both in [1] and [6], the results are consistent;

$\mathcal{F}_2$, $\mathcal{F}_3$: the work [1] suggests calculating non-generic solutions and implies generic solutions exist (with no proof, however). The paper [6] proves the existence of generic TLS solutions for $\mathcal{F}_2$, but provides no way of calculating them, and shows generic solutions do not exist for $\mathcal{F}_3$.

1.6 Key Ideas of Constructing Solutions

In this section we will present the key ideas lying behind constructing the TLS solutions. We will limit ourselves to the problems where the singular values distributions have a pretty pivot or a long tale. The section is based on the paper [6, Sections 3.1–3.2]. Another description is presented in [1, Chapter 3].

One could perceive the general approach to constructing the TLS solutions as follows. According to the Rouché-Capelli theorem (Theorem 3), a system of linear equations $AX = B$ is compatible if and only if $rank(B, A) = rank(A)$, i.e. the columns of B are linearly dependent on the columns of A. We face the situation $rank(B, A) > rank(A)$ when it comes to solving the TLS problem $AX \approx B$. It then seems natural to reduce the value $rank(B, A)$ so that the equality is reached. When reducing the matrix rank, we want the perturbations to be as small as possible, which leads us to using Eckart-Young-Mirsky theorem (Theorem 2).

1.6.1 Pretty Pivot Case

We will start with looking at the most straight-forward case when $\sigma_n > \sigma_{n+1}$. We could vaguely describe it as "we need to get rid of the last d singular values of the extended matrix (A, B) and there is only one way to choose them".

Theorem 9 (Pretty pivot case properties)**.** *Consider the setting from Definition 3. Let* (1.13) *be the SVD of the extended matrix* (B, A) *and let* (1.17)–(1.18) *be the corresponding e-q partitioning of the matrix* V*.*

Assume $\mathbf{V}_1$ *is square and non-singular, and the problem has the pretty pivot property (i.e.* $\sigma_n > \sigma_{n+1}$*). Then*

1. $q = 0$, $p = n$;
2. *the problem belongs to the set* $\mathcal{F}_1$*;*
3. *the TLS correction, corrected matrices, and TLS solution are correspondingly given by the following equations:*

$$(\Delta B, \Delta A) = -U\Sigma \begin{pmatrix} 0 & \mathbf{V}_1 \\ 0 & \mathbf{V}_2 \end{pmatrix}^{\mathsf{T}}, \tag{1.22}$$

$$(B', A') = U\Sigma \begin{pmatrix} V_{11} & 0 \\ V_{12} & 0 \end{pmatrix}^{\mathsf{T}}, \tag{1.23}$$

$$X' = -\mathbf{V}_2\mathbf{V}_1^{-1}. \tag{1.24}$$

Proof. 1. Since $\sigma'_n > \sigma_{n+1}$, $q = 0$ by definition. Consequently, we obtain $p := n - q = n$.

2. The assumption that $\mathbf{V}_1$ is non-singular ensures that the problem belongs to the set $\mathcal{F}_1$.

3. (a) Note that the correction effectively nullifies the last d singular values (by turning the associated right-hand singular vectors to zero). With the help of the dyadic decomposition (Definition 2) we obtain:

$$\begin{aligned} (B, A) &= \sum_{i=1}^{n+d} \sigma_i u_i {v_i}^{\mathsf{T}}, \\ (\Delta B, \Delta A) &= -\sum_{i=n+1}^{n+d} \sigma_i u_i {v_i}^{\mathsf{T}}, \\ (B', A') &= (B, A) + (\Delta B, \Delta A) \\ &= \sum_{i=1}^{n} \sigma_i u_i {v_i}^{\mathsf{T}} = U\Sigma \begin{pmatrix} V_{11} & 0 \\ V_{12} & 0 \end{pmatrix}^{\mathsf{T}}. \end{aligned}$$

   (b) Eckart-Young-Mirsky theorem (Theorem 2) ensures that (1.23) is the unique rank n approximation of (B, A) with minimal $\|(\Delta B, \Delta A)\|_F$.

   (c) $\mathcal{N}(B', A') = \mathit{span}\,(\mathbf{V}_1^{\mathsf{T}}, \mathbf{V}_2^{\mathsf{T}})^{\mathsf{T}}$. Since $\mathbf{V}_1$ is square and non-singular,

$$(B', A') \begin{pmatrix} -\mathbf{V}_1 \\ -\mathbf{V}_2 \end{pmatrix} = 0$$

$$(B', A') \begin{pmatrix} -I \\ -\mathbf{V}_2\mathbf{V}_1^{-1} \end{pmatrix} = 0.$$

Hence $X' = -\mathbf{V}_2\mathbf{V}_1^{-1}$.

$\square$

Remark 16. The alternative description of the proof/construction of the solution for this special case could be found in [1, pp.51–53] and [6, p.751, Theorem 3.1].

Remark 17. Observing that by definition $q = 0$ and using Theorem 4, the assumptions $\sigma_n > \sigma_{n+1}$ and $\mathbf{V}_1$ is square and non-singular could be substituted with the assumption $\sigma'_n > \sigma_{n+1}$. For $d = 1$, it is known as the Golub-Van Loan (GVL) condition [4, Theorem 6.3.1] ensuring a unique TLS solution existence.

1.6.2 Long Tail Case

We will now proceed with applying the same approach to the case where

$$\sigma_1 \geq \ldots \geq \sigma_p > \sigma_{p+1} = \ldots = \sigma_{n+1} = \ldots = \sigma_{d+n} \geq 0, \tag{1.25}$$

which means that $e = d$. The setting could be roughly described as "we want to get rid of the last d singular values of the extended matrix (B, A); however, we have $d + q$ equal singular values to choose from". Observe that:

1. Two degenerated cases are possible within this setting:
    (a) If $p = n$ (i.e. $q = 0$), the problem is reduced to the previous case.
    (b) If $p = 0$ (i.e. $q = n$), all singular values of the extended matrix (B, A) are equal to each other. Hence, $\mathcal{R}(B) \perp \mathcal{R}(A)$ and the only reasonable solution is $X := 0$. This case *does not* satisfy the non-triviality condition $A^{\intercal}B = 0$.
2. Now let $0 < p < n$ (i.e. $n > q > 0$):
    (a) We could choose any d singular values from the last $d + q$ and subsequently use exactly the same approach as in the proof of Theorem 9 to construct the solution and corresponding correction matrix.
    (b) The 2- and Frobenius norms are unitary invariant. Therefore, we could apply an orthogonal linear transformation to the last $d + q$ right singular vectors and then use the approach from the previous point.
    (c) The latter means that the problem has *infinitely many* solutions. This in turn leads us to the necessity of constructing the TLS solution minimal in norm.

Theorem 10. *Consider the setting from Definition 3. Let* (1.13) *be the SVD of the extended matrix* (B, A) *and let* (1.17)–(1.18) *be the corresponding e-q partitioning of the matrix* V.

Assume $\mathbf{V}_1$ *is of full row rank (i.e. the problem is of the first class),* $q < n$, *and* $\sigma_p > \sigma_{p+1} = \ldots = \sigma_{n+d}$.

Let $Q_1 \in \mathbb{R}^{(d+q)\times q}$, $Q_2 \in \mathbb{R}^{(d+q)\times d}$ *be matrices such that*

$$Q := (Q_1, Q_2) \;\in\; \mathbb{R}^{(d+q)\times(d+q)}$$

is an orthogonal matrix and

$$(v_{p+1}, \dots, v_{n+d})Q =: \begin{pmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{pmatrix} (Q_1, Q_2) = \begin{pmatrix} 0 & \Gamma \\ Y & Z \end{pmatrix}. \tag{1.26}$$

Then the TLS correction, corrected matrices, and TLS solution minimal in 2- and Frobenius norms are correspondingly given by the following equations:

$$(\Delta B, \Delta A) = -(B, A) \begin{pmatrix} \Gamma \\ Z \end{pmatrix} \begin{pmatrix} \Gamma \\ Z \end{pmatrix}^\intercal, \tag{1.27}$$

$$(B', A') = (B, A) \left(I_{d+n} - \begin{pmatrix} \Gamma \\ Z \end{pmatrix} \begin{pmatrix} \Gamma \\ Z \end{pmatrix}^\intercal \right), \tag{1.28}$$

$$X' = -\mathbf{V}_2 \mathbf{V}_1^\dagger = (A^\intercal A - \sigma_{n+1}^2 I_n)^\dagger A^\intercal B. \tag{1.29}$$

Proof (certain highlights). For the full proof, see [6, Section 3.2] and [1, pp.60-64, Theorems 3.9-3.10]. Several aspects however deserve a closer look.

