NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCES
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

PROGRAM OF POSTGRADUATE STUDIES

PhD THESIS

Robust Algorithms for Linear and Nonlinear Regression via Sparse Modeling Methods: Theory, Algorithms and Applications to Image Denoising

George K. Papageorgiou

ATHENS

MARCH 2016

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Εύρωστοι Αλγόριθμοι Αραιής Μοντελοποίησης για το Πρόβλημα της Γραμμικής και μη Γραμμικής Παλινδρόμησης: Θεωρία, Αλγόριθμοι και Εφαρμογές στην Αποθορύβωση Εικόνας

Γεώργιος Κ. Παπαγεωργίου

ΑΘΗΝΑ

ΜΑΡΤΙΟΣ 2016

PhD THESIS

Robust Algorithms for Linear and Nonlinear Regression via Sparse Modeling Methods:
Theory, Algorithms and Applications to Image Denoising

George K. Papageorgiou

SUPERVISOR: Sergios Theodoridis, Professor UoA

THREE-MEMBER ADVISORY COMMITTEE:
Sergios Theodoridis, Professor UoA
Nicholas Kalouptsidis, Professor UoA
Leoni Euaggelatou-Dalla, Professor UoA

SEVEN-MEMBER EXAMINATION COMMITTEE

Sergios Theodoridis,
Professor UoA

Nicholas Kalouptsidis,
Professor UoA

Leoni Euaggelatou-Dalla,
Professor UoA

Eleftherios Kofidis,
Assistant Professor UniPi

Stavros Perantonis,
Research Director NCSR-Demokritos

Athanasios Rontogiannis,
Senior Researcher NOA

Konstantinos Koutroubas,
Senior Researcher NOA

Examination Date 09/03/2016

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Εύρωστοι Αλγόριθμοι Αραιής Μοντελοποίησης για το Πρόβλημα της Γραμμικής και μη Γραμμικής Παλινδρόμησης: Θεωρία, Αλγόριθμοι και Εφαρμογές στην Αποθορύβωση Εικόνας

Γεώργιος Κ. Παπαγεωργίου

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: Σέργιος Θεοδωρίδης, Καθηγητής ΕΚΠΑ

ΤΡΙΜΕΛΗΣ ΕΠΙΤΡΟΠΗ ΠΑΡΑΚΟΛΟΥΘΗΣΗΣ:
Σέργιος Θεοδωρίδης, Καθηγητής ΕΚΠΑ
Νικόλαος Καλουπτσίδης, Καθηγητής ΕΚΠΑ
Λεώνη Ευαγγελάτου-Δάλλα, Καθηγήτρια ΕΚΠΑ

ΕΠΤΑΜΕΛΗΣ ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ

Σέργιος Θεοδωρίδης,
Καθηγητής ΕΚΠΑ

Νικόλαος Καλουπτσίδης,
Καθηγητής ΕΚΠΑ

Λεώνη Ευαγγελάτου-Δάλλα,
Καθηγήτρια ΕΚΠΑ

Ελευθέριος Κοφίδης,
Επίκουρος Καθηγητής ΠΑΠΕΙ

Σταύρος Περαντώνης,
Ερευνητής Α΄ ΕΚΕΦΕ Δημόκριτος

Αθανάσιος Ροντογιάννης,
Ερευνητής Β΄ Αστερ. Αθηνών

Κωνσταντίνος Κουτρούμπας,
Ερευνητής Β΄ Αστερ. Αθηνών

Ημερομηνία εξέτασης 09/03/2016

ABSTRACT

The task of robust regression is of particular importance in signal processing, statistics and machine learning. Ordinary estimators, such as the Least Squares (LS) one, fail to achieve sufficiently good performance in the presence of outliers. Although the problem has been addressed many decades ago and several methods have been established, it has recently attracted more attention in the context of sparse modeling and sparse optimization techniques. The latter is the line that has been followed in the current dissertation. The reported research, led to the development of a novel approach in the context of greedy algorithms. The model adopts the decomposition of the noise into two parts: a) the inlier noise and b) the outliers, which are explicitly modeled by employing sparse modeling arguments. Based on this rationale and inspired by the popular Orthogonal Matching Pursuit (OMP), two novel efficient greedy algorithms are established, one for the linear and another one for the nonlinear robust regression task.

The proposed algorithm for the linear task, i.e., Greedy Algorithm for Robust Denoising (GARD), alternates between a Least Squares (LS) optimization criterion and an OMP selection step, that identifies the outliers. The method is compared against state-of-the-art methods through extensive simulations and the results demonstrate that: a) it exhibits tolerance in the presence of outliers, i.e., robustness, b) it attains a very low approximation error and c) it has relatively low computational requirements. Moreover, due to the simplicity of the method, a number of related theoretical properties are derived. Initially, the convergence of the method in a finite number of iteration steps is established. Next, the focus of the theoretical analysis is turned on the identification of the outliers. The case where only outliers are present has been studied separately; this is mainly due to the following reasons: a) the simplification of technically demanding algebraic manipulations and b) the “articulation” of the method’s interesting geometrical properties. In particular, a bound based on the Restricted Isometry Property (RIP) constant guarantees that the recovery of the signal via GARD is exact (zero error). Finally, for the case where outliers as well as inlier noise coexist, and by assuming that the inlier noise vector is bounded, a similar condition that guarantees the recovery of the support for the sparse outlier vector is derived. If such a condition is satisfied, then it is shown that the approximation error is bounded, and thus the denoising estimator is stable.

For the robust nonlinear regression task, it is assumed that the unknown nonlinear function belongs to a Reproducing Kernel Hilbert Space (RKHS). Due to the existence of outliers, common techniques such as the Kernel Ridge Regression (KRR), or the Support Vector Regression (SVR) turn out to be inadequate. By employing the aforementioned noise decomposition, sparse modeling arguments are employed so that the outliers are estimated according to the greedy approach. The proposed robust scheme, i.e., Kernel Greedy Algorithm for Robust Denoising (KGARD), alternates between a KRR task and an OMP-like selection step. Theoretical results regarding the identification of the outliers are provided. Moreover, KGARD is compared against other cutting edge methods via extensive simulations, where its enhanced performance is demonstrated. Finally, the proposed robust estimation framework is applied to the task of image denoising, where the advantages of the proposed method are unveiled. The experiments verify that KGARD improves the denoising process significantly, when outliers are present.

SUBJECT AREA: Signal Processing

KEYWORDS: robust regression, greedy algorithm for robust denoising GARD, robust nonlinear regression in RKHS, kernel greedy algorithm for robust denoising KGARD, image denoising

ΠΕΡΙΛΗΨΗ

Η εύρωστη παλινδρόμηση κατέχει έναν πολύ σημαντικό ρόλο στην Επεξεργασία Σήματος, τη Στατιστική και τη Μηχανική Μάθηση. Συνήθεις εκτιμητές, όπως τα «Ελάχιστα Τετράγωνα», αποτυγχάνουν να εκτιμήσουν σωστά παραμέτρους, όταν στα δεδομένα υπεισέρχονται ακραίες παρατηρήσεις, γνωστές ως “outliers”. Το πρόβλημα αυτό είναι γνωστό εδώ και δεκαετίες, μέσα στις οποίες διάφορες μέθοδοι έχουν προταθεί. Παρόλα αυτά, το ενδιαφέρον της επιστημονικής κοινότητας για αυτό αναζωπυρώθηκε όταν επανεξετάστηκε υπό το πρίσμα της αραιής μοντελοποίησης και των αντίστοιχων τεχνικών, η οποία κυριαρχεί στον τομέα της μηχανικής μάθησης εδώ και δύο δεκαετίες. Αυτή είναι και η κατεύθυνση η οποία ακολουθήθηκε στην παρούσα διατριβή. Το αποτέλεσμα αυτής της εργασίας ήταν η ανάπτυξη μιας νέας προσέγγισης, βασισμένης σε άπληστες τεχνικές αραιής μοντελοποίησης. Το μοντέλο που υιοθετείται βασίζεται στην ανάλυση του θορύβου σε δύο συνιστώσες: α) μια για το συμβατικό (αναμενόμενο) θόρυβο και β) μια για τις ακραίες παρατηρήσεις (outliers), οι οποίες θεωρήθηκε ότι είναι λίγες (αραιές) σε σχέση με τον αριθμό των δεδομένων. Με βάση αυτή τη μοντελοποίηση και τον γνωστό άπληστο αλγόριθμο “Orthogonal Matching Pursuit” (OMP), δύο νέοι αλγόριθμοι αναπτύχθηκαν, ένας για το γραμμικό και ένας για το μη γραμμικό πρόβλημα της εύρωστης παλινδρόμησης.

Ο προτεινόμενος αλγόριθμος για τη γραμμική παλινδρόμηση ονομάζεται “Greedy Algorithm for Robust Demoising” (GARD) και εναλλάσσει τα βήματά του μεταξύ της μεθόδου Ελαχίστων Τετραγώνων (LS) και της αναγνώρισης των ακραίων παρατηρήσεων, τεχνικής που βασίζεται στον OMP. Στη συνέχεια, ακολουθεί η σύγκριση της νέας μεθόδου με ανταγωνιστικές της. Συγκεκριμένα, από τα αποτελέσματα παρατηρείται ότι ο GARD: α) δείχνει ανοχή σε ακραίες τιμές (εύρωστος), β) καταφέρνει να προσεγγίσει τη λύση με πολύ μικρό λάθος και γ) απαιτεί μικρό υπολογιστικό κόστος. Επιπλέον, προκύπτουν σημαντικά θεωρητικά ευρήματα, τα οποία οφείλονται στην απλότητα της μεθόδου. Αρχικά, αποδεικνύεται ότι η μέθοδος συγκλίνει σε πεπερασμένο αριθμό βημάτων. Στη συνέχεια, η μελέτη επικεντρώνεται στην αναγνώριση των ακραίων παρατηρήσεων. Το γεγονός ότι η περίπτωση απουσίας συμβατικού θορύβου μελετήθηκε ξεχωριστά, οφείλεται κυρίως στα εξής: α) στην απλοποίηση απαιτητικών πράξεων και β) στην ανάδειξη σημαντικών γεωμετρικών ιδιοτήτων. Συγκεκριμένα, προέκυψε κατάλληλο φράγμα για τη σταθερά της συνθήκης «Περιορισμένης Ισομετρίας» (“Restricted Isometry Property” - (RIP)), το οποίο εξασφαλίζει ότι η ανάκτηση του σήματος μέσω του GARD είναι ακριβής (μηδενικό σφάλμα). Τέλος, για την περίπτωση όπου ακραίες τιμές και συμβατικός θόρυβος συνυπάρχουν και με την παραδοχή ότι το διάνυσμα του συμβατικού θορύβου είναι φραγμένο, προέκυψε μια αντίστοιχη συνθήκη η οποία εξασφαλίζει την ανάκτηση του φορέα του αραιού διανύσματος θορύβου (outliers). Δεδομένου ότι μια τέτοια συνθήκη ικανοποιείται, αποδείχθηκε ότι το σφάλμα προσέγγισης είναι φραγμένο και άρα ο εκτιμητής GARD ευσταθής.

Για το πρόβλημα της εύρωστης μη γραμμικής παλινδρόμησης, θεωρείται, επιπλέον, ότι η άγνωστη μη γραμμική συνάρτηση ανήκει σε ένα χώρο Hilbert με αναπαραγωγικούς πυρήνες (RKHS). Λόγω της ύπαρξης ακραίων παρατηρήσεων, τεχνικές όπως το Kernel Ridge Regression (KRR) ή το Support Vector Regression (SVR) αποδεικνύονται ανεπαρκείς. Βασισμένοι στην προαναφερθείσα ανάλυση των συνιστωσών του θορύβου και χρησιμοποιώντας την τεχνική της αραιής μοντελοποίησης, πραγματοποιείται η εκτίμηση των ακραίων παρατηρήσεων σύμφωνα με τα βήματα μιας άπληστης επαναληπτικής διαδικασίας. Ο προτεινόμενος αλγόριθμος ονομάζεται “Kernel Greedy Algorithm for Robust Denoising” (KGARD), και εναλλάσσει τα βήματά μεταξύ ενός εκτιμητή KRR και της αναγνώρισης ακραίων παρατηρήσεων, με βάση τον OMP. Αναλύεται θεωρητικά η ικανότητα του αλγορίθμου να αναγνωρίσει τις πιθανές ακραίες παρατηρήσεις. Επιπλέον, ο αλγόριθμος KGARD συγκρίνεται με άλλες μεθόδους αιχμής μέσα από εκτεταμένο αριθμό πειραμάτων, όπου και παρατηρείται η σαφώς καλύτερη απόδοσή του. Τέλος, η προτεινόμενη μέθοδος για την εύρωστη παλινδρόμηση εφαρμόζεται στην αποθορύβωση εικόνας, όπου αναδεικνύονται τα σαφή πλεονεκτήματα της μεθόδου. Τα πειράματα επιβεβαιώνουν ότι ο αλγόριθμος KGARD βελτιώνει σημαντικά την διαδικασία της αποθορύβωσης, στην περίπτωση όπου στον θόρυβο υπεισέρχονται ακραίες παρατηρήσεις.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Επεξεργασία Σήματος

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: εύρωστη γραμμική παλινδρόμηση, άπληστος αλγόριθμος για εύρωστη γραμμική παλινδρόμηση, εύρωστη μη γραμμική παλινδρόμηση σε χώρους Hilbert με αναπαραγωγικούς πυρήνες, άπληστος αλγόριθμος για εύρωστη γραμμική παλινδρόμηση σε χώρους Hilbert με αναπαραγωγικούς πυρήνες, αποθορύβωση εικόνας

To my beloved wife and child...

