Subspace Segmentation Problem

Augusto José Rabelo Almeida Santos

Dissertation submitted for obtaining the degree of
Master in Electrical and Computer Engineering

Jury

President: Carlos Jorge Ferreira Silvestre
Supervisor: João Manuel de Freitas Xavier
Member: Alessio Del Bue

October 2008
Acknowledgements

I would like to thank Professor João Xavier for accepting me in this project. Moreover, the time he spent, his critics and suggestions were crucial to reduce the incoherence I’ve introduced into the thesis and also to accelerate my learning process mainly in mathematical subjects.

I would like to thank all my friends from IST, specially to João Rodrigues, Gonçalo Cabral and Filipe Aguiar.

I would like to thank ISR for the excellent conditions I was given. I would like to thank all my colleagues from ISR for the excellent environment created and a special thanks to Nuno Pinho for his advices to improve some points on the thesis.

And the most important part of my life: my core family. All my efforts are made mainly for them.
Abstract

Segmenting a mixture of linear subspaces in the presence of noise and outliers is a challenging problem which finds direct applications in computer vision, e.g., motion segmentation, structure-from-motion and object recognition.

We present two optimization formulations for the subspace segmentation problem together with the associated numerical algorithms to solve them.

In the first formulation, the segmentation problem is cast as the maximization of a convex function over a compact convex set. We propose a monotone optimization algorithm which loops between a Minimization-Maximization (MM) routine and a Escape from only-Local Maximizers (ELM) routine. The MM routine locates a local maximizer and ELM tries to escape from it towards the global one, re-initializing the MM routine when successful. For the MM routine, convergence is proved to be achieved in a finite number of steps (for a certain cost function). Our technique shows a competitive performance with both synthetic and real data image sequences, which includes the hard articulated scenario. The main drawback is that the number of subspaces to be segmented and their dimensions are required beforehand.

In the second formulation, the segmentation problem is approached through successive minimization of a $l_1$-norm based convex function over the unit-norm sphere. The goal is to construct, recursively, the orthogonal complements of the subspaces. Prior knowledge of the number of subspaces or their dimensions is not required. To handle the optimization problems over the unit-sphere, we present a locally-convergent optimization algorithm with quadratic rate of convergence (Newton method on a Riemannian manifold). The performance of our approach is tested through computer simulations.

Finally, we introduce a novel theoretical framework to analyse the subspace segmentation problem. This is based on optimality criteria for convex optimization and subdifferential calculus. This framework shows that, in the absence of outliers, the subspace segmentation problem can be solved by a certain random linear program, with probability one. Incorporation of outliers lowers the probability of success, but our framework enables its theoretical computation and offers an intuitive geometrical interpretation.

**Keywords:** subspace segmentation, convex analysis, convex optimization, subdifferential, singular value decomposition.
Resumo

A segmentação de uma união arbitrária de subespaços lineares na presença de ruído e de outliers é um problema desafiador e que encontra aplicação directa em visão computacional, nomeadamente em segmentação de imagens por movimento, structure-from-motion e reconhecimento de objectos.

Nesta tese, propomos duas formulações em forma de programas de optimização para o problema da segmentação de subespaços. Implementamos também os algoritmos numéricos associados para os resolver.

Na primeira formulação, o problema da segmentação é modelado através da maximização de uma função convexa restrita a um conjunto convexo e compacto. É proposto um algoritmo de optimização monotono que alterna entre uma rotina Minimization-Maximization (MM) e uma rotina de escape de maximizantes locais (ELM). A rotina MM localiza um máximo local e a ELM tenta escapar deste em direcção ao máximo global, re-inicializando a MM quando houver sucesso no escape. Provamos que a rotina MM converge em um número finito de iteracções (para uma certa função de custo). Nossa técnica apresentou um desempenho competitivo ao lidar com dados sintéticos ou sequências de imagens reais, incluindo o problemático caso articulado. A principal desvantagem deste método é que ele requer um conhecimento prévio acerca da dimensão e do número de subespaços em jogo.

Na segunda formulação, o problema da segmentação é abordado através da sucessiva minimização de uma função convexa (baseada na norma $l_1$) sobre a esfera unitária. O objectivo é construir recursivamente o complemento ortogonal dos subespaços. Para esta abordagem não é necessário conhecimento prévio acerca da quantidade de subespaços e suas dimensões. Para tratar o problema de optimização sobre a esfera, nós apresentamos um algoritmo de optimização localmente convergente, com velocidade de convergência quadrática (método de Newton sobre uma variedade Riemanniana). O desempenho do algoritmo é testado através de simulações por computador.

Finalmente, apresentamos uma nova abordagem para analisar o problema da segmentação. Esta é baseada em um critério de optimalidade para a optimização convexa e o cálculo subdiferencial. Esta abordagem mostra que, na ausência de outliers, o problema da segmentação de subespaços pode ser resolvido através de um certo programa linear aleatório, com probabilidade um. A incorporação de outliers diminui a probabilidade de sucesso, porém nossa abordagem permite calculá-la a partida e oferece um interpretação geométrica intuitiva deste caso.

Keywords: segmentação de subespaços, análise convexa, optimização convexa, subdiferencial, decomposição em valores singulares.
# Contents

<table>
<thead>
<tr>
<th>Notation</th>
<th>xi</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Context</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Segmentation Principle</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Thesis Outline</td>
<td>4</td>
</tr>
<tr>
<td>1.4 Contributions</td>
<td>5</td>
</tr>
<tr>
<td>2 Subspace Segmentation - Orthogonal Subspaces</td>
<td>6</td>
</tr>
<tr>
<td>2.1 Chapter Summary</td>
<td>6</td>
</tr>
<tr>
<td>2.2 Motivation for the Orthogonal Case Approach</td>
<td>9</td>
</tr>
<tr>
<td>2.3 Data Model</td>
<td>10</td>
</tr>
<tr>
<td>2.4 Algorithm</td>
<td>16</td>
</tr>
<tr>
<td>2.4.1 MM Routine</td>
<td>18</td>
</tr>
<tr>
<td>2.4.2 Qualitative Analysis and the ELM Algorithm</td>
<td>26</td>
</tr>
<tr>
<td>2.5 Log-Version of the $f_{tr}$ Objective Function</td>
<td>27</td>
</tr>
<tr>
<td>2.6 Final Algorithm</td>
<td>33</td>
</tr>
<tr>
<td>2.7 Synthetic Results</td>
<td>35</td>
</tr>
<tr>
<td>2.7.1 Computer Simulations: Noiseless Case</td>
<td>35</td>
</tr>
<tr>
<td>2.7.2 Computer Simulations: Noisy Case</td>
<td>37</td>
</tr>
<tr>
<td>2.8 Advantage From the ELM or From More Trials?</td>
<td>40</td>
</tr>
<tr>
<td>2.9 Real Data Results</td>
<td>41</td>
</tr>
<tr>
<td>2.10 Conclusion</td>
<td>42</td>
</tr>
<tr>
<td>3 Subspace Segmentation - General Formulation</td>
<td>44</td>
</tr>
<tr>
<td>3.1 Chapter Summary</td>
<td>44</td>
</tr>
<tr>
<td>3.2 Data Model</td>
<td>46</td>
</tr>
<tr>
<td>3.3 Algorithm</td>
<td>48</td>
</tr>
<tr>
<td>3.3.1 Recursive Strategy</td>
<td>48</td>
</tr>
<tr>
<td>3.3.2 Optimization over the sphere: NS algorithm</td>
<td>51</td>
</tr>
</tbody>
</table>
List of Figures

1.1 Illustration of the object jointly with the feature points. The camera is the origin of the world’s coordinate system. One feature point has been chosen as the origin of the object’s coordinate system. .................................................. 3

2.1 Draft depicting the simple data pre-processing. .................................................. 10

2.2 \( \mathcal{L}_{\text{ACT1}} \) and \( \mathcal{L}_{\text{ACT2}} \) represent activated subspaces, but \( \mathcal{L}_{\text{OUT}} \) does not, in light of the definition. Note that \( \#(\mathcal{L}_{\text{OUT}} \cap X) = 2 < h = 5 \). .................................................. 11

2.3 Illustration of three orthogonal straight lines in \( \mathbb{R}^3 \). \( \mathcal{L}_1 = \mathcal{L}_2 \oplus \mathcal{L}_3 \). .................................................. 12

2.4 Illustration of the convex hull of a set. The resulting set \( \text{co} S \) is the "smallest" convex set containing \( S \). .................................................. 14

2.5 Graph of the function \( \phi_{tr}(s) = |s - \frac{1}{2}| \). .................................................. 15

2.6 Illustration of the subset of extreme points of a set \( P \). .................................................. 15

2.7 By choosing a subgradient \( s \) in \( \partial f(x) \) we can ensure that any point \( y \) in the dashed zone is such that \( f(y) \geq f(x) \). .................................................. 16

2.8 Illustration of a convex function being touched by an affine lower bound. \( S \in \partial f(\Pi_0) \). .................................................. 17

2.9 Normal cone \( N_C(x) \). .................................................. 18

2.10 cone \( P \) represents the set of all conical combinations of the set \( P \) (blue). .................................................. 19

2.11 Representation of the tangent cone \( T_C(x_k) \) and normal cone \( N_C(x_k) \). In this case the point \( x_k \) is a local maximizer because \( \partial f(x_k) \subseteq N_C(x_k) \). .................................................. 20

2.12 Geometric illustration of the MM algorithm achieving a local maximizer. The algorithm only walks over extreme points which are the projectors in this case. \( \Pi_3 \) is a local maximizer. .................................................. 22

2.13 Flowchart of the algorithm MM with the ELM .................................................. 26

2.14 Fig.(a) represents "\( b_i^T \Pi_0 b_i \) vs \( i \)" where \( \Pi_0 \) is an initialization (a projector of rank 4). Fig.(b) represents "\( b_i^T \Pi^* b_i \) vs \( i \)" where \( \Pi^* \) is a maximizer returned by the MM. The first four \( b_i \)'s represent an orthogonal basis for \( \mathcal{L}_1 \) .................................................. 27

2.15 Flowchart sketching the MM-ELM loop. .................................................. 28

2.16 Flow chart illustrating the inner working of the ELM routine. .................................................. 29
2.17 Geometric illustration of the parameters in the experience. \( \Pi \) represents a projector of rank 1, \( \theta \) is the angle between \( \text{span}\{\Pi\} \) and the canonical vector \( e_1 \) (and is also the variable of \( f_{tr} \)) and \( \theta_i \) is the angle between \( L_i \) and \( e_1 \).  

2.18 Fig.(a) and (b) depict the graph of \( f_{tr} \) in the orthogonal case with 30% and 60% of outliers, respectively.  

2.19 Fig.(a) and (b) depicts the graph of \( f_{tr} \) assuming 30% of outliers.  

2.20 Graph of \( f_{log} \) with \( \varepsilon = 10^{-5} \).  

2.21 Graph of \( f_{log} \) with \( |\theta_1 - \theta_2| = 45^\circ \) and 20% of outliers.  

2.22 Graph of \( f_{log} \) with \( |\theta_1 - \theta_2| = 45^\circ \) and 20% of outliers.  

2.23 Illustration of the two possible solutions \( \Pi_1^* \) and \( \Pi_2^* \) when the restriction set is chosen to be \( \text{Proj}_3(2) \). The yellow points represent outliers. \( \text{span}\{\Pi_1^*\} \) and \( \text{span}\{\Pi_2^*\} \) represents the subspaces in red and blue, respectively. One of the yellow points lies within \( \text{span}\{\Pi_1^*\} \) jointly with the straight line \( r \) (green).  

2.24 One activated Subspace: avarage of correct classifications for 5000 trials.  

2.25 Two oblique subspaces with trivial intersection: avarage of correct classifications for 5000 trials.  

2.26 Rights vs number of trial. Curve (a) is associated to the scape and curve (b) is associated to the random shootings routine.  

2.27 First frame of the tested sequences from the Hopkins155 database. In all of them, 20% of outliers were added to the dataset.  

3.1 Identifying two straight lines in \( \mathbb{R}^3 \). First, one identifies \( L_1 \), and then one identifies \( L_{11} \) and \( L_{12} \) inside \( L_1 \).  

3.2 \( U_1 \sigma \) represents a linear application provided by the SVD of \( X_1 \). Thus, identifying the red and green lines in (b), we are also identifying \( L_{11} \) and \( L_{12} \).  

3.3 Recursion tree. The sons represent subspaces that span their respective fathers. The numbers, within the red balls on the edges, represent the order in which the recursion is executed.  

3.4 Illustration of the intrinsic Newton's direction. In our case, \( M \) is the unit-sphere, \( T_p M \) is the tangent plane of \( M \) at \( p \), \( v_p \) is the Newton's direction (intrinsicly computed in the tangent plane) and \( \text{Exp} v_p \) represents the geodesic shotted in the \( v_p \) direction from \( p \).  

3.5 Illustration of the curves: rights vs number of trial.  

4.1 Image of \( x_a = \alpha e_2 \), for \( \alpha > 0 \), by \( \partial f \).  

4.2 Illustration of the mapping of subsets of \( \mathbb{R}^2 \) by \( \partial f \) into the dual space. The right figure represents \( \text{dom} f \), and the left one represents \( \partial f(\mathbb{R}^2) \). The vertex \( \partial I \) represents the image of the whole region I in \( \text{dom} f \) by \( \partial f \) and \( \partial e_2 \) (straight line segment connecting \( \partial IV \) to \( \partial I \)) represents the image of each point over the ordinates axes.
4.3 Figures (a) and (b) illustrate the intersection \((-\partial f) \cap \text{span}\{c\}\) for two different choices of \(c\). For the choice in (a) we have that the minimizers of (4.10) are those points \(x^*\) s.t. \((-\partial f(x)) \cap \text{span}\{c\} \neq \emptyset\), that is, will be the points lying in \(\text{span}\{e_2\}\). In (b), the minimizers will be the points in \(\text{span}\{e_1\}\).

4.4 Configuration of the subdifferential \(\partial f\) when one outlier is added.
List of Tables

2.1 **MM routine**: \( n_1 \) represents the dimension of one of the (two pairwise orthogonal) subspaces, \( X \) is the data matrix, \( \Pi_k \) is the \( k \)th projector in the sequence defined by the algorithm and \( \Pi^* \) is the local maximizer. ............................................................ 25

2.2 Recursive version of the MM-ELM algorithm. \( I \) and \( J \) are sets of indexes. \( X_I \) is the matrix constructed from the data matrix \( X \) extracting those points \( x_i \), s.t. \( i \in I \). The output of the algorithm is the set of indexes labeling the points embedded in \( L_1 \) (activated \( n_1 \)-dimensional subspace). In this case, we are assuming that the restriction set is \( \text{Proj}_{n_1}(n_1) \) (we could also choose \( \text{Proj}_{n_1}(n-n_1) \) ). \( \text{size}(X,1) \) is the number of rows of \( X \), that is, the current ambient space dimension. ................................. 34

2.3 Comparison: MM-only vs MM-ELM. One 4-dimensional activated subspace with cost function \( f_{\text{log}} \). ............................................................... 36

2.4 Comparison: \( f_{\text{log}} \) versus \( f_{\text{tr}} \). Two 4-dimensional orthogonal activated subspaces. ... 36

2.5 Comparison: \( f_{\text{log}} \) versus \( f_{\text{tr}} \). Two 4-dimensional oblique activated subspaces. ...... 36

2.6 Comparison: \( f_{\text{log}} \) versus \( f_{\text{tr}} \). One 4-dimensional activated subspaces. .................. 37

2.7 Comparison: two 4-dimensional oblique subspaces subject to \( \dim L_1 \cap L_2 = 1 \) versus two 4-dimensional subspaces with trivial intersection. Cost function used was \( f_{\text{log}} \). ... 37

2.8 One point from the yellow group and another from the red group have been misclassified as outliers. The green points represent random trajectories (outliers) added. .............. 41

2.9 Thirteen points from the red group and twenty points from the red group have been misclassified as outliers. The green points represent random trajectories (outliers) added. 41

2.10 Fifty two points from the red group and fifty five points from the green group have been misclassified as outliers. The blue points represent random trajectories (outliers) added. 42

3.1 Strategy using NS. The parameter \( \text{INI} \) represents the number of initializers. ............... 52

3.2 NS-only. One 4-dimensional subspace. The cost function used was \( f_{\text{sq}} \). ....................... 52

3.3 NS-only. Two 4-dimensional oblique subspaces. The cost function used was \( f_{\text{sq}} \). ....... 53

3.4 Comparison: NS.log vs NS.sq. One 4-dimensional subspace. ................................. 53

3.5 Comparison: NS.log vs NS.sq. Two 4-dimensional oblique subspaces. ....................... 53

3.6 NS-ELM Algorithm. The parameter \( \text{INI} \) represents the number of points for initialization. 54

3.7 Comparison: NS-ELM vs NS. One 4-dimensional subspace. ................................. 55
3.8 Comparison: NS-ELM vs NS. Two 4-dimensional oblique subspaces. ....................... 55
3.9 NS-ELM.log. Two 4-dimensional oblique subspaces subject to \( \dim L_1 \cap L_2 = 1 \) ........ 55
3.10 Comparison: NS-ELM.sq vs NS.sq. One 4-dimensional subspace. ......................... 56
3.11 Comparison: NS-ELM.sq vs NS.sq. Two 4-dimensional oblique subspaces. ............. 56
3.12 Comparison: NS-ELM.log vs MM-ELM.log. Two 4-dimensional oblique subspaces. ... 56

4.1 Strategy to achieve activated subspaces by solving a linear program. .................... 63
4.2 Results. The 100% in reliability means that the solutions obtained by solving the problem (4.10) (with an arbitrary choice for \( c \)) was always interesting. ......................... 63
## Notation

In the following, we list the symbols and acronyms frequently used throughout this thesis. In each row, the first and second entries introduce the symbol or acronym and define its meaning, respectively. The third entry contains the number of the page where it appears for the first time.