1. $\mathbf{V}_1$ is of full row rank. Therefore, there exists the matrix Q: for instance, we could use the QR-decomposition of the matrix $\mathbf{V}_1^\intercal$.

2. We will now focus on the correction matrix:

$$\begin{aligned} (\Delta B, \Delta A) &= -(B, A) \begin{pmatrix} \Gamma \\ Z \end{pmatrix} \begin{pmatrix} \Gamma \\ Z \end{pmatrix}^\intercal \\ &= -U\Sigma V^\intercal \begin{pmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{pmatrix} Q_2 Q_2^\intercal \begin{pmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{pmatrix}^\intercal \\ &= -U\Sigma \left(v_1, \dots, v_{d+n} \right)^\intercal \left(v_{n-q+1}, \dots, v_{n+d} \right) Q_2 Q_2^\intercal \begin{pmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{pmatrix}^\intercal \\ &= -U\Sigma \begin{pmatrix} 0 \\ I_{d+q} \end{pmatrix} Q_2 Q_2^\intercal \begin{pmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{pmatrix}^\intercal. \end{aligned}$$

   Note that again the correction effectively nullifies the last $d + q$ singular values. The fact that they are all equal allows us to rewrite it as follows:

$$= -\sigma_{n+1} \left(u_{p+1}, \dots, u_{n+d} \right) Q_2 Q_2^\intercal \begin{pmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \end{pmatrix}^\intercal. \tag{1.30}$$

□

Remark 18. It could be shown that the construction does not depend on the particular choice of the matrix Q. Hence, the minimal norm TLS solution X' is *unique.*

Remark 19. Again the condition $\sigma'_p > \sigma_{p+1} = \ldots = \sigma_{d+n}$ is sufficient for applying the theorem and therefore for existence of a unique TLS solution minimal in 2- and Frobenius norms.

1.6.3 General Distribution Case and Summary

We could deal with the $\mathcal{F}_1$ problems with the general distribution of singular values in the same manner. The details could be found in [6, Section 3.3].

The following theorem summarises the information about the behaviour of the $\mathcal{F}_1$ problems.

Theorem 11. *Consider the setting from Definition 3. Let* (1.13) *be the SVD of the extended matrix* (B, A) *and let* (1.17) *and* (1.18) *be the corresponding e-q partitioning of the matrix* V*. Let the problem be from the* $\mathcal{F}_1$ *set:* $rank(\mathbf{V}_1) = d$, $rank(V_{12}) = e$, *and* $rank(V_{13}) = d - e$.

Let $Q_1 \in \mathbb{R}^{(d+q)\times q}$, $Q_2 \in \mathbb{R}^{(d+q)\times d}$ *be matrices such that*

$$Q := (Q_1, Q_2) \in \mathbb{R}^{(d+q)\times(d+q)}$$

is an orthogonal matrix with the property (1.26)*. Then it holds that:*

1. $X' := -\mathbf{V}_2\mathbf{V}_1^{\dagger}$ *is the TLS solution minimal in 2- and Frobenius norms,*
2. *the corresponding correction is given by* $(\Delta B, \Delta A) = -(u_{p+1}, \ldots, u_{n+d})diag(\sigma_{p+1}, \ldots, \sigma_{n+d})Q_2Q_2^{\intercal}(v_{p+1}, \ldots, v_{n+d})^{\intercal}$.

Proof. For the proof, see [6, p. 762, Theorem 4.4]. ☐

Remark 20. The closed-form expressions for the TLS solutions are mainly of theoretical interest. When it comes to solving the TLS problems on practice, it is often a better idea to calculate the solution via solving the corrected system.

1.7 One-Dimensional ($d = 1$) Case

Originally, the concept of one-dimensional TLS problem was presented and analysed in the paper [3]. In this section we will show how the theory of multidimensional TLS problems we described above matches with the well-known one-dimensional case.

First of all, we should point out that for $d = 1$, $\mathbf{V}_1 \in \mathbb{R}^{1\times(1+q)}$, i.e. the matrix $\mathbf{V}_1$ is a row vector, the number of its elements equals the multiplicity of $\sigma_{d+n} := \sigma_{n+1}$. Secondly, it is easy to observe that there are only two possible scenarios for the problem's singular values distribution: it either has a pretty pivot, or a long tale. We will now explore the two cases.

In the **pretty pivot** case, both $q = e = 0$. Hence the matrix $\mathbf{V}_1 \in \mathbb{R}^{1\times(1+0)}$ is reduced to a real number (which is in fact the first element of the last right singular vector). Two configurations are possible:

$\mathbf{V}_1 \neq 0$ means the problem belongs to the first class. The problem has a pretty pivot property, hence it belongs to the set $\mathcal{F}_1$ and, as follows from the multidimensional theory, the problem has a unique solution (see Theorem 5).

$\mathbf{V}_1 = 0$ implies that by definition (see Definition 7) the problem belongs to the second class and has no solution, a non-generic solution is constructed instead.

If the problem's singular value distribution has a **long tale**, then:

1. $e = 1$;
2. $d - e = 1 - 1 = 0$, hence:
    (a) the matrix V_{13} has zero columns, $rank(V_{13}) = 0$;
    (b) $\mathbf{V}_1 = V_{12}$;
3. Depending on the value $rank(V_{12}) = rank(\mathbf{V}_1)$:

    $rank(V_{12}) \neq 0$ means V_{12} has a non-zero element and the problem is of the first class. Then there are two ways to show the problem belongs to the set $\mathcal{F}_1$. We could either observe that $rank(V_{12}) = 1 = e$, hence the problem is from $\mathcal{F}_1$ by definition (see Definition 8); or we could say that the problem is of the first class and it has a long tail property, thus it is from the set $\mathcal{F}_1$ by Theorem 5. The problem then has infinitely many solutions and a unique solution minimal in 2- and Frobenius norms.

    $rank(V_{12}) = 0$ means V_{12} consists of zeros. The problem then belongs to the second class $\mathcal{S}$ and has no solution; a non-generic solution could be constructed.

As we can see, the results derived from the multidimensional theory fully coincide with the wide-known results for the one-dimensional TLS problems. In particular, we also see that if $d = 1$, then $\mathcal{F} = \mathcal{F}_1$, i.e. for any problem of the first class, there exists a TLS solution, and the solution minimal in norm is unique.

2. Core Data Reduction

We have introduced the multidimensional TLS problem and summarised its properties. In this chapter, we will focus on the following question: do we need all data from the matrix (B, A) to solve the TLS problem? Or, perhaps, some data are redundant and we could potentially benefit from removing them? As we will see, it was shown first in [7] and [6] that the necessary and sufficient data can be extracted from the data matrix (B, A) in the form known as the core problem.

In the first part of the chapter we will give the basic idea behind the core data reduction (CDR). The second part of the chapter focuses on the description of the so-called *classic CDR algorithm* in the SVD form. A cheat sheet summarising the key steps of CDR is available in Appendix C. We will also compare the multidimensional case with the case $d = 1$ (described in [7]). This part is based mostly on the paper [8].

2.1 Preliminaries

Consider a TLS problem

$$AX \approx B \tag{2.1}$$

as in Definition 3. We want to transform the problem in such a way that its dimensions will be reduced, without changing the possible TLS solutions. The 2- and Frobenius norms are orthogonally invariant, thus we will limit ourselves to orthogonal transformations only.

We are search for such orthogonal matrices P, Q, R, that they separate the necessary and sufficient information from the redundancies, as follows:

$$P^{\mathsf{T}}(B, A)\begin{pmatrix} R & 0 \\ 0 & Q \end{pmatrix} = (P^{\mathsf{T}}BR, P^{\mathsf{T}}AQ) =: \left(\begin{array}{c||c|c} B_1 & A_1 & 0 \\ \hline 0 & 0 & A_2 \end{array}\right), \tag{2.2}$$

which corresponds to solving the approximation problem

$$\widehat{A}\widehat{X} \approx \widehat{B}, \tag{2.3}$$

$$\widehat{A} := P^{\mathsf{T}}AQ, \qquad \widehat{X} := Q^{\mathsf{T}}XR, \qquad \widehat{B} := P^{\mathsf{T}}BR. \tag{2.4}$$

This effectively splits the original TLS problem to two independent subproblems:

$$A_1\widehat{X_1} \approx B_1, \tag{2.5}$$

$$A_2\widehat{X_2} \approx 0. \tag{2.6}$$

The homogenous subproblem (2.6) is trivial and we set $\widehat{X_2} = 0$ (see [7] for a detailed explanation); so the initial problem is then reduced to (2.5). We naturally want the matrix A_1 to be of minimal dimensions and we call the problem (2.5) a *core problem* if the minimality condition is fulfilled. Here is the formal definition of a core problem.