Acknowledgements

This dissertation would have not been possible without the help of many people who supported me during the demanding process.

Foremost, I would like to express my sincere gratitude to my supervisor, Prof. Sergios Theodoridis, for introducing me to the "world" of scientific research and giving me the opportunity to pursue a doctoral degree. His knowledge and continuous guidance during the time of research and the writing of the thesis has been a significant learning experience to me. I feel extremely fortunate that I have worked with him and I will always be inspired by his amount of enthusiasm, patience and motivation for research.

Besides my supervisor, I am also grateful to Dr. Pantelis Bouboulis for the continuous support. His expertise improved my research skills and also prepared me for new challenges.

My sincere thanks also goes to the rest of my thesis committee: Prof. Leoni Euaggelatou-Dalla, who was the spark for me to start out, and Prof. Nikolaos Kalouptsidis for his insightful comments that kept me busy for quite some time.

My work has also benefited from my collaboration with Dr. Yannis Kopsinis, with whom I have had numerous long discussions over several topics. I would also like to thank Dr. Konstantinos Themelis for his assistance with the Bayesian codes.

Finally, I would like to thank my friends and colleagues, Dr. Symeon Chouvardas and Dr. Dimitris Manatakis, with whom I have spent countless pleasant hours working in the lab.

List of Publications

Refereed Journal Papers

1. G. Papageorgiou, P. Bouboulis, and S. Theodoridis, "Robust Linear Regression Analysis - A Greedy Approach", IEEE Transactions on Signal Processing, Vol. 63, Issue: 15, p. 3872-3887, 2015.

2. G. Papageorgiou, P. Bouboulis, and S. Theodoridis, "Robust non-linear Regression: A Greedy Approach Employing Kernels", under review, IEEE Transactions on Signal Processing, 2016.

Refereed Conference Papers

1. G. Papageorgiou, P. Bouboulis, and S. Theodoridis, "Robust Regression in RKHS - An Overview", In Proceedings of the European Signal Processing Conference (EUSIPCO), Nice, Cote d'Azur, France, 31st August – 4th September, 2015.

2. G. Papageorgiou, P. Bouboulis, and S. Theodoridis, "Robust Linear Regression Analysis The Greedy Way", In Proceedings of the European Signal Processing Conference (EUSIPCO), Lisbon, Portugal, September 1 – 5, 2014 (among the top five shortlisted for best paper award).

3. P. Bouboulis, G. Papageorgiou, and S. Theodoridis, "Robust Image Denoising in RKHS via Orthogonal Matching Pursuit", International Workshop on Cognitive Information Processing (CIP), Copenhagen, Denmark, May 26 – 28, 2014.

4. G. Papageorgiou, P. Bouboulis, and S. Theodoridis, "Robust Kernel-Based Regression Using Orthogonal Matching Pursuit", IEEE International Workshop on Machine Learning for Signal Processing (MLSP), Southampton, United Kingdom, September 22 – 25, 2013.

Συνοπτική Παρουσίαση της Διδακτορικής Διατριβής

Η ανάλυση παλινδρόμησης ή απλά παλινδρόμηση βρίσκεται στην καρδιά της Μηχανικής Μάθησης. Σε ένα κλασσικό πρόβλημα παλινδρόμησης, δίνεται ένα σύνολο δεδομένων εκπαίδευσης, όπου στόχος είναι η εκμάθηση μιας σειράς από άγνωστες παραμέτρους. Αυτό έχει ως αποτέλεσμα να μπορούμε να πραγματοποιήσουμε προβλέψεις ή να εξάγουμε χρήσιμες πληροφορίες σχετικά με το βαθμό εξάρτησης μεταξύ της ανεξάρτητης και της εξαρτημένης μεταβλητής. Η σχέση αυτή μπορεί βεβαίως να είναι γραμμική ή μη γραμμική, που είναι και η πιο γενική κατηγορία.

Η πρώτη γραμμική παλινδρόμηση πραγματοποιήθηκε με τη μέθοδο «Ελαχίστων Τετραγώνων», η οποία δόθηκε στη δημοσιότητα από τον Legendre το 1805 και από τον Gauss το 1809. Δεδομένου ότι η πραγματική διαδικασία παραγωγής των δεδομένων είναι γενικά άγνωστη, η ανάλυση παλινδρόμησης εξαρτάται συχνά και σε μεγάλο βαθμό από τις υποθέσεις μας σχετικά με αυτή τη διαδικασία. Αν και κάποιες μέθοδοι χρησιμοποιούνται ακόμα σε περιπτώσεις όπου κάποιες υποθέσεις παραβιάζονται μερικώς, η ακρίβεια τους δεν είναι καλή. Ο πιο σημαντικός παράγοντας που επηρεάζει την απόδοση μιας μεθόδου εκτίμησης παραμέτρων, υπό την προϋπόθεση ότι το μοντέλο είναι σωστό, είναι το είδος του θορύβου που υπεισέρχεται στις παρατηρήσεις μας. Για παράδειγμα, στη γραμμική παλινδρόμηση και παρουσία λευκού Gaussian θορύβου, η μέθοδος Ελαχίστων Τετραγώνων είναι βέλτιστη, υπό την έννοια της μέγιστης πιθανοφάνειας (ML). Δυστυχώς όμως, αυτό δεν είναι ισχύει στην περίπτωση που ο θόρυβος ακολουθεί άλλες κατανομές, όπως για παράδειγμα μια κατανομή με μακριές ουρές. Σε μια τέτοια περίπτωση, ο εκτιμητής Ελαχίστων Τετραγώνων αποτυγχάνει σημαντικά να δώσει αξιόπιστη εκτίμηση. Συνεπώς, η απόδοση μιας μεθόδου δεν μπορεί να εξασφαλιστεί κάτω από οποιεσδήποτε συνθήκες.

Μια από τις σημαντικότερες προκλήσεις για το πρόβλημα της παλινδρόμησης είναι η ανάπτυξη εύρωστων (robust) μεθόδων, δηλαδή βασισμένων σε τεχνικές οι οποίες δεν είναι ευάλωτες σε σημαντικά εσφαλμένες μετρήσεις, οι οποίες ονομάζονται «ακραίες παρατηρήσεις» (outliers). Παραδόξως, αν και όλα αυτά τα χρόνια έχει γίνει προσπάθεια να δοθεί ένας ακριβής ορισμός για τις ακραίες παρατηρήσεις, αυτό δεν κατέστη δυνατό, καθότι είναι άμεσα εξαρτώμενο από τα δεδομένα. Ο πιο συνηθισμένος χαρακτηρισμός για μια ακραία παρατήρηση είναι ότι μοιάζει αταίριαστη με το υπόλοιπο σύνολο δεδομένων ή το γενικό μοτίβο κατανομής τους. Θεωρούνται ως εσφαλμένες μετρήσεις οι οποίες προήλθαν από διαφορετική πηγή και συχνά παρεκκλίνουν σημαντικά από τις υπόλοιπες παρατηρήσεις.

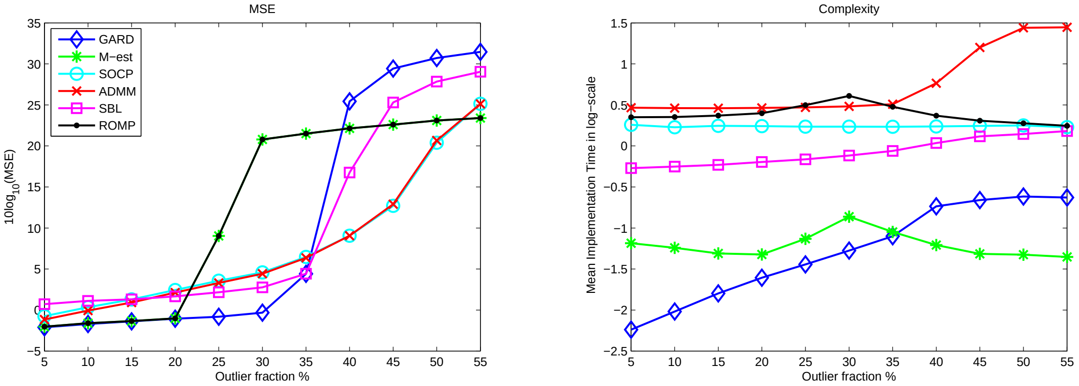
Οι μέθοδοι που έχουν αναπτυχθεί για την επίλυση προβλημάτων εύρωστης παλινδρόμησης χωρίζονται σε δύο σημαντικές κατηγορίες: α) τις τεχνικές Διάγνωσης και β) εκείνες της Εύρωστης Παλινδρόμησης. Παρόλο που οι δύο αυτές τεχνικές έχουν κοινό στόχο, αυτός προσεγγίζεται με την αντίστροφη σειρά. Με τη χρήση διαγνωστικών εργαλείων, αρχικά προσπαθούμε να αναγνωρίσουμε και να εξαιρέσουμε τις ακραίες παρατηρήσεις από το σύνολο των δεδομένων, ώστε στη συνέχεια να εκτιμήσουμε τις παραμέτρους με μια κλασσική μέθοδο, π.χ. Ελαχίστων Τετραγώνων. Από την άλλη πλευρά, μια μέθοδος εύρωστης παλινδρόμησης πρώτα ταιριάζει κατά προσέγγιση τα δεδομένα χρησιμοποιώντας κάποιο εκτιμητή, και στη συνέχεια εκμεταλλεύεται την αρχική αυτή εκτίμηση για τον εντοπισμό των ακραίων παρατηρήσεων. Και οι δύο κατηγορίες έχουν μελετηθεί συστηματικά για πάνω από μισό αιώνα και αποτελούν τα θεμέλια της Εύρωστης Στατιστικής.

Στο επίκεντρο αυτής της διατριβής βρίσκεται η μελέτη του προβλήματος της γραμμικής και μη γραμμικής παλινδρόμησης και η ανάπτυξη εύρωστων αλγορίθμων, υπό το πρίσμα των νέων τεχνικών αραιής μοντελοποίησης. Για πάνω από μια δεκαετία, η επιστημονική κοινότητα της Επεξεργασίας Σήματος εστίασε το ενδιαφέρον της στην αραιή μοντελοποίηση, η οποία ακόμα και σήμερα εξακολουθεί να είναι μια ερευνητικά ενεργή περιοχή. Η αραιότητα σχετίζεται στενά με την επάρκεια για μια οικονομική αναπαράσταση, έναν μηχανισμό που εναρμονίζεται με τη φύση, η οποία τείνει να είναι φειδωλή. Τα προβλήματα βελτιστοποίησης με αραιές αναπαραστάσεις χωρίζονται σε δύο βασικές κατηγορίες. Στην πρώτη κατηγορία ανήκουν εκείνες οι μέθοδοι που επιδιώκουν την ελαχιστοποίηση της ℓ_0(ψεύδο)-νόρμας, η οποία ισούται με τον αριθμό των μη μηδενικών συντεταγμένων ενός διανύσματος. Λόγω ότι η ℓ_0(ψεύδο)-νόρμα είναι μη κυρτή, έχει αποδειχθεί ότι τα προβλήματα αυτά δεν επιλύονται σε πολυωνυμικό χρόνο (NP-Hard). Ωστόσο, έχουν αναπτυχθεί τεχνικές οι οποίες μπορούν να παρακάμψουν τη συνδυαστική φύση αυτών των προβλημάτων και κάτω υπό ορισμένες προϋποθέσεις να οδηγήσουν στη λύση τους. Μια πολύ σημαντική τέτοια τεχνική είναι αυτή των «άπληστων» (greedy) αλγορίθμων. Η δεύτερη κατηγορία η οποία είναι και δημοφιλέστερη, αποτελείται από κυρτά προβλήματα ελαχιστοποίησης της ℓ_1-νόρμας.