### List of Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A^\top$</td>
<td>Transpose of $A$</td>
<td>12</td>
</tr>
<tr>
<td>$A^{-1}$</td>
<td>Inverse of $A$</td>
<td>47</td>
</tr>
<tr>
<td>$\mathcal{L}^\perp$</td>
<td>Orthogonal complement of the subspace $\mathcal{L}$</td>
<td>26</td>
</tr>
<tr>
<td>$\partial f(x)$</td>
<td>Subdifferential of $f$ at the point $x$</td>
<td>16</td>
</tr>
<tr>
<td>$\nabla f(x)$</td>
<td>Gradient of $f$ at the point $x$</td>
<td>18</td>
</tr>
<tr>
<td>$\langle A, B \rangle$</td>
<td>Inner product $\langle A, B \rangle = \text{tr}(A^\top B)$</td>
<td>16</td>
</tr>
<tr>
<td>$\text{Gr}(k, n)$</td>
<td>Grassmann manifold: set of all $k$-dimensional subspaces of $\mathbb{R}^n$</td>
<td>??</td>
</tr>
<tr>
<td>$|A|$</td>
<td>Frobenius norm of $A$</td>
<td>24</td>
</tr>
<tr>
<td>$|v|_0$</td>
<td>$L_0$-norm of the real vector $v$</td>
<td>44</td>
</tr>
<tr>
<td>$|v|_1$</td>
<td>$L_1$-norm of the real vector $v$</td>
<td>44</td>
</tr>
<tr>
<td>$|v|_\infty$</td>
<td>$L_\infty$-norm of the real vector $v$</td>
<td>47</td>
</tr>
<tr>
<td>$</td>
<td>\cdot</td>
<td>$</td>
</tr>
<tr>
<td>$S_{n \times n}$</td>
<td>Set of $n \times n$ symmetric matrices</td>
<td>71</td>
</tr>
<tr>
<td>$\text{dom } f$</td>
<td>Domain of the map $f$</td>
<td>17</td>
</tr>
<tr>
<td>$\text{Proj}_{n_1}(n)$</td>
<td>Set of $n \times n$ orthogonal projectors of rank $n_1$</td>
<td>14</td>
</tr>
<tr>
<td>$\text{co}(-)$</td>
<td>Convex hull operation</td>
<td>14</td>
</tr>
<tr>
<td>$\text{cl}(-)$</td>
<td>Closure operation. $\text{cl}(S)$ is the “smallest” closed set containing $S$</td>
<td>19</td>
</tr>
<tr>
<td>$\text{Conv } \mathbb{R}^n$</td>
<td>Set of convex functions with domain in $\mathbb{R}^n$</td>
<td>14</td>
</tr>
<tr>
<td>$\text{span}{v_1, \ldots, v_n}$</td>
<td>Subspace spanned by the vectors $v_1, \ldots, v_n$</td>
<td>47</td>
</tr>
<tr>
<td>$\text{span}{A}$</td>
<td>Subspace spanned by the column vectors of the matrix $A$</td>
<td>21</td>
</tr>
<tr>
<td>$#(-)$</td>
<td>$#(S)$ represents the cardinality of the set $S$</td>
<td>??</td>
</tr>
<tr>
<td>$\mathbb{E}{\cdot}$</td>
<td>Expected value of a random object</td>
<td>38</td>
</tr>
<tr>
<td>$\mathcal{N}(\mu, \Sigma)$</td>
<td>Gaussian distribution with mean $\mu$ and covariance matrix $\Sigma$</td>
<td>37</td>
</tr>
<tr>
<td>$\mathbb{N}$</td>
<td>Set of natural numbers ${0,1,2,\ldots}$</td>
<td>18</td>
</tr>
<tr>
<td>$\mathbb{R}^n$</td>
<td>Set of $n$-dimensional real vectors</td>
<td>2</td>
</tr>
<tr>
<td>$\mathbb{R}^{n \times m}$</td>
<td>Set of $n \times m$ real matrices</td>
<td>13</td>
</tr>
<tr>
<td>$\text{rank}(A)$</td>
<td>Rank of the matrix $A$</td>
<td>14</td>
</tr>
<tr>
<td>$\text{tr}(A)$</td>
<td>Trace of the matrix $A$</td>
<td>21</td>
</tr>
<tr>
<td>$\oplus$</td>
<td>Orthogonal direct sum</td>
<td>11</td>
</tr>
</tbody>
</table>
# Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>MM</td>
<td>Minimization-Maximization</td>
<td>6</td>
</tr>
<tr>
<td>NS</td>
<td>Newton-Sphere</td>
<td>45</td>
</tr>
<tr>
<td>SVD</td>
<td>Singular Value Decomposition</td>
<td>4</td>
</tr>
<tr>
<td>EVD</td>
<td>Eigenvalue Decomposition</td>
<td>70</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal-to-Noise Ratio</td>
<td>38</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
<td>34</td>
</tr>
<tr>
<td>ELM</td>
<td>Escape from only-Local Maximizer</td>
<td>7</td>
</tr>
<tr>
<td>RS</td>
<td>Random Sampling</td>
<td>40</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Context

Today, a huge amount of videos are made to record events and study the structure of the world. Many applications require methods for segmenting these image sequences into meaningful events or scenes for easy access, analysis or editing [1].

Humans can easily discriminate independently moving objects without retrieving their specific identities. It roughly suggests that motion is an important clue for segmenting objects.

The first step for discriminating different moving objects in a sequence of images is the tracking process. Essentially, it consists in detecting specific features of the objects in each frame and arrange them into trajectories across the images.

In fact, several local features arise from the object's movement. For instance, assume a sequence of images with the same background and one object moving around. Assume that from one frame to the next the edges of the object shift by a few pixels. Subtracting these two consecutive frames, the result is a frame in which the pixels differing significantly from zero are those ones corresponding to the aforementioned edges. Actually, we are interested in more local features. Neither edges nor lines, but specific points in the objects so that we can reconstruct their rigid structure. There are several techniques and criteria to detect features in a sequence of images. However, these are out of the scope of this work. For more details, see [1].

The tracking process even when successfully done is just the beginning of the story. Once the features’ trajectories are produced, it is necessary to figure out which trajectories belong to which objects, so that we can process the information concerning each object separately.

Segmenting a moving object in a sequence of images, given the tracked features’ trajectories, is
equivalent to identifying an associated 4-dimensional or 3-dimensional subspace living in a certain Euclidean ambient space. That is, each subspace identified conveys trajectory points regarding each object. This fact was firstly pointed out by Costeira and Kanade [2]. This is the principle behind most of currents works on motion segmentation. Kanatani [3] provided a simple proof of this paradigm which is not a general proof since it assumes the simplest camera model (orthographic), but this fact holds for all affine camera models [4]. For completeness, we now present Kanatani’s proof.

1.2 Segmentation Principle

Consider a 3-D rigid object on which $N$ feature points have been marked. A video sequence of this body (possibly moving around) is made by an orthographic camera. The video sequence consists of $M$ frames (2-D images), which are obtained as follows.

Adopt the camera’s coordinate system as the world’s coordinate system with the $z$-axis along the optical axis. Choose any of the feature points as the origin of the object’s coordinate system and attach to it an orthonormal system. This orthonormal system encodes the object’s attitude. Let $t_m \in \mathbb{R}^3$ and $Q_m = [q_{1m} \ q_{2m} \ q_{3m}] \in \mathbb{R}^{3\times3}$ denote the position of the object’s origin and the orthonormal system, respectively, w.r.t the world coordinate system in the $m$th frame. See Fig. 1.1.

Then, the $n$th tracked point in the $m$th frame can be expressed as:

$$r_{mn} = t_m + Q_m \begin{bmatrix} a_n \\ b_n \\ c_n \end{bmatrix} \in \mathbb{R}^3. \quad (1.1)$$

Taking the orthographic projection of $r_{mn}$ onto the camera’s plane $\{(x,y,z) \in \mathbb{R}^3 : z = 0\}$, yields:

$$p_{mn} = \hat{t}_m + \hat{Q}_m \begin{bmatrix} a_n \\ b_n \\ c_n \end{bmatrix} \in \mathbb{R}^2, \quad (1.2)$$

where $\hat{t}_m$ and $\hat{Q}_m$ are obtained from $t_m$ and $Q_m$ by chopping their last row.

The 2-D trajectory of the $n$th point can be represented by the vector:

$$p_n = [p_{1n}^T \ p_{2n}^T \ \cdots \ p_{Mn}^T]^T \in \mathbb{R}^{2M} \quad (1.3)$$
Figure 1.1: Illustration of the object jointly with the feature points. The camera is the origin of the world's coordinate system. One feature point has been chosen as the origin of the object's coordinate system. Using (1.2) there holds:

\[ p_n = t + Q \begin{bmatrix} a_n \\ b_n \\ c_n \end{bmatrix} \]  \hspace{1cm} (1.4)

where \( t := [\hat{t}_1^T \hat{t}_2^T \ldots \hat{t}_M^T]^T \in \mathbb{R}^{2M} \) and \( Q := \begin{bmatrix} \hat{Q}_1 \\ \hat{Q}_2 \\ \vdots \\ \hat{Q}_M \end{bmatrix} = [q_1 \ q_2 \ q_3] \in \mathbb{R}^{2M \times 3} \).

Equation (1.4) shows that \( p_n \) lies within the 4-dimensional subspace spanned by \( \{ t, q_1, q_2, q_3 \} \). However, in the special cases where the object performs planar movements, namely, whether it translates along the \( x \) and \( y \) directions or it rotates around the \( z \)-axis, there holds \( q_{3m} = 0 \) for all \( m \) and therefore, \( q_3 = 0 \). Thus, the subspace’s dimension would reduce by one.

In sum, for the general case, rigid objects’ movement activates 4-dimensional subspaces, which contains the trajectory (as defined above) of any given tracked point.

In practice, some points are deviated from their corresponding features in the object during the tracking process. This failure produces flawed trajectory points which do not need to lie in the same 4-dimensional subspace of the inliers trajectories. These kind of points are usually called outliers. Therefore, whenever a method for discriminating trajectories is developed, it is crucial to require robustness to outliers, as well.

In this thesis, we will be addressing the subspace segmentation problem, that is, we will be con-
cerned in identifying activated subspaces given a set of trajectory points. We will not deal with tracking issues.

1.3 Thesis Outline

Orthogonal Subspaces. Chapter 2 provides a formulation of the subspace segmentation problem assuming orthogonal subspaces and no outliers. We show that, when there is no outliers, it is equivalent to handle the orthogonal or oblique case in the sense that one can switch from an oblique disposition of the subspaces to an orthogonal configuration by a simple pre-processing of the data set. We obtain an optimization program whose solutions correspond to the underlying subspaces. Roughly speaking, the algorithm created to seek the solutions jumps from one iteration to another by evaluating SVD's turning the process computationally efficient in achieving a solution. For the performance test, we will handle two cost functions; one that comes naturally from the mathematical formulation and the latter that is a log-version of this previous one and shown to be more robust to outliers in the oblique case. The algorithm presented an interesting performance when dealing with outliers and noise. The main drawback is that it needs prior information regarding the number and dimensions of the subspaces.

Segmentation with Outliers. In chapter 3 we segment subspaces by incrementally building their orthogonal complements. In contrast with the previous method, this algorithm does not need prior knowledge of the number of subspaces nor their dimensions. Executing a recursive strategy it can group all of the points lying in each subspace. The drawback here is that the algorithm seems to be rather sensitive to noise and it is not as computationally efficient as the algorithm developed in the previous chapter.

Subspace Segmentation without Outliers: Linear Program. In chapter 4 we characterize the subspace segmentation problem without outliers through a linear program. That is, by solving a properly linear program, we can find an activated subspace. We also provide a geometrical approach to study the outliers case through tools from convex analysis.
1.4 Contributions

- A main contribution of this thesis is the formulation of the subspace segmentation as the maximization of a convex function over a compact convex set. This special structure allowed the derivation of efficient locally-convergent numerical algorithms.

- We proved the convergence of one of the proposed algorithms in a finite number of steps. The line of proof can be exploited in the future to prove the convergence of the other proposed algorithms.

- We also introduce a novel approach to tackle the subspace segmentation problem. This is based on optimality criteria for convex optimization and subdifferential calculus. This approach shows that, in the absence of outliers, the subspace segmentation problem can be solved by a certain random linear program. Incorporation of outliers lowers the probability of success, but our framework allows us to compute it beforehand and offers an intuitive geometrical interpretation.
Chapter 2

Subspace Segmentation - Orthogonal Subspaces

2.1 Chapter Summary

Section 2.2 [Motivation for the Orthogonal Case Approach]. In this section, we show that, without outliers, there is no loss of generality in treating the subspace segmentation problem assuming orthogonal subspaces. That is, the oblique case can be easily transported to the orthogonal case by a simple pre-processing of the data set through an SVD.

Section 2.3 [Data Model]. This section formulates the subspace-segmentation problem for the case of mutually orthogonal subspaces. Basically, it consists in identifying pairwise orthogonal linear subspaces given a finite set of data points drawn from them. The primary target is to characterize such a problem through the maximization of a convex function restricted to a convex and compact set:

$$\max_{x \in C} f(x).$$

(2.1)

The characterization is in the sense that all of the activated subspaces are codified in the solutions of (2.1) and conversely, all the solutions represent activated subspaces\(^1\).

We shall highlight that the program we achieve here is not convex, since it is the maximization of a convex function. However, we can explore such a structure in solving that problem.

Section 2.4 [Algorithm]. In this section, the convex structure of the program obtained in the previous section will be explored. Our first goal is to implement the Minimization-Maximization technique (MM)

\(^1\)Though in presence of outliers this latter claim is no longer valid.
in order to lead us to a maximizer.

In presence of outliers there is no guarantee that a solution of our optimization problem correspond to an activated subspace. Thus, an heuristic called ELM (Escape from only-Local Maximizers) will be developed that tries to make the algorithm escape from only-local maximizers. This heuristic was inspired by observations of the qualitative behavior of the MM routine.

To sum up, our algorithm (MM-ELM) loops through two routines, MM and ELM. The MM routine is responsible to find a local maximizer. The ELM routine is responsible to try to escape from this proposed solution. If it succeeds, it re-initializes the MM routine. If not, the algorithm’s loop is aborted and the iterations stop.

Section 2.5 [Log-Version of the \( f_{tr} \) Objective Function]. So far, we have a cost function (call it \( f_{tr} \)) that arises naturally from the optimization formulation of the orthogonal subspace segmentation problem. In fact, in the presence of outliers, the equivalence between the orthogonal case and the oblique case aforementioned is no longer valid and we will have to face the oblique case. Our aim in this section is to study the behavior of \( f_{tr} \) in this latter case through a set of simple experiments. We conclude that in the oblique case the maximizers of \( f_{tr} \) are in general displaced relatively to those points codifying activated subspaces. Thus, we construct a new cost function in order to bypass this default.

Section 2.6 [Final Algorithm]. Here we present the final version of the algorithm MM-ELM which will be tested in the following sections.

Section 2.7 [Synthetic Results]. Results from simulations with synthetic data will be presented. Several subspace configurations will be tested within a given ambient space. Namely: (i.) two orthogonal subspaces (ii.) two oblique subspaces (with trivial and nontrivial intersection) (iii.) three subspaces and (iv.) one subspace. All configurations will be tested with outliers. The performance of the algorithm will be evaluated as function of the outliers’ concentration. There will be also some experiences with noise added to the data set.

Section 2.8 [Advantage from The ELM or from More Trials?]. We will present some results that strongly suggest that the gain in performance of the MM-ELM loop relative to the MM comes directly from the heuristic behind the ELM.
Section 2.9 [Real Data Results]. The algorithm will be tested also over real data (from indoor or outdoor scenes). For this purpose, we have used three sequences of images provided by the Hopkins155 database and publicly available in [19].
2.2 Motivation for the Orthogonal Case Approach

Consider $X = [x_1 \ldots x_m]$ as a set of data points in $\mathbb{R}^n$ distributed into $d$ linear subspaces $\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_d$ with dimensions $\dim \mathcal{L}_i = n_i$ and s.t. $\mathbb{R}^n = \mathcal{L}_1 \oplus \ldots \oplus \mathcal{L}_d$. Let $m_i$ be the number of points of $X$ lying in $\mathcal{L}_i$. Hence, we can rewrite $X$ (up to a right permutation) as:

$$X = \hat{P}_{n \times n} \begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_d \end{bmatrix}$$

(2.2)

Here, $\hat{P}$ is a $n \times n$ full rank matrix: its first $n_1$ columns span $\mathcal{L}_1$, the next $n_2$ columns span $\mathcal{L}_2$ and so on. The matrix $W_i$ is the $i$th $n_i \times m_i$ submatrix of the block diagonal matrix $W$ and contains the coordinates of the points in $\mathcal{L}_i$ w.r.t the basis in $\hat{P}$. We can evaluate the Gram-Schmidt orthogonalization of the rows of each block yielding $W_i = R_i Q_i^T$, where $R_i$ is upper-triangular and $Q_i$ has orthonormal columns. Thus, we can rewrite (2.2) as:

$$X = \hat{P} R_1 R_2 \ldots R_d Q_1 Q_2 \ldots Q_d = P Q$$

(2.3)

where $P := \hat{P} R$. We can also evaluate the SVD of $X$:

$$X = U \Sigma V^T$$

(2.4)

and we can prove that the linear transformation that takes the column points of $Q$ into the column points of $V^T$ is an isometry, that is, it can be characterized through an $n \times n$ orthogonal matrix $M$ (its proof is given in appendix A.1.1):

$$V^T = MQ$$

(2.5)

This means that the subspaces underlying the column points in $V^T$ are pairwise orthogonal. In other words, even when the subspaces are mutually oblique in the data matrix, we can pre-process this data through an SVD obtaining an orthogonal configuration (see Fig. 2.1).

In sum, when there is no outliers, there is no loss of generality in treating the subspace segmenta-
tion problem assuming orthogonal subspaces. That is, if we are dealing with the oblique case, we can just pre-process the data through an SVD in order to recover the orthogonal structure.

2.3 Data Model

First of all, we shall define the concept of outlier, to be used through out this work.

**Definition.** Let $X$ denote a finite set of points in $\mathbb{R}^n$. Given an integer $h$, an element $p$ of the set $X$ is called an outlier in $X$ if there is no a sum decomposition $\mathbb{R}^n = L_1 \oplus \ldots \oplus L_d$ with pre-defined dimensions $\dim L_i = n_i$ s.t.:

$$\#(X \cap L_i) \geq h \text{ and } \exists i : p \in L_i$$

(2.6)

where $\#(S)$ denotes the cardinality of the set $S$.

In other words, assuming that part of the points of $X$ are distributed into $d$ subspaces, $p$ is an outlier in $X$ if it does not lie in no one of these nontrivial subspaces containing $h$ or more points of $X$. Of course, the definition is attached to the value of $h$. The set of outliers in $X$ will be denoted by $O_X$.

Furthermore, we shall consider a subspace to be activated if it contains at least $h$ elements of $X$. Thus, the threshold $h$ has to be settled up a priori in order to make sense of this terminology.

Basically, our aim in this work is to identify activated subspaces. Fig. 2.2 illustrates the rough idea.
Problem Formulation. Let \( X = \{ x_i \in \mathbb{R}^n : i = 1, \ldots, m \} \) be a given data set s.t. \( O_X = \emptyset \). Consider that there exists a direct sum decomposition \( \mathbb{R}^n = L_1 \perp \ldots \perp L_d \) respecting:

\[
\#(L_i \cap X) \geq h
\]

and \( \dim L_i \) are known, for all \( i \). Find the subspaces \( L_i \), for all \( i \).

In this section, we will assume that \( d = 2 \) and that the points \( x_i \) lie over the unit-sphere. This entails no loss of generality. Indeed, if a set of points spans a subspace \( L \), so do their unit-norm scaled versions. The fact that there is no loss of generality in considering \( d = 2 \) is in the sense that if we find two orthogonal subspaces \( L_1 \) and \( L_2 \) s.t. \( \mathbb{R}^n = L_1 \perp L_2 \) then the remaining subspaces live within this couple of subspaces. Specifically, if we were dealing with \( d \) linear pairwise orthogonal subspaces \( L_1, \ldots, L_d \), then, there holds:

\[
L_1 \perp L_2 \perp \ldots \perp L_d = L_1 \perp (L_2 \perp \ldots \perp L_d)
\]

That is, the \( d \) linear subspaces are conveyed within two orthogonal subspaces. By achieving \( L_1 \) and \( L_1^\perp \), we can investigate for more activated subspaces within \( L_1^\perp \) in order to achieve all particular activated subspaces. Let’s see an example.

Assume we have to identify three orthogonal straight lines \( L_1, L_2 \) and \( L_3 \) in \( \mathbb{R}^3 \) (see Fig. 2.3) subject to \( \mathbb{R}^3 = L_1 \perp L_2 \perp L_3 \).

Then, we can find the subspaces \( L_1 \) and \( L_1^\perp \). Since we know previously the dimensions of the activated subspaces, we can infer that one subspace has been identified and the remaining \( L_2 \) and \( L_3 \)

\[
\begin{align*}
\text{Problem Formulation.} & \quad \text{Let } X = \{ x_i \in \mathbb{R}^n : i = 1, \ldots, m \} \text{ be a given data set s.t. } O_X = \emptyset. \\
& \quad \text{Consider that there exists a direct sum decomposition } \mathbb{R}^n = L_1 \perp \ldots \perp L_d \text{ respecting:} \\
& \quad \quad \#(L_i \cap X) \geq h \quad (2.7) \\
& \quad \text{and } \dim L_i \text{ are known, for all } i. \text{ Find the subspaces } L_i, \text{ for all } i. \\
& \quad \text{In this section, we will assume that } d = 2 \text{ and that the points } x_i \text{ lie over the unit-sphere. This entails no loss of generality. Indeed, if a set of points spans a subspace } L, \text{ so do their unit-norm scaled versions. The fact that there is no loss of generality in considering } d = 2 \text{ is in the sense that if we find two orthogonal subspaces } L_1 \text{ and } L_2 \text{ s.t. } \mathbb{R}^n = L_1 \perp L_2 \text{ then the remaining subspaces live within this couple of subspaces. Specifically, if we were dealing with } d \text{ linear pairwise orthogonal subspaces } L_1, \ldots, L_d, \text{ then, there holds:} \\
& \quad \quad L_1 \perp L_2 \perp \ldots \perp L_d = L_1 \perp (L_2 \perp \ldots \perp L_d) \quad (2.8) \\
& \quad \text{That is, the } d \text{ linear subspaces are conveyed within two orthogonal subspaces. By achieving } L_1 \text{ and } L_1^\perp, \text{ we can investigate for more activated subspaces within } L_1^\perp \text{ in order to achieve all particular activated subspaces. Let’s see an example.} \\
& \quad \text{Assume we have to identify three orthogonal straight lines } L_1, L_2 \text{ and } L_3 \text{ in } \mathbb{R}^3 \text{ (see Fig. 2.3) subject to } \mathbb{R}^3 = L_1 \perp L_2 \perp L_3. \\
& \quad \text{Then, we can find the subspaces } L_1 \text{ and } L_1^\perp. \text{ Since we know previously the dimensions of the activated subspaces, we can infer that one subspace has been identified and the remaining } L_2 \text{ and } L_3}
\end{align*}
\]
lie within $L_1^\perp$. Then, we should look at those points of the data matrix $X$, that lie over $L_1^\perp$. Let $I$ be the set of indexes corresponding to those points of $X$ in $L_1^\perp$. Thus, we can build up another matrix $X_I$ with the referred points and evaluate the SVD in order to reduce the ambient space dimension:

$$X_I = U_I \Sigma_I V_I^\top,$$  \hspace{1cm} (2.9)

That is, we can work over $V_I^\top$ (whose columns are points in $\mathbb{R}^2$) instead of $X_I$ (whose columns are points in $\mathbb{R}^3$). As explained in section 2.2 the underlying activated subspaces in $V^\top$ are also mutually orthogonal. Then, we can again find more two subspaces say $L_2$ and $L_2^\perp$ and the process continues up to peel all subspaces.