Definition 10 (Core problem, core data reduction)**.** Consider an approximation problem $AX \approx B$ as in Definition 3. Consider orthogonal matrices P, Q, R transforming the problem as in (2.2). If (B_1, A_1) is minimally dimensioned and A_2 is of maximal dimensions, then we call the subproblem $A_1X_1 \approx B_1$ the corresponding *core problem*. *Core data reduction (CDR)* is then the process of extracting the core problem from the data matrix (B, A).

The definition was proposed in the paper [8, Definition 5.2].

Remark 21. The core problem is not unique. We will see that it could be derived in various forms; the so-called SVD and QR forms are of main interest. In this work we however focus on the CDR in SVD form and the phrase “classic CDR algorithm” will refer to the one based on SVD.

Remark 22. In general case when B is not of full column rank, the transformed matrix has the form:

$$\left(\begin{array}{c|c||c|c} B_1 & 0 & A_1 & 0 \\ \hline 0 & 0 & 0 & A_2 \end{array}\right). \tag{2.7}$$

We then get four independent subproblems, three of them will be homogenous, see [8, p. 925, equations (3.22)–(3.23)]. In this work we will limit ourselves to the case when B is of full rank, thus the zero column disappears. The cheat sheet in Appendix C however describes the general situation.

2.2 Classic CDR Algorithm

In this section we will summarise the classic algorithm for extracting the core problem presented in [8, Section 3].

We construct the matrices P, Q, R in four steps:

1. preprocessing the matrix of observations B; this step ensures the matrix of observations is of full rank; achieving some special properties (diagonal, low triangular echelon form) may also be the purpose;
2. transforming the system matrix A; in this step we transform the system matrix to a diagonal form;
3. transforming the matrix of observations B; in this step we maximize the number of zero rows in the observations matrix.;
4. final permutation regroups the elements so that the diagonal structure as in (2.2) is revealed.

Remark 23. In 2015, an article [14] was published, presenting an iterative CDR algorithm based on generalised Golub-Kahan band diagonalization. By saying *classic* CDR algorithm we stress that we are talking about the algorithm based on orthogonal transformations.

2.2.1 Preprocessing Matrix B

For general data, the main purpose of this step is to ensure the matrix B is of full column rank. Recall that in our setting we already assume:

$$B \in \mathbb{R}^{m \times d}, \qquad rank(B) = d. \tag{2.8}$$

Thus in our case we preprocess the right-hand side to achieve certain properties. General case however does not differ much from ours; for more details refer to [8, Section 3.1]. We will present two alternative ways of preprocessing the observations matrix.

Preprocessing in SVD form

We start with the SVD of the matrix B:

$$B = S\Theta R^{\mathsf{T}}, \tag{2.9}$$

$$S \in \mathbb{R}^{m\times d}, \qquad \Theta \in \mathbb{R}^{d\times d}, \qquad R \in \mathbb{R}^{d\times d}. \tag{2.10}$$

Multiplying the problem (2.1) by R from the right side yields:

$$A(XR) \approx BR \overset{def}{\iff} AY \approx C, \tag{2.11}$$

$$BR = S\Theta R^{\mathsf{T}}R = S\Theta =: C \in \mathbb{R}^{m\times d}, \qquad XR =: Y \in \mathbb{R}^{n\times d}. \tag{2.12}$$

Note that the matrix C has mutually orthogonal columns.

Preprocessing in LU (QR) form

We start with the LU decomposition of the matrix B:

$$B = LU, \tag{2.13}$$

$$L \in \mathbb{R}^{m\times d}, \qquad U \in \mathbb{R}^{d\times d}, \tag{2.14}$$

where L is a lower triangular matrix and U is orthogonal. Multiplying the problem (2.1) by U^{T} from the right side yields:

$$A(XU^{\mathsf{T}}) \approx BU^{\mathsf{T}} \overset{def}{\iff} AY \approx C, \tag{2.15}$$

$$BU^{\mathsf{T}} = LUU^{\mathsf{T}} = L =: C \in \mathbb{R}^{m\times d}, \qquad XU^{\mathsf{T}} =: Y \in \mathbb{R}^{n\times d}. \tag{2.16}$$

Matrix C in this case is lower triangular.

2.2.2 Transforming Matrix A

We now work with the problem $AY \approx C$. In this step we transform the system matrix A to diagonal form. Consider the SVD of the matrix A given by (1.1). We then obtain:

$$(U'^{\mathsf{T}}AV')(V'^{\mathsf{T}}Y) \approx U^{\mathsf{T}}C, \tag{2.17}$$

$$Z := V'^{\mathsf{T}}Y, \qquad F := U'^{\mathsf{T}}C, \qquad U'^{\mathsf{T}}AV' = \Sigma', \tag{2.18}$$

$$\Sigma' Z \approx F. \tag{2.19}$$

Note that now the approximation problem has the system matrix Σ' in diagonal form and the matrix of observations F is of full column rank (and has mutually orthogonal columns if we used the preprocessing in the SVD form).

2.2.3 Transforming Matrix of Observations

In this step we maximise the number of zero rows in the right-hand side matrix. We begin with the singular values σ_i' of the matrix A (i.e. with the generalised diagonal of the matrix Σ'). Let A have k distinct singular values with multiplicities $m_i, i = 1, \ldots, k+1$. The index $k+1$ here stands for the zero singular value.

We proceed with partitioning of the matrix F:

$$F =: \left(F_1^\intercal, \ldots, F_k^\intercal, F_{k+1}^\intercal\right)^\intercal, \tag{2.20}$$

$$F_i \in \mathbb{R}^{m_i \times d}, \qquad r_i := rank(F_i) \leq min\{m_i, d\}, \tag{2.21}$$

$$i = 1, \ldots, k, k+1. \tag{2.22}$$

We cut the matrix F by rows in correspondence with the multiplicities of σ_i'; for each σ_i' we allocate a block of m_i rows in F. We denote the ranks of F_i as r_i. It may happen that a certain singular value has a multiplicity greater than d, hence the *min* appearing in the expression for the upper boundary for r_i.

Transformation in SVD Form

Consider the SVD of F_i in the following form:

$$F_i = S_i \Theta_i W_i^\intercal, \qquad S_i \in \mathbb{R}^{m_i \times m_i}, \qquad \Theta_i \in \mathbb{R}^{m_i \times r_i}, \qquad W_i \in \mathbb{R}^{r_i \times d}. \tag{2.23}$$

We then define the following transformation matrices:

$$S_L := diag(S_1, S_2, \ldots, S_{k+1}), \qquad S_R := diag(S_1, S_2, \ldots, I_{n-rank(A)}). \tag{2.24}$$

The transformation matrices differ in the last block only; we have substituted the S_{k+1} with the identity matrix. Also note the dimensions:

$$m = \sum_{i=1}^{k+1} m_i, \tag{2.25}$$

$$S_L \in \mathbb{R}^{m \times d(k+1)}, \qquad S_R \in \mathbb{R}^{m \times (kd + m_{k+1})}. \tag{2.26}$$

The transformation of (2.19) then yields:

$$(S_L^\intercal \Sigma' S_R^\intercal)\,(S_R^\intercal Z) \approx S_L^\intercal F. \tag{2.27}$$

It could be shown that $(S_L^\intercal \Sigma S_R^\intercal) = \Sigma$. We then obtain the following approximation problem:

$$\Sigma' T \approx G,$$

$$T := S_R^\intercal Z, \qquad G := S_L^\intercal F. \tag{2.28}$$

The extended data matrix is of the following form:

$$(G, \Sigma') := (S_L^\intercal F, \Sigma') = \left(\begin{array}{c|cccc} \Theta_1 W_1^\intercal & \sigma_1' I_{m_1} & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \Theta_k W_k^\intercal & 0 & \ldots & \sigma_k' I_{m_k} & 0 \\ \Theta_{k+1} W_{k+1}^\intercal & 0 & \ldots & 0 & 0 \end{array}\right) \in \mathbb{R}^{m \times (n+d)} \tag{2.29}$$

Note that the extended matrix (G, Σ) has the same dimensions as the original extended matrix (B, A).