Η συνεισφορά της εργασίας που ακολουθεί είναι η ανάπτυξη μια νέας προσέγγισης για την εύρωστη παλινδρόμηση, η οποία βασίζεται στην αξιοποίηση «άπληστων» τεχνικών αραιής μοντελοποίησης. Θεωρώντας ότι οι ακραίες παρατηρήσεις είναι αραιές, δηλαδή λίγες σε σχέση με το σύνολο των δεδομένων, πραγματοποιείται η ανάλυση του διανύσματος του θορύβου σε δύο συνιστώσες, μια για τον αναμενόμενο θόρυβο, που αναπόφευκτα υπεισέρχεται στις μετρήσεις μας, και μια για τις ακραίες παρατηρήσεις που ενδέχεται να υπάρχουν. Συγκεκριμένα, αναπτύχθηκαν δύο αλγόριθμοι βασισμένοι στον βασικό άπληστο (greedy) αλγόριθμο Orthogonal Matching Pursuit (OMP). Ένας για τη γραμμική και ένας δεύτερος για τη μη γραμμική παλινδρόμηση, όπου έγινε και χρήση χώρων Hilbert με αναπαραγωγικούς πυρήνες. Η μελέτη και για τα δύο προβλήματα έγινε τόσο σε θεωρητικό όσο και σε πρακτικό επίπεδο. Τέλος, ο δεύτερος αλγόριθμος που υλοποιήθηκε για το μη γραμμικό πρόβλημα, εφαρμόστηκε για την αποθορύβωση εικόνας.

Εύρωστη Γραμμική Παλινδρόμηση

Για το πρόβλημα της γραμμικής παλινδρόμησης θεωρήσαμε ότι στα δεδομένα εξόδου υπεισέρχεται αναμενόμενος θόρυβος και ακραίες παρατηρήσεις, τις οποίες θεωρούμε λίγες σε σχέση με τον αριθμό των δεδομένων. Επιπλέον, υποθέσαμε έναν ικανοποιητικό αριθμό, N, από δεδομένα, μεγαλύτερο των άγνωστων παραμέτρων προς εκτίμηση, M. Η προτεινόμενη μέθοδος για την εύρωστη γραμμική παλινδρόμηση βασίστηκε στο επαναληπτικό σχήμα του άπληστου αλγόριθμου OMP. Ο νέος αλγόριθμος, Greedy Algorithm for Robust Denoising (GARD), εναλλάσσεται μεταξύ ενός βήματος Ελάχιστων Τετραγώνων και της αναγνώρισης και επιλογής μιας μόνο ακραίας παρατήρησης, μέσα από το υπόλοιπο της μεθόδου Ελαχίστων Τετραγώνων. Πρόκειται για έναν αποδοτικό αλγόριθμο, ο οποίος συνδυάζει τις τεχνικές της διάγνωσης και της εύρωστης παλινδρόμησης.



(α): Διάσταση αγνώστου $M = 100$.

(β): Διάσταση αγνώστου $M = 100$.

Σχήμα 1: (α): Το μέσο τετραγωνικό σφάλμα (MSE) σε λογαριθμική κλίμακα συναρτήσει του ποσοστού ακραίων παρατηρήσεων στα δεδομένα εξόδου. (β): Λογαριθμική κλίμακα του μέσου χρόνου σύγκλισης κάθε μεθόδου για το ίδιο πείραμα. Ο αριθμός κάθε δείγματος αποτελείται από $N = 600$ παρατηρήσεις.

Η απλότητα του νέου αλγόριθμου GARD πηγάζει από τον OMP, και αποδείχθηκε κλειδί για την εκτενή μελέτη των ιδιοτήτων του. Συγκεκριμένα, τα θεωρητικά αποτελέσματα που προέκυψαν είναι τα εξής:

- Η απόδειξη της σύγκλισης του GARD σε πεπερασμένο αριθμό βημάτων.

- Για την περίπτωση όπου μόνο ακραίες παρατηρήσεις υπεισέρχονται στις μετρήσεις μας, η απόδειξη μιας (ανισοτικής) συνθήκης-φράγματος για τη σταθερά της συνθήκης Περιορισμένης Ισομετρίας (Restricted Isometry Property - (RIP)), η οποία εξασφαλίζει ότι η μέθοδος επιτυγχάνει με επιτυχία να αναγνωρίσει τις ακραίες παρατηρήσεις, όπου εμφανίζονται. Επιπλέον, στην περίπτωση αυτή, εξασφαλίζεται ότι ο GARD καταφέρνει να ανακτήσει την ακριβή λύση του προβλήματος, χωρίς σφάλμα.

- Η θεμελίωση ενός δεύτερου φράγματος για τη σταθερά της συνθήκης Περιορισμένης Ισομετρίας (RIP), το οποίο εξασφαλίζει ότι η μέθοδος επιτυγχάνει με επιτυχία να αναγνωρίσει τις ακραίες παρατηρήσεις, παρουσία και αναμενόμενου θορύβου, η ενέργεια του οποίου όμως είναι φραγμένη.

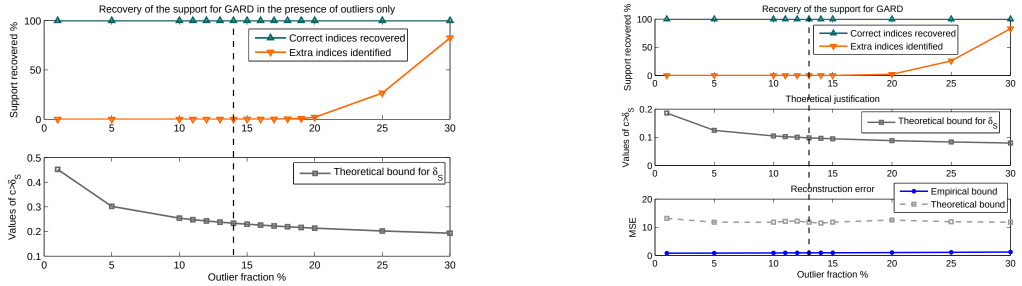
- Για την προηγούμενη περίπτωση, προέκυψε, επιπλέον, ένα φράγμα για το σφάλμα της προσέγγισης, το οποίο μας εξασφαλίζει τη σταθερότητα του αλγόριθμου GARD.

Αξίζει να σημειωθεί ότι, η ανάκτηση των θέσεων των ακραίων παρατηρήσεων μέσα από το παραπάνω φράγμα είναι ένα αποτέλεσμα που προέκυψε για πρώτη φορά στα πλαίσια της εύρωστης παλινδρόμησης.

Στη συνέχεια, ακολούθησε μια σειρά από εκτεταμένα πειράματα στα οποία ο GARD συγκρίθηκε με ανταγωνιστικούς του αλγόριθμους. Για κάθε μέθοδο, μετρήθηκε το μέσο τετραγωνικό σφάλμα (MSE) και ο χρόνος σύγκλισης, συναρτήσει του ποσοστού των ακραίων παρατηρήσεων που υπεισέρχονται στα δεδομένα εξόδου. Παρατηρήθηκε ότι ο αλγόριθμος GARD:

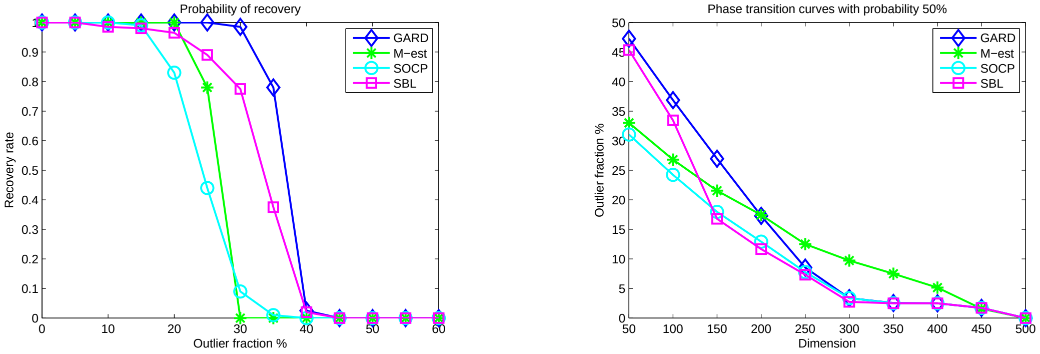
- Προσεγγίζει τη λύση του προβλήματος παλινδρόμησης με το χαμηλότερο μέσο τετραγωνικό σφάλμα (MSE).

- Παρουσιάζει τη μεγαλύτερη ευρωστία από κάθε άλλη μέθοδο.



(α): Μόνο ακραίες παρατηρήσεις. (β): Ακραίες παρατηρήσεις και θόρυβος φραγμένης ενέργειας.

Σχήμα 2: Η αναγνώριση των ακραίων παρατηρήσεων και η συσχέτιση με το θεωρητικό φράγμα της συνθήκης Περιορισμένης Ισομετρίας δ_S. (α): Ο θόρυβος αποτελείται μόνο από ακραίες παρατηρήσεις. (β): Ο θόρυβος αποτελείται από ακραίες παρατηρήσεις και συμβατικό φραγμένης ενέργειας. Δίνεται επίσης το εκτιμώμενο και το πραγματικό σφάλμα της προσέγγισης που επιτυγχάνει ο GARD (κάτω δεξιά).



(α): Διάσταση αγνώστου $M = 100$. (β): Μετάβαση από την επιτυχία στην αποτυχία με πιθανότητα 50%.

Σχήμα 3: (α): Η πιθανότητα για σωστή εκτίμηση σε σχέση με το ποσοστό των ακραίων παρατηρήσεων. (β): Καμπύλες μετάβασης από σωστή σε λανθασμένη εκτίμηση, συναρτήσει του αριθμού των αγνώστων για σταθερό αριθμό $N = 600$ παρατηρήσεων.

- Απαιτεί το μικρότερο μέσο χρόνο σύγκλισης.

Στο Σχήμα 1, παρατηρείται η απόδοση του προτεινόμενου αλγόριθμου, KGARD, σε σχέση με άλλες μεθόδους αιχμής, ενώ στο Σχήμα 2 παρουσιάζεται η ανάκτηση των θέσεων των ακραίων παρατηρήσεων από τον GARD.

Στο Σχήμα 3 (α), διαμορφώνεται η πιθανότητα για σωστή εκτίμηση για κάθε μέθοδο, καθώς αυξάνεται το ποσοστό των ακραίων παρατηρήσεων. Στο Σχήμα 3 (β), δίνονται οι καμπύλες μετάβασης από σωστή σε λανθασμένη εκτίμηση για το ποσοστό 50%, μεταβάλλοντας τον αριθμό των αγνώστων. Δηλαδή, για παράδειγμα στη διάσταση $M = 100$ (Σχήμα (α)), η οριζόντια ευθεία που διέρχεται από το 0.5 δίνει τα αντίστοιχα ποσοστά των ακραίων παρατηρήσεων που βρίσκονται από τα σημεία τομής των καμπύλων και της κατακόρυφης ευθείας που περνά από τη διάσταση 100 του Σχήματος 3 (β). Είναι εμφανές ότι ο αλγόριθμος GARD, εκμεταλλεύεται τον διαθέσιμο αριθμό των παρατηρήσεων καλύτερα από τις άλλες μεθόδους.

Τέλος, στον Πίνακα 1 δίνεται το μετρούμενο μέσο τετραγωνικό σφάλμα για κάθε μέθοδο, με τον θόρυβο να ακολουθεί μια πιο γενική κατανομή. Στις στήλες A, B και C ο θόρυβος προκύπτει από μια κατανομή με μακριές ουρές, την alpha-stable της Lévy, ενώ στη στήλη D χρησιμοποιήθηκε θόρυβος με ακραίες παρατηρήσεις και αναμενόμενος, προερχόμενος από δύο ανεξάρτητες Gaussian κατανομές.