Just as a note, we will only use the $n_1$-truncated SVD throughout this thesis, where $n_1$ is the rank of the entry matrix of the SVD. That is, given $X$ s.t. $\text{rank}(X) = n_1$, then we will just consider the first $n_1$ columns of $U$ and $V$ in the representation of the SVD of $X$.

Given that $\mathbb{R}^n = L_1 \oplus L_2$ with $\text{dim } L_i = n_i$, it follows that the set $X = [x_1 \ x_2 \ \ldots \ x_m]$ can be rotated by an orthogonal matrix $Q$ such that a special pattern appears in the rotated data. More precisely:

$$\exists Q \in O(n) : Q^\top \begin{bmatrix} x_1 & x_2 & \ldots & x_m \end{bmatrix} = \begin{bmatrix} * & * & 0 & \ldots & \star & \ldots & 0 \\ \star & \star & 0 & \ldots & \star & \ldots & \star \\ 0 & 0 & \star & \ldots & 0 & \ldots & \star \end{bmatrix} \begin{bmatrix} \downarrow n_1 \\ \downarrow n_2 \end{bmatrix},$$  \hspace{1cm} (2.10)

in which $\star$ represents entries not necessarily zero and $O(n)$ is the orthogonal matrix group of order $n$:

$$O(n) = \{ Q \in \mathbb{R}^{n \times n} : Q^\top Q = I_n \}$$  \hspace{1cm} (2.11)
For such a matrix $Q$ in (2.10), we can see that the first $n_1$ columns of $Q$ span $L_1$ and the remaining $n_2$ span $L_2$ (or vice-versa). Indeed, we can multiply (on the left) both sides of (2.10) by $Q$, yielding:

$$[x_1 \ x_2 \ \ldots \ \ x_m] = Q \begin{bmatrix} * & * & 0 & \ldots & \ldots & 0 \\ 0 & 0 & * & \ldots & \ldots & 0 \\ \end{bmatrix}. \quad (2.12)$$

It is clear that we have an amount of points of $X$ lying in the subspace spanned by the first $n_1$ columns of $Q$ and the remaining points over the subspace spanned by the last $n_2$ columns. Thus, by identifying $Q$ we obtain basis for the unknown subspaces.

Finding an orthogonal matrix inducing the configuration in (2.10) is equivalent to solving the following optimization problem:

$$Q^* \in \arg \min_{Q=\{Q_1, Q_2\} \in O(n)} f(Q) := \sum_{i=1}^{m} \min\{||Q_1^T x_i||^2, ||Q_2^T x_i||^2\}, \quad (2.13)$$

Equivalence here is in the sense that a solution of (2.13) respects (2.10) and a matrix $Q$ that respects (2.10) is a solution of (2.13). Indeed, the objective function $f(Q)$ is non-negative, thus:

$$f(Q) = 0 \Leftrightarrow \min\{||Q_1^T x_i||^2, ||Q_2^T x_i||^2\} = 0 \text{ for all } i. \quad (2.14)$$

That is, the points rotated by $Q$ have nonzero components in only one subspace, which means that part of them lies in a subspace of dimension $\dim L_1$ and the remaining lies in a subspace of dimension $\dim L_2$. The converse is straightforward.

Note that neither $f(Q)$ nor $O(n)$ are convex objects, but we can reformulate the problem (2.13) as a maximization problem s.t. both the objective function and the constrain set are convex objects. This is explained in theorem 2.3.1 below. Before proceeding to the theorem, we recall the concepts of convex combination and convex hull.

**Definition.** A convex combination of the elements $x_1, \ldots, x_m$ in $\mathbb{R}^n$ is an element of the form:

$$x = \sum_{i=1}^{m} \alpha_i x_i \quad (2.15)$$

where $\sum_{i=1}^{m} \alpha_i = 1$ and $\alpha_i \geq 0$ for $i = 1, \ldots, m$.

**Definition.** Given a set $S$, the convex hull of $S$ (denoted by $\text{co } S$) is the set of all convex combinations of elements of $S$. 

13
Another equivalent and more intuitively definition of convex hull [7] is:

\[ \text{co } S = \cap \{ C : C \text{ is convex and contains } S \} \]  

(2.16)

Figure 2.4: Illustration of the convex hull of a set. The resulting set \( \text{co } S \) is the "smallest" convex set containing \( S \).

We now present the theorem (its proof is given in appendix A.1.6).

**Theorem 2.3.1.** Let \( X = [x_1 \cdots x_m] \) be the data matrix, \( Q = [Q_1 \; Q_2] \), and \( C = \text{co Proj}_{n_1}(n) \), then:

\[
\min_{Q \in O(n)} \sum_{i=1}^{m} \min \{ ||Q_1^T x_i||^2, ||Q_2^T x_i||^2 \} \iff \max_{\Pi \in C} \sum_{i=1}^{m} \left| (x_i^T \Pi x_i) - \frac{1}{2} \right| \]

(2.17)

in which \( \text{co Proj}_{n_1}(n) \) is the convex hull of the orthogonal projectors set of order \( n \) and rank \( n_1 \).

We are assuming in theorem 2.3.1 that those two programs are equivalent in the sense that if \( Q^* = [Q_1^* \; Q_2^*] \) is a solution of the first problem, then \( \Pi^* = Q_1^* Q_1^{\top} \) is solution of the second one and if \( \Pi^* = Q_1^* Q_1^{\top} \) is a solution of the second problem then \( Q^* = [Q_1^* \; Q_2^*] \) is solution of the first one.

Such theorem allows us to transport the problem (2.13) to a more compact and structurally attractive program with cost function:

\[
f_{tr}(\Pi) = \sum_{i=1}^{m} \phi_{tr}(x_i^T \Pi x_i) \in \text{Conv } \mathbb{R}^{n \times n}
\]

(2.18)

where \( \phi_{tr} : \mathbb{R} \to \mathbb{R} \) is defined as \( \phi_{tr}(s) = |s - \frac{1}{2}| \) (see its graph in Fig. 2.5) and with a restriction set that is convex and compact (convexity and compactness are shown in appendix A.1.2 and A.1.5). Therefore, without loss of optimality, it is sufficient to restrict our attention to the subset of extreme points of \( C \).
Definition. The point \( x \in C \) is an extreme point of \( C \) when there are no two distinct points \( x_1 \) and \( x_2 \) in \( C \) and no \( \alpha \in (0, 1) \) such that:
\[
x = \alpha x_1 + (1 - \alpha) x_2.
\] (2.19)

We denote the set of extreme points of \( C \) by \( \text{ext} \ C \).

Fig. 2.6 illustrates this concept.

We can prove that \( \text{Proj}_n(n_1) = \text{ext}(\text{co Proj}_n(n_1)) \) (see appendix A.1.6) and therefore, we can restrict attention to the set \( \text{Proj}_n(n_1) \), that is, we will be seeking for projectors with a pre-defined rank.

Note that \( f_{tr} \) (2.18) is nonsmooth. The gradient of \( f_{tr} \) does not exist at every point. However, a generalization of it, the subdifferential does. The subdifferential is a crucial concept which comes up in
algorithms for nonsmooth convex optimization.

**Definition.** The subdifferential of \( f \in \text{Conv} \mathbb{R}^n \) at the point \( x \) is the set of vectors \( s \in \mathbb{R}^n \) which satisfy:

\[
f(y) \geq f(x) + \langle s, y - x \rangle \quad \text{for all } y \in \mathbb{R}^n.
\]

(2.20)

It is denoted by \( \partial f(x) \). An element \( s \in \partial f(x) \) is called a subgradient of \( f \) at \( x \).

![Figure 2.7: By choosing a subgradient \( s \) in \( \partial f(x) \) we can ensure that any point \( y \) in the dashed zone is such that \( f(y) \geq f(x) \).](image)

Fig. 2.7 highlight an important fact derived from the definition. Assume we are maximizing a convex function \( f \) subject to the restriction set \( C \). By choosing a subgradient \( s \) in \( \partial f(x) \), we see that any point \( \hat{y} \) of \( C \) in the half-space \( \{ y \in \mathbb{R}^n : \langle s, y - x \rangle > 0 \} \) beats the point \( x \). Specifically, returning to our case, given \( \Pi_k \in C \), it follows that:

\[
\langle S, Y - \Pi_k \rangle_D > 0 \Rightarrow f(Y) > f(\Pi_k) \quad \text{for any } S \in \partial f(\Pi_k) \text{ and } Y \in C.
\]

(2.21)

In other words, by local analysis, we can predict one direction \( D \) in which the function always grows. The subdifferential here allows us to broaden this local property even to nonsmooth (but convex) functions. In the next section we will explore this statement in more detail.

**2.4 Algorithm**

Based on the formulation developed in the previous section, we are dealing with the following optimization program:

\[
\max_{\Pi \in C} f(\Pi),
\]

(2.22)
in which $C$ is convex compact and $f \in \text{Conv} \mathbb{R}^{n \times n}$. The problem as stated is hard. Nevertheless, we can explore the convex structure of the objective function and derive a single Minimization-Maximization (MM) routine. Indeed, $f$ can be lowerbounded by an affine function at any chosen point. It touches this lower bound at the given point. Specifically (see also Fig. 2.8):

$$\forall S \in \partial f(\Pi_0) : f(\Pi) \geq f(\Pi_0) + \langle S, \Pi - \Pi_0 \rangle$$

(2.23)

Figure 2.8: Illustration of a convex function being touched by an affine lower bound. $S \in \partial f(\Pi_0)$

Hence, given an initial point $\Pi_0$, one can maximize the affine lower bound function (call it $f_{\text{low}}$) instead of $f$. This maximization results in a local maximizer $\Pi_1$ for $f_{\text{low}}$ and $f(\Pi_1) \geq f(\Pi_0)$ holds. Then, one re-starts the process. We develop another affine lower bound function $f_{\text{low}}$ at $\Pi_1$ and maximize it again. The process continues until a local maximizer for $f$ is located.

To sum up, we can construct an iterative routine, which we call MM, that iteratively increases the objective function by solving a sequence of single linear programs:

$$\Pi_{k+1} \in \arg \max_{\Pi \in C} f(\Pi_k) + \langle S_k, \Pi - \Pi_k \rangle$$

(2.24)

where $S_k \in \partial f(\Pi_k)$. Moreover, we have that:

$$\Pi_{k+1} \in \arg \max_{\Pi \in C} f(\Pi_k) + \langle S_k, \Pi - \Pi_k \rangle \Rightarrow f(\Pi_{k+1}) \geq f(\Pi_k) + \langle S_k, \Pi_{k+1} - \Pi_k \rangle \geq f(\Pi_k)$$

which formally shows that the sequence of objective values $(f(\Pi_k))$ is non-decreasing.
We have no guarantee that the sequence \((\Pi_k)_{k \in \mathbb{N}}\) converges to a global maximizer. Thus, we should build up a method for trying to escape from only-local maximizers.

To sum up, we will need a first step of optimization responsible to provide a local maximizer, followed by a second step responsible to escape from this one towards the global maximizer.

### 2.4.1 MM Routine

In this subsection we explain in more detail the inner working of the MM.

As a preliminary note, let \(f : \mathbb{R}^n \to \mathbb{R}\) be a differentiable function restricted to a regular surface \(C\) (of dimension \(n - 1\)). A necessary condition for a point \(x\) to be a maximizer of \(f\) is (from Lagrange multipliers theorem):

\[
\nabla f(x) = \lambda n_C(x) \tag{2.25}
\]

in which \(n_C(x)\) represents a vector orthogonal to \(C\) at \(x\), that is, a vector orthogonal to the tangent space of \(C^2\) at \(x\).

This is naturally extended to the case in which neither the restriction set nor the function are smooth objects. At a nonregular point \(x\) of \(C\), instead of a unique orthogonal vector, we have a set of vectors and we call it the normal cone of \(C\) at \(x\) and denote it by \(N_C(x)\). See Fig. 2.9 for an illustration.

![Figure 2.9: Normal cone \(N_C(x)\).](image)

The corresponding generalization of the gradient of a nonsmooth convex function is the subdifferential \(\partial f(x)\) (already defined in the previous section). The condition (2.25) now becomes (it will be

\footnote{If this surface could be characterized as the regular level set of a smooth function \(g : \mathbb{R}^n \to \mathbb{R}\), then by the implicit function theorem we can take \(n_C(x) = \nabla g(x)\) [6].}
more evident latter):

$$\partial f(x) \subset N_C(x) \quad (2.26)$$

We now recall the definitions of a normal cone and tangent cone.

**Definition.** Given $x_1, \ldots, x_m \in \mathbb{R}^n$, we say that $x$ is a conical combination of these elements if:

$$x = \sum_{i=1}^{m} \alpha_i x_i, \quad \alpha_i \geq 0, \text{ for all } i = 1, \ldots, m. \quad (2.27)$$

The set of all conical combinations of a set $S$ will be denoted by $\text{cone } S$. Such a set is in fact a cone (see Fig. 2.10). We can also write:

$$\text{cone } (S) = \mathbb{R}^+ (\text{co } S) \quad (2.28)$$

![Figure 2.10: cone $P$ represents the set of all conical combinations of the set $P$ (blue).](image)

**Definition.** The tangent cone of $C$ at $x$ is defined as follows:

$$T_C(x) = \text{cl}(\text{cone } (C - \{x\})) = \text{cl}\{d \in \mathbb{R}^n : d = \alpha(y - x), y \in C, \alpha \geq 0\} \quad (2.29)$$

**Definition.** The normal cone of $C$ at $x$ is defined as follows:

$$N_C(x) = \{s \in \mathbb{R}^n : \langle s, g \rangle \leq 0 \forall g \in T_C(x)\} \quad (2.30)$$

A necessary condition for a point $\Pi_k$ to be a maximizer is, then (see Fig. 2.11):

$$\langle G_k, S \rangle \leq 0 \forall S \in T_C(\Pi_k), g_k \in \partial f(\Pi_k) \Leftrightarrow \partial f(\Pi_k) \subset N_C(\Pi_k). \quad (2.31)$$
Figure 2.11: Representation of the tangent cone $T_C(x_k)$ and normal cone $N_C(x_k)$. In this case the point $x_k$ is a local maximizer because $\partial f(x_k) \subset N_C(x_k)$.

Therefore, we can check if a point $\Pi_k$ satisfies the necessary condition for optimality (2.31) by verifying whether:

$$\exists G_k \in \partial f(\Pi_k) : \max_{S \in T_C(\Pi_k)} \langle G_k, S \rangle > 0$$

(2.32)

is true or not. To solve this problem we will first assume that $f$ is smooth, so that $\partial f(\Pi_k) = \{ \nabla f(\Pi_k) \}$ for all $k$. Then, equation (2.32) becomes:

$$\max_{S \in T_C(\Pi_k)} \langle \nabla f(\Pi_k), S \rangle > 0$$

(2.33)

We can rewrite (2.33) as follows:

$$\max_{S \in T_C(\Pi_k)} \langle \nabla f(\Pi_k), S \rangle = \max_{Y \in C} \langle \nabla f(\Pi_k), Y - \Pi_k \rangle$$

$$= \max_{Y \in C} \langle \nabla f(\Pi_k), Y \rangle - \langle \nabla f(\Pi_k), \Pi_k \rangle$$

$$= \max_{Y \in \text{Proj}_n(n_1)} \langle \nabla f(\Pi_k), Y \rangle - \langle \nabla f(\Pi_k), \Pi_k \rangle,$$

(2.34)

in which $C = \text{co Proj}_n(n_1)$. The last equality comes from the fact that a solution of a maximization program with a convex cost function restricted to a convex and compact set $C$ can be found in the set of extreme points of $C$. Remark that $\text{Proj}_n(n_1) = \text{ext(co Proj}_n(n_1))$.

The solution of the latter program in (2.34) is the projector spanning the dominant eigenspace of
\( \nabla f(\Pi_k) \) with \( \dim = n_1 \), that is, it is the projector onto the subspace spanned by the \( n_1 \) eigenvectors associated to the \( n_1 \) highest eigenvalues of \( \nabla f(\Pi_k) \). In order to prove this claim, we shall invoke a well known theorem from linear algebra [12]:

**Theorem 2.4.1.** Let \( A \in S_n \) be a \( n \times n \) symmetric matrix, \( r \in \{1, ..., n\} \) and \( \lambda_n(A) \geq \ldots \geq \lambda_1(A) \), then:

\[
\lambda_n(A) + \ldots + \lambda_{n-r+1}(A) = \max_{Q^TQ = I_r} \text{tr}(Q^T AQ),
\]

(2.35)

where the solution \( Q^* \) has as columns the eigenvectors associated to the eigenvalues \( \lambda_n \geq \ldots \geq \lambda_{n-r+1} \) of \( A \).

From the equation below:

\[
\text{tr}(Q^T \nabla f(\Pi_k)Q) = \text{tr}(\nabla f(\Pi_k)QQ^T)
\]

(2.36)

and looking back to the last equality in equation (2.34) and to (2.35), we have that ( remarking that \( QQ^T \) represents geometrically the orthogonal projector onto \( \text{span}\{Q\} \)):

\[
\max_{Y \in \text{Proj}_{n}(n_1)} \langle \nabla f(\Pi_k), Y \rangle = \lambda_n(\nabla f(\Pi_k)) + \ldots + \lambda_{n-r+1}(\nabla f(\Pi_k))
\]

(2.37)

Hence, we can conclude that a point \( \Pi_k \) is a local maximizer when:

\[
\sum_{i=n-r+1}^{n} \lambda_i(\nabla f(\Pi_k)) - \langle \nabla f(\Pi_k), \Pi_k \rangle < 0.
\]

(2.38)

In other words, we just need to know the sum of the \( n_1 \) highest eigenvalues of \( G_k = \nabla f(\Pi_k) \) in order to verify whether a point satisfies the necessary condition for being a local maximizer.

Still, in light of the theorem 2.4.1, the next iteration of the program (2.24) is the projector spanning the dominant subspace of \( G_k \) with \( \dim = n_1 \). Thus, the characterization of \( T_C \) would be superfulous in this case. In fact, we are not working over the whole restriction set \( C \), the algorithm just works over the extreme points of this one.

**Remark:** It is worth to mention that 2.4.1 is the generalization of a well-known theorem which states that all of the critical points in:

\[
\max_{x \in S^n} x^T Ax
\]

(2.39)

in which \( A \) is a symmetric matrix and \( S^n \) represents the unit n-sphere, are the eigenvectors of \( A \), and
of course the global maximizer is the eigenvector associated to the highest eigenvalue. This theorem can be proved through the implicit function and lagrange multipliers theorem [6].

We have constructed a method to efficiently characterize the sequence (2.24) when \( f \) is a smooth function. But \( f_{tr} = \sum_{i=1}^{m} |x_i^\top \Pi x_i - \frac{1}{2}| \) is rather nonsmooth and in this case we need first to characterize \( \partial f_{tr} \). The nonregular points of \( f_{tr} \) correspond to those ones where at least one of the terms \( |x_i^\top \Pi x_i| \) is zero.