Remark 24. This step could be also conducted using QR-decomposition, yielding the transformation in the QR form, see [8, p. 926, Remark 3.1].

2.2.4 Final Permutation

Let us now focus on the sub-matrices $\Theta_i W_i^\intercal$. If $m_i > r_i$ (i.e. the number of rows is larger than the matrix rank), the bottom rows will be zero:

$$\Theta_i W_i^\intercal =: \begin{pmatrix} \Phi_i \\ 0_i \end{pmatrix}, \qquad \Phi_j \in \mathbb{R}^{r_i \times d}. \tag{2.30}$$

In order to reveal the diagonal structure in the matrix (G, Σ), we do the following:

1. We move the rows of (G, Σ), corresponding to the zero blocks 0_i in (2.30) to the bottom of the matrix.
2. The corresponding columns with blocks $\sigma_i I_{m_i - r_i}$ are then moved to the right (r_i rows are non-zero, hence the dimension $m_i - r_i$ of the identity matrix).

This can be written as follows:

$$\Pi_L^\intercal (G, \Sigma') \begin{pmatrix} I_d & 0 \\ 0 & \Pi^R \end{pmatrix}. \tag{2.31}$$

The formal description of the permutations Π_L and Π^R may be found in [8, Section 3.4].

2.2.5 Summary of Transformations (SVD Form)

The classic CDR algorithm for our setting could be summed up as follows:

$$\begin{gathered} (P^\intercal A Q)(Q^\intercal X R) \approx P^\intercal B R, \\ P := U' S_L \Pi_L, \qquad Q := V' S_R \Pi_R. \end{gathered} \tag{2.32}$$

Matrices P, Q, R are orthogonal. Denote

$$m^* := \sum_{i=1}^{k+1} r_i, \qquad n^* := \sum_{i=1}^{k} r_i. \tag{2.33}$$

Then the sub-matrices of the transformed data matrix (B, A) have the following dimensions:

$$(\widehat{B}, \widehat{A}) := P^\intercal (B, A) \begin{pmatrix} R & 0 \\ 0 & Q \end{pmatrix} = \tag{2.34}$$

$$\left(\begin{array}{c||c|c} B_1 & A_1 & 0 \\ \hline \underbrace{\quad 0 \quad}_{\mathtt{d}} & \underbrace{\quad 0 \quad}_{\mathtt{n^*}} & \underbrace{\quad A_2 \quad}_{\mathtt{n - n^*}} \end{array}\right) \begin{array}{l} \}\,\mathtt{m^*} \\ \}\,\mathtt{m - m^*} \end{array} \tag{2.35}$$

The matrix of variables is transformed as follows:

$$\widehat{X} := Q^\intercal X R = \begin{pmatrix} \widehat{X}_1 \\ \widehat{X}_2 \end{pmatrix}, \tag{2.36}$$

$$\widehat{X}_1 \in \mathbb{R}^{n^* \times d}, \qquad \widehat{X}_2 \in \mathbb{R}^{(n - n^*) \times d}. \tag{2.37}$$

The transformations described here are then called *the classic CDR algorithm in the SVD form.* The following theorem states the key property of the CDR algorithm and justifies the name “core problem”.

Theorem 12 (Core problem minimality). *Consider a TLS problem $AX \approx B$ as in Definition 3 and its subproblem $A_1\widehat{X}_1 \approx B_1$ yielded by the classic CDR algorithm. Then the matrix (B_1, A_1) has minimal dimensions over all matrices obtained by the orthogonal transformations of the form (2.2). Consequently, the subproblem $A_1\widehat{X}_1 \approx B_1$ is a core problem.*

Proof. For a more general formulation and proof, see [8, Section 4]. □

Let $\widehat{X}_1 \in \mathbb{R}^{n^* \times d}$ be a TLS solution of the core problem $A_1\widehat{X}_1 \approx B_1$. Corresponding TLS solution X' of the original problem (B, A) can be reconstructed by reverting the transformations applied to the variables X:

$$X' = Q \begin{pmatrix} \widehat{X}_1 \\ 0 \end{pmatrix} P^\intercal, \qquad 0 \in \mathbb{R}^{(n-n^*)\times d}. \tag{2.38}$$

2.3 One-Dimensional Case

In this section we are going to describe how the steps of the classic CDR algorithm are simplified in case $d = 1$. For more details on the CDR algorithm for the case of one-dimensional observation please refer to [7] and [8, Section 2].

Consider a TLS problem

$$\begin{gathered} Ax \approx b, \\ A \in \mathbb{R}^{m\times n}, \qquad x \in \mathbb{R}^n, \qquad b \in \mathbb{R}^m. \end{gathered} \tag{2.39}$$

The CDR procedure then goes as follows:

Preprocessing the right-hand side. Trivially, there is nothing to preprocess. By Definition 3, $A^\intercal b \neq 0$. Hence $b \neq 0$ and the right-hand side is of full rank 1. The matrix R from the equations (2.2) and (2.32) reduces to the constant $1 \in \mathbb{R}$.

Transforming the matrix A. The step does not differ from the case $d > 1$. We use the SVD of the matrix A in order to diagonalize it. The step yields the problem with the data matrix (f, Σ'). The problem then has the form:

$$\begin{gathered} \Sigma' z \approx f \\ \Sigma' \in \mathbb{R}^{m\times n}, \qquad V'^\intercal x =: z \in \mathbb{R}^n, \qquad U'^\intercal b =: f \in \mathbb{R}^m. \end{gathered} \tag{2.40}$$

Transforming the observations. We proceed with the partitioning as in equations (2.20)–(2.22). In our case however we slice the vector f, not a matrix, hence the result is of the form:

$$f =: (f_1^\intercal, \ldots, f_k^\intercal, f_{k+1}^\intercal)^\intercal, \tag{2.41}$$

Let σ_i' has the multiplicity m_i. Then $f_i \in \mathbb{R}^{m_i}$.

Here comes the first substantial simplification. In the multidimensional case we used the SVDs of sub-matrices F_i in order to maximize the number of zero rows. Since we are now dealing with sub-vectors f_i, we want to maximize the number of zero entries and it suffices to find a Householder reflection S_i such that $S_i^\intercal f_i = \|f_i\| e_1$, where e_1 is the first canonical base vector of the same dimension as f_i.

The transformations S_L and S_R are constructed as in the equation (2.24), again $S_L^\intercal \Sigma' S_R = \Sigma'$. After this step, the data matrix has the form $(S_L^\intercal f, \Sigma')$.

Final permutation. The permutations Π_L and Π_R are constructed exactly in the same way as for the multidimensional case.

As we can see, the two main simplifications include:

1. for any data matrix (b, A) the right-hand side is always of the full rank and does not need any preprocessing;
2. there is no need to compute the SVD in order to transform the right-hand side b, it could be transformed using only one Householder reflection for each sub-vector f_i.

3. Numerical Experiments

A proof-of-concept programme was developed as a part of this work. The source code is available on the GitHub repository [15].

When developing the programme, the focus was on the setting used and discussed in the previous chapter without attempting to cover all possible input data matrices (B, A). We particularly assumed that the right-hand side B is of full rank. The following algorithms were implemented:

1. the TLS algorithm for the problems from $\mathcal{F}_1$ discussed in Section 1.6;

2. the classic CDR algorithm described in Section 2.2.

The chapter is organised as follows. In the first part of the chapter, we discuss the technical details behind the programme implementation. We then proceed with describing numerical experiments and analysing their results.

3.1 Technical Aspects

3.1.1 PC and Software Specification

The experiments were conducted on a notebook with the following specifications:

```
Model: ASUS ROG STRIX G G731GU-EV032T

Processor: Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz
Installed RAM: 16.0 GB
System type: 64-bit operating system, x64-based processor
GPU: GeForce GTX 1660 Ti
```

Programming languages and libraries used to develop the programme:

1. Python programming language, version 3.7.6, and its standard modules, see [16];

2. SciPy library, version 1.4.1, see [17];

3. NumPy library, version 1.18.1, see [18].

3.1.2 Programming Details

Essentially the programme consists of two parts: the core library that implements the algorithm and ensures computations, and a number of programming scripts with computational experiments. The Python programming language was used both for writing the core library and running experiments. The library NumPy was used as a basic tool for all routines related to linear algebra, e.g. representing matrices and operations with them, computing the SVD and QR decompositions, computing matrix norms and numeric ranks etc. SciPy was used for working with block diagonal matrices (namely the `scipy.linalg.block_diag()` function).