Πίνακας 1: Το μετρούμενο μέσο τετραγωνικό σφάλμα (MSE) για πιο γενικές περιπτώσεις θορύβου. Για της στήλες A,B και C ο θόρυβος προκύπτει από μια κατανομή με μακριές ουρές, την alpha-stable της Lévy. Για την στήλη D χρησιμοποιήθηκε θόρυβος που αποτελείται από ακραίες παρατηρήσεις και επιπλέον από δύο ανεξάρτητες Gaussian κατανομές.

Algorithm	Test A	Test B	Test C	Test D
GARD	0.1772	0.0180	0.0586	0.690
M-est	0.2248	0.2859	1.844e+06	0.704
SOCP	0.4990	0.3502	5.852e+05	1.011
SBL	0.9859	58.3489	2.165e+06	1.292
ROMP	0.2248	0.2859	1.844e+06	0.704

Εύρωστη Μη Γραμμική Παλινδρόμηση

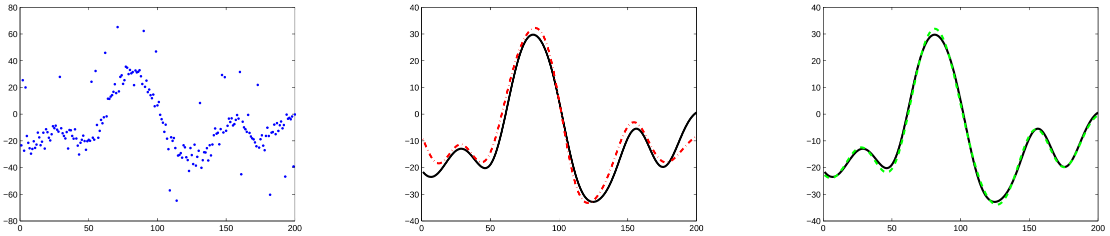
Για τη μελέτη της εύρωστης μη γραμμικής παλινδρόμησης θεωρήσαμε ότι η συνάρτηση η οποία παράγει τα δεδομένα χωρίς θόρυβο ανήκει σε ένα χώρο Hilbert με αναπαραγωγικούς πυρήνες (RKHS). Χρησιμοποιώντας τους πυρήνες, καταφεύγουμε σε απλές πράξεις (γραμμικές) με την αντικατάσταση της μήτρας παλινδρόμησης από μια μήτρα πυρήνων. Με αυτό τον τρόπο και κάνοντας χρήση του Θεωρήματος Αναπαράστασης (Representer Theorem), καταλήξαμε σε ένα επαναληπτικό σχήμα, επίσης βασισμένο στον αλγόριθμο OMP. Αξίζει να σημειωθεί, ότι το πρόβλημα αυτό είναι μη παραμετρικό, σε αντίθεση με το αντίστοιχο γραμμικό πρόβλημα. Συνεπώς, ο προτεινόμενος αλγόριθμος διαφέρει σημαντικά από τον GARD για το γραμμικό μοντέλο. Τέλος, στο μοντέλο που χρησιμοποιήσαμε για το μη γραμμικό πρόβλημα θεωρήσαμε ότι στις μετρήσεις μας υπεισέρχεται αναμενόμενος θόρυβος (π.χ. Gaussian) και λίγες ακραίες παρατηρήσεις (αραιές).

Ο προτεινόμενος αλγόριθμος ονομάστηκε Kernel Greedy Algorithm for Robust Denoising (KGARD) και εναλλάσσεται μεταξύ ενός βήματος Ελαχίστων Τετραγώνων με αντισταθμιστή και ενός βήματος αναγνώρισης και επιλογής μιας ακραίας παρατήρησης μέσα από το υπόλοιπο. Η προσθήκη του όρου αντιστάθμισης στο μοντέλο οδηγεί σε μια πιο σύνθετη θεωρητική ανάλυση, αλλά είναι αναπόφευκτη, λόγω της αναζήτησης για μια σχετικά ομαλή συνάρτηση. Παρόλα αυτά, προέκυψαν σημαντικά θεωρητικά ευρήματα για τον αλγόριθμο KGARD, τα οποία συνοψίζονται ακολούθως:

- Η λύση του προβλήματος Ελαχίστων Τετραγώνων με αντισταθμιστή σε κάθε βήμα είναι μοναδική.

- Αποδείχθηκε ένα φράγμα της μέγιστης ιδιάζουσας τιμής (singular value) του πίνακα των πυρήνων, το οποίο, αν ικανοποιείται, εξασφαλίζει ότι η μέθοδος πρώτα θα αναγνωρίσει τις σωστές θέσεις των ακραίων παρατηρήσεων, στην περίπτωση όπου δεν υπεισέρχεται άλλος θόρυβος στις μετρήσεις μας.

Η ανάλυση πραγματοποιήθηκε για την περίπτωση όπου στο θόρυβο υπεισέρχονται μόνο ακραίες παρατηρήσεις. Εντούτοις, όπως παρατηρείται στα πειράματα, η μέθοδος επιτυγχάνει να ανακτήσει τη σωστές θέσεις των ακραίων παρατηρήσεων και σε πολλές περιπτώσεις όπου το θεωρητικό αποτέλεσμα δεν ισχύει. Αυτό οδηγεί στο συμπέρασμα ότι η συνθήκη αυτή είναι αυστηρή. Στην πράξη, όταν στις μετρήσεις υπεισέρχεται επιπλέον και αναμενόμενος θόρυβος, η μέθοδος καταφέρνει να αναγνωρίσει την πλειοψηφία των θέσεων των ακραίων παρατηρήσεων. Ο λόγος για τον οποίο η ανάλυση δεν πραγματοποιείται για την περίπτωση αυτή είναι η δυσκολία των υπολογισμών, καθότι η ανάλυση



(α): Παρατηρήσεις με θόρυβο. (β): Εκτίμηση με Ελάχιστα Τετράγωνα με αντισταθμιστή. (γ): Εκτίμηση με τον αλγόριθμο KGARD.

Σχήμα 4: Ο ρόλος της εύρωστης μη γραμμικής παλινδρόμησης. (α): Δεδομένα με Gaussian θόρυβο και ακραίες παρατηρήσεις. (β): Η εκτίμηση μέσω μιας μη εύρωστης τεχνικής (κόκκινη διακεκομμένη γραμμή) απέχει σημαντικά από την πραγματική μη γραμμική συνάρτηση (μαύρη γραμμή). Το μέσο τετραγωνικό σφάλμα υπολογίζεται στο 10.79. (γ): Η εκτίμηση μέσω του KGARD (πράσινη διακεκομμένη γραμμή) είναι πολύ πιο ακριβής και το μέσο τετραγωνικό σφάλμα υπολογίζεται στο 1.21.

γίνεται ιδιαίτερα σύνθετη. Η απουσία του αναμενόμενου θορύβου καθιστά την ανάλυση ευκολότερη και δίνει έμφαση σε θεωρητικές πτυχές που καταδεικνύουν τους λόγους για τους οποίους η μέθοδος λειτουργεί. Τέλος, αξίζει να σημειωθεί ότι μια τέτοια θεωρητική ανάλυση παρουσιάζεται για πρώτη φορά στη σχετική βιβλιογραφία. Η σπουδαιότητα της εύρωστης μη γραμμικής παλινδρόμησης φαίνεται στο Σχήμα 4, όπου γίνεται σύγκριση του αλγορίθμου με τον εκτιμητή Ελαχίστων Τετραγώνων με αντισταθμιστή.

Στη συνέχεια ελέγξαμε την απόδοση του αλγόριθμου KGARD πειραματικά, μέσα από μια σειρά μετρήσεων. Οι ανταγωνιστικοί αλγόριθμοι με τους οποίους συγκρίθηκε επίσης βασίζονται σε αραιές αναπαραστάσεις και χρησιμοποιούν αναπαραγωγικούς πυρήνες χώρων Hilbert. Η μέθοδος RAM βασίζεται στην ελαχιστοποίηση της ℓ_1-νόρμας ενός διανύσματος, ενώ η μέθοδος RB-RVM κάνει χρήση Bayesian modeling τεχνικών. Στον Πίνακα 2, παρουσιάζεται το εκτιμώμενο μέσο τετραγωνικό σφάλμα (MSE) για την προσέγγιση της συνάρτησης $\mathrm{f}(x) = 20sinc(2\pi x)$. Επιπλέον, δίνονται το ποσοστό σωστών και λάθος θέσεων που κάθε μέθοδος καταχώρισε ως ακραία παρατήρηση και ο μέσος χρόνος σύγκλισης, για κάθε επίπεδο αναμενόμενου Gaussian θορύβου (σε dB) και ποσοστό ακραίων παρατηρήσεων. Παρατηρείται ότι για τις περισσότερες περιπτώσεις, εκτός από την περίπτωση ισχυρού θορύβου με 20% ακραίες παρατηρήσεις, ο KGARD παρουσιάζει τη μικρότερη τιμή τετραγωνικού σφάλματος, ενώ παράλληλα καταφέρνει με επιτυχία να αναγνωρίσει τις θέσεις των ακραίων παρατηρήσεων. Τέλος, αξίζει να σημειωθεί ότι η απόδοση της μεθόδου ήταν ανάλογη και για την περίπτωση όπου τα δεδομένα εισόδου ανήκουν στον $\mathbb{R}^2$.

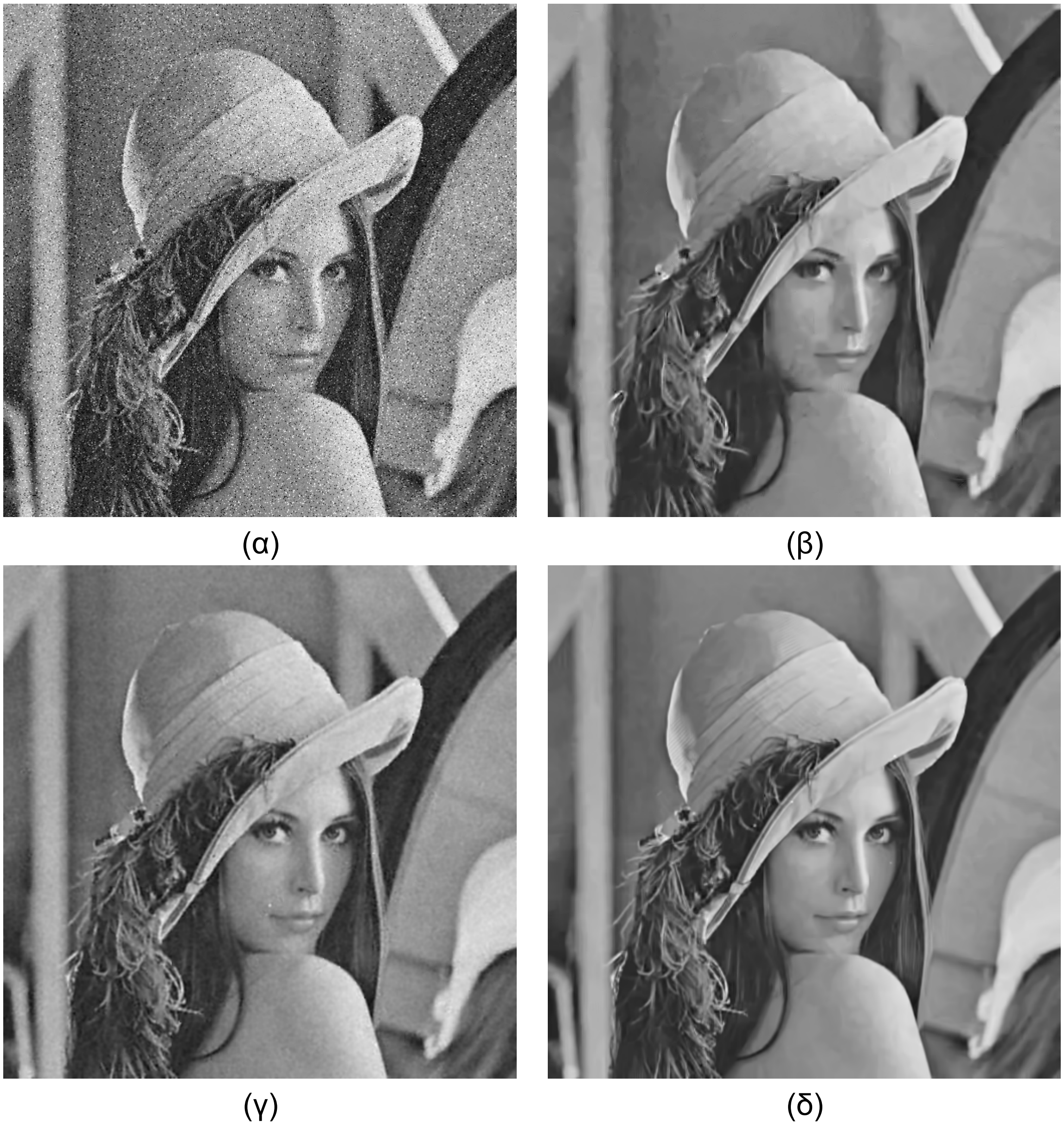
Πίνακας 2: Μετρούμενο μέσο τετραγωνικό σφάλμα (MSE) για την προσέγγιση της συνάρτησης $\mathrm{f}(x) = 20sinc(2\pi x)$ για $x \in \mathbb{R}$, για τα σύνολα εκπαίδευσης και επιβεβαίωσης. Επιπλέον, δίνονται το ποσοστό σωστών και λάθος θέσεων που κάθε μέθοδος καταχώρισε ακραία παρατήρηση και ο μέσος χρόνος σύγκλισης, για κάθε επίπεδο αναμενόμενου Gaussian θορύβου (σε dB) και ποσοστό ακραίων παρατηρήσεων.