We have that [7]:

\[
\partial f_{tr}(\Pi) = \partial \sum_{i=1}^{m} \left| x_i^\top \Pi x_i - \frac{1}{2} \right| = \sum_{i=1}^{m} \partial \left| x_i^\top \Pi x_i - \frac{1}{2} \right| \tag{2.40}
\]

Since each term \( f_{tr}^i(\Pi) = |x_i^\top \Pi x_i - \frac{1}{2}| \) is the composition of the function \( g : \mathbb{R} \rightarrow \mathbb{R} \) given by \( g(x) = |x| \) with the affine function \( A : \mathbb{R}^{n \times n} \rightarrow \mathbb{R} \) given by \( A(\Pi) = x_i^\top \Pi x_i - \frac{1}{2} \), that is, \( f_{tr}^i(\Pi) = (g \circ A)(\Pi) \), we have from well-known convex analysis [7] that:

\[
\partial (g \circ A)(\Pi) = (\nabla A). (\partial g(A(\Pi))) \tag{2.41}
\]

We know (also from convex analysis) that \( \partial g(x) = \text{sgn}(x) \), in which \( \text{sgn} \) is the multifunction defined as follows:

\[
\text{sgn}(x) = \begin{cases} 
1 & \text{if } x > 0 \\
[-1, 1] & \text{if } x = 0 \\
-1 & \text{if } x < 0 
\end{cases} \tag{2.42}
\]
Therefore, $\partial f^i_tr(\Pi) = \text{sgn}(x_i^\top \Pi x_i - \frac{1}{2})x_i x_i^\top$, since $\nabla A(\Pi) = x_i x_i^\top$. Then, to answer (2.32), we have to solve:

$$\max\limits_{G_k} \max\limits_{S \in T_C(\Pi_k)} \langle G_k, S \rangle$$

and verify if it is greater than zero or not. But (2.43) can be written as (see (2.34) and (2.36)):

$$\max\limits_{G_k} \max\limits_{S \in T_C(\Pi_k)} \text{tr}(Q^\top G_k Q)$$

Since $G_k$ is an element of $\partial f(\Pi_k)$, it can be written as $G_k(\Pi) = \sum i \alpha_i x_i x_i^\top$ with $\alpha_i \in \text{sgn}(x_i^\top \Pi x_i - \frac{1}{2})$. If $\Pi$ lies in a region s.t. $f^i_tr(\Pi) = |x_i^\top \Pi x_i - \frac{1}{2}|$ is differentiable, then $\alpha_i = 1$ or $\alpha_i = -1$ depending whether $f^i_tr(\Pi)$ is greater or less than zero, respectively. If $\Pi$ is a point in which $f^i_tr(\Pi)$ is nonsmooth, then $\alpha_i = [-1, 1]$. In order to choose one $G_k \in \partial f(\Pi_k)$, we have to choose $\alpha_i \in [-1, 1]$ for those terms $f^i_tr$ nonsmooth. To sum up, we have to choose properly the coefficients $\alpha_i$ of the nondifferentiable terms in order to maximize (2.44).

Given:

$$\max\limits_{G_k} \max\limits_{Q Q^\top \in \text{Proj}_n(n_1)} \sum_i \alpha_i \text{tr}(Q^\top x_i x_i^\top Q)$$

we can observe that $\text{tr}(Q^\top x_i x_i^\top Q) \geq 0$ for all $i$ and therefore, this implies that the function defined as:

$$f(\alpha_1, \ldots, \alpha_m) = \max\limits_{Q Q^\top \in \text{Proj}_n(n_1)} \sum_{i=1}^m \alpha_i \text{tr}(Q^\top x_i x_i^\top Q)$$

is s.t. $f(\alpha_1, \ldots, \alpha_m) \geq f(\beta_1, \ldots, \beta_m)$ if $\alpha_i > \beta_i \forall i$. Thus, the solution of (2.44) is constructed by choosing the coefficients (associated to the nonsmooth terms $(x_i^\top \Pi x_i - \frac{1}{2})$ as $\alpha_i = 1$. That is, we can construct $G_k$ by choosing $\alpha_i = 1$ for all those nonsmooth terms and then we can solve the problem:

$$\max\limits_{Y \in \text{Proj}_n(n_1)} \langle G_k, Y \rangle$$

which has already been done. Then we proceed to the next iteration.

Furthermore, we can prove that the sequence $(\Pi_k)_{k \in \mathbb{N}}$ aforementioned is convergent when the cost function is $f_tr(\Pi) = \sum_{i=1}^m |(x_i^\top \Pi x_i) - \frac{1}{2}|$ (its proof is provided in A.1.7):
Theorem 2.4.2. Let $X = [x_1 \ x_2 \ \cdots \ x_m]$ be the data matrix. The MM routine with objective function $f_R(\Pi) = \sum_{i=1}^m |(x_i^\top \Pi x_i) - \frac{1}{2}|$ produces a sequence:

$$\Pi_{k+1} \in \text{span}_{n_1} \{XD(\Pi_k)X^\top\}$$

where $\text{span}_{n_1} \{M\}$ denotes the set of projectors spanning the dominant eigenspace of $M$ with $\dim = n_1$, and:

$$D(\Pi_k) = \begin{bmatrix} \alpha_1(\Pi_k) \\ \alpha_2(\Pi_k) \\ \vdots \\ \alpha_m(\Pi_k) \end{bmatrix}$$

with

$$\alpha_i(\Pi_k) = \begin{cases} 1 & \text{if } x_i^\top \Pi_k x_i \geq \frac{1}{2} \\ -1 & \text{if } x_i^\top \Pi_k x_i < \frac{1}{2} \end{cases}$$

If $x_i^\top \Pi^* x_i \neq \frac{1}{2}$ for all $i$, then $\Pi_k \to \Pi^*$ and $\Pi^*$ satisfies the necessary conditions for being a local maximizer.

In conclusion, we have developed a (computationally efficient) way to verify whether a point is a local maximizer and also a scheme to compute the next iteration of (2.24) (computationally quick, as well).

In order to initialize our algorithm, we can choose:

$$\Pi_0 \in \arg \min_{\Pi \in \text{Proj}_{n_1}} \|X - \Pi X\|.$$  

(2.50)

where $X$ is the data matrix and $\|\cdot\|$ is the Frobenius norm.

The solution of (2.50) is the projector spanning the $n_1$-dimensional dominant eigenspace of $X^\top X$ (see A.1.4). In other words, to initialize the algorithm we shall first evaluate the SVD of $X$ yielding $X = U \Sigma V^\top$ and then, we extract the first $n_1$ columns of the Stiefel matrix $U = [u_1 \ \cdots \ u_{\text{rank} X}]^3$ obtaining $\Pi_0 = \sum_{i=1}^{n_1} u_i u_i^\top$.

We present in Fig. 2.13 a top level summary of the MM routine coupled with the ELM strategy. Also, we present in table 2.1 a more detailed sketch of the MM routine.

---

3Remark that we are assuming the $n_1$-truncated SVD (see pg. 12).
In: \( n_1, X \)

step 1) Evaluate the \( n_1 \)-truncated SVD of \( X \): \( X = U \Sigma V^T \)

step 2) \( k \leftarrow 0, \Pi_0 \leftarrow UU^T \)

step 3) Construct \( G_k = \sum^m_{i=1} \alpha_i x_i x_i^T \), where:

\[
\alpha_i = \begin{cases} 
1 & \text{if } |x_i^T \Pi_k x_i - \frac{1}{2}| \geq 0 \\
-1 & \text{if } |x_i^T \Pi_k x_i - \frac{1}{2}| < 0 
\end{cases}
\]

step 4) Check if \( \Pi_k \) satisfies the necessary condition of local maximizer \( (\partial f(\Pi_k, X) \subset N_C(\Pi_k)) \):

\[
\sum_{i=n-r+1}^n \lambda_i(G_k) - \langle G_k, \Pi_k \rangle < 0
\]

If yes, \( \Pi^* \leftarrow \Pi_k \) and go to out. If not, go to 5).

step 5) If \( k > 0 \), construct \( G_k = \sum^m_{i=1} \alpha_i x_i x_i^T \), where:

\[
\alpha_i = \begin{cases} 
1 & \text{if } |x_i^T \Pi_k x_i - \frac{1}{2}| \geq 0 \\
-1 & \text{if } |x_i^T \Pi_k x_i - \frac{1}{2}| < 0 
\end{cases}
\]

else, continue to step 6).

step 6) Choose \( \Pi_{k+1} \) such that

\[
\Pi_{k+1} \in \text{arg max } \langle G_d, \Pi \rangle \quad \Pi \in C
\]

Do \( k \leftarrow k + 1 \) and jump to 3).

Out: \( \Pi^* \in \text{Proj}_n(n_1) \) (local maximizer)

Table 2.1: MM routine: \( n_1 \) represents the dimension of one of the (two pairwise orthogonal) subspaces, \( X \) is the data matrix, \( \Pi_k \) is the \( k \)th projector in the sequence defined by the algorithm and \( \Pi^* \) is the local maximizer.

25
2.4.2 Qualitative Analysis and the ELM Algorithm

The local maximizers returned typically by the MM routine convey relevant geometric information (even in presence of outliers). In general, these local maximizers use to be “near” from $L_1$ or $L_2$.

To illustrate, let $\{b_1, \ldots, b_{10}\}$ be an orthonormal basis for the ambient space $\mathbb{R}^{10}$. Let’s construct a constellation over a 4-dimensional subspace $L_1$ spanned by $\{b_1, \ldots, b_4\}$. Assume we are also in the presence of outliers. We can run the MM routine to hunt this subspace and show the result in Fig. 2.14 (which represents also a typical case).

We can verify that in Fig. 2.14(a) there is no pattern in the energy profile. Whereas in Fig. 2.14(b) we can see clearly an assimmetry in the energy profile pointing out the basis $\{b_1, \ldots, b_4\}$ that spans $L_1$.

This observation suggests that given a maximizer $\Pi^*$, probably, those points $x_i$ with lower or greater values of $h(x_i) = x_i^\top \Pi^* x_i$ (call it dominant points) could be in fact in one of the subspaces $L_1^\perp$ or $L_1$, respectively. Then, as an heuristic, we can construct a new projector $\Pi_{\text{new}}$ onto the subspace spanned by the points with greater or lower value of $h(x_i)$, re-initializing the MM with this new projector if $f(\Pi_{\text{new}}) > f(\Pi^*)$.

To sum up, we can sort the points, based on its values of $h(x_i)$, in increasing order and decreasing order. Then, we can construct a projector onto $L_{\text{new}} = \text{span}\{x_1, \ldots, x_n : h(x_1) \leq \ldots \leq h(x_n)\}$. The routine is illustrated in Fig. 2.15.

Fig. 2.16 illustrates the routine in more detail. Note that in the strategy proposed before, we have
Figure 2.14: Fig.(a) represents \( b_i^\top \Pi_0 b_i \) vs \( i \) where \( \Pi_0 \) is an initialization (a projector of rank 4). Fig.(b) represents \( b_i^\top \Pi^* b_i \) vs \( i \) where \( \Pi^* \) is a maximizer returned by the MM. The first four \( b_i \)'s represent an orthogonal basis for \( L_1 \).

There is no guarantee that the first \( n_1 \) points from the sorted set \( X \) lies within the activated subspace, but nevertheless we expect that the majority comes from \( L_1 \). Thus, as an heuristic, if the projector constructed as explained above does not beat the current one, we can add one more dominant point to the set \( \{x_1, ..., x_{n_1} : h(x_1) \leq ... \leq h(x_{n_1})\} \) or to the set \( \{x_1, ..., x_{n_1} : h(x_1) \geq ... \geq h(x_{n_1})\} \) yielding a new dominant set \( X_{27} = \{x_1, ..., x_{n_1+i}\} \). Then, we can construct a projector that at least minimizes \( \|X - \Pi X\| \). We can continue with this strategy up to find a better projector or to end a pre-defined number of iterations.

The algorithm depicted in Fig. 2.16 seems heavy from the computational view point in the sense that there is an SVD per each point \( x_i \) added in the loop. However, there are fast algorithms to update SVD’s when a column is added to the matrix without having to compute from scratch each time a column is added [8]. Thus, the algorithm can exploit this recursive form of SVD to reduce the computational load.

### 2.5 Log-Version of the \( f_{tr} \) Objective Function

Our aim is to build up an algorithm to achieve a global solution in presence of outliers and for a general configuration of the activated subspaces. Thus, a necessary condition for a good performance is that the stronger maximizers of the objective function should codify the activated subspaces.

In fact, \( f_{tr} \) comes up by the assumption that the subspaces are orthogonal. As we have seen, this configuration is not a limitation when there is no outliers, since we can transport the oblique case to the orthogonal case by a simple pre-processing of the data matrix. But, in the presence of outliers this
pre-processing is no longer valid and we have in fact to deal with oblique subspaces.

Through a simple example we can have an insight about the behavior of $f_{tr}$ in the orthogonal and oblique cases.

Assume we have two activated subspaces $\mathcal{L}_1$ and $\mathcal{L}_2$ with $\dim \mathcal{L}_1 = \dim \mathcal{L}_2 = 1$ and the ambient space is $\mathbb{R}^2$. The idea here is to plot the graph of $f_{tr}$ with domain in $\text{Proj}_2(1)$. Fig. 2.17 illustrates the main parameters that will characterize this experience. Actually, we will reparametrize $f_{tr}$ w.r.t $\theta$.

Fig. 2.18 presents the graph of $f_{tr}$ as function of the angle $\theta$. In this case, the subspaces $\mathcal{L}_1$ and $\mathcal{L}_2$ are assumed to be orthogonal. We can observe that by increasing the concentration of outliers, the position of the maximizers start becoming fuzzy and come out local maximizers.

Now let’s take a look at the oblique case. Fig. 2.19 depicts the graph of $f_{tr}$ for this case. We can see that there is a displacement between the position of the maximizers and the points codifying the activated subspaces. Furthermore, Fig. 2.19 shows that when $|\theta_1 - \theta_2| = 45^\circ$, the activated subspaces are minimizers of $f_{tr}$.

In conclusion, the objective function $f_{tr}$ fails to be inherently immune to outliers in the oblique case in the sense that it does not reach its maximum value at an activated subspace, necessarily. In fact, the displacement between the maximizers of $f_{tr}$ and $\theta_1$ and $\theta_2$ can vary significantly and is a function of this latter angles.

To bypass this default, we can propose another cost function $\phi(\cdot)$ to replace $\phi_{tr}(s) = |s - \frac{1}{2}|$. We will consider the function $\phi_{\log} : [0, 1] \rightarrow \mathbb{R}$ defined by $\phi_{\log}(s) = \log(s(1-s)+\varepsilon)$ and therefore, our new
cost function is given by:

\[
f_{\log}(\Pi) = \sum_{i=1}^{m} \phi_{\log}(x_i^T \Pi x_i) = -\sum_{i=1}^{m} \log((x_i^T \Pi x_i)(1 - x_i^T \Pi x_i) + \varepsilon) \quad (2.51)
\]

One can expect that with \( \varepsilon \) low enough, the global solution of the model with \( f_{\log} \) represents an activated subspace.

Fig. 2.21 contains the graphs of \( f_{\log} \) for the same angles \( \theta_1 \) and \( \theta_2 \) considered for the \( f_{tr} \) function. One can observe that the maximizers in this cases are better defined and are not displaced from the activated subspaces.

The only drawback we can observe with this cost function right from the start is that greater the concentration of outliers, greater the concentration of non-interesting maximizers. Fig. 2.22 illustrates the graph of \( f_{\log} \) when the concentration of outliers is 20\%.
Figure 2.17: Geometric illustration of the parameters in the experience. \( \Pi \) represents a projector of rank 1, \( \theta \) is the angle between \( \text{span}(\Pi) \) and the canonical vector \( e_1 \) (and is also the variable of \( f_{tr} \)) and \( \theta_i \) is the angle between \( \mathcal{L}_i \) and \( e_1 \).

Figure 2.18: Fig.(a) and (b) depict the graph of \( f_{tr} \) in the orthogonal case with 30% and 60% of outliers, respectively.
Figure 2.19: Fig.(a) and (b) depicts the graph of $f_{tr}$ assuming 30% of outliers.

Figure 2.20: Graph of $\phi_{\log}$ with $\varepsilon = 10^{-5}$. 
Figure 2.21: Fig.(a) and (b) depicts the graph of $f_{\log}$ assuming 30% of outliers.

Figure 2.22: Graph of $f_{\log}$ with $|\theta_1 - \theta_2| = 45^\circ$ and 20% of outliers.
2.6 Final Algorithm

The algorithm presented so far is prepared to seek for a couple of pairwise orthogonal subspaces even in the presence of outliers.

In order to turn the algorithm applicable in more general configurations (e.g., oblique case), the strategy is to catch iteratively single subspaces. Thus, we can run the algorithm as many times as the number of activated subspaces and in each iteration, the algorithm would return a subspace.

Therefore, we should concentrate our attention to the single subspace segmentation problem (in presence of outliers). And we can see that an inconvenient still remains in the model. We can illustrate the problem through an example.

Suppose we have to identify a straight line \( r \) in presence of outliers in \( \mathbb{R}^3 \). Then, we have to solve:

\[
\max_{\Pi \in \text{Proj}_3(1)} f(\Pi) \quad \text{or} \quad \max_{\Pi \in \text{Proj}_3(2)} f(\Pi)
\]  

(2.52)

![Diagram](image.png)

Figure 2.23: Illustration of the two possible solutions \( \Pi_1^* \) and \( \Pi_2^* \) when the restriction set is chosen to be \( \text{Proj}_3(2) \). The yellow points represent outliers. \( \text{span}\{\Pi_1^*\} \) and \( \text{span}\{\Pi_2^*\} \) represents the subspaces in red and blue, respectively. One of the yellow points lies within \( \text{span}\{\Pi_1^*\} \) jointly with the straight line \( r \) (green).

If we choose to solve the second program in (2.52) it means that the solution will be a projector \( \Pi^* \) of rank 2, since \( \text{Proj}_3(2) \) is the restriction set. Actually, there would be two strong solutions \( \Pi_1^* \) and \( \Pi_2^* \)
such that \( \text{span}\{\Pi_1^*\} \supset r \) and \( \text{span}\{\Pi_2^*\} = r^\perp \). In the latter case, we obtain the expected straight line. In the former case, the plane \( \text{span}\{\Pi_1^*\} \) would contain the inliers over the straight line, but it would also contain one more point not lying in the straight line \( r \). Thus, since the activated solution we are seeking for is a straight line, we shall reduce the ambient space dimension. That is, we can take the SVD of the data matrix \( X_I: X_I = U \Sigma V^\top \) consisting of those points of \( X \) lying within \( \text{span}\{\Pi_1^*\} \). Then, we can work over the columns of \( V^\top \). Essentially, we are performing here a PCA of the data \( X_I \). Now the ambient space has dimension 2 and we are seeking for a straight line. We have to solve:

\[
\max_{\Pi \in \text{Proj}_2(1)} f(\Pi) \tag{2.53}
\]

in which \( V^\top \) is the new data matrix and it is conveyed in the cost function \( f \). Equation (2.53) has as solution a rank 1 projector onto the straight line \( \hat{r} \) or onto \( \hat{r}^\perp \). Then \( r \) can be recovered by looking at the column indexes of \( V^\top \) that lies in \( \hat{r} \) or \( \hat{r}^\perp \). The same reasoning could be done if we had chosen the right hand side program in (2.52).

We present in table 2.2 the recursive version of the method for identifying a single subspace.

<table>
<thead>
<tr>
<th>In: ( n_1, X )</th>
</tr>
</thead>
<tbody>
<tr>
<td>step 1) ( \Pi^* \leftarrow \text{MM-ELM}(X, n_1) )</td>
</tr>
<tr>
<td>step 2) Choose ( I ) s.t. ( x_i^\top \Pi^* x_i \leq 10^{-5} \forall i \in I ) and ( J ) s.t. ( (1 - x_j^\top \Pi^* x_j) \leq 10^{-5} \forall j \in J )</td>
</tr>
<tr>
<td>step 3) If ( (#(I) &lt; n_1 + l \text{ and } #(J) &lt; n_1 + l) ) return empty.</td>
</tr>
<tr>
<td>step 4) If ( #(J) \geq n_1 + l ) return ( J ).</td>
</tr>
<tr>
<td>step 5) If ( (\text{size}(X, 1) - n_1 = n_1) ) return ( I ).</td>
</tr>
<tr>
<td>step 6) ( X_I = U \Sigma V^\top )</td>
</tr>
<tr>
<td>step 7) ( X \leftarrow V^\top \text{(PCA)}. \text{Jump to step 1.} )</td>
</tr>
</tbody>
</table>

Out: Set of Indexes of those points lying within \( L_1 \).