Since Python has an interactive shell, we decided to limit ourself to writing a library without developing a user interface and turning it into a stand-alone programme. It is meant to be just a proof-of-concept, hence the computational efficiency (both in terms of memory and time) was not a priority, the focus was rather on the source code readability. We believe the source code is reasonably well organised, commented and documented, so working with it should not pose major challenges.

The programme was developed using the objective-oriented programming paradigm. The class `TLS_Problem`:

1. represents the concept of TLS problems,
2. encapsulates all data related to it (for instance, the data matrix (B, A) and its SVD, *e*-*q* numbers and matrices yielded by the corresponding partitioning (see Definitions 5, 6) etc.),
3. and provides an interface for common routines related to the TLS and CDR (e.g. computing the corrections, extracting the core problem etc.).

Certain basic tests were conducted, but creating appropriate unit tests and implementing them in code is beyond the scope of the present work.

3.1.3 Computational Details

In this subsection we will review various computational aspects that arose in the process of developing the programme and seem to be important or interesting.

Tolerances and thresholds

Our algorithm requires computing numerical ranks and comparing singular values numerically, e.g. when determining *e*-*q* numbers (see Definition 5) or when slicing the matrix F in the CDR (see equations (2.20)–(2.22). Such procedures demand using a carefully selected tolerance. In our programme it is called `sigma_threshold` and may be chosen separately for different problems. The default value of the constant is set to 10^{-6}. The question of how to choose tolerances requires further investigation.

TLS algorithm

As it was already mentioned, our library deals only with problems from $\mathcal{F}_1$ with the right-hand side B of full rank and the programme raises an exception if any of the conditions is not met. Essentially the algorithm determines whether the problem has a special property (see Definition 9) and then uses the appropriate formulae from the Section 1.6 to compute the TLS correction or corrected data matrix.

When it comes to computing the TLS solution, the library offers two ways.

Closed formula. The first one is to compute the TLS solution based on the appropriate closed formula (the `solution_closed()` method). The formula is not applied in a straight-forward way, rather it is transformed in a way leading to solving a system of equations, which should be more numerically stable. For instance, in the pretty pivot case, the closed formula (1.24) $X' = -\mathbf{V}_2\mathbf{V}_1^{-1}$ is rewritten in the form $\mathbf{V}_1^\intercal X'^\intercal = -\mathbf{V}_2^\intercal$. In case with the general formula (1.29), the same trick is applied, but the least squared method is used to solve the resulting system.

Solving corrected system. Another way is solving the corrected system. The library computes the corrected system using an appropriate closed formula and returns its least squares solution.

Currently, when it comes to long-tail problems, the library uses general closed formulae. The reason is explained below.

Singular values regularisation

Both in the long-tail case and when slicing the matrix F we end up with submatrices of Σ or Σ', that contain numerically close singular values. It may happen however that they are not equal, and two approaches are possible:

Do nothing. This is the library's current behaviour.

Enforce equality by applying regularisation rules.

The question of choosing the regularisation rules and their effect on the whole process requires further investigation as well.

3.2 Numerical Experiments

The source codes for all experiments are available at the repository [15]. A whole series of computational experiments were conducted using the developed test scripts. Here we include two selected examples as an illustration. In this section we will use the following notation:

1. X_0 denotes the real solution (if known);
2. X' refers to the solutions obtained by applying the TLS algorithm;
3. X^* refers to the solution reconstructed from the solution of the corresponding core problem;
4. X'_a and X^*_a denote the solutions yielded by the closed formulae;
5. X'_b and X^*_b stand for the solutions of the corrected system.

3.2.1 Trivial experiment (`exp1.py`)

We begin with the following system:

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 5 & 4 \\ 3 & 2 \\ 0 & 0 \end{pmatrix} X \approx \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 5 & 4 \\ 3 & 2 \\ 1 & 1 \end{pmatrix} \tag{3.1}$$

Note the both sides are of full rank, and the system is almost consistent: the right-hand side differs only in the last row. Hence $X_0 = I_2$.

The programme states, that the problem is from $\mathcal{F}_1$ and has a pretty pivot special property, which is easy to check. All numerical solutions that we track are equal and the following holds:

$$X' = \begin{pmatrix} 0.91393909 & -0.08606091 \\ 0.1334653 & 1.1334653 \end{pmatrix} \tag{3.2}$$

$$\|I_2 - X'\|_F = \|I_2 - X'\|_2 = 0.224586 \tag{3.3}$$

As we can see, the trivial test is passed.

3.2.2 Increasing number of rows m (`exp2.py`)

In the second experiment we tracked how the number of rows m affects the TLS solutions. The experiment is designed as follows.

1. The script allows to choose the parameters m_{max}, n, d, for our experiment they were set respectively to $700, 5, 4$.
2. The matrix $A_0 \in \mathbb{R}^{m_{max} \times n}$ is populated with random integers from 0 to 49 sampled from a discrete uniform distribution.
3. The matrix $X_0 \in \mathbb{R}^{n \times d}$ is populated with random integers -5 to 5 sampled from the same distribution.
4. The matrix B_0 is calculated as a product of A_0 and X_0.
5. The data matrix (B, A) is obtained from (B_0, A_0) by adding noise matrices (B_e, A_e) whose elements were populated from $N(0, \sigma^2)$.
6. The matrix $(B^m, A^m), m \in n + d, m_{max}$ is obtained by taking the first m rows of (B, A). We then use (B^m, A^m) to compute the TLS solutions. In each iteration we track:
    - the norms of the deviations between the TLS solutions and X_0;
    - the signal-to-noise ratio $\frac{\|B_0^m, A_0^m\|_F}{\|B_e^m, A_e^m\|_F}$.

Three sub-experiments were conducted with the parameter σ increasing from 5 to 20 and 40. The results are presented in the Figure 3.1. For the sake of keeping the figure transparent, only graphs referring to X'_a and X^*_b were plotted.

As we can see, the TLS and CDR solutions behave in the same way, which proves that the classic CDR algorithm could be used for solving TLS problems. Secondly, as it could be expected, the larger is the parameter σ, more data is needed (i.e. greater m) to obtain a reliable solution. Nevertheless, despite the loss of stability for small m, the deviations eventually converge to zero.

The solutions for the parameter $\sigma = 40$ could be pretty unstable for small m. More detailed graphs are presented in the Figure 3.2. Such behaviour leads to an idea: when applying the TLS method in practice, it might be a good idea to track the norm of the solution for different number of rows. This technique may allow to detect abnormal behaviour. The exact reason why the solutions are so unstable is to be further investigated.