Algorithm	MSE_{tr}	MSE_{val}	Cor. ind.	Wr. ind.	MIT (sec)	Inlier - Outlier
RB-RVM	0.0850	0.0851	-	-	0.298	20 dB - 5%
RAM ($\lambda = 0.07, \mu = 2.5$)	0.0344	0.0345	100 %	0.2 %	0.005	20 dB - 5%
KGARD ($\lambda = 0.2, \varepsilon = 10$)	**0.0285**	**0.0285**	100 %	0 %	0.004	20 dB - 5%
RB-RVM	0.0911	0.0912	-	-	0.298	20 dB - 10%
RAM ($\lambda = 0.07, \mu = 2.5$)	0.0371	0.0372	100 %	0.1 %	0.007	20 dB - 10%
KGARD ($\lambda = 0.2, \varepsilon = 10$)	**0.0305**	**0.0305**	100 %	0 %	0.008	20 dB - 10%
RB-RVM	0.0992	0.0994	-	-	0.299	20 dB - 15%
RAM ($\lambda = 0.07, \mu = 2$)	0.0393	0.0393	100 %	0.6 %	0.008	20 dB - 15%
KGARD ($\lambda = 0.3, \varepsilon = 10$)	**0.0330**	**0.0330**	100 %	0 %	0.012	20 dB - 15%
RB-RVM	0.1189	0.1184	-	-	0.305	20 dB - 20%
RAM ($\lambda = 0.07, \mu = 2$)	**0.0421**	**0.0422**	100 %	0.4 %	0.010	20 dB - 20%
KGARD ($\lambda = 1, \varepsilon = 10$)	0.0626	0.0626	100 %	0 %	0.017	20 dB - 20%
RB-RVM	0.3630	0.3631	-	-	0.327	15 dB - 5%
RAM ($\lambda = 0.15, \mu = 5$)	0.1035	0.1036	100%	0.7 %	0.005	15 dB - 5%
KGARD ($\lambda = 0.3, \varepsilon = 15$)	**0.0862**	**0.0862**	100 %	0.1 %	0.005	15 dB - 5%
RB-RVM	0.3828	0.3830	-	-	0.319	15 dB - 10%
RAM ($\lambda = 0.15, \mu = 5$)	0.1117	0.1118	100%	0.4 %	0.006	15 dB - 10%
KGARD ($\lambda = 0.3, \varepsilon = 15$)	**0.0925**	**0.0925**	100 %	0 %	0.008	15 dB - 10%
RB-RVM	0.4165	0.4166	-	-	0.317	15 dB - 15%
RAM ($\lambda = 0.15, \mu = 5$)	0.1186	0.1186	100%	0.3 %	0.007	15 dB - 15%
KGARD ($\lambda = 0.3, \varepsilon = 15$)	**0.1001**	**0.1003**	100 %	0 %	0.012	15 dB - 15%
RB-RVM	0.4793	0.4798	-	-	0.312	15 dB - 20%
RAM ($\lambda = 0.15, \mu = 4$)	**0.1281**	**0.1282**	100%	1.4 %	0.008	15 dB - 20%
KGARD ($\lambda = 0.7, \varepsilon = 15$)	0.1340	0.1349	100 %	0 %	0.016	15 dB - 20%

Εφαρμογή στην αποθορύβωση Εικόνας

Τέλος, έμφαση δόθηκε σε εφαρμογές της μεθόδου KGARD για την αποθορύβωση εικόνας, παρουσία ακραίων τιμών. Ο θόρυβος που χρησιμοποιήθηκε είναι Gaussian για τον αναμενόμενο και αλάτι-πιπέρι (salt and pepper noise) για τις ακραίες παρατηρήσεις.

Η συμβολή μας στην εύρωστη αποθορύβωση ήταν μέσω του αλγόριθμου KGARD. Συγκεκριμένα, προτείνονται δύο διαφορετικές μέθοδοι αποθορύβωσης για αυτό το είδος του θορύβου. Η πρώτη βασίζεται άμεσα στον αλγόριθμο KGARD. Η δεύτερη μέθοδος ολοκληρώνεται σε δύο στάδια: αρχικά πραγματοποιείται ο εντοπισμός και η απομάκρυνση των ακραίων παρατηρήσεων μέσω του KGARD. Στη συνέχεια ακολουθεί η αφαίρεση του εναπομείναντος θορύβου μέσω μιας μεθόδου αιχμής βασισμένης σε wavelets (BM3D). Τα αποτελέσματα που προέκυψαν είναι βάσει του μετρούμενου PSNR (Peak singal-to-noise-ratio). Στον Πίνακα 3 δίνονται ορισμένες τιμές για την αποθορύβωση της εικόνας της *Lena* και στο Σχήμα 5 παρατηρείται το αποτέλεσμα. Τέλος, στον Πίνακα 4 δίνονται οι σημαντικότερες τιμές για το μετρούμενο PSNR στην αποθορύβωση της εικόνας του καραβιού *boat*. Στο Σχήμα 6 παρατηρούμε ότι η συνδυαστική μέθοδος KGARD-BM3D δίνει ένα πολύ καλό αποτέλεσμα.



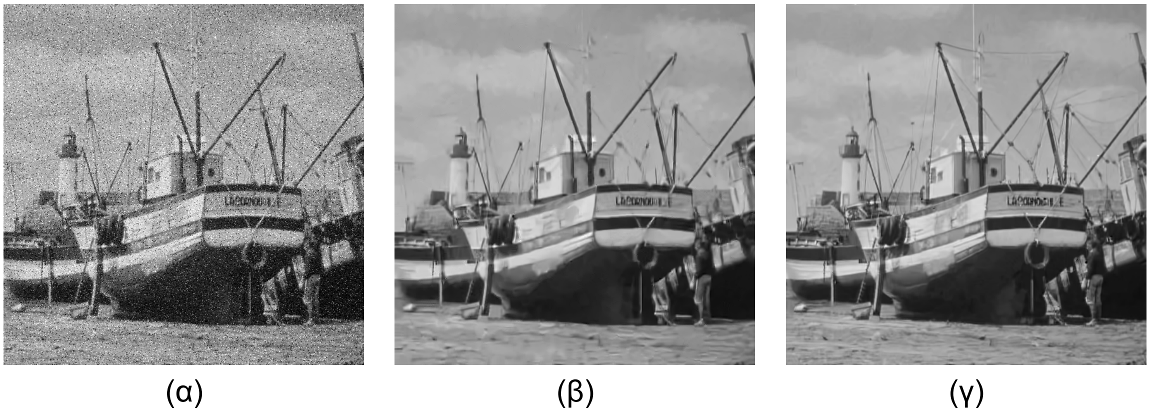
Σχήμα 5: (α): Η εικόνα της *Lena* με 20 dB Gaussian θορύβου και 10% ακραίες παρατηρήσεις (salt and pepper noise). (β): Αποθορύβωση με τη wavelet μέθοδο BM3D (PSNR=30.66 dB). (γ) Αποθορύβωση με τον αλγόριθμο KGARD (PSNR=31.94 dB). (δ) Αποθορύβωση με τον αλγόριθμο KGARD-BM3D (PSNR=33.81 dB).

Πίνακας 3: Αφαίρεση θορύβου από την εικόνα *Lena* για διάφορα επίπεδα Gaussian θορύβου και ποσοστά ακραίων παρατηρήσεων. Σύγκριση των μεθόδων BM3D, RB-RVM, KGARD και KGARD-BM3D.

Method	Parameters	Gaussian Noise	Impulses (± 100)	PSNR
BM3D	$s = 30$	25 dB	10%	30.84 dB
RB-RVM	$\sigma = 0.3$	25 dB	10%	31.25 dB
KGARD	$\sigma = 0.3$, $\lambda = 1$	25 dB	10%	33.49 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 10$	25 dB	10%	**35.67 dB**
BM3D	$s = 35$	20 dB	10%	30.66 dB
RB-RVM	$\sigma = 0.4$	20 dB	10%	29.09 dB
KGARD	$\sigma = 0.3$, $\lambda = 1$	20 dB	10%	31.94 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 15$	20 dB	10%	**33.81 dB**
BM3D	$s = 40$	15 dB	10%	29.94 dB
RB-RVM	$\sigma = 0.4$	15 dB	10%	25.85 dB
KGARD	$\sigma = 0.3$, $\lambda = 2$	15 dB	10%	28.47 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 25$	15 dB	10%	**30.77 dB**

Πίνακας 4: Αφαίρεση θορύβου από την εικόνα *boat* για διάφορα επίπεδα Gaussian θορύβου και ποσοστά ακραίων παρατηρήσεων, για τις μεθόδους BM3D και KGARD-BM3D.

Method	Parameters	Gaussian Noise	Impulses (± 100)	PSNR
BM3D	$s = 25$	25 dB	5%	30.57 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 10$	25 dB	5%	**34.61 dB**
BM3D	$s = 35$	20 dB	10%	28.97 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 15$	20 dB	10%	**31.52 dB**
BM3D	$s = 50$	20 dB	20%	27.49 dB
KGARD-BM3D	$\sigma = 0.4$, $\lambda = 1$, $s = 15$	20 dB	20%	**29.7 dB**



Σχήμα 6: (α): Η εικόνα *boat* με 20 dB Gaussian θορύβου και 10% ακραίες παρατηρήσεις. (β): Αφαίρεση θορύβου με τη μέθοδο BM3D (PSNR=28.97 dB). (γ): Αφαίρεση θορύβου με τη συνδιασμένη μέθοδο KGARD-BM3D (PSNR=31.52 dB).