Table 2.2: Recursive version of the MM-ELM algorithm. \( I \) and \( J \) are sets of indexes. \( X_I \) is the matrix constructed from the data matrix \( X \) extracting those points \( x_i \) s.t. \( i \in I \). The output of the algorithm is the set of indexes labeling the points embedded in \( L_1 \) (activated \( n_1 \)-dimensional subspace). In this case, we are assuming that the restriction set is \( \text{Proj}_{n_1}(n_1) \) (we could also choose \( \text{Proj}_{n_1}(n - n_1) \)). \( \text{size}(X, 1) \) is the number of rows of \( X \), that is, the current ambient space dimension.

In conclusion, the strategy adopted require prior knowledge regarding the number and dimensions of the activated subspaces. Given this information, the algorithm returns each subspace per iteration using the recursive strategy illustrated in the example above to come out with the subspace with the
desired dimension. Note that with this strategy the condition $\mathbb{R}^n = \mathcal{L}_1 \oplus \ldots \oplus \mathcal{L}_d$ in the problem formulation can be relaxed to $\mathbb{R}^n \supset \mathcal{L}_1 \oplus \ldots \oplus \mathcal{L}_d$.

### 2.7 Synthetic Results

We now present some results from synthetic simulations with the algorithm MM-ELM.

The conditions of the simulations in this section were: $n_1 = \dim \mathcal{L}_1 = 4$, $n_2 = \dim \mathcal{L}_2 = 4$, $n = 10$ (ambient space dimension) and $m = 100$ (number of data points). We have chosen 4-dimensional subspaces because our main motivation lies in the motion segmentation and as we have seen in section 1.2, for general motions, the subspaces have dimension 4.

All simulations were made subject to $\#(X \cap \mathcal{L}_1) = \#(X \cap \mathcal{L}_2)$, that is, the number of elements of $X$ in $\mathcal{L}_1$ is equal to the number of elements of $X$ in $\mathcal{L}_2$.

#### 2.7.1 Computer Simulations: Noiseless Case

Before present the computer simulation results, we introduce some definitions used for the terminology in the heading of the tables of results.

The term outlier has the same definition as that one stated in the beginning of the chapter. The term Reliability in the tables is defined as $\frac{N_{\text{success}}}{N_{\text{Monte-Carlo}}}$, where $N_{\text{success}}$ is the number of times the algorithm found all of the activated subspaces in case and $N_{\text{monte-carlo}}$ is the total number of simulations.

We will use also the terminology i one subspace ii Two Orthogonal Subspaces iii Two Oblique Subspaces (with trivial or nontrivial intersection) to specify what kind of configuration have been adopted to obtain the results in each table. When there is a nontrivial intersection, we will assume that $\dim \mathcal{L}_1 \cap \mathcal{L}_2 = 1$.

A total of 10,000 Monte-Carlos were used to obtain the reliability index in all of the tables.
• **One Subspace: MM-only versus MM-ELM.**

We start by presenting in table 2.3 a comparison of the performance of the MM-only vs MM-ELM, in order to highlight the important gain in performance provided by the ELM. This is the only experience with the pure MM routine, the remaining results have been obtained with the MM-ELM algorithm.

<table>
<thead>
<tr>
<th>Outliers[%]</th>
<th>Elements in $\mathcal{L}_1$ [%]</th>
<th>Reliability MM-only [%]</th>
<th>Reliability MM-ELM [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>50</td>
<td>87.57</td>
<td>100</td>
</tr>
<tr>
<td>60</td>
<td>40</td>
<td>80.31</td>
<td>100</td>
</tr>
<tr>
<td>70</td>
<td>30</td>
<td>63.24</td>
<td>100</td>
</tr>
<tr>
<td>80</td>
<td>20</td>
<td>28.43</td>
<td>99.93</td>
</tr>
<tr>
<td>85</td>
<td>15</td>
<td>*</td>
<td>96.93</td>
</tr>
<tr>
<td>90</td>
<td>10</td>
<td>*</td>
<td>52.13</td>
</tr>
</tbody>
</table>

Table 2.3: Comparison: MM-only vs MM-ELM. One 4-dimensional activated subspace with cost function $f_{\log}$.

• **Two Orthogonal Subspaces: $f_{\log}$ versus $f_{tr}$.**

<table>
<thead>
<tr>
<th>Outliers[%]</th>
<th>Elements in $\mathcal{L}_1$ [%]</th>
<th>Elements in $\mathcal{L}_2$ [%]</th>
<th>Reliability $f_{\log}$ [%]</th>
<th>Reliability $f_{tr}$ [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>50</td>
<td>50</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
<td>35</td>
<td>100</td>
<td>99.99</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>25</td>
<td>98.94</td>
<td>96.85</td>
</tr>
<tr>
<td>60</td>
<td>20</td>
<td>20</td>
<td>94.22</td>
<td>74.50</td>
</tr>
<tr>
<td>70</td>
<td>15</td>
<td>15</td>
<td>78.09</td>
<td>33.12</td>
</tr>
<tr>
<td>75</td>
<td>13</td>
<td>12</td>
<td>64.56</td>
<td>*</td>
</tr>
<tr>
<td>80</td>
<td>10</td>
<td>10</td>
<td>18.14</td>
<td>*</td>
</tr>
</tbody>
</table>

Table 2.4: Comparison: $f_{\log}$ versus $f_{tr}$. Two 4-dimensional orthogonal activated subspaces.

• **Two Oblique Subspaces: $f_{\log}$ versus $f_{tr}$.**

<table>
<thead>
<tr>
<th>Outliers[%]</th>
<th>Elements in $\mathcal{L}_1$ [%]</th>
<th>Elements in $\mathcal{L}_2$ [%]</th>
<th>Reliability $f_{\log}$ [%]</th>
<th>Reliability $f_{tr}$ [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>50</td>
<td>50</td>
<td>100</td>
<td>*</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
<td>35</td>
<td>100</td>
<td>*</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>25</td>
<td>99.63</td>
<td>*</td>
</tr>
<tr>
<td>60</td>
<td>20</td>
<td>20</td>
<td>97.98</td>
<td>*</td>
</tr>
<tr>
<td>70</td>
<td>15</td>
<td>15</td>
<td>83.45</td>
<td>*</td>
</tr>
<tr>
<td>75</td>
<td>13</td>
<td>12</td>
<td>69.87</td>
<td>*</td>
</tr>
<tr>
<td>80</td>
<td>10</td>
<td>10</td>
<td>32.63</td>
<td>*</td>
</tr>
</tbody>
</table>

Table 2.5: Comparison: $f_{\log}$ versus $f_{tr}$. Two 4-dimensional oblique activated subspaces.
• One Subspace: \( f_{\log} \) versus \( f_{\text{tr}} \).

<table>
<thead>
<tr>
<th>Outliers[%]</th>
<th>Elements in ( L_1 )[%]</th>
<th>Reliability ( f_{\text{tr}} )[%]</th>
<th>Reliability ( f_{\log} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>50</td>
<td>94.27</td>
<td>100</td>
</tr>
<tr>
<td>60</td>
<td>40</td>
<td>72.83</td>
<td>100</td>
</tr>
<tr>
<td>70</td>
<td>30</td>
<td>*</td>
<td>100</td>
</tr>
<tr>
<td>80</td>
<td>20</td>
<td>*</td>
<td>99.93</td>
</tr>
<tr>
<td>85</td>
<td>15</td>
<td>*</td>
<td>96.28</td>
</tr>
<tr>
<td>90</td>
<td>10</td>
<td>*</td>
<td>52.13</td>
</tr>
</tbody>
</table>

Table 2.6: Comparison: \( f_{\log} \) versus \( f_{\text{tr}} \). One 4-dimensional activated subspaces.

• Two Oblique Subspaces with Nontrivial Intersection vs Two Oblique Subspaces with trivial Intersection.

<table>
<thead>
<tr>
<th>Outliers[%]</th>
<th>Elements in ( L_1 )[%]</th>
<th>Elements in ( L_2 )[%]</th>
<th>Reliability Nontrivial[%]</th>
<th>Reliability Trivial</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>50</td>
<td>50</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
<td>35</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>25</td>
<td>97.24</td>
<td>99.63</td>
</tr>
<tr>
<td>60</td>
<td>20</td>
<td>20</td>
<td>77.06</td>
<td>97.98</td>
</tr>
<tr>
<td>70</td>
<td>15</td>
<td>15</td>
<td>34.51</td>
<td>83.45</td>
</tr>
</tbody>
</table>

Table 2.7: Comparison: two 4-dimensional oblique subspaces subject to \( \dim L_1 \cap L_2 = 1 \) versus two 4-dimensional subspaces with trivial intersection. Cost function used was \( f_{\log} \).

2.7.2 Computer Simulations: Noisy Case

Now we consider the case where the data set is corrupted by noise. In order to emulate this environment we can generate synthetically two groups of points over two subspaces \( L_1 \) and \( L_2 \) and additionally a group of outliers. After this step, we can perturb a little bit each point in a random direction. To \( L_1 \) and \( L_2 \) we call the groundtruth subspaces.

Thus, the data set \( X \) is such that \( X = Y \cup O_X \), where:

\[
Y = \{ y_i = p_i + \eta_i \in \mathbb{R}^n : \eta_i \sim \mathcal{N}(0, \sigma^2 I), p_i \sim B_j \mathcal{N}(0, I), B_j^T B_j = I \text{ for } j=1,2 \text{ and for } i = 1, \ldots, m \}
\]

and

\[
O_x = \{ o_i \in \mathbb{R}^n : o_i \sim \mathcal{N}(0, I) \}
\]

(2.54)
The signal-to-noise ratio for the inliers is defined as:

$$SNR = \frac{E\{\|p_i\|^2\}}{E\{\|\eta_i\|^2\}}$$

(2.55)

which is equivalent to:

$$SNR = \frac{E\{tr(p_ip_i^\top)\}}{E\{tr(\eta_i\eta_i^\top)\}} = \frac{tr(E\{p_ip_i^\top\})}{tr(E\{\eta_i\eta_i^\top\})}$$

(2.56)

Considering that $p_i \in L_1 \forall i$ (the case $p_i \in L_2$ is similar), we can write $p_i = Bs_i$, where $n_1 = \dim L_1$ and $B : n \times n_1$ is a Stiefel matrix, that is, $B^\top B = I$ (we will denote it by $B$ to compact notation). Thus,

$$tr(p_ip_i^\top) = tr(Bss_i^\top B^\top) = tr(s_is_i^\top)$$

(2.57)

Therefore:

$$tr(E\{p_ip_i^\top\}) = tr(E\{s_is_i^\top\}) = tr(\sigma_n^2 I_{n_1 \times n_1}) = n_1$$

(2.58)

remarking that $\sigma_n^2 = 1$ since $B(\cdot)$ is an isometry and $\sigma_p^2 = 1$.

Finally, we have that:

$$SNR = \frac{n_1}{tr(\sigma_n^2 I)} = \frac{n_1}{n_1 \sigma_n^2}$$

(2.59)

Equation 2.59 states the closed form relation between the variance $\sigma_n^2$ and the SNR. In conclusion, we can generate points $p_i$ lying within an $n_1$-dimensional subspace such that $\Sigma_p = I_{n_1 \times n_1}$ and then, we can perturb this points $p_i$:

$$x_i = p_i + \eta_i$$

(2.60)

where $\eta_i \sim \mathcal{N}(0, \sigma^2_n I)$

Thus, in order to classify the data points, we can measure the distance of each point to the subspaces returned by the MM-ELM. Then, based on a previously defined threshold $\epsilon$, we can segment the set $X$ into groups of points. In conclusion, we say that a point $x_i$ comes from the subspace $L_1 = \text{span}\{\Pi^*\}$ where $\Pi^* \in \text{Proj}_{n}(n_1)$ (returned by the MM-ELM), if:

$$\|x_i - \Pi x_i\| \leq \epsilon$$

(2.61)

We can then, compare the classifications we have made with the groundtruth information and evaluate the number of correct classifications. Note that it just makes sense to consider a point as correct classified if it lies in a group in which the majority of the points comes from the same subspace. For the simulations that follows, we adopted a strict criterion. If a group segmented by the MM-ELM contains
points from different activated subspaces or even outliers, then we considered all classifications as wrongs.

The conditions for the simulations were again $n_1 = 4$, $n_2 = 4$, $n = 10$ and $m = 100$ data points (including the outliers). The threshold adopted for the following simulations was $\varepsilon = 0.001$. The cost function used in the MM-ELM algorithm was $f_{\log}$.

![Figure 2.24: One activated Subspace: average of correct classifications for 5000 trials.](image)

![Figure 2.25: Two oblique subspaces with trivial intersection: average of correct classifications for 5000 trials.](image)
2.8 Advantage From the ELM or From More Trials?

We could ask whether the gain in performance of the MM-ELM algorithm (see section 2.7) comes out directly from the heuristic incorporated in the ELM routine or if it is simply due to the fact that the ELM is just a routine exploring the search space through sampling. Thus, it is fair to compare the ELM routine with an algorithm just consisting of random trial points, which we refer to as RS (random sampling) routine. That is, a routine that tries to escape from the local maximizers by choosing random projectors and checking if they beat the current local maximizer. We shall investigate how much trials the ELM and RS routines require to escape from a maximizer (in average).

We expect that a good heuristic turns out to be fast in escaping, that is, could escape within the first trials whereas a random shooting (in general) does not have to have any pattern. Then, we will perform the following experience: we will run 1000 times the ELM routine from a local maximizer. Thus, we will count in each monte-carlo, how many trials such a routine needed to escape. The RS routine will try up to 200 times per monte-carlo. Then, we can construct an histogram with the “number of escaping’s vs number of the trial”. The results are presented in Fig. 2.26

![Fig. 2.26: Rights vs number of trial. Curve (a) is associated to the scape and curve (b) is associated to the random shootings routine.](image)

The conditions in this experience were: 85% of outliers, $n = 10$, $m = 100$, $\dim \mathcal{L}_1 = 4$ (only one subspace). The objective function was $f_{\log}$. We can see that the ELM could escape at first trial 826 times out of 1000. Looking to Fig. 2.26(b) we can see that the RS routine could never escape in any of the 200 trials per monte-carlo. This fact suggests that the ELM plays a critical role in the overall
performance of the algorithm. We can compare, as well, the reliability of both algorithms in this case:

\[
\text{Reliability}_{\text{ELM}} = 92.52\% \\
\text{Reliability}_{\text{RS}} = 47.5\%
\] (2.62)

2.9 Real Data Results

We have tested our approach on three sequence of images provided by the Hopkins155 database [19]: three bodies, arm and car sequences. In Fig. 2.27 we present the first frame of the films that have been chosen to test our algorithm. The three bodies’ sequence is an indoor sequence consisting of three linked bodies, forming one articulated object, with 150 inlier trajectories over 31 frames. The arm sequence has 77 inlier trajectories over 30 frames. The car sequence has 307 inlier trajectories over 20 frames. Additionally, 20% of outlier trajectories have been added to each sequence.

The results are shown in tables 2.8, 2.9 and 2.10. We shall highlight that no outlier has been misclassified as inlier.

<table>
<thead>
<tr>
<th>Three Bodies</th>
<th>Yellow</th>
<th>red</th>
<th>blue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of points in the sequence</td>
<td>68</td>
<td>37</td>
<td>45</td>
</tr>
<tr>
<td>Number of points correctly estimated</td>
<td>67</td>
<td>36</td>
<td>45</td>
</tr>
</tbody>
</table>

Table 2.8: One point from the yellow group and another from the red group have been misclassified as outliers. The green points represent random trajectories (outliers) added.

In the articulated case we can see that only two points, one from the yellow group and the other from the red group, have been misclassified as outliers. The remaining points have been correctly classified.

In the arm case the performance has decreased. Nevertheless, no points from one group have been classified as being from the other group, that is, all of the misclassified inliers had been classified as outliers.

<table>
<thead>
<tr>
<th>Arm</th>
<th>red</th>
<th>blue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of points in the sequence</td>
<td>32</td>
<td>45</td>
</tr>
<tr>
<td>Number of points correctly estimated</td>
<td>19</td>
<td>25</td>
</tr>
</tbody>
</table>

Table 2.9: Thirteen points from the red group and twenty points from the red group have been misclassified as outliers. The green points represent random trajectories (outliers) added.

The same happened in the car case.
Table 2.10: Fifty two points from the red group and fifty five points from the green group have been misclassified as outliers. The blue points represent random trajectories (outliers) added.

2.10 Conclusion

In order to tackle the subspace-segmentation problem, an algorithm (MM-ELM) has been created. The MM-ELM algorithm cycles through 2 routines: MM reaches a maximizer and ELM tries to escape from this maximizer (re-initializing the MM). From the computer simulation results, we have observed that the algorithm with $f_{\text{log}}$ is indeed more immune to outliers in the oblique case than $f_{\text{tr}}$. In fact, the algorithm cannot rely on $f_{\text{tr}}$ when dealing with the oblique case. We could also see that the articulated case in the absence of noise is not a trouble maker with this approach comparing to other methods, for instance [2], even in presence of outliers.
Figure 2.27: First frame of the tested sequences from the Hopkins155 database. In all of them, 20% of outliers were added to the dataset.
Chapter 3

Subspace Segmentation - General Formulation

3.1 Chapter Summary

Section 3.2 [Data Model]. In this section, we will formulate the subspace-segmentation problem with a new approach. Our strategy will consist in seeking activated subspaces by finding vectors in their orthogonal complements. We will show that it is possible to characterize this problem through the model:

$$\min \|X^T u\|_0, \quad \|u\| = 1$$

(3.1)

where $X$ is the data matrix and $\| \cdot \|_0$ represents the $l_0$-norm. We will show that the global minimizer of (3.1) is orthogonal to a subspace that contains activated subspaces.

However, finding the solutions of the previous model is a hard combinatorial problem. A good heuristic (in a sense to be defined ahead) to approximate this program is:

$$\min \|X^T u\|_1, \quad \|u\| = 1$$

(3.2)

where $\| \cdot \|_1$ represents the $l_1$-norm.

Despite the fact that the objective function is convex, the restriction set is not convex (it is the unit-sphere). So local minimizers are to be expected.

Just as a note, we will adopt here the same definitions for outliers and activated subspaces as in
Section 3.3 [Algorithm]. In this section, the aim is to tackle (3.2) through two algorithms.

The first algorithm named Newton-sphere (NS), minimizes a generic smooth function restricted to the unit-sphere. The NS algorithm is initialized from many different points over one of the hemispheres of the sphere (the objective function is symmetric). The best minimizer is then selected.

The second algorithm NS-ELM arises from the fact that the NS is usually trapped in local minimizers which are not usually interesting. The NS-ELM algorithm is constructed through an adaptation of the ELM routine\(^1\) (see chapter 2) to the NS. For both the NS and the NS-ELM, it is necessary to evaluate the second order derivative of the objective function which in the case (3.2) is not even differentiable. This fact lead us to work with a smooth (approximate) version of the cost function \(f(u) = \|X^T u\|_1\).

We will also consider another cost function for (3.2). The idea is the same as in previous chapter: construct a log-version of \(\|X^T u\|_1\), which is expected to turn the algorithm more robust to outliers.

Section 3.4 [Qualitative Analysis]. We will illustrate that the concept behind the ELM routine is responsible for the gain in performance of the NS only. In order to justify this claim, we compare the NS-ELM with a RS routine, that is, a routine that tries to escape from local minimizers by choosing a random point in the ambient space and then verifying if it is better than the current one. This experience is similar to the one in section 2.8 of the previous chapter.

\(^{1}\)In this chapter this acronym shall be read as escape from only-local minima.
3.2 Data Model

Problem Formulation. Let \( X = \{ x_i \in \mathbb{R}^n : i = 1, \ldots, m + r \} \) be a finite set of points, s.t.:

\[
x_i \in \bigcup_{j=1}^{d} L_j \text{ for all } i = 1, \ldots, m,
\]

(3.3)

where \( L_1, \ldots, L_d \) represent linear subspaces in which:

\[
\#(L_j \cap X) \geq h_j \quad \text{for all } j = 1, \ldots, d,
\]

(3.4)

Then, given \( X \) and \( h \), find the subspaces \( L_j \) \( \forall j = 1, \ldots, d \).