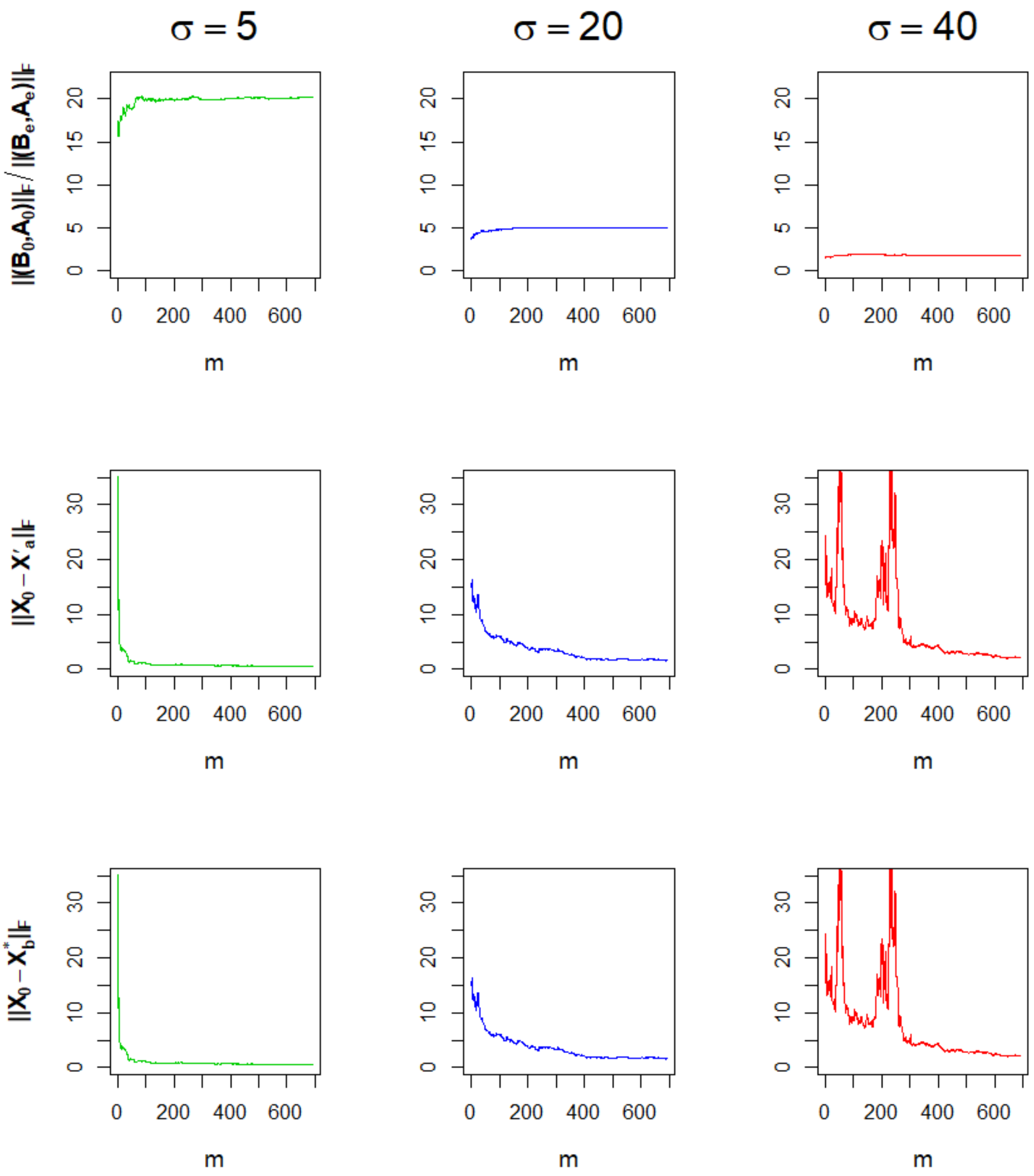

Figure 3.1: The figure presents the results of the second experiment for different parameters σ. The graphs in the first row track the behaviour of the signal-to-noise ratio with growing m. Graphs in the second row track the distance between the real solution and the closed formula TLS solution. Graphs in the third row track the deviation of the CDR solution obtained by solving the corrected system from X_0. Note that the deviation graphs in the third column have outliers that are not plotted.

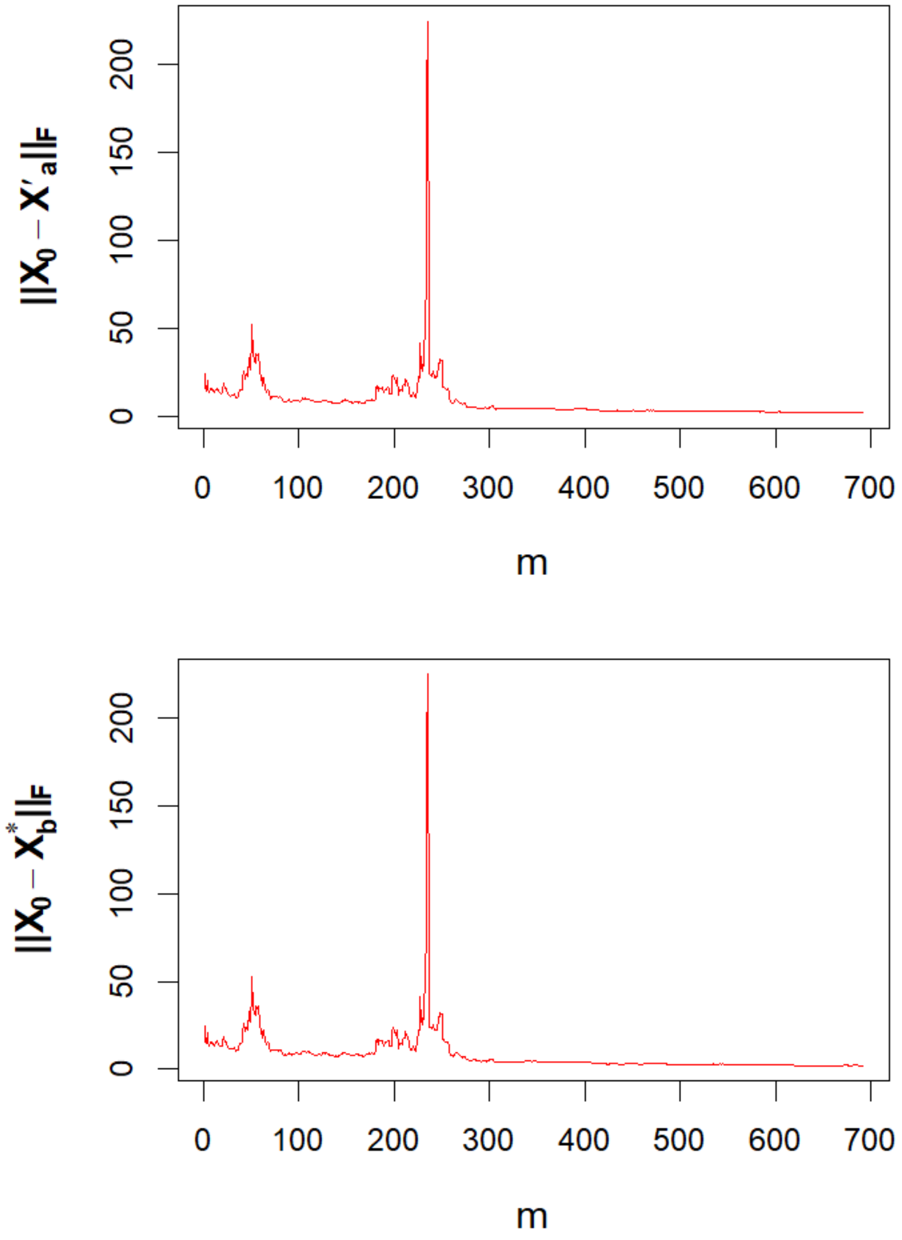

Figure 3.2: The complete graph of deviations for the parameter $\sigma = 40$. As we can see, the solution may be quite unstable if m is small.

Conclusion

In this work we have summarised the basic information related to the TLS problems with multiple right-hand side available in literature. In particular, we have described the classification of the TLS problems and discussed the properties of each problem class. We explained that certain TLS problems may have no solution, or infinitely many solutions minimal in norm may exist. We have introduced terms for the special cases of the TLS problems that are of particular interest, i.e. the pretty pivot case and the long tail case. We have also compared the TLS problems with single and multiple right-hand side and demonstrated that indeed the single observation problems are just a special case of the multidimensional problems.

The information on the TLS is spread across various sources, the notation and approaches are not consistent and differ from author to author, which posed one of the key challenges. Hence we have included a section comparing the approaches, see also a cheat sheet that helps to work with different sources consistently in Appendix B.

A brief description of the core data reduction (CDR) has been given. CDR allows to extract the information that is necessary and sufficient for solving the TLS problem in the form of a so-called *core problem.* We have summarised certain basic properties of the CDR and compared core problems for the cases of single and multiple right-hand side. Also see a cheat sheet on CDR in Appendix C.

Core problem properties have so far been studied solely for the case of exact arithmetic. Hence it was a question if CDR is applicable for solving TLS problems on computers. A proof-of-concept programme has been written implementing the classic CDR algorithm. A series of basic numerical experiments has been conducted, giving certain hope that the CDR approach in principle could be used for solving TLS problems numerically. It must be said the experiments are merely a small first step in this direction and certainly the topic requires further thorough studies. As it often happens, an attempt to create a programme implementation of a mathematical algorithm yielded a number of computational questions (e.g. treating numerically close singular values; choosing drop tolerances etc.). These questions are also to be further investigated.

In 2015, an article [14] describing an iterative version of the core data reduction procedure was published. Studying the numerical properties of the iterative algorithm and comparing it with the classic CDR procedure is one more potential way for future research.

Bibliography

[1] Sabine Van Huffel and Joos Vandewalle. *The total least squares problem: computational aspects and analysis.* Society for Industrial and Applied Mathematics, Philadelphia, 1991. ISBN 0-89871-275-0. doi: 10.1137/1.9781611971002.

[2] Gene H. Golub and Christian Reinsch. Singular value decomposition and least squares solutions. *Numerische Mathematik*, 14(5):403–420, 1970. ISSN 0029-599X. doi: 10.1007/bf02163027.