Contents

1 Introduction 43

1.1 Regression in Machine Learning 43

1.2 Estimation in the Presence of Outliers 45

1.2.1 Robust Regression versus Outlier Diagnostics 46

1.2.2 Robust Regression via Sparse Modeling 47

1.3 Contributions and Novelty of the Thesis 48

1.4 Outline of the Thesis 50

2 Robust Methods for Linear Regression 51

2.1 Introduction 51

2.2 Least Squares and the Quest for Robust Methods 51

2.3 Robust Methods 55

2.3.1 Maximum Likelihood Type Estimates (M-est) 55

2.3.2 Robust Estimators with Higher Breakdown Point 58

3 Robust Linear Regression via Sparse Modeling: Greedy Methods and ℓ_1-norm Minimization 63

3.1 Introduction 63

3.2 Searching for Sparse Representations 64

3.2.1 Greedy Methods - Orthogonal Matching Pursuit (OMP) 66

3.2.2 Convex Relaxation to the ℓ_1-norm 66

3.3 Sparse Outlier Modeling for Robust Linear Regression 67

3.4 Related Works 68

3.4.1 Robust Denoising via the Minimization of the ℓ_1-norm 68

3.4.2 A Probabilistic Approach for the Robust Denoising Task 69

3.4.3 A Combined OMP Selection-based and M-est Method 70

4 Greedy Algorithm for Robust Denoising 73

4.1 Introduction 73

4.2 Greedy Algorithm for Robust Denoising (GARD) 73

4.3 Theoretical Analysis 76

4.3.1 General Results 76

4.3.2 The Presence of Outliers Only 77

4.3.3 The Presence of Both Inlier and Outlier Noise 84

4.4 Experiments 88

4.4.1 Mean-Square-Error (MSE) Estimation 90

4.4.2 Complexity Evaluation for Large Data Sets 92

4.4.3 Support Recovery Test 93

4.4.4 Evaluating the Probability of an Accurate Estimation 96

4.4.5 Experiments with Noise of More General Forms 97

5 Nonlinear Regression in RKHS and the Pursuit for Robustness 99

5.1 Introduction 99

5.2 Reproducing Kernel Hilbert Spaces (RKHS) 99

5.3 Properties of RKHS 101

5.3.1 Examples of Kernel Functions 102

5.3.2 Basic Theorems 107

5.4 Kernel Ridge Regression (KRR) 108

5.5 The Pursuit for Robustness - Robust Kernel Ridge Regression (RKRR) 110

5.6 Related Works 112

5.6.1 Convex Relaxation: Refined Alternating Directions Method of Multipliers (RAM) 112

5.6.2 Sparse Bayesian learning: Robust Relevance Vector Machine (RB-RVM) 114

6 Robust Nonlinear Regression: A Greedy Approach Employing Kernels 117

6.1 Introduction 117

6.2 Kernel Greedy Algorithm for Robust Denoising (KGARD) 117

6.2.1 Efficient Implementations 121

6.2.2 Further Improvements on KGARD's Performance 122

6.3 Theoretical Analysis 122

6.3.1 Non-singularity of the Inversion Matrix 123

6.3.2 Robust Enhancement via the Regularization Term 124
6.3.3 Identification of the Outliers for the Noiseless Case 125
6.4 Experiments 131
6.4.1 Testing the Recovery of the Sparse Outlier vector's Support 131
6.4.2 Evaluation of the Method: Mean-Square-Error (MSE) 133

7 Applications to Image Denoising 137
7.1 Introduction 137
7.2 Modeling the Image and the Noise 137
7.2.1 Dividing the Image into ROIs 138
7.2.2 Robust Modeling 138
7.2.3 Implementation 139
7.2.4 Parameter Selection 144
7.3 Experiments on Images Corrupted by Synthetic Noise 145

8 Summary of the Thesis and Conclusions 151
8.1 The Linear Regression Task 151
8.2 The Nonlinear Regression Task 153
8.3 Future Directions 154
8.3.1 Generalization of the Method with Noise in the Input Data 154
8.3.2 GARD in a Distributed Network Environment 154

Appendix 159

References 163

Index 171

List of Tables

2.1 Various types of *robust cost functions*, ρ, with their derivatives, $\psi = \rho'$, given. . 58

2.2 The respective weight functions. 58

4.1 Computational costs for the methods that deal with the linear regression task. For GARD, k is the number of times the algorithm identifies an outlier. For GARD and SOCP total complexity is given, while for the rest of the methods the total complexity depends on the number of iterations for convergence. For ROMP, many parameters are involved, thus its complexity is not readily computable in closed form. 89

4.2 Computed MSE, for various experiments. In tests A, B and C, the noise is drawn from the heavy-tailed distribution alpha-stable of Lévy distribution. In test D, noise consists of a sum of two vectors, drawn from 2 independent Gaussian distributions with different variance, plus an outlier noise vector of impulsive noise. 98

6.1 Percentage of correct and wrong indices that KGARD has selected, for outlier values $\underline{u}$ ranging from 50 to 1000. The correct support (second column) corresponds to true outliers (indices in $\mathcal{S}$), while the wrong support (third column) corresponds to locations which are wrongly classified as outliers (thus do not belong to $\mathcal{S}$). In the final column the minimum value $\underline{u}$ of outliers for which the support recovery condition is valid, is listed. 132

6.2 Computed MSE for $\underline{f}(x) = 20sinc(2\pi x)$ over the training and validation set. Additionally, the percentage of correct and wrong indices that each method has classified as outliers and the Mean Implementation Time (MIT), for various levels of inlier and outlier noise, are evaluated. 133

6.3 Performance evaluation for each method, where $\underline{f} \in \mathcal{H}$ is considered as a linear combination of a few kernels with the input data lying on the 1-dimensional space. The inlier noise is considered random Gaussian with $\sigma = 4$ and for various fractions of outliers we have computed: the training and validation MSE, the percentage of correct and wrong indices that each method has classified as outliers and the Mean Implementation Time (MIT). 134

6.4 Performance evaluation for each method, where $\underline{f} \in \mathcal{H}$ is considered as a linear combination of a few kernels with the input data lying on the 2-dimensional space. The inlier noise is considered white Gaussian with $\sigma = 3$ and for various fractions of outliers we have computed: the training and validation MSE,the percentage of correct and wrong indices that each method has classified as outliers and the Mean Implementation Time (MIT). 135

7.1 Denoising performed on the *Lena* image corrupted by various types and intensities of noise using the proposed methods, the robust RVM (RB-RVM) approach and the state-of-the-art wavelet method BM3D. 147

7.2 Denoising performed on the *boat* image corrupted by various types and intensities of noise using the state-of-the-art wavelet method BM3D with and without outlier detection. 149

7.3 Denoising performed on the *Barbara* image corrupted by various types and intensities of noise using the state-of-the-art wavelet method BM3D with and without outlier detection. 150

List of Figures

1.1 Given the training data (gray dots), the task of regression is to learn the underlying structure, i.e., the regression function f. In (a) the regression function is linear, while in (b) the regression function is nonlinear. 44

1.2 A data set contaminated by outliers (red dots). 45

1.3 The *Lena* image contaminated by outliers (salt and pepper noise). 49

2.1 Estimation performed for a data set of five points via the LS method. The blue line is the LS estimate for the outlier-free data set (gray points). A single outlier may cause the LS estimator to collapse. (a) The outlier in the y direction (green dot) affects the LS estimation (green dashed line), but not significantly. (b) The outlier in the x direction (red dot) greatly affects the estimation (red dashed line). The estimator is said to “break down” and thus the method is rendered unreliable. 54

2.2 Robust loss function graphs for various types. From left to right, the $\rho(r)$, the $\psi(r)$ and the $\mathrm{w}(r) = \psi(r)/r$ (weight) functions. 57

3.1 The function $\mathrm{f}(s) = |s|^p$ for various values of p. As $p \to 0$, f approaches the indicator function, which is 0 for $s = 0$ and 1 elsewhere. Convexity is retained for $p \geq 1$, while for $p < 1$ the epigraph is non-convex. 64

3.2 The 2-dimensional unit balls for the ℓ_p-norms: a) $p = +\infty$ (max-norm), b) $p = 2$, c) $p = 1$, d) $p = 0.5$ and e) $p = 0$ (ℓ_0-norm). As $p \to 0$ the unit sphere covers the axes with the exception of the center $(0, 0)$. Observe that for $p < 1$ the unit balls are non-convex. 65

4.1 (a), (c) and (e): The attained MSE versus the outlier fraction, for various dimensions of the unknown vector $\underline{\boldsymbol{\theta}}$. (b), (d) and (f): Log-scale of the Mean Implementation Time (MIT) versus the outlier fraction for each dimension of the unknown vector. For all dimensions, $N = 600$ observations are used. 91

4.2 (a): The attained MSE for the dimension $M = 100$ of the unknown vector, $\underline{\boldsymbol{\theta}}$, versus the fraction of the outliers, with their values drawn randomly from the set $[-50, -25] \cup [25, 50]$. (b): The respective Mean Implementation Time (MIT) in logarithmic scale. The number of data is $N = 600$. 92

4.3 Large scale complexity test for dimension of the unknown vector set at $M = 100$. While varying the number of observations, the MSE (top) and the Mean Implementation Time (MIT) in log-scale (bottom), is shown for each method. It is clear that GARD attains the lowest MSE, whilst being the most efficient. . . . 93

4.4 Recovery of the support and relation to the bound of δ_S, for the noiseless case. For outlier fraction of less than 14%, the bound for δ_S in (4.16), is guaranteed, hence the recovery is exact. 94

4.5 Recovery of the support and relation to the bound of δ_S, for the case inlier and outlier noise coexist. For an outlier fraction of less than 13%, the bound for δ_S (4.16) is guaranteed, hence the recovery of the support is exact, while the computed MSE is valid under the bound that inequality (4.47) suggests. 95

4.6 Recovery of the support for GARD, in the case where outlier and heavy inlier noise of approx. 20 dB coexist. Although, the support is not entirely recovered, the MSE is relatively low. 96

4.7 (a): The probability of recovery while varying the fraction of outliers, for the the dimension $M = 100$ of the unknown vector, $\underline{\boldsymbol{\theta}}$, and $N = 600$ observations. As the fraction of the outliers increases, the probability for an accurate estimation drops. (b): Transition from success to failure with probability $p = 0.5$. A vertical line at $M = 100$ indicates the percentage of outliers (for each method respectively) that correspond to the values of the x-axis for probability $p = 0.5$, in (a). 97

5.1 The mapping from the original low-dimensional input space $\mathcal{X}$ to a linear one in the high-dimensional RKHS $\mathcal{H}$. Employing the kernel trick, inner product operations are efficiently performed via function evaluations on the original low-dimensional space $\mathcal{X}$. 100

5.2 (a) The Gaussian kernel for $\mathcal{X} = \mathbb{R}$ and $\sigma = 1.5$. (b) The element $\phi(0) = \kappa(\cdot, 0)$ for different values of σ. 103

5.3 The element $\phi(0) = \kappa(\cdot, 0)$ of the Gaussian kernel for $\mathcal{X} = \mathbb{R}^2$ and for various values of the parameter σ. (a) $\sigma = 0.7$, (b) $\sigma = 1$, (c) $\sigma = 1.5$, (d) $\sigma = 2$. 104

5.4 The element $\phi(0) = \kappa(\cdot, 0)$ of the Laplacian kernel for $\mathcal{X} = \mathbb{R}^2$ and for various values of the parameter t. (a) $t = 0.5$, (b) $t = 1$, (c) $t = 1.5$, (d) $t = 2$. 105

5.5 The polynomial kernel for $\mathcal{X} = \mathbb{R}$: (a) inhomogeneous $d = 1, c = 3$, (c) inhomogeneous $d = 2, c = 3$ and (e) homogeneous $d = 2, c = 0$. (b), (d), (f) The respective element $\phi(x_0) = \kappa(\cdot, x_0)$ for various values of x_0. 106

5.6 (a) The Laplacian kernel for $\mathcal{X} = \mathbb{R}$ and $t = 1$. (b) The element $\phi(0) = \kappa(\cdot, 0)$ of for different values of t. 107

5.7 (a) Noisy data of 17 dB inlier noise and 10% outliers. (b) Black line: the uncorrupted data and red dashed line: the estimated data. Estimation is performed with the classic KRR method. It is clear that the performance of the estimation is greatly affected by the presence of outliers (MSE = 10.79). 111

6.1 The significance of robust estimation: (a) Data corrupted by both inlier and 10% of outlier noise. (b) The black and the red dashed lines correspond to the uncorrupted data and the non-robust estimation performed, respectively. The MSE over the training set is 10.79. (c) The black and the green dashed lines correspond to the uncorrupted data and the robust estimation performed with KGARD, respectively. The MSE over the training set is 1.21. 120

6.2 Percentage of the correct (green pointing up) and wrong (orange pointing down) indices that KGARD has selected, while varying the values $\pm\underline{u}$ of the outliers at the fixed fraction of 10%. Although the condition (6.16) is valid only for values greater than ±600 (and with high probability valid for values 400-599), the support of the sparse outlier vector has been correctly identified for much smaller values of outlier noise, too. 132

7.1 (a) A square $N \times N$ region of intest (ROI). (b) Rearranging the pixels of a ROI. 138

7.2 Two consecutive $N \times N$ ROIs. Observe that the two ROIs overlap. 140

7.3 (a) The algorithm has reached the right end of the image, hence it moves L pixels below. (b) In this example, $L = 8$, $N = 12$. The image has been padded using 2 pixels in all dimensions. The Figure shows the 8 first rROIs. 140

7.4 Histograms of the residual vectors used in the automatic computation of ϵ. . . . 142

7.5 The KGARD-BM3D denoising method. First the KGARD applied to the noisy image extracts the outliers and then the BM3D removes the remaining of the noise.144

7.6 (a) The *Lena* image corrupted by 20 dB of Gaussian noise and 10% outliers. (b) Denoising with BM3D (30.66 dB). (c) Denoising with KGARD (31.94 dB). (d) Denoising with joint KGARD-BM3D (33.81 dB). 148

7.7 (a) The *boat* image corrupted by 20 dB of Gaussian noise and 10% outliers. (b) Denoising with BM3D (28.97 dB). (c) Denoising with joint KGARD-BM3D (31.52 dB). 149