Assume that the constellation \( X \) lies over the unit-sphere. Let’s assume also that \( d = 1 \), that is, we will seek for one subspace \( L_1 \) in the presence of outliers. This is not a limitation since the recursive approach we will present in the next section can return all of the nontrivial subspaces.

We can represent such a constellation by the columns of the \( n \times (m + r) \) matrix \( X \):

\[
X = \begin{bmatrix}
| & | & \cdots & | \\
| & | & \cdots & | \\
x_1 & x_2 & \cdots & x_{m+r}
\end{bmatrix}
\]

Then, we can write:

\[
X = A_{n \times n} \hat{X}_{n \times (m+r)} \Pi,
\]

(3.5)

in which

\[
\hat{X} = \begin{bmatrix}
* & \cdots & * & \cdots & \cdots & * \\
0 & \cdots & 0 & \cdots & \cdots & * \\
\end{bmatrix} \uparrow n_1,
\]

\( A = [A_1 A_2] \) represents a full rank matrix with \( L_1 = \text{span}\{A_1\} \), \( \Pi \) represents a permutation matrix and \( * \) in \( \hat{X} \) represents entries not necessarily zero.

The expression (3.5) is just to help us to legitimate (3.1) as a model for the subspace segmentation problem.

We can segment \( L_1 \) by searching for its orthogonal complement. From (3.5) we can seek for special
vectors respecting the following configuration:

\[
v = (A^T)^{-1} \begin{bmatrix}
0 \\
\vdots \\
0 \\
* \\
\vdots \\
*
\end{bmatrix}
\]

(3.6)

so that we have:

\[
X^T v = \Pi^T \hat{X}^T A^T (A^T)^{-1} \begin{bmatrix}
0 \\
\vdots \\
0 \\
* \\
\vdots \\
*
\end{bmatrix} = \Pi^T \begin{bmatrix}
0 \\
\vdots \\
0 \\
* \\
\vdots \\
*
\end{bmatrix}
\]

(3.7)

Thus, the zeros in the right hand side of the last equality in (3.7) point out those points of \(X\), lying in \(\text{span}\{v\}^\perp\) which is a \((n - 1)\)-linear subspace. From equations (3.5) and (3.6) we can conclude that \(\mathcal{L}_1 \subset \text{span}\{v\}^\perp\). Hence, finding such vectors is thus equivalent to solve:

\[
\min \|X^T u\|_0 \quad \|u\| = 1
\]

(3.8)

The problem (3.8) is hard since it is combinatorial with such an objective function. A good heuristic to replace the \(l_0\)-norm is the \(l_1\)-norm (commonly used in these cases), resulting in the model:

\[
\min \|X^T u\|_1 \quad \|u\| = 1
\]

(3.9)

The \(l_1\)-norm is a good heuristic in the sense that it is the best convex envelop for the \(l_0\)-norm within the domain \(B_\infty(0, 1)\), where we have defined \(B_\infty(a, r) = \{x \in \mathbb{R}^n : \|x - a\|_\infty \leq r\}\). That is, for any convex underestimator \(f : B_\infty(0, 1) \to \mathbb{R}\), we have that:

\[
f(x) \leq \|x\|_1 \leq \|x\|_0 \quad \forall x \in B_\infty(0, 1)
\]

(3.10)

which holds in our case, since \(X^T u \in B_\infty(0, 1) \forall u : \|u\| = 1\). For completeness we provide the proof.
of this claim in appendix A.1.8. It is worth to mention also that the error $||X^T u||_0 - ||X^T u||_1||$, in our case, is rather bounded.

Once we found a $(n-1)$-subspace that contains $L_1$, we have to work within $\text{span} \{u\}^\perp$ in order to find a $(n-2)$-subspace and so on, up to find $L_1$. In other words, we proceed recursively with this reduction of dimensionality until an activated subspace cannot be found anymore. The recursive strategy is better explained in the next section.

### 3.3 Algorithm

The problem (3.9) is not convex, because the restriction set is the unit-sphere. Obviously, we cannot relax such a set (taking its convex hull), since in this case we would always obtain $u^* = 0$ as solution. We will see some approaches for the treatment of this problem. But before proceed, we will present the recursive strategy employed in the algorithms.

#### 3.3.1 Recursive Strategy

By solving (3.9), and if it succeeds, we obtain a solution $u^*$ that is orthogonal to a $(n-1)$-dimensional hyperplane containing activated subspaces. In order to achieve such subspaces (with arbitrary dimensions) it is necessary to reduce the dimension of the ambient space following a recursive strategy as explained in what follows.

As an example, assume we have to identify two straight lines in $\mathbb{R}^3$ as depicted in Fig. 3.1.

Denote the data matrix as $X_1$. The algorithm would start by solving:

$$
\min \|X_1^T u\|_0, \quad \|u\| = 1
$$

(3.11)

and if it succeeds it will find the solution $u^*$ s.t. $\text{span}\{u^*\} = L_1^\perp$. This corresponds to a first level of recursion.

Up to here, we have a solution pointing out a subspace $L_1$ that eventually contains more activated subspaces (in this case it contains two straight lines). Then, we could construct a new data matrix $X_2$ as follows: i. Evaluate the $(n-1)$-truncated SVD of the matrix $X_2$ consisting of those points of $X_1$ lying in $L_1$, yielding $X_2 = U\Sigma V^T$, ii. Define $X_2 = V^T$. 

48
Figure 3.1: Identifying two straight lines in $\mathbb{R}^3$. First, one identifies $L_1$, and then one identifies $L_{11}$ and $L_{12}$ inside $L_1$.

Note that if we try to solve the problem without reducing the dimensionality of the ambient space:

$$\min \|X_{I}^T u\|_0, \quad \|u\| = 1 \tag{3.12}$$

then, the global solution would be the same as before: $u^*$ s.t. $\text{span}\{u^*\} = L_1^\perp$.

Thus, we can transport the problem of identifying $L_{11}$ and $L_{12}$ in $\mathbb{R}^3$ to a subspace of dimension $\dim L_1 = 2$.

We should solve:

$$\min \|X_{2}^T u\|_0, \quad \|u\| = 1 \tag{3.13}$$

where $u$ is now a 2-dimensional vector and $X_2 := V^\top$. This corresponds to the second level of recursion.

Hence, through the possible solutions in (3.13) we can obtain the set of points lying in one of the straight lines $L_{11}$ or $L_{12}$. These points are labeled by a set of indexes $I$ s.t. $\langle \hat{x}_i, u^* \rangle = 0 \ \forall i \in I$.

Once we achieve a straight line (suppose the red one in Fig. 3.2), we cannot go deeper in the recursion, then we try to peel all possible subspaces within this level of recursion (second level so far).
That is, we can construct a matrix $X_2^c$ corresponding to those points of $X_2$ that do not lie over the red line. We can solve:

$$\min \|X_2^c u\|_0, \quad \|u\| = 1$$

in order to find the green straight line. By finding this subspaces in $\mathbb{R}^2$ we can identify the activated subspaces in $\mathbb{R}^3$ through the map $U \Sigma (\cdot)$.

To conclude, we can proceed recursively with this strategy up to find all activated subspaces. This recursion stops when there is no more subspaces with more than $h$ points (see Fig. 3.3).

Figure 3.2: $U_1 \sigma$ represents a linear application provided by the SVD of $X_1$. Thus, identifying the red and green lines in (b), we are also identifying $L_{11}$ and $L_{12}$.

Figure 3.3: Recursion tree. The sons represent subspaces that span their respective fathers. The numbers, within the red balls on the edges, represent the order in which the recursion is executed.
3.3.2 Optimization over the sphere: NS algorithm

We can tackle the program (3.9) through a minimization algorithm over the unit-sphere.

The NS needs to evaluate the second order derivative of the objective function, which in this case is not even differentiable\footnote{We could try to treat the problem through a second order analysis of nonsmooth convex functions, however this is not in the scope of this thesis. For more details, see [7].}. One possible way to bypass such a problem is smoothing the function \( f(u) = \|X^T u\|_1 \) turning it differentiable, but as near as possible from the nonsmooth version:

\[
\min_{\|u\| = 1} f_{sq} := \sum_{i=1}^{m} \left( x_i^T u \right)^2 + \epsilon
\]  

(3.15)

Roughly speaking, the NS intrinsically builds the Newton’s direction inside the tangent plane at a given point. From this direction, the algorithm shoots a geodesic (through an exponential map) over the sphere as we can see in Fig. 3.4.

\[\]  

Figure 3.4: Illustration of the intrinsic Newton’s direction. In our case, \( M \) is the unit-sphere, \( T_p M \) is the tangent plane of \( M \) at \( p \), \( v_p \) is the Newton’s direction (intrinsically computed in the tangent plane) and \( \text{Exp} \ v_p \) represents the geodesic shotted in the \( v_p \) direction from \( p \).

It is well-known that Newton’s algorithm is quadratically convergent (under appropriate conditions) only in a neighborhood of a minimizer. Since the problem we are dealing with has many local minimizers, we shall initialize the algorithm from many different points and select the best minimizer. See table 3.1.

It is not clear right from the start, how many points should be used to initialize the NS. The performance of the algorithm will in fact depend on it. We will initialize the NS with 60 points. First, because we decided to choose the same number for all of the computer simulations in this chapter. Second, because with this number of initializers, the algorithms spends between 30 seconds up to two minutes per Monte-Carlo in average and apparently their reliability starts to saturate, that is, no more gain in performance is significantly provided by increasing the number of initializations.
In: \( X, \text{INI} \)

**step 1**  Generate INI points over one of the hemispheres of the unit-sphere.

**step 2**  Solve the program:

\[
    u^* \in \min_{\|u\| = 1} f(u) \tag{3.16}
\]

for each of the INI initial points generated in step 1.

**step 3**  Choose the best minimizer among the INI previous solutions.

Out: \( u^*_{\text{best}} \).

Table 3.1: Strategy using NS. The parameter INI represents the number of initializers.

Next, we will show some results from the application of this algorithm with cost function \( f_{\text{sq}} \), initialized by 60 points over one of the hemispheres of the sphere (the objective function is symmetric \( f(x) = f(-x) \)). The other configuration parameters are: \( n = 10, n_1 = \text{dim} \mathcal{L}_1 = 4 \) and \( n_2 = \text{dim} \mathcal{L}_2 = 4 \). Four thousand Monte-Carlos have been performed in order to compute each reliability in the tables of results that follows. These will be, indeed, the conditions for all simulations that follows in this chapter.

Just as a preliminary note, almost all the terminologies within the tables will have the same significance as those in chapter 2 (see section 2.7). We will denote by NS.sq, the routine NS with cost function \( f_{\text{sq}} \). Furthermore, the symbol Rel.NS.sq in the head of the tables of results means the reliability of the NS routine with objective function \( f_{\text{sq}} \).

**• NS-only with cost function \( f_{\text{sq}} \): one subspace and two oblique subspaces.**

<table>
<thead>
<tr>
<th>One Subspace</th>
<th>Outliers [%]</th>
<th>Elements in ( \mathcal{L}_1 ) [%]</th>
<th>Rel.NS.sq [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>30</td>
<td>70</td>
<td>99.28</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>50</td>
<td>95.43</td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>40</td>
<td>81.6</td>
</tr>
<tr>
<td></td>
<td>70</td>
<td>30</td>
<td>55.87</td>
</tr>
<tr>
<td></td>
<td>80</td>
<td>20</td>
<td>26.62</td>
</tr>
</tbody>
</table>

Table 3.2: NS-only. One 4-dimensional subspace. The cost function used was \( f_{\text{sq}} \).

As we can see, the algorithm with this objective function is still inefficient with outliers and mainly when the number of activated subspaces is greater than one. One possible way to turn it more robust...
Two Oblique Subspaces

<table>
<thead>
<tr>
<th>Outliers[%]</th>
<th>Elements in $\mathcal{L}_1$[%]</th>
<th>Elements in $\mathcal{L}_2$[%]</th>
<th>Rel. NS.sq[%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>50</td>
<td>50</td>
<td>91.9</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>40</td>
<td>71.13</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
<td>35</td>
<td>57.93</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>25</td>
<td>31.15</td>
</tr>
</tbody>
</table>

Table 3.3: NS-only. Two 4-dimensional oblique subspaces. The cost function used was $f_{sq}$.

is replace the cost function $f_{sq}$ (as done in the previous chapter) by rather $f_{logns}$:

$$\min_{||u|| = 1} f_{logns} := \sum_{i=1}^{m} \log((x_i^T u)^2 + \epsilon)$$

(3.17)

in order to give greater weights to zero entries of the vector $X^T u$. We denote by NS.log the NS routine with $f_{logns}$ cost function.

We present some computer simulation results with this cost function. We also compare it with the performance of the NS.sq.

• **NS.log vs NS.sq: One subspace and two oblique subspaces.**

<table>
<thead>
<tr>
<th>One Subspace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outliers[%]</td>
</tr>
<tr>
<td>--------------</td>
</tr>
<tr>
<td>30</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>60</td>
</tr>
<tr>
<td>70</td>
</tr>
<tr>
<td>80</td>
</tr>
</tbody>
</table>

Table 3.4: Comparison: NS.log vs NS.sq. One 4-dimensional subspace.

<table>
<thead>
<tr>
<th>Two Oblique Subspaces</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outliers[%]</td>
</tr>
<tr>
<td>--------------</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>30</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>60</td>
</tr>
</tbody>
</table>

Table 3.5: Comparison: NS.log vs NS.sq. Two 4-dimensional oblique subspaces.
We can see a significative improvement of the reliability in the oblique case. Nevertheless these are still modest results when compared with those obtained from the MM-ELM routine in the first chapter.

### 3.3.3 NS-ELM: NS Coupled with ELM

In fact, the algorithm presented in the previous section is not much robust to outliers. Seemingly to the previous chapter, there are local solutions that represent no interesting minimizers. Then, we can adapt the ELM routine to the NS in order to escape from local minimizers towards the interesting ones. The paradigm is essentially the same as faced in chapter 2: we have got a routine that achieves a local minimizer (NS) and we use the ELM to try to escape from these proposed minimizers, re-initializing the NS (if it succeeds) at a better point.

The idea for the ELM routine here is the same as in chapter 2: given $u^*$ a local minimizer, we can sort the points of $X$ in increasing order of their values $|⟨x_i, u^*⟩|$. Then, we can construct a vector $u_{new} \in \text{span}\{x_1, \ldots, x_k\}^\perp$. This new point works as a re-initialization for the NS if $f_{\text{obj}}(x_{\text{new}}) < f_{\text{obj}}(x)$. If this latter inequality does not hold, then we can construct $x_{\text{new}} \in \text{span}\{x_1, \ldots, x_{k+1}\}^\perp$ and so on, for the same reason presented in chapter 2. The NS-ELM is presented in table 3.6.

<table>
<thead>
<tr>
<th>In:</th>
<th>$X$, INI</th>
</tr>
</thead>
<tbody>
<tr>
<td>step 1</td>
<td>Generates INI points to initialize the algorithm: $Y_{\text{INI}} = {y \in S^{n-1} : y^1 &gt; 0}$, $k \leftarrow 0$.</td>
</tr>
<tr>
<td>step 2</td>
<td>If $k = \text{INI}$, Finish. Else, Do $k \leftarrow k + 1$, $u \leftarrow \text{NS}(y_k, X)$, $j \leftarrow 2$, $r \leftarrow 2$.</td>
</tr>
<tr>
<td>step 3</td>
<td>Sort the data set $X = {x_i : i = 1, \ldots, m}$ in increasing order of its values $</td>
</tr>
<tr>
<td>step 4</td>
<td>If $j &lt; m$, then construct $u^* \in \text{span}{x_1, \ldots, x_j}^\perp$, else jump to 6.</td>
</tr>
<tr>
<td>step 5</td>
<td>If $F_{\text{obj}}(u) &lt; F_{\text{obj}}(u^*)$ then do $j \leftarrow j + 1$ and jump to 4. Else jump to 2.</td>
</tr>
<tr>
<td>step 6</td>
<td>Sort the set $X$ in decreasing order of its values $</td>
</tr>
<tr>
<td>step 7</td>
<td>If $r &lt; m$ then construct $u^* \in \text{span}{x_1, \ldots, x_r}^\perp$, else jump to 2.</td>
</tr>
<tr>
<td>step 8</td>
<td>If $F_{\text{obj}}(u) &lt; F_{\text{obj}}(u^*)$ then do $r \leftarrow r + 1$ and jump to 7. Else jump to 2.</td>
</tr>
<tr>
<td>Out:</td>
<td>$u^*$ is the minimizer.</td>
</tr>
</tbody>
</table>

Table 3.6: NS-ELM Algorithm. The parameter INI represents the number of points for initialization.

Next, we present some relevant computer simulation results.
• **NS-ELM.log vs NS.log:** One subspace and two oblique subspaces.

<table>
<thead>
<tr>
<th>Outliers [%]</th>
<th>Elements in $L_1$ [%]</th>
<th>Elements in $L_2$ [%]</th>
<th>Rel.NS-ELM.log [%]</th>
<th>Rel.NS.log [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>70</td>
<td>100</td>
<td>100</td>
<td>95.17</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>100</td>
<td>100</td>
<td>93.03</td>
</tr>
<tr>
<td>60</td>
<td>40</td>
<td>100</td>
<td>100</td>
<td>90.65</td>
</tr>
<tr>
<td>70</td>
<td>30</td>
<td>97.05</td>
<td>86.2</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>20</td>
<td>89.4</td>
<td>67.93</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>10</td>
<td>18.75</td>
<td></td>
<td>*</td>
</tr>
</tbody>
</table>

Table 3.7: Comparison: NS-ELM vs NS. One 4-dimensional subspace.

<table>
<thead>
<tr>
<th>Outliers [%]</th>
<th>Elements in $L_1$ [%]</th>
<th>Elements in $L_2$ [%]</th>
<th>Rel.NS-ELM.log [%]</th>
<th>Rel.NS.log [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>50</td>
<td>50</td>
<td>100</td>
<td>94.57</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>40</td>
<td>100</td>
<td>91.25</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
<td>35</td>
<td>100</td>
<td>89</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>25</td>
<td>99.1</td>
<td>77.18</td>
</tr>
<tr>
<td>60</td>
<td>20</td>
<td>20</td>
<td>88</td>
<td>65.62</td>
</tr>
<tr>
<td>70</td>
<td>15</td>
<td>15</td>
<td>70.5</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.8: Comparison: NS-ELM vs NS. Two 4-dimensional oblique subspaces.

• **NS-ELM.log:** two oblique subspaces with nontrivial intersections.

<table>
<thead>
<tr>
<th>Outliers [%]</th>
<th>Elements in $L_1$ [%]</th>
<th>Elements in $L_2$ [%]</th>
<th>Rel.NS-ELM.log [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>50</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>40</td>
<td>100</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
<td>35</td>
<td>100</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>25</td>
<td>93.4</td>
</tr>
<tr>
<td>60</td>
<td>20</td>
<td>20</td>
<td>68.75</td>
</tr>
<tr>
<td>70</td>
<td>15</td>
<td>15</td>
<td>23.3</td>
</tr>
</tbody>
</table>

Table 3.9: NS-ELM.log. Two 4-dimensional oblique subspaces subject to $\dim L_1 \cap L_2 = 1$
• **NS-ELM.sq vs NS.sq:** one subspace and two oblique subspaces.

<table>
<thead>
<tr>
<th>Outliers[%]</th>
<th>Elements in ( L_1 )[%]</th>
<th>Rel.NS-ELM.sq[%]</th>
<th>Rel.NS.sq [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>80</td>
<td>99.85</td>
<td>99.28</td>
</tr>
<tr>
<td>30</td>
<td>70</td>
<td>96.48</td>
<td>95.43</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td>85.12</td>
<td>81.6</td>
</tr>
<tr>
<td>60</td>
<td>40</td>
<td>57.4</td>
<td>55.87</td>
</tr>
<tr>
<td>70</td>
<td>20</td>
<td>*</td>
<td>*</td>
</tr>
</tbody>
</table>

Table 3.10: Comparison: NS-ELM.sq vs NS.sq. One 4-dimensional subspace.