[3] Gene H. Golub and Charles F. Van Loan. An analysis of the total least squares problem. *SIAM journal on numerical analysis*, 17(6):883–893, 1980.

[4] Gene H. Golub and Charles F. Van Loan. *Matrix computations.* The Johns Hopkins University Press, Baltimore, 2013. ISBN 9781421408590.

[5] Shijian Yan and Kaibin Huang. The original TLS solution sets of the multidimensional TLS problem. *International journal of computer mathematics*, 73(3):349–359, 2000.

[6] Iveta Hnětynková, Martin Plešinger, Diana Maria Sima, Zdeněk Strakoš, and Sabine Van Huffel. The total least squares problem in $AX \approx B$: a new classification with the relationship to the classical works. *SIAM Journal on Matrix Analysis and Applications*, 32(3):748–770, 2011. ISSN 0895-4798. doi: 10.1137/100813348.

[7] Chris C. Paige and Zdeněk Strakoš. Core problem in linear algebraic systems. *SIAM Journal on Matrix Analysis and Applications*, (27):861–875, 2006.

[8] Iveta Hnětynková, Martin Plešinger, and Zdeněk Strakoš. The core problem within a linear approximation problem $AX \approx B$ with multiple right-hand sides. *SIAM Journal on Matrix Analysis and Applications*, 34(3):917–931, 2013. ISSN 0895-4798. doi: 10.1137/120884237.

[9] Roger A. Horn and Charles R. Johnson. *Matrix analysis.* Cambridge University Press, Cambridge, second edition, 2013. ISBN 978-0-521-54823-6.

[10] Carl Eckart and Gale Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936. ISSN 0033-3123. doi: 10.1007/bf02288367.

[11] L. Mirsky. Symmetric gauge functions and unitarily invariant norms. *The Quarterly Journal of Mathematics. Oxford. Second Series*, 11:50–59, 1960. ISSN 0033-5606. doi: 10.1093/qmath/11.1.50.

[12] Igor R. Shafarevich and Alexey O. Remizov. *Linear Algebra and Geometry.* Springer Berlin Heidelberg, 2012. ISBN 3642309933.

[13] Xue-Feng Wang. Total least squares problem with the arbitrary unitarily invariant norms. *Linear and Multilinear Algebra*, 65(3):438–456, 2017.

[14] Iveta Hnětynková, Martin Plešinger, and Zdeněk Strakoš. Band generalization of the Golub-Kahan bidiagonalization, generalized Jacobi matrices, and the core problem. *SIAM Journal on Matrix Analysis and Applications*, 36 (2):417–434, 2015. ISSN 0895-4798. doi: 10.1137/140968914.

[15] Matvei G. Slavenko. TLS problems. GitHub source code repository, 2020. URL `https://github.com/slavenkofm/tls_problems`. Accessed on 27.07.2020.

[16] Guido VanRossum. *The Python language reference.* Python Software FoundationSoHo Books, Hampton, NHRedwood City, Calif, 2010. ISBN 1441412697.

[17] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, and Paul van Mulbregt. SciPy 1.0: fundamental algorithms for scientific computing in python. *Nature Methods*, 17(3):261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[18] Stéfan van der Walt, S Chris Colbert, and Gaël Varoquaux. The NumPy array: A structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22–30, 2011. doi: 10.1109/mcse.2011.37.

A. TLS Problem Cheat Sheet

TLS

$$A \in \mathbb{R}^{m\times n}, \qquad X \in \mathbb{R}^{n\times d}, \qquad B \in \mathbb{R}^{m\times d} \tag{A.1}$$

$$rank(B) = d, \qquad m \geq d + n, \qquad A^{\intercal}B \neq 0 \tag{A.2}$$

$$\mathcal{R}(A) \not\subseteq \mathcal{R}(B) \tag{A.3}$$

$$AX \approx B \iff (B, A)\begin{pmatrix} -I_d \\ X \end{pmatrix} \approx 0 \tag{A.4}$$

$$\underset{A',B'}{minimize} \; \|(B', A') - (B, A)\|_{(F|2)} \quad \text{subject to} \quad \mathcal{R}(B') \subseteq \mathcal{R}(A') \tag{A.5}$$

Any X' such that $A'X' = B'$ is a *(generic) solution.*
$(\Delta B, \Delta A) := (B', A') - (B, A)$ is the corresponding *TLS correction.*

SVD for (B, A)

$$U \in \mathbb{R}^{m\times m}, \quad V \in \mathbb{R}^{(d+n)\times(d+n)}; \tag{A.6}$$

$$\Sigma = \begin{pmatrix} \Sigma_r \\ 0 \end{pmatrix} \in \mathbb{R}^{m\times(d+n)}; \tag{A.7}$$

$$\Sigma_r = diag(\sigma_1, \dots, \sigma_{d+n}), \tag{A.8}$$

$$\sigma_1 \geq \dots \geq \sigma_n \geq \sigma_{n+1} \geq \dots \geq \sigma_{n+d} \geq 0, \tag{A.9}$$

$$(B, A) = U\Sigma V^{\intercal}, \tag{A.10}$$

$$\sigma_{n-q} > \underbrace{\sigma_{n-q+1} = \dots = \sigma_n}_{q} = \underbrace{\sigma_{n+1} = \dots = \sigma_{n+e}}_{e} > \sigma_{n+e+1} \geq \dots \geq \sigma_{d+n} \tag{A.11}$$

$$\begin{array}{l} p := n - q \\ \sigma_p := \sigma_{n-q} \\ \sigma_n \neq \sigma_{n+1} \Rightarrow q := 0 =: e \\ \mathbf{V}_1 := (V_{12}, V_{13}) \in \mathbb{R}^{d\times(d+q)} \\ \mathbf{V}_2 := (V_{22}, V_{23}) \in \mathbb{R}^{n\times(d+q)} \end{array} \qquad \mathrm{V} = \overbrace{\begin{pmatrix} V_{11} & V_{12} & V_{13} \\ V_{21} & V_{22} & V_{23} \end{pmatrix}}^{\mathtt{d+n}} \begin{array}{l} \}\,\mathtt{d} \\ \}\,\mathtt{n} \end{array} \atop \qquad \underbrace{\quad}_{\mathtt{n-q}} \quad \underbrace{\quad}_{\mathtt{e+q}} \quad \underbrace{\quad}_{\mathtt{d-e}} \tag{A.12}$$

Property	σ_i distribution
Pretty pivot	$\sigma_n > \sigma_{n+1}$
Long tail	$\sigma_p > \sigma_{p+1} = \dots = \sigma_{n+1} = \dots = \sigma_{d+n}$
General distribution	other cases

(A.13)

B. Notation Comparison Cheat Sheet

<table>
<tr><th>S. Van Huffel et al.
([1])</th><th>I. Hnětynkova et al.
([6])</th><th>This work</th></tr>
<tr><td colspan="3">Extended matrices</td></tr>
<tr><td>$(A, B)\begin{pmatrix} X \\ -I_d \end{pmatrix} \approx 0$</td><td colspan="2">$(B, A)\begin{pmatrix} -I_d \\ X \end{pmatrix} \approx 0$</td></tr>
<tr><td colspan="3">Matrix V partitioning</td></tr>
<tr><td>$\begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix} \begin{array}{l} \}\,\mathtt{n} \\ \}\,\mathtt{d} \end{array}$
$\underbrace{\hphantom{V_{11}V_{11}}}_{\mathtt{n}} \quad \underbrace{\hphantom{V_{12}V_{12}}}_{\mathtt{d}}$</td><td>$\begin{pmatrix} V_{11}^{(q)} & V_{12}^{(q)} \\ V_{21}^{(q)} & V_{22}^{(q)} \end{pmatrix} \begin{array}{l} \}\,\mathtt{d} \\ \}\,\mathtt{n} \end{array}$
$\underbrace{\hphantom{V_{11}V_{11}}}_{\mathtt{n}-\mathtt{q}} \quad \underbrace{\hphantom{V_{12}V_{12}}}_{\mathtt{d}+\mathtt{q}}$
$V_{12}^{(q)} = \left(\underbrace{W^{(q,e)}}_{\mathtt{q}+\mathtt{e}} \quad \underbrace{V_{12}^{(q)}}_{\mathtt{d}-\mathtt{e}} \right) \}\,\mathtt{d}$</td><td>$\overbrace{\hphantom{V_{11}\quad V_{12}\quad V_{13}}}^{\mathtt{d}+\mathtt{n}}$
$\begin{pmatrix} V_{11} & V_{12} & V_{13} \\ V_{21} & V_{22} & V_{23} \end{pmatrix} \begin{array}{l} \}\,\mathtt{d} \\ \}\,\mathtt{n} \end{array}$
$\underbrace{\hphantom{V_{11}}}_{\mathtt{n}-\mathtt{q}} \quad \underbrace{\hphantom{V_{12}}}_{\mathtt{e}+\mathtt{q}} \quad \underbrace{\hphantom{V_{13}}}_{\mathtt{d}-\mathtt{e}}$</td></tr>
</table>

Table B.1: The table compares extended matrices and the matrix V partitionings used in different texts analysing the TLS problems.