7.8 (a) The *Barbara* image corrupted by 20 dB of Gaussian noise and 10% outliers. (b) Denoising with BM3D (29.2 dB). (c) Denoising with joint KGARD-BM3D (30.43 dB). 150

Notation

The following mathematical notation has been used throughout the thesis. The symbols are summarized:

- Scalars are denoted with lowercase or uppercase italics, e.g., ε or N.
- The floor function of any non-negative real number x is denoted as $\lfloor x \rfloor$ and is equal to the largest integer less than or equal to x.
- Sets are denoted with capital calligraphic letters, e.g., $\mathcal{S}$, where $\mathcal{S}^c$ denotes the complement of $\mathcal{S}$ and $|\mathcal{S}|$ denotes its cardinality.
- Vectors are denoted with **boldface** lowercase letters, e.g., $\boldsymbol{\theta}$ (column vector). Moreover, the i-th element of vector $\boldsymbol{\theta}$ is denoted by θ_i.
- Matrices are denoted with **boldface** capital letters, e.g., $\boldsymbol{X}$, and the symbol $\cdot^T$ denotes the transpose of the respective matrix/vector. In addition, the j-th column of matrix $\boldsymbol{X}$ is denoted by $\boldsymbol{x}_j$ and the element of the i-th row and j-th column of matrix $\boldsymbol{X}$ by x_{ij}.
- Functions/operators are denoted with lowercase or uppercase roman letters (english - greek), e.g., f or F - ψ or Ψ, except for the expectation operator, where $\mathbb{E}[\cdot]$ is employed and the probability of a random variable, which is denoted by $p(\cdot)$.
- The operator supp($\boldsymbol{u}$) is reserved for the support set of the vector $\boldsymbol{u}$.
- The operator diag($\boldsymbol{a}$) denotes the respective square diagonal matrix (this matrix has the vector's coefficients on its *diagonal*, while all other entries are equal to zero).
- An arithmetic index in parenthesis, i.e., (k), $k = 0, 1, \ldots,$ is reserved to declare an iterative (algorithmic) process, e.g., on matrix $\boldsymbol{X}$ and vector $\boldsymbol{r}$ the iteratively generated matrix and vector are denoted by $\boldsymbol{X}_{(k)}$ and $\boldsymbol{r}_{(k)}$, respectively. Following this rationale, $r_{(k),i}$ is reserved for the i-th element of the iteratively generated vector $\boldsymbol{r}_{(k)}$.
- The identity matrix of dimension $N \times N$ will be denoted as $\boldsymbol{I}_N$ where $\boldsymbol{e}_j$ is its j-th column vector. The zero matrix of dimension $N \times K$ is denoted as $\boldsymbol{O}_{N\times K}$, while the vector of zero elements, for appropriate dimension, as $\mathbf{0}$.
- The matrix that comprises the columns of $\boldsymbol{X}$ whose indices belong to the ordered index set $\mathcal{S} = \{j_1, \ldots, j_S\}$ is denoted by $\boldsymbol{X}_{\mathcal{S}}$. As a special case, the columns of matrix $\boldsymbol{I}_N$ restricted over the set $\mathcal{S}$, as $\boldsymbol{I}_{\mathcal{S}}$. Moreover, for an augmented matrix of the form $\boldsymbol{A} = [\boldsymbol{B}\ \boldsymbol{C}]$ the notation $\boldsymbol{A}_{:|\mathcal{S}}$ is reserved for the restriction of columns over its second part only, i.e.,

matrix $\boldsymbol{C}$, over the set $\mathcal{S}$. For example, let $\boldsymbol{A} = [\boldsymbol{X}\ \boldsymbol{I}_N]$; the restriction implies that $\boldsymbol{A}_{:|\mathcal{S}} = [\boldsymbol{X}\ \boldsymbol{I}_{\mathcal{S}}]$. In other words, all the columns of the first matrix are included and only the columns of the second one are restricted. Finally, the submatrix that comprises rows and columns of $\boldsymbol{X}$ over the set, $\mathcal{S}$, is denoted by $\boldsymbol{X}_{\mathcal{S},\mathcal{S}}$.

- The notation $\boldsymbol{W}_{(\mathcal{S})}$ denotes the dependency of the matrix $\boldsymbol{W}$ on the given set, $\mathcal{S}$. Thus, it should be not confused with the restriction of its columns where no parenthesis is employed.

- The linear operator $\mathrm{F}_{\mathcal{S}}(\boldsymbol{v}) := \boldsymbol{I}_{\mathcal{S}}\boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{v}$ is used on a vector $\boldsymbol{v}$ of $\mathbb{R}^N$ over the index set $\mathcal{S} \subseteq \mathcal{J} = \{1, \dots, N\}$ and has identical coordinates with $\boldsymbol{v}$ in all indices of $\mathcal{S}$ and zero everywhere else. It is clear that for the sparse vector $\boldsymbol{u}$ with support set $\mathcal{S}$, leads to $\mathrm{F}_{\mathcal{S}}(\boldsymbol{u}) = \boldsymbol{I}_{\mathcal{S}}\boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{u} = \boldsymbol{u}$. Moreover, for the set $\mathcal{S}$ with $|\mathcal{S}| = S \leq N$, $\boldsymbol{v}_{\mathcal{S}}$ denotes the restriction of the vector $\boldsymbol{v} \in \mathbb{R}^N$ over the set. Thus, $\boldsymbol{v}_{\mathcal{S}} \in \mathbb{R}^S$ is a (lower dimensional) vector with entries the elements of $\boldsymbol{v}$ in indices that belong to the set $\mathcal{S}$; that is, $\boldsymbol{v}_{\mathcal{S}} = \boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{v}$. Hence, $\mathrm{F}_{\mathcal{S}}(\boldsymbol{v}) = \boldsymbol{I}_{\mathcal{S}}\boldsymbol{v}_{\mathcal{S}}$, directly follows.

Preface

This research has been co-financed by the European Union (European Social Fund – ESF) and Greek national funds through the Operational Program “Education and Lifelong Learning” of the National Strategic Reference Framework (NSRF) - Research Funding Program: Aristeia I - 621.

Chapter 1

Introduction

1.1 Regression in Machine Learning

Learning from data is not only at the heart of any scientific field, but it is also closely associated to what we call human intelligence. From our early days of birth, a large part of our life is devoted to (mostly empirical) learning. Besides, not only humans learn. Other entities, such as mammals, also rely on their ability to “learn” and adapt to their environment in order to survive.

In sciences, the need of humans to learn from data is what led to the development of different techniques that unveil the hidden structures and patterns associated with their generation mechanism. This information, in turn, provides us with an understanding of the nature of the data and paves the way to take actions or make *predictions* for the future. *Machine Learning* revolves around the development of efficient algorithms and techniques pointing in such direction. Although Machine Learning has its roots mainly in mathematical disciplines that have been established over centuries ago, it is only recently, over the last few decades, that has found widespread applications due to the advent and advances in computers. Nowadays, it comprises a major part in various disciplines such as Statistics, Pattern Recognition, Biostatistics, Signal and Image Processing, Computer Science, Industrial Automation and Computer-Aided Medical Diagnosis, just to name a few. The common and principal goal for any ML method is to use a computer (machine) in order to to first learn from the available data and then use the acquired “knowledge” to perform prediction.

At the heart of Machine Learning is the task of *regression* or *regression analysis*. In a classic regression task, given a set of training data, the goal is to learn a set of unknown parameters in order to make predictions. In simple words, the task could be seen as a curve fitting problem. Consider a set of training points $(y_i, \boldsymbol{x}_i)$, $y_i \in \mathbb{R}$ and $\boldsymbol{x}_i \in \mathbb{R}^M$ for $i = 1, \dots, N$. The task is to estimate a function, f, whose graph fits the data. The target function, f, of the independent variables, $\boldsymbol{x}$, is called the *regression function*. The difference between regression and classification is that in regression the dependent variable belongs to an interval in the real axis (or region in the complex plane), while in classification it is a discrete variable, see[1].

Regression analysis is widely used for prediction and forecasting. It is also used as a means to extract information concerning the degree of dependence among the dependent (output) and

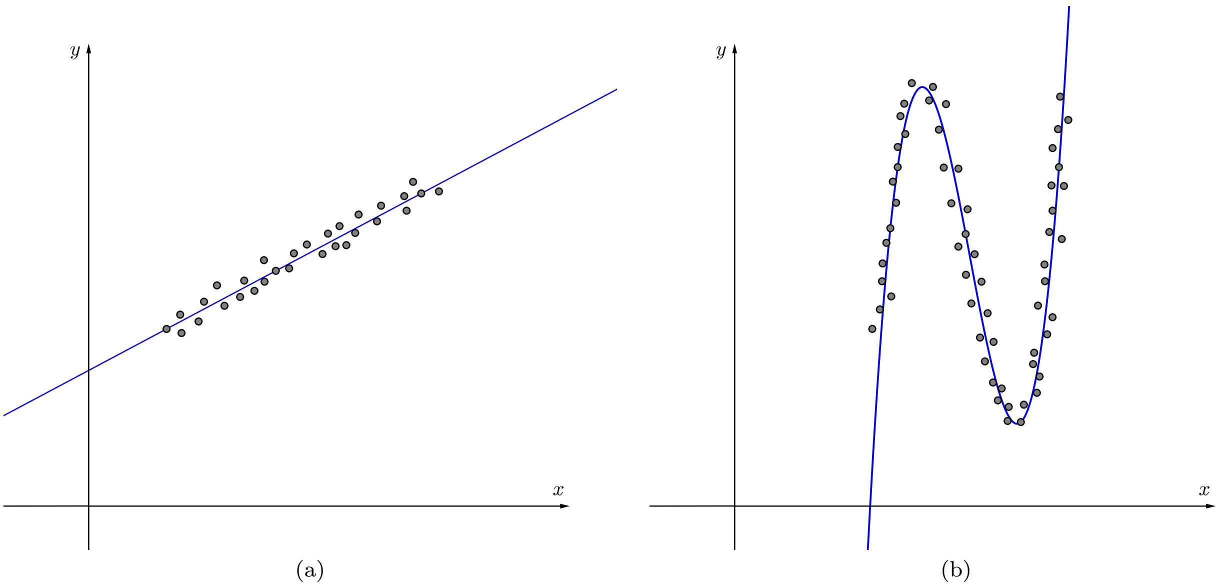


Figure 1.1: Given the training data (gray dots), the task of regression is to learn the underlying structure, i.e., the regression function f. In (a) the regression function is linear, while in (b) the regression function is nonlinear.

the independent (input) variables. Thus, useful information and related implications of such dependencies can be revealed.

The earliest form of regression was the method of Least Squares (LS), which was published by Legendre in 1805 and by Gauss in 1809, see [2], [3]. Legendre and Gauss both applied the method to the problem of determining the orbits of comets, based on astronomical observations. Gauss published a further development of the theory of the LS in 1821 including a version of the Gauss-Markov theorem, see [4].

Many techniques that perform regression analysis have been developed, since then. Familiar methods such as linear regression and ordinary Least Squares regression belong to the parametric class of learning techniques; that is, the model function is defined in terms of a finite number of unknown parameters that are estimated from the data, as demonstrated in Figure 1.1 (a). In contrast, nonparametric regression refers to techniques that bypass the need for explicit parameterization of the unknown functional dependence. For example, the regression function can be assumed to lie in a specific set of functions, which may also be infinite-dimensional. A popular example, that will be adopted in the current thesis, is to assume that the regression function lies in a Reproducing Kernel Hilbert Space (RKHS), as shown in Figure 1.1 (b).

The performance of regression analysis methods in practice depends on the form of the data generating process and how this relates to the regression model being used. Since the true form of the data-generating process is generally unknown, regression analysis often depends, to a large extent, on making assumptions concerning this process. Regression models, that are designed for prediction, are often useful even when the assumptions are moderately violated, although they may not perform optimally. However, if our goal is to make accurate predictions, we should look for a model/method that is *robust* enough, i.e., it can tolerate abnormalities on the data so that the estimation is not significantly affected.