<table>
<thead>
<tr>
<th>Outliers[%]</th>
<th>Elements in ( L_1 )[%]</th>
<th>Elements in ( L_2 )[%]</th>
<th>Rel.NS-ELM.sq[%]</th>
<th>Rel.NS.sq [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>50</td>
<td>50</td>
<td>93.55</td>
<td>91.9</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>40</td>
<td>74.92</td>
<td>71.13</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
<td>35</td>
<td>60</td>
<td>57.93</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>25</td>
<td>34.68</td>
<td>31.15</td>
</tr>
</tbody>
</table>

Table 3.11: Comparison: NS-ELM.sq vs NS.sq. Two 4-dimensional oblique subspaces.

We also present in table 3.12 the comparison between the NS-ELM and the MM-ELM for the oblique case with cost functions \( f_{logs} \) and \( f_{log} \), respectively.

• **NS-ELM.log vs MM-ELM.log:** two oblique subspaces.

<table>
<thead>
<tr>
<th>Outliers[%]</th>
<th>Elements in ( L_1 )[%]</th>
<th>Elements in ( L_2 )[%]</th>
<th>Rel.NS-ELM.log[%]</th>
<th>Rel.MM-ELM.log</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>50</td>
<td>50</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>20</td>
<td>40</td>
<td>40</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>30</td>
<td>35</td>
<td>35</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>50</td>
<td>25</td>
<td>25</td>
<td>99.1</td>
<td>99.63</td>
</tr>
<tr>
<td>60</td>
<td>20</td>
<td>20</td>
<td>88</td>
<td>97.98</td>
</tr>
<tr>
<td>70</td>
<td>15</td>
<td>15</td>
<td>70.5</td>
<td>83.45</td>
</tr>
</tbody>
</table>

Table 3.12: Comparison: NS-ELM.log vs MM-ELM.log. Two 4-dimensional oblique subspaces.

### 3.4 Qualitative Analysis

We could ask whether the gain in performance of the algorithm NS-ELM comes out from the heuristic associated to the ELM routine or if it is due to the number of trials that the ELM eventually try before escaping from a minimizer. Thus, we compare the routine NS-ELM with a RS routine. The idea is the same as approached in section 2.8.

We will perform the following experience: we will run for 11000 times the ELM routine from a local
maximizer. Thus, we will count in each Monte Carlo, how many trials the routine needed to escape. The RS will try up to 200 times per Monte Carlo. Then, we can construct an histogram with "the number of escaping's vs number of the trial". The results are presented in Fig. 3.5.

![Figure 3.5: Illustration of the curves: rights vs number of trial.](image)

This plots have been made for the oblique case, with 50% of outliers and with $f_{\log}$ as cost function. By the Fig. 3.5 we can conclude that more than a half of the times the ELM routine could improve the cost function at first trial. The RS routine presents a noisiness plot, as expected. This fact suggests that this heuristic plays an important role in the gain of performance of the algorithm.

We can compare, as well, the reliability of both algorithms in this case:

$$\text{Reliability}_{ELM} = 99.1\%$$
$$\text{Reliability}_{RS} = 78.35\%$$

(3.18)

The reliability of the NS.log routine in the oblique case, with 50% of outliers is 77.18%. Thus, we can see that the RS routine returned no significant gain.
3.5 Conclusion

We adopted in this chapter a segmentation strategy that needs no prior information about the dimensions and number of subspaces. The NS algorithm has been used with different cost functions. Since such an algorithm use to be trapped in local minimizers, the ELM routine has been adapted to this case. As we could see in the computer simulation results and even in the qualitative analysis section, such a routine brings a significant gain to the performance of the NS.

The drawback here is that such a method showed itself to be sensitive to noise and computationally more inefficient when compared with the MM-ELM. In fact, the MM-ELM is just initialized at one point whereas the NS-ELM is initialized at 60 points.
Chapter 4

Subspace Segmentation - Linear Program

4.1 Chapter Summary

Section 4.2 [Purpose]. We state the optimality condition supporting the theory developed in this chapter.

Section 4.3 [Example Without Outliers]. Our purpose in this section is to show that, in the absence of outliers in the data matrix $Y = [y_1 \cdots y_m]^\top$, the solutions of:

$$\min_{c^\top u = 1} \|Yu\|_1$$

are included in the solution set of:

$$\min_{\|u\| = 1} \|Yu\|_0$$

for almost all choices of $c$. Thus, the combinatorial problem (4.2) can be solved by a single linear program 4.1. Moreover, any solution of (4.1) codifies an activated subspace. Varying $c$, all of the interesting solutions can be achieved.

We will present a pictorial proof (through an example) of this fact rather than a strict mathematical proof. We will be looking to the graph of $\partial \|Yu\|_1$ in order to extract minimality conditions.

Section 4.4 [Example: Outliers Effect]. In this section we interpret geometrically (through convex analysis) the effect of adding outliers in the system.
4.2 Purpose

From convex analysis, we know that for a convex program:

\[
\min_{x \in C} f(x),
\]

(4.3)

with \( f \in \text{Conv} \mathbb{R}^n \) and \( C \) a closed convex set, we have that (see \[7\], chap. 7):

\[
0 \in \partial f(x^*) + N_C(x^*)
\]

(4.4)

in which \( x^* \) represents a solution of (4.3). Condition 4.4 can be rewritten as:

\[
(-\partial f(x^*)) \cap N_C(x^*) \neq \emptyset
\]

(4.5)

The condition (4.5) allows us to decide whether a point is a minimizer by characterizing the first order elements \( \partial f \) and \( N_C \) of the cost function and constrain set, respectively. This is the condition behind the purpose of this chapter.

4.3 Example Without Outliers

Let \( f(x) = \|Yx\|_1 \), where:

\[
Y = \begin{bmatrix}
1 & 0 \\
1 & 0 \\
1 & 0 \\
1 & 0 \\
0 & 1 \\
0 & 1
\end{bmatrix}
\]

(4.6)

In this case we have to identify two subspaces: \( \mathcal{L}_1 = \text{span}\{e_1\} \) and \( \mathcal{L}_2 = \text{span}\{e_2\} \), where \( e_1 = (1, 0) \) and \( e_2 = (0, 1) \) represent the canonical vectors of \( \mathbb{R}^2 \). There holds \( f(x) = 4|e_1^T x| + 2|e_2^T x| \). So:

\[
\partial f(x) = 4\partial|e_1^T x| + 2\partial|e_2^T x|
\]

(4.7)

Moreover, we have \( \partial|y_i^T x| = \text{sgn}(y_i^T x)y_i \), where:

\[
\text{sgn}(p) = \begin{cases}
1, & \text{if } p > 0 \\
[-1, 1], & \text{if } p = 0 \\
-1, & \text{if } p < 0
\end{cases}
\]

(4.8)
Thus, we can write:

$$\partial f(x) = 4 \text{sgn}(e_1^\top x)e_1 + 2 \text{sgn}(e_2^\top x)e_2$$  \hspace{1cm} (4.9)$$

We should underline that $\partial f$ is rather a multifunction, that is, $\partial f(x)$ is a set for each $x$. For instance, for a point $x_a = \alpha e_2$, with $\alpha > 0$, we have $\partial f(x_a) = 4[-1, 1]e_1 + 2e_2$, which represents the set pictured in Fig. 4.1.

![Figure 4.1: Image of $x_a = \alpha e_2$, for $\alpha > 0$, by $\partial f$.](image)

In Fig. 4.2 we show how $\mathbb{R}^2$ is mapped by $\partial f$. One can easily verify that this graph is equal to the one of the multifunction $-\partial f$.

![Figure 4.2: Illustration of the mapping of subsets of $\mathbb{R}^2$ by $\partial f$ into the dual space. The right figure represents $\text{dom } f$, and the left one represents $\partial f(\mathbb{R}^2)$. The vertex $\partial I$ represents the image of the whole region I in $\text{dom } f$ by $\partial f$ and $\partial e_2$ (straight line segment connecting $\partial IV$ to $\partial I$) represents the image of each point over the ordinates axes.](image)

In this example, we can note that for any straight line $r$ crossing the origin in the dual plane $\partial f$, we have $r \cap \partial f(\mathbb{R}^2 \setminus \{0\}) \neq \emptyset$, that is, $r$ always crosses the boundary of the rectangle depicted in Fig. 4.2.
Since the hyperplane \( \Pi_c = \{ u \in \mathbb{R}^2 : c^\top u = 1 \} \) has a straight line \( r = \text{span} \{ c \} \) as normal cone, then it will be possible to study minimality conditions of:

\[
\min_{c^\top u = 1} \| Y u \|_1,
\]

looking to the intersection \((-\partial f(x)) \cap N_{C}(x) \neq \emptyset \) or \((-\partial f(x)) \cap \text{span} \{c\} \neq \emptyset \) (see Fig. 4.3).

![Diagram](image_url)

Figure 4.3: Figures (a) and (b) illustrate the intersection \((-\partial f) \cap \text{span} \{ c \}\) for two different choices of \( c \). For the choice in (a) we have that the minimizers of (4.10) are those points \( x^* \) s.t. \((-\partial f(x)) \cap \text{span} \{ c \} \neq \emptyset \), that is, will be the points lying in \( \text{span} \{ e_2 \} \). In (b), the minimizers will be the points in \( \text{span} \{ e_1 \} \).

We can see that (for the choices \( c \) represented in 4.3) the solutions of (4.10) represent activated subspaces. In fact, the only problem is when we choose \( c \) such that \( \text{span} \{ c \} \) intersects one of the vertices of the rectangle in the dual space \( \partial f \). In this case, the solution \( x^* \) would be in one of the regions \( I, II, III, IV \) (see Fig. 4.2) which does not represent an interesting solution, since it is not orthogonal to any of those activated subspaces. But if we generate \( c \) randomly (as in a roulette wheel),
the probability that \( r \) crosses one of the vertices is zero, then all the solutions from (4.10) evaluated in this way will be always interesting.

In table 4.1 we summarize the strategy.

<table>
<thead>
<tr>
<th>In:</th>
<th>( Y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 1)</td>
<td>Choose ( c \in \mathbb{R}^n ) randomly.</td>
</tr>
<tr>
<td>step 2 2)</td>
<td>Solve the linear program:</td>
</tr>
</tbody>
</table>
|         | \[
\min_{c^\top x = 1} \|Yx\|_1
\] (4.11) |
| Out:   | \( x^* \) minimizer. |

Table 4.1: Strategy to achieve activated subspaces by solving a linear program.

It is important to note that the example given above for orthogonal subspaces is extensible to the oblique case. In this last case we obtain a general quadrilateral in the dual space instead of a rectangle. The trouble makers are still the vertices. This can also be generalized to \( \mathbb{R}^n \).

We present some results in table 4.2 that confirm (latus sensus) our analysis.

<table>
<thead>
<tr>
<th>Two Oblique Subspaces</th>
<th>Elements in ( \mathcal{L}_1 ) [%]</th>
<th>Elements in ( \mathcal{L}_2 ) [%]</th>
<th>reliability [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>50</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>70</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>80</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>20</td>
<td>100</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.2: Results. The 100% in reliability means that the solutions obtained by solving the problem (4.10) (with an arbitrary choice for \( c \)) was always interesting.

The results in table 4.2 have been obtained by solving (4.10) through the Sedumi toolbox of matlab. We have made 10000 Monte-Carlos and in each simulation a value of \( c \) was chosen arbitrarily. The computer simulation results show that for any choice of \( c \) randomly generated, the minimizer achieved was always interesting.
4.4 Example: Outliers Effect

When there are outliers, the vertices of the rectangle in the dual space are widened as we now illustrate.

Let:

\[ Y = \begin{bmatrix}
1 & 0 \\
1 & 0 \\
1 & 0 \\
1 & 0 \\
0 & 1 \\
0 & 1 \\
1 & 1 \\
\end{bmatrix} \]  

(4.12)

in which \( e = \begin{bmatrix} 1 & 1 \end{bmatrix} \) is clearly an outlier.

In this case we have \( \partial f(x) = 4\text{sgn}(e_1^T x) + 2\text{sgn}(e_2^T x) + \text{sgn}(e^T x) \). The graph of \( \partial f \) is presented in Fig. 4.4.

Figure 4.4: Configuration of the subdifferential \( \partial f \) when one outlier is added.
When \( \text{span}\{c\} \) crosses the "widened vertices", that is, the line segments \( \partial II \partial III \) and \( \partial IV \partial V \), we can say that the associated minimizer of (4.10) is not an interesting point. In fact, such a solution would not be orthogonal to the activated subspaces, but it would be rather orthogonal to the outlier. Choosing \( c \) arbitrarily, we have that the greater the number of outliers, the lower the probability to match interesting minimizers, since it will come up more "widened" zones. Actually, we can even estimate the likelihood of matching an interesting minimizer shooting \( c \) randomly. We have:

\[
p_a = 1 - \frac{2\theta}{360},
\]

We can evaluate the value of \( \theta \) by doing (see Fig. 4.4) \( \theta = 90 - 45 - \arctan(\frac{1}{5}) = 33.69^\circ \), which results in \( p_a = 81.28\% \).

We can proceed with a simulation of this example through the *Sedumi* and we can compare the theoretical and the experimental value of the reliability in matching interesting solutions:

\[
\begin{align*}
\text{Reliability}_{\text{experimental}} &= 80.3\% \\
\text{Reliability}_{\text{theoretical}} &= 81.28\%
\end{align*}
\]

(4.13)

where it has been performed 10000 Monte-Carlos. For each Monte-Carlo an arbitrary unit-vector \( c \) has been chosen.

### 4.5 Conclusion

In this section we illustrated a particular case (when there is no outliers in the data set) where a combinatorial problem (4.2) can be solved by a linear program. We also provided a geometrical interpretation of the problem when an amount of outliers is added to the data set. We performed two experiences that empirically confirms the theory presented.
Chapter 5

Conclusion and Future Approaches

We have presented two algorithms for segmenting arbitrary unions of linear subspaces: the MM-ELM routine in Chapter 2 and the NS-ELM routine in chapter 3. Both approaches differ from most of current works by searching for activated subspaces by solving nicely structured optimization problems. The MM-ELM algorithm showed an interesting performance in dealing with the articulated case and even with real data. The main drawback is that the dimensions and number of subspaces are needed beforehand. In contrast, the NS-ELM algorithm does not need prior information regarding the number or dimensions of the subspaces. Following a recursive strategy it can peel all of the activated subspaces. It presented a reasonable performance in segmenting subspaces in the articulated case. However, this latter algorithm showed to be rather sensitive to noise and is not as computationally efficient as the MM-ELM algorithm in segmenting subspaces. We shall remark that the NS-ELM is initialized from 60 points whereas the MM-ELM is just initialized from one point following a suboptimal approach. Roughly speaking, the MM-ELM can in average segment two subspaces in about 3 seconds whereas the NS-ELM can spend up to 2 minutes in some cases.

We have presented also that the subspace segmentation problem can be partially solved through a linear program. We provided also an alternative geometrical approach of the problem through convex analysis.

As future work, the contraction operation from geometric algebra will be tested as cost function in the MM-ELM algorithm. Also, a richer characterization of the Grassmann manifold will be tried, that is, we will try to explore Riemannian properties of $\text{Proj}_{\alpha}(n_1)$ as a manifold. Some features from the physical problem will be taken into account in order to eliminate some physically non-feasible solutions from the model. A more mathematically rigorous analysis of the latter chapter is ongoing.
Bibliography


Appendix A

Proofs

A.1 Appendix: Proofs

A.1.1 $W$ Is Isometric to $V^\top$

Consider the data matrix $X$ where its points are distributed into $m$ subspaces s.t. $\mathbb{R}^n = L_1 \oplus \ldots \oplus L_d$. We can write $X$ (up to a left permutation) as:

$$X = \hat{P}_{n \times n} \begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_m \end{bmatrix} (A.1)$$

Through the Gram-Schmidt orthogonalization, we can rewrite A.1 as:

$$X = \hat{P} \begin{bmatrix} R_1 \\ R_2 \\ \vdots \\ R_m \end{bmatrix} \begin{bmatrix} Q_1^\top \\ Q_2^\top \\ \vdots \\ Q_m^\top \end{bmatrix} = PQ (A.2)$$

Taking the SVD of $X$, yields $X = U \Sigma V^\top$.

We claim that:

$$V^\top = MQ (A.3)$$

where $M$ is an orthogonal matrix.

PROOF.
We have that:

\[ X = PQ \]  \hspace{1cm} (A.4)

\[ X = U\Sigma V^T \]  \hspace{1cm} (A.5)

Thus, it follows that \( PQ = U\Sigma V^T \), that is:

\[ V^T = \Sigma^{-1}U^TPQ \]  \hspace{1cm} (A.6)

We shall prove that the matrix \( M = \Sigma^{-1}U^TP \) is orthogonal. We have that:

\[ M^TM = P^TU\Sigma^{-2}U^TP \]  \hspace{1cm} (A.7)

Hence, we have to prove that \( P^TU\Sigma^{-2}U^TP = I \). Indeed, note that if \( X = U\Sigma V^T \) then:

\[ XX^T = U\Sigma V^TV\Sigma U^T = U\Sigma^2U^T \]  \hspace{1cm} (A.8)

We can note also that

\[ (XX^T)^{-1} = U\Sigma^{-2}U^T \]  \hspace{1cm} (A.9)

and

\[ XX^T = PQ^TP^T = PP^T \]  \hspace{1cm} (A.10)

In conclusion:

\[ M^TM = P^TU\Sigma^{-2}U^TP = P^T(P^T)^{-1}P^{-1}P = I \]  \hspace{1cm} (A.11)

Note that permuting the columns of \( W \) results in the permutation of the columns of \( V^T \). The isometry \( M \) remains the same.

**A.1.2** \( f_{tr}(\Pi) \in \text{Conv } \mathbb{R}^{n \times n} \)

Let \( f_{tr} : \mathbb{R}^{n \times n} \to \mathbb{R} \) be defined as:

\[ f_{tr}(\Pi) = \sum_i |x_i^T\Pi x_i - \frac{1}{2}| \]  \hspace{1cm} (A.12)

then \( f_{tr} \) is convex.

**PROOF.** We know that the function absolute value \( |\cdot| : \mathbb{R} \to \mathbb{R} \) is convex. We know also that given
\(g\), an affine function, and \(h\), a convex function, s.t. \(f = h \circ g\), then \(f\) is convex. Moreover, the sum of convex functions is also convex.

**A.1.3** \(f_{\log}(\Pi) \in \text{Conv} \mathbb{R}^{n \times n}\)

\[f_{\log}(\Pi) = -\log((x_i^T \Pi x_i)(1 - x_i^T \Pi x_i))\] is convex.

**PROOF.**

\[f_{\log}(\Pi) = -\log((x_i^T \Pi x_i)(1 - x_i^T \Pi x_i)) = -\log(x_i^T \Pi x_i) - \log(1 - x_i^T \Pi x_i)\] (A.13)

Thus, by the same reasons stated in the latter proof we conclude that \(f_{\log}\) is convex.

**A.1.4 Minimization of the Frobenius Norm**

Given the data matrix \(X \in \mathbb{R}^{n \times m}\) and considering its SVD \(X = U \Sigma V^T\), the global solution of:

\[
\text{arg min}_{\Pi \in \text{Proj}_{n_1}(n)} \|X - \Pi X\|^2 \quad (A.14)
\]

is the projector onto the \(n_1\)-dimensional dominant eigenspace of \(X\), that is, it is given by \(\Pi = U_{n \times n_1} U_{n \times n_1}^T\), in which the matrix \(U_{n \times n_1}\) is constructed from \(U\) by extracting its \(n_1\) first columns.