S. Van Huffel et al. ([1])	I. Hnětynkova et al. ([6])	This work
V_{12}	V_{22}	$\mathbf{V}_2$
V_{22}	V_{12}	$\mathbf{V}_1$
—	$W^{(q,e)}$	V_{12}
—	$V_{12}^{(-e)}$	V_{13}

Table B.2: The table gives equivalents for selected parts of the matrix V used in different texts that analyse the TLS problems.

C. Classic CDR Algorithm Cheat Sheet

$$A \in \mathbb{R}^{m\times n}, \qquad X \in \mathbb{R}^{n\times d}, \qquad B \in \mathbb{R}^{m\times d} \tag{C.1}$$

$$d^* := rank(B), \qquad m \geq d+n, \qquad A^\intercal B \neq 0 \tag{C.2}$$

$$r := rank(A). \tag{C.3}$$

$$\mathcal{R}(A) \nsubseteq \mathcal{R}(B) \tag{C.4}$$

$$P^\intercal (B, A) \begin{pmatrix} R & 0 \\ 0 & Q \end{pmatrix} = \tag{C.5}$$

$$(P^\intercal BR, P^\intercal AQ) =: \left(\begin{array}{c|c||c|c} B_1 & 0 & A_1 & 0 \\ \hline 0 & 0 & 0 & A_2 \end{array} \right). \tag{C.6}$$

P, Q, R orthogonal. Core problem: $\widehat{A}\widehat{X_1} \approx \widehat{B}_1$.

Preprocessing B using SVD

Make the right-hand side of full rank, with orthogonal columns.

$$B = S\Theta R^\intercal, \qquad S \in \mathbb{R}^{m\times d^*}, \qquad \Theta \in \mathbb{R}^{d^*\times d}, \qquad R \in \mathbb{R}^{d\times d}, \tag{C.7}$$

$$A(XR) \approx BR \overset{def}{\iff} AY \approx C, \tag{C.8}$$

$$BR = S\Theta R^\intercal R = S\Theta =: (C, 0) \in \mathbb{R}^{m\times d^*}, \qquad XR =: Y \in \mathbb{R}^{n\times d^*}. \tag{C.9}$$

We get two subproblems, one of them is homogenous. We then focus on the subproblem:

$$AY \approx C. \tag{C.10}$$

Transforming A using SVD

Making A diagonal.

$$A = U'\Sigma' V'^\intercal, \qquad U \in \mathbb{R}^{m\times m}, \qquad \Sigma' \in \mathbb{R}^{m\times n}, \qquad V \in \mathbb{R}^{n\times n}, \tag{C.11}$$

$$(U'^\intercal AV')(V'^\intercal Y) \approx U^\intercal C \overset{def}{\iff} \Sigma' Z \approx F, \tag{C.12}$$

$$Z := V'^\intercal Y, \qquad F := U'^\intercal C, \qquad U'^\intercal AV' = \Sigma'. \tag{C.13}$$

We now work with $\Sigma' Z \approx F$.

Transforming matrix of observations using SVD

Let k distinct non-zero singular values σ'_i of the matrix A (i.e. the generalised diagonal of Σ') have multiplicities $m_i, i \in \{1, \ldots, k, k+1\}$. The index $k+1$ stands for the zero singular value.

$$F =: \left(F_1^\intercal, \ldots, F_k^\intercal, F_{k+1}^\intercal\right)^\intercal, \tag{C.14}$$

$$F_i \in \mathbb{R}^{m_i \times d^*}, \qquad r_i := rank(F_i) \leq min\{m_i, d^*\}, \tag{C.15}$$

$$i = 1, \ldots, k, k+1. \tag{C.16}$$

Consider the SVD of F_i in the following form:

$$F_i = S_i\Theta_i W_i^\intercal, \qquad S_i \in \mathbb{R}^{m_i\times m_i}, \qquad \Theta_i \in \mathbb{R}^{m_i\times r_i}, \qquad W_i \in \mathbb{R}^{r_i\times d^*}. \tag{C.17}$$

We then define the transformation matrices:

$$S_L := diag(S_1, S_2, \ldots, S_{k+1}), \qquad S_R := diag(S_1, S_2, \ldots, I_{n-r}). \tag{C.18}$$

$$\begin{aligned} (S_L^\intercal \Sigma' S_R^\intercal)\,(S_R^\intercal Z) &\approx S_L^\intercal F, \qquad &(C.19)\\ (S_L^\intercal \Sigma' S_R^\intercal) &= \Sigma' \Longrightarrow &(C.20)\\ \Sigma'\,(S_R^\intercal Z) \approx S_L^\intercal F &\overset{def}{\Longleftrightarrow} \Sigma' T \approx G. &(C.21)\end{aligned}$$

We now work with $\Sigma' T \approx G$.

Final permutation

$$(G, \Sigma') := (S_L^\intercal F, \Sigma') = \left(\begin{array}{c|cccc} \Theta_1 W_1^\intercal & \sigma_1' I_{m_1} & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \Theta_k W_k^\intercal & 0 & \ldots & \sigma_k' I_{m_k} & 0 \\ \Theta_{k+1} W_{k+1}^\intercal & 0 & \ldots & 0 & 0 \end{array}\right), \tag{C.22}$$

$$m_i > r_i \Longrightarrow \Theta_i W_i^\intercal =: \begin{pmatrix} \Phi_i \\ 0_i \end{pmatrix}, \qquad \Phi_j \in \mathbb{R}^{r_i\times d^*}. \tag{C.23}$$

In order to reveal the diagonal structure in the matrix (G, Σ), we do the following:

1. We move the rows of (G, Σ), corresponding to the zero blocks 0_i in (2.30) to the bottom of the matrix.
2. The corresponding columns with blocks $\sigma_i I_{m_i - r_i}$ are then moved to the right (r_i rows are non-zero, hence the dimension $m_i - r_i$ of the identity matrix).

This can be written as:

$$\Pi_L^\intercal (G, \Sigma') \begin{pmatrix} I_d & 0 \\ 0 & \Pi^R \end{pmatrix}. \tag{C.24}$$

Summary

$$\begin{aligned} (P^\intercal A Q)(Q^\intercal X R) &\approx P^\intercal B R, &(C.25)\\ P := U' S_L \Pi_L, &\qquad Q := V' S_R \Pi_R, &(C.26)\\ d^* := rank(B), \qquad m^* := \sum_{i=1}^{k+1} r_i, &\qquad n^* := \sum_{i=1}^{k} r_i, &(C.27)\\ (\widehat{B}, \widehat{A}) := P^\intercal (B, A) &\begin{pmatrix} R & 0 \\ 0 & Q \end{pmatrix} = &(C.28)\end{aligned}$$

$$= \left(\begin{array}{c|c||c|c} B_1 & 0 & A_1 & 0 \\ \hline 0 & 0 & 0 & A_2 \\ \underbrace{\qquad}_{\mathtt{d}^*} & \underbrace{\qquad}_{\mathtt{d}-\mathtt{d}^*} & \underbrace{\qquad}_{\mathtt{n}^*} & \underbrace{\qquad}_{\mathtt{n}-\mathtt{n}^*} \end{array}\right) \begin{array}{l} \}\,\mathtt{m}^* \\ \}\,\mathtt{m}-\mathtt{m}^* \\ \; \end{array} \tag{C.29}$$

Reconstructing the original solution:

$$X' = Q \begin{pmatrix} \widehat{X_1} & 0 \\ 0 & \mathcal{O} \end{pmatrix} P^\intercal, \tag{C.30}$$

$$\widehat{X_1} \in \mathbb{R}^{n^* \times d^*}, \qquad \mathcal{O} \in \mathbb{R}^{(n-n^*)\times(d-d^*)}. \tag{C.31}$$