**PROOF.**

\[
\|X - \Pi X\|^2 = \|\Pi^\perp X\|^2 = \text{tr}((\Pi^\perp X)(\Pi^\perp X)) = \text{tr}(X^T \Pi^\perp \Pi^\perp X) = \text{tr}(X^T \Pi^\perp X) \quad (A.15)
\]

Since the EVD of \(\Pi\) is given by \(\Pi = QQ^T\), in which \(Q\) is a \(n \times n_1\) Stiefel matrix, we can rewrite (A.14) as:

\[
\text{arg min}_{Q^T Q = I_{n_1 \times n_1}} \text{tr}(Q^T X X^T Q), \quad (A.16)
\]

By the theorem 2.4.1 in chapter 2, the solution \(Q^*\) of (A.16) is the matrix consisting of the \(n_1\) eigenvectors associated to the \(n_1\) highest eigenvalues of \(XX^T\). But since \(X = U \Sigma V^T\), it follows that:

\[
XX^T = U \Sigma V^T V S U^T = US^2 U^T \quad (A.17)
\]

The latter term in (A.17) represents the EVD of the symmetric matrix \(XX^T\). Thus, the matrix \(U_{n \times n_1}\)
is the solution of (A.16) and therefore, \( \Pi^* = U_{n \times n} U_{n \times n}^T \) represents the solution of (A.14).

### A.1.5 \( \text{Proj}_n(n_1) \) is compact

Given \( \text{Proj}_n(n_1) = \{ \Pi \in S_{n \times n} : \Pi^2 = \Pi, \text{tr}(\Pi) = n_1 \} \) the set of projectors of dimension \( n \times n \) and rank \( n_1 \), we have that \( \text{Proj}_n(n_1) \) is compact.

**PROOF.**

- \( \text{Proj}_n(n_1) \) is a bounded Set:

  Let \( \Pi \in \text{Proj}_n(n_1) \), then \( \text{tr}(\Pi^2) = \text{tr}(\Pi) = n_1 \) and in light of the norm \( ||X|| = \sqrt{\text{tr}(X^T X)} \), it follows that: \( \text{Proj}_n(n_1) \subset B(0, r) \), where we have defined \( B(A, r) = \{ X \in \mathbb{R}^{n \times n} : ||X - A|| < r \} \).

- \( \text{Proj}_n(n_1) \) is a Closed Set:

  \( \text{Proj}_n(n_1) \) is the intersection of closed sets as we can see:

  Define \( g(\Pi) = \Pi^2 - \Pi \), then:

  \( P_1 = \{ \Pi \in S_{n \times n} : \Pi^2 = \Pi \} = g^{-1}(0) \), in which \( g^{-1}(0) \) is the pullback of \( \{0\} \) by \( g \). Of course, \( g \) is continuous and \( \{0\} \) is a closed set\(^1\).

  \( P_2 = \{ \Pi \in S_{n \times n} : \text{tr}(\Pi) = n_1 \} = \text{tr}^{-1}(n_1) \), in which \( \text{tr}^{-1}(n_1) \) is the pullback of \( \{n_1\} \) by \( \text{tr} \) which is continuous and \( \{n_1\} \) is closed.

  \( P_3 = \{ \Pi \in \mathbb{R}^{n \times n} \} \) is a vector space.

### A.1.6 Proof of Theorem 2.3.1

Let \( Q = [Q_1, Q_2] \), and \( C = \text{co Proj}_n(n_1) \), then:

\[
\min_{Q \in O(n)} \sum_{i=1}^{k} \min \{||Q_1^T x_i||^2, ||Q_2^T x_i||^2\} \Leftrightarrow \max_{\Pi \in C} \sum_{i=1}^{k} \left| (x_i^T \Pi x_i) - \frac{1}{2} \right|^2 \quad (A.18)
\]

in which \( \text{co Proj}_n(n_1) \) is the convex hull of the projectors set of order \( n \) and rank \( n_1 \).

**PROOF.** We have that:

\[
||Q_1^T x_k||^2 = x_k^T Q_1 Q_1^T x_k \quad (A.19)
\]

\(^1\mathbb{R}^n \) with the standard topology is Hausdorff [18].
And since:

\[ QQ^\top = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} \]

\[ = Q_1Q_1^\top + Q_2Q_2^\top = I \] (A.20)

it follows that:

\[ ||Q_2^\top x_k||^2 = x_k^\top Q_2Q_2^\top x_k = x_k^\top (I - Q_1Q_1^\top) x_k = 1 - x_k^\top Q_1Q_1^\top x_k \] (A.21)

Then:

\[ \arg\min_{Q_1Q_1^\top \in \text{Proj}_n(n_1)}\sum_{i=1}^{k}\min\{ (x_i^\top Q_1Q_1^\top x_i), 1 - (x_i^\top Q_1Q_1^\top x_i) \} = \]

\[ = \arg\min_{Q_1Q_1^\top \in \text{Proj}_n(n_1)}\sum_{i=1}^{k}\left( \frac{1}{2} - |(x_i^\top Q_1Q_1^\top x_i) - \frac{1}{2}| \right) = \]

\[ = \arg\max_{Q_1Q_1^\top \in \text{Proj}_n(n_1)}\sum_{i=1}^{k}|(x_i^\top Q_1Q_1^\top x_i) - \frac{1}{2}|, \] (A.24)

At this point we would rather relax the restriction set \( \text{Proj}_n(n_1) \) to its convex hull, in order to work with a convex restriction set. Since:

\[ \text{Proj}_n(n_1) = \text{ext}(\text{co} \text{Proj}_n(n_1)), \] (A.25)

the below theorem legitimates such relaxation:

**Theorem A.1.1.** Let \( C \) be a closed and convex set and let \( f \) be a convex function, then:

\[ \max_{x \in C} f(x) \iff \max_{x \in \text{ext}(C)} f(x), \] (A.26)

Let’s then prove equality A.25.

**Corollary A.1.2.** \( \text{Proj}_n(n_1) = \text{ext}(\text{co} \text{Proj}_n(n_1)) \)

**PROOF:**

Let \( \Pi, \Pi_1, \Pi_2, \ldots, \Pi_n' \in \text{Proj}_n(n_1) \) and suppose that \( \Pi = \sum_{i=1}^{n'} \alpha_i \Pi_i \) in which \( \sum_{i=1}^{n'} \alpha_i = 1, \alpha_i >\)
\[ \forall i. \text{ It follows that:} \]
\[ \Pi^2 = \left( \sum_{i=1}^{n'} \alpha_i \Pi_i \right) \left( \sum_{j=1}^{n'} \alpha_j \Pi_j \right) = \sum_{i,j} \alpha_i \alpha_j \Pi_i \Pi_j \]
(A.27)

But \( \Pi^2 = \Pi \), since \( \Pi \in \text{Proj}_n(n_1) \). Then:
\[ \sum_{i,j} \alpha_i \alpha_j \Pi_i \Pi_j = \Pi = \sum_{k} \alpha_k \Pi_k \]
(A.28)

Applying the \text{trace} function in both sides of equation (A.28), we obtain:
\[ \sum_{i,j} \alpha_i \alpha_j \text{tr}(\Pi_i \Pi_j) = n_1 \]
(A.29)

But by the Cauchy-Shwarz inequality, we have:
\[ \sum_{i,j} \alpha_i \alpha_j \text{tr}(\Pi_i \Pi_j) \leq \sum_{i,j} \alpha_i \alpha_j n_1 = n_1 \left( \sum_i \alpha_i \right)^2 = n_1 \]
(A.30)

The equality holds if, and only if, \( \Pi_i \) is parallel to \( \Pi_j \) \( \forall i,j \) \( \Rightarrow \Pi_1 = \Pi_2 = \ldots = \Pi_{n'} \), because \( \text{tr}(\Pi_i) = n_1 \forall i \). That is, the equality holds only if the projectors are parallels, however as long as all of them have the same norm, this is equivalent to say that the equality holds when they are equal.

In conclusion, no projector can be constructed by the convex combination of other projectors.

\[ \supseteq \]

Let \( \Pi \in \text{co Proj}_{n_1}(n_1) = \{ \Pi \in S_{n \times n} : 0 \leq \Pi \leq 1, \text{tr}(\Pi) = n_1 \} \) and assuming that:
\[ \exists \lambda_i(\Pi) < 1 \]
(A.31)

and moreover:
\[ \exists \epsilon > 0 : \lambda_i(\Pi) + \epsilon < 1 , \lambda_i(\Pi) - \epsilon > 0 \]
(A.32)

\(^2\)The Cauchy-Schwarz inequality holds for all dot product with a norm defined as: \( |x| = \sqrt{\langle x, x \rangle} \). In fact, \( \text{tr}(A^\top B) \) holds the axioms of the dot product and defining \( |A| = \sqrt{\text{tr}(A^\top A)} \), for \( \Pi_1 \) and \( \Pi_2 \) \( \in \text{Proj}_n(n_1) \), it follows that: \( \text{tr}(\Pi_1 \Pi_2) \leq \sqrt{\text{tr}(\Pi_1^2)} \sqrt{\text{tr}(\Pi_2^2)} = n_1 \).
We can construct two matrices given by:

\[
\Lambda_1 = \begin{bmatrix}
\lambda_1 & 0 & \ldots & \ldots & 0 \\
0 & \lambda_2 & \ddots & \cdots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \lambda_i + \epsilon \ddots & \ddots & \vdots \\
0 & \ldots & \ldots & 0 & \lambda_n
\end{bmatrix}
\]

\[
\Lambda_2 = \begin{bmatrix}
\lambda_1 & 0 & \ldots & \ldots & 0 \\
0 & \lambda_2 & \ddots & \cdots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \lambda_i - \epsilon \ddots & \ddots & \vdots \\
0 & \ldots & \ldots & 0 & \lambda_n
\end{bmatrix}
\]

And then we have:

\[
\Pi_1 = S\Lambda_1 S^T, \quad \Pi_2 = S\Lambda_2 S^T \in \text{co Proj}_n(n_1)
\]  \hfill (A.33)

in which we assumed \( S \) as a modal matrix\(^3\) of \( P \). Then, it is easy to verify that:

\[
\Pi = \frac{\Pi_1 + \Pi_2}{2}
\]  \hfill (A.34)

To sum up, we have proved that:

\[
\Pi \in \text{co Proj}_n(n_1) \setminus \text{Proj}_n(n_1) \Rightarrow \exists \Pi_1, \Pi_2 \in \text{co Proj}_n(n_1), \alpha \in [0,1] : \Pi = \alpha \Pi_1 + (1 - \alpha) \Pi_2,
\]  \hfill (A.35)

that is, if \( \Pi \) belongs to the convex hull and it is not a projector, then it admits a convex combination of elements in this set, which is equivalent to say that (by the counter-positive\(^4\)) if \( \Pi \) could not be represented as a convex combination of others elements of the set (it would be an extreme point of this one), then \( \Pi \) is a projector.

If we consider \( C = \text{co Proj}_n(n_1) \), then we have:

\[
\max_{x \in \text{co Proj}_n(n_1)} f(x) \Leftrightarrow \max_{x \in \text{Proj}_n(n_1)} f(x),
\]  \hfill (A.36)

---

\(^3\) Matrix in which the columns represent the eigenvectors

\(^4\) \( (a \Rightarrow b) \Leftrightarrow (b \Rightarrow a) \)
A.1.7 Algorithm Convergence

Theorem A.1.3. Let $X = [x_1 \ x_2 \ \cdots \ x_m]$ be the data matrix. The MM routine with objective function $f_{tr}(\Pi) = \sum_{i=1}^{m} |(x_i^\top \Pi x_i) - \frac{1}{2}|$ produces a sequence:

$$\Pi_{k+1} \in \text{span}_{n_1}\{XD(\Pi_k)X^\top\} \quad (A.37)$$

where $\text{span}_{n_1}\{M\}$ denotes the set of projectors spanning the dominant eigenspace of $M$ with dim $= n_1$, and:

$$D(\Pi_k) = \begin{bmatrix} \alpha_1(\Pi_k) & \alpha_2(\Pi_k) & \cdots & \alpha_m(\Pi_k) \end{bmatrix} \quad (A.38)$$

with

$$\alpha_i(\Pi_k) = \begin{cases} 1, & \text{if } x_i^\top \Pi_k x_i \geq \frac{1}{2} \\ -1, & \text{if } x_i^\top \Pi_k x_i < \frac{1}{2} \end{cases}$$

If $x_i^\top \Pi^* x_i \neq \frac{1}{2}$ for all $i$, then $\Pi_k \to \Pi^*$ and $\Pi^*$ satisfies the necessary conditions for being a local maximizer.

PROOF.

Part I:

A point $\Pi^*$ satisfies the necessary conditions for being a local maximizer of:

$$\max_{\Pi \in C} f(\Pi). \quad (A.39)$$

If and only if:

$$\Pi^* \in \text{span}_{n_1}\{XD(\Pi^*)X^\top\} \quad (A.40)$$

Part II:

$(\Pi_k)_{k \in \mathbb{N}}$ has a convergent subsequence, say $(\Pi_k)_{k \in \mathbb{N}_1}$, since $C$ is compact.

We will assume that $\Pi^*$ satisfies: 1) $x_i^\top \Pi^* x_i \neq \frac{1}{2}$ and that 2) $\text{span}_{n_1}\{XD(\Pi^*)X^\top\}$ is a singleton.

Part III:
Since 1) holds, there exists \( \varepsilon > 0 \) s.t.:

\[
\|\Pi - \Pi^*\| \leq \varepsilon \Rightarrow \alpha_i(\Pi) = \alpha_i(\Pi^*) \quad \forall i
\]  

(A.41)

**Part IV:**

Part III implies that: \( \Pi \in B(\Pi^*, \varepsilon) \Rightarrow XD(\Pi)X^\top = XD(\Pi^*)X^\top \)

**Part V:**

\[
f(\Pi_1) \geq f(\Pi_0) + \langle G_0, \Pi_1 - \Pi_0 \rangle
\]

\[
f(\Pi_2) \geq f(\Pi_1) + \langle G_1, \Pi_2 - \Pi_1 \rangle
\]

\[
\vdots
\]

\[
f(\Pi_{k+1}) \geq f(\Pi_k) + \langle G_k, \Pi_{k+1} - \Pi_k \rangle
\]

(A.42)

where \( G_i := XD(\Pi_i)X^\top \)

Taking the sum of the right hand side of all equations in (A.42) and also the left hand side, yields:

\[
f(\Pi_{k+1}) \geq f(\Pi_0) + \sum_{i=0}^{k} (\langle G_i, \Pi_{k+1} \rangle - \langle G_i, \Pi_k \rangle)
\]

(A.43)

We have that \( \langle G_k, \Pi_k \rangle = (\lambda_1 + \ldots + \lambda_n)(G_k) \). Since \( f \) is a continuous function and \( C \) is compact then, there exists \( B \in \mathbb{R} \) s.t. \( f(\Pi_k) \leq B \forall k \). It follows that:

\[
B \geq f(\Pi_0) + \sum_{i=0}^{k} \lambda(G_k) - \langle G_k, \Pi_k \rangle \forall k
\]

(A.44)

Letting \( k \to \infty \), yields \( \lambda(G_k) - \langle G_k, \Pi_k \rangle \to 0 \)

Without loss of generality, assume \( (\Pi_k)_{k \in \mathbb{N}_1} \subset B(\Pi^*, \varepsilon) \). From **Part IV:**

\[
\forall_{k \in \mathbb{N}_1} : \langle G_k, \Pi_k \rangle = \langle G^*, \Pi_k \rangle
\]

(A.45)

then,

\[
\langle G^*, \Pi_k \rangle = \langle G^*, \Pi^* \rangle
\]

(A.46)

**Part XI:**

From **Part VIII** and \( k \in \mathbb{N}_1 \):

\[
\lambda(G^*) - \langle G^*, \Pi_k \rangle \to 0 \Rightarrow \langle G^*, \Pi^* \rangle \to \lambda(G^*)
\]

(A.47)
Part XII:

\[
\begin{cases}
\text{Part X} & \Rightarrow \lambda(G^*) = \langle G^*, \Pi^* \rangle \\
\text{Part XI}
\end{cases}
\] (A.48)

Part XIII

\[
\text{Part XII} \rightarrow \Pi^* = \text{span}_{\Pi^*} \{XD(\Pi^*)X^T\}
\] (A.49)

Therefore:

\[k \in \mathbb{N}_1 \Rightarrow XD(\Pi_k)X^T = XD(\Pi^*)X^T\] (A.50)

And it follows that:

\[
\text{span}_{\Pi^*} \{XD(\Pi_k)X^T\} = \text{span}_{\Pi^*} \{XD(\Pi^*)X^T\}
\] (A.51)

A.1.8 \(\| \cdot \|_1\) is the best convex envelop to \(\| \cdot \|_0\)

For all \(f \in \text{Conv } \mathbb{R}^n\) with \(f : B_\infty(0, 1) \rightarrow \mathbb{R}\) and \(f(x) \leq \|x\|_0 \forall x \in B_\infty(0, 1)\), we have that:

\[f(x) \leq \|x\|_1 \leq \|x\|_0 \forall x \in B_\infty(0, 1)\] (A.52)

PROOF.

In this proof, we will denote \(B_\infty(0, 1)\) as \(B_\infty\), just to compact notation.

Firstly, we will prove the last inequality in (A.52).

Define \(| \cdot |_0 : \mathbb{R} \rightarrow \mathbb{R} \) as:

\[|x|_0 = \begin{cases} 1, & \text{se } x \neq 0 \\ 0, & \text{se } x = 0 \end{cases}\] (A.53)

Then, \(\forall x \in B_\infty : \|x\|_1 = \sum_i |x_i| \leq \sum_i |x_i|_0 = \|x\|_0\).

Now let’s start proving the first inequality in (A.52). Define the set:

\[\text{Vertex} := \{v \in \mathbb{R}^n : v = \sum_{i=1}^n \delta_i e_i, \delta_i \in \{-1, 1\}\}\] (A.54)

We have that: \(\|x\|_0 = \|x\|_1 \forall x \in \text{Vertex}\)

We shall prove that any \(x \in B_\infty\) can be written as a convex combination of elements of \(\text{Vertex}\), that is, \(B_\infty \subset \text{co} \text{Vertex}\). We can prove this, showing that \(\text{ext}(B_\infty) \subset \text{Vertex}\). In fact, H. Minkowski’s theorem [7] states that given a compact convex set \(C\) (in this case \(C = B_\infty^5\)) in \(\mathbb{R}^n\), then \(C\) is the

\(^5\)Every ball in \(\mathbb{R}^n\) is convex.
convex hull of its extreme points: \( C = \text{co ext}(C) \). Therefore, we have that \( \text{co ext}(B_\infty) = B_\infty \). That is

\[
\text{ext}(B_\infty) \subset \text{Vertex} \Rightarrow \text{co ext}(B_\infty) \subset \text{co Vertex} \Leftrightarrow B_\infty \subset \text{co Vertex} \tag{A.55}
\]

In other words, we have to prove the fact: \( x \in \text{ext}(B_\infty) \Rightarrow x \in \text{Vertex} \), which is equivalent to \( x \in B_\infty \setminus \text{Vertex} \Rightarrow x \in B_\infty \setminus \text{ext}(B_\infty) \).

Let \( x \in B_\infty \setminus \text{Vertex} \), we have that \( \exists i = 1, \ldots, n : -1 < \pi_i(x) < 1 \) and (by construction) it follows that \( \exists x_a, x_b \in B_\infty : \pi_i(x_a) = 1, \pi_i(x_b) = -1 \) and \( \pi_j(x_a) = \pi_j(x_b) = \pi_j(x) \forall j \neq i \).

Thus, \( \exists x = \alpha x_a + (1 - \alpha) x_b, \) that is, \( x \in B_\infty \setminus \text{ext}(B_\infty) \).

Therefore, any \( x \in B_\infty \) can be written as:

\[
x = \sum_{i=1}^{n+1} \alpha_i v_i, \quad \sum_j \alpha_j = 1, \quad \alpha_i \in (0, 1), \quad v_i \in \text{Vertex} \forall i \tag{A.56}
\]

Actually, in equation (A.56) we have used, implicitly, the Carathéodory’s theorem \[7\].

Thus, since \( f \in \text{Conv} \mathbb{R}^n \) and \( f(x) \leq \|x\| \forall x \in B_\infty \), we have that:

\[
f(x) \leq \sum \alpha_i f(v_i) \leq \sum \alpha_i \|v_i\|_0 = \sum \alpha_i \|v_i\|_1 = \|x\|_1 \tag{A.57}
\]

that is, we proved that:

\[
f(x) \leq \|x\|_1 \forall x \in B_\infty \tag{A.58}
\]

Just as an additional note, we have that \( B_\infty = \text{co}(\text{Vertex}) \). In effect we have \( \text{Vertex} \subset B_\infty \Rightarrow \text{co Vertex} \subset \text{co } B_\infty = B_\infty \). Then \( \text{co}(\text{Vertex}) \subset B_\infty \).