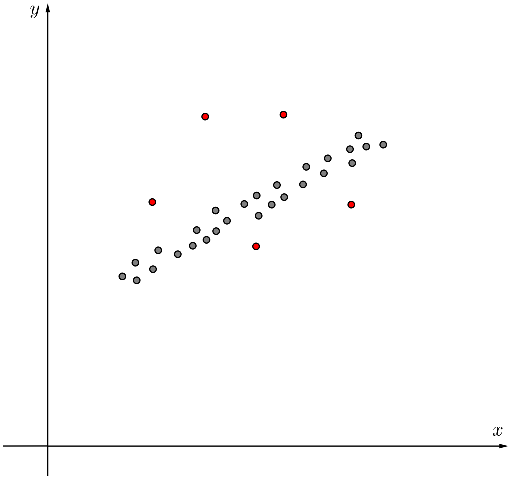


Figure 1.2: A data set contaminated by outliers (red dots).

1.2 Estimation in the Presence of Outliers

We have already pointed out that the accuracy of the estimation in a regression model strongly depends on the data generating process. Undoubtedly, the most common threat that reflects to the performance of an estimation method is the type of noise, that corrupts the data. For example, in the task of linear regression, the Least Squares (LS) estimator is optimal, in the Maximum Likelihood (ML) sense, if the noise is white Gaussian. Unfortunately, this is not the case if the noise follows other distributions, such as a heavy tailed one. Thus, it is not expected that a specific model will work under any circumstances. Indeed in some cases, where the data are contaminated by heavy-tailed noise, certain methods collapse. Thus, the direction we should take is to seek for robust methods, which match the problem at hand.

The notion of robustness, i.e., the efficiency of a method to solve a learning task from data under noise uncertainties of various types, has been a major issue in the scientific community for over half a century [5, 6]. Regardless the nature of the problem, e.g., classification or regression, the goal is to minimize the effect of the observations that have been corrupted by unexpected high values of noise, known as *outliers*. A typical example of observations that are contaminated by outliers is given in Figure 1.2. Surprisingly, no exact definition of an outlier exists, although many authors have attempted to give a definition over the years. A few typical characterizations are:

- “An outlier is an observation that deviates so much from other observations as to arouse suspicions that is was generated by a different mechanism” (Hawkins, 1980), [7].
- “An outlier is an observation which appears to be inconsistent with the remainder of the data set” (Barnet and Lewis, 1994), [8].
- “An outlier is an observation that lies outside the overall pattern of a distribution” (Moore and McCabe, 1999), [9].
- “An outlier in a set of data is an observation or a point that is considerably dissimilar or inconsistent with the remainder of the data” (Ramasmawy, 2000), [10].

- "Outliers are those data records that do not follow any pattern in an application" (Chen, 2002), [11, 12, 13].

Outliers are often regarded as erroneous measurements that deviate greatly from the rest of the observations. This is due to the fact that: either its values are heavily influenced by another source or that these observations are generated by a different distribution. The focus of this dissertation is the establishment of reliable estimates for the regression task in the presence of outliers, both in its parametric (linear) and non-parametric (nonlinear) formulations. Furthermore, emphasis is given on the application front in the context of the image denoising task.

In such tasks, classic estimators, e.g., the Least Squares, are known to fail to perform well, see [14]. This problem was originally addressed since the 1950s, in [5, 6] and it was actually solved more than a decade later by Huber, in [15, 14, 16, 17]. Eventually, this led to the development of a new field in Statistics, known as *Robust Statistics*. However, the need for development of robust estimators was not only limited within the Statistics scientific community. Similar tasks (involving robust estimators) emerged in the context of many fields such as Physics, Medicine, Biology, Engineering and Computer Science, to name a few.

1.2.1 Robust Regression versus Outlier Diagnostics

The variety of methods that have been developed to handle outliers can be classified into two major categories. The first one includes tools that rely on the use of *diagnostics*, whereas the second direction is based on *robust regression* methods. Diagnostics and robust regression have the same goals, only obtained in the opposite order. When using diagnostic tools, one first tries to delete the outliers and then to fit the "good" data by a common estimator, e.g., Least Squares. On the other hand, a robust regression (or regression analysis) method first fits the data, using a rough approximation, and then exploits the original estimation to identify the outliers as those points which possess large residuals. Both approaches have a long history in field of Robust Statistics.

Diagnostics are statistics generally based on classical estimates that aim at giving numerical or graphical clues for the detection of data points that "deviate" significantly from the assumed model. There is a considerable literature on outlier diagnostics, and a good outlier diagnostic is clearly better than doing nothing. However, these methods present two drawbacks. The first is that, they are in general not very reliable in detecting outliers. The other is that, once suspicious observations have been flagged, the actions to be taken with them remain the analyst's personal decision, and thus there is no objective way to establish the properties of the result of the overall procedure.

Methods developed under the robust regression framework attempt to device estimators that are not so strongly affected by outliers. The first great steps forward occurred in the 1960s, and the early 1970s with the fundamental work of John Tukey (1960, 1962), Peter Huber (1964, 1967) and Frank Hampel (1971, 1974). One of the pioneering research works at that time was the development of Huber's M-est, which is a fairly good estimator with relatively low computational requirements, see [18, 19, 14]. This is accomplished via the use of appropriate (robust) functions of the residual norm (instead of the square function), in order to penalize large values of the residual. The applicability of the new robust methods proposed by these researchers accelerated by the increased speed and accessibility of computers.

Another approach under the robust regression framework was the development of combinatorial optimization algorithms, such as Hampel's Least Median of Squares Regression (LMedS) [20], Rousseeuw's Least Trimmed Squares (LTS) [19, 20] and Fischler's - Bolles's RANdom SAmple Consensus (RANSAC) [21]. Combinatorial optimization methods seemed to display enhanced robustness at that time. However, they were never adopted by the community, due to the increased computational requirements. In contrast, the desire for low complexity efficient algorithms has constantly been rising. Nowadays, where the dimensionality of the data can be inherently large, combinatorial methods are prohibited.

1.2.2 Robust Regression via Sparse Modeling

The revolution of *sparse modeling* and optimization techniques, marked a major research path in the machine learning and signal processing communities, since the 2000s. The recent development of methods in the spirit of robust analysis owes a lot to the emergence of sparse optimization methods, during the past decade.

Sparsity-aware learning and related optimization techniques have been at the forefront of the research in signal processing, encompassing a wide range of topics, such as compressed sensing, signal denoising and approximation techniques, see [22, 23, 24, 25, 26, 1]. The attribute of sparsity seems to fit in many more applications than initially anticipated. Sparsity is closely related to sufficiency or economy of a representation, a mechanism that harmonizes with nature, which tends to be parsimonious. Despite the fact that similar techniques, such as the minimization of cost functions involving ℓ_1-norms, have been used since the 1970s, it is only recently that it has become the focus of attention of a massive volume of research in the context of compressed sensing. At the heart of this problem lies an underdetermined set of linear equations, which, in general, accepts an infinite number of solutions. Imposing sparsity, is interpreted as seeking for a solution where only a few of the unknown coordinates, which we attempt to estimate, are nonzero.

There are two major paths, towards modeling sparse vectors/signals. The first one focuses on minimizing the ℓ_0(pseudo)-norm of a vector, which equals the number of its nonzero coordinates. However, since this is a non-convex optimization task, approximate methods have been established. The family of algorithms that have been developed to address problems involving the ℓ_0 (pseudo)-norm, comprises *greedy* methods, which have been shown to provide the solution of the related minimization task, under certain reasonable assumptions, [27, 28, 29, 30, 31]. Even though, in general, this is an NP-Hard task, it has been shown that such methods can efficiently recover a solution in polynomial time. On the other hand, the family of algorithms developed around the methods that employ the ℓ_1-norm, embraces convex optimization, providing a broader set of tools and stronger guarantees for convergence [29, 22, 32, 23, 33, 1]. Both methods have been shown to generate sparse solutions.

A more recent application of sparse modeling and optimization methods, which is also the focus of this work, is that of signal denoising. There, one is interested in recovering the original signal, which apart from the standard inlier noise, e.g., Gaussian, has also been corrupted by outliers. The key to this modeling is to assume that the outliers comprise only a small fraction of the entire data set, thus the outlier vector is modeled as a sparse one.

1.3 Contributions and Novelty of the Thesis

The scientific contributions that appear in the present work exhibit a novel approach to the robust regression task. Our first contribution is the development of a greedy scheme that can be used for the task of robust linear regression. The proposed algorithm is called *Greedy Algorithm for Robust Denoising* (GARD) and is published in [34, 35]. Following this path, an efficient estimator for the task of robust nonlinear regression is proposed, where it is also assumed that the (unknown) nonlinear function lies in a Reproducing Kernel Hilbert Space (RKHS). The estimator, called *Kernel Greedy Algorithm for Robust Denoising* (KGARD), is published in [36, 37, 34, 38]. Both of these methods are based on one among the most popular greedy schemes; that is, the Orthogonal Matching Pursuit (OMP). However, they exhibit many dissimilarities, due to the specific nature of the two regression tasks (parametric for the linear task vs nonparametric for the nonlinear).

For the task of linear regression the main results are: (a) the proof of convergence for the GARD scheme in a finite number of steps, (b) the establishment of a bound based on the Restricted Isometry Property (RIP) constant, see [39, 40], which guarantees that the proposed robust scheme (after convergence) successfully identifies the sparse outlier vector's support for the case where the data are contaminated by outlier and bounded inlier noise and (c) the establishment of performance bounds on the approximation. It should be noted that, (b) is a result that has been derived for the first time in the robust regression framework. Finally, the case where only outliers are present (no inlier noise) is also treated separately. In such a case, it has also been proven that GARD succeeds to recover both the exact regression solution and the sparse outlier vector, under the existence and uniqueness conditions. Although this result is mostly of theoretical importance, it justifies the reasons of the method's obtained performance. It should also be noted that, in the early years at the respective literature of linear regression, there has been an argument about the adequacy of the Least Squares estimator to detect outliers. As a matter of fact, this assessment was valid, although it was not clear when or why the detection could fail. Within this work, via the bounds based on the RIP condition, we believe to have shed some further light into this matter. The derived bounds are strong and unveil the hidden geometrical structures, which enables us to perform a detection based on the residual of the Least Squares estimate.

In the sequel, a significant amount of effort was invested on the performance evaluation of KGARD for the task of robust nonlinear regression employing kernels. Typically, by assuming that the function to be estimated lies in an RKHS, we resort to simple manipulations by replacing the regression matrix with a kernel one. However, since this is a nonparametric estimation task, the proposed algorithm had to be modified again (with respect to GARD). The proposed scheme alternates between a Kernel Ridge Regression (KRR) task and an OMP-like selection step. The addition of a regularization term at the estimation steps was inevitable and eventually led to a more complex theoretical analysis for the method. Thus, a different path, than the previously reported one for the linear case, had to be followed. The obtained results are: (a) the proof of the method's convergence in a finite number of steps, and (b) the establishment of a bound on the maximum singular value of the kernel matrix, which guarantees that the method identifies the correct locations of all the outliers, first. The analysis has been carried out for the case where only outliers exist in the noise. However, as demonstrated in the experiments, the method manages to recover the correct support of the sparse outlier vector in many cases where the

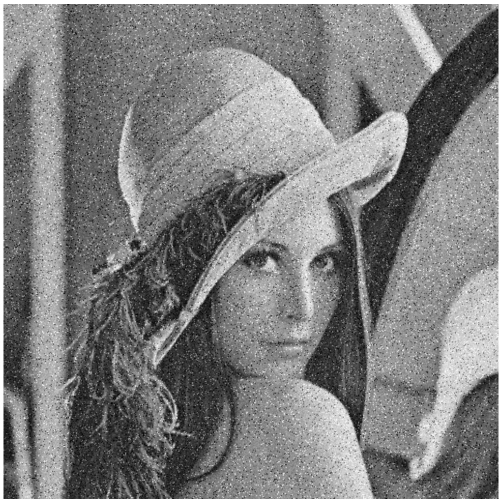


Figure 1.3: The *Lena* image contaminated by outliers (salt and pepper noise).

theoretical result does not hold. This leads to the conclusion that the provided conditions can be loosen up significantly in the future. Moreover, in practice, where inlier noise also exists, the method succeeds to correctly identify the majority of the outliers. The reason that the analysis is carried out for the case where inlier noise is not present is due to the fact that the analysis gets highly involved. The absence of the inlier noise makes the analysis easier and it highlights some theoretical aspects on why the method works. It must be emphasized that, such a theoretical analysis appears for the first time in the related bibliography.

Finally, a lot of effort was invested in the application of KGARD to the task of image denoising. In Figure 1.3 the popular image of *Lena*, which is widely used in the field of image processing since 1973, is corrupted by Gaussian noise plus salt and pepper noise (white and black pixels). The task of image denoising resorts to successfully removing the noise from the image. However, since salt and pepper noise is regarded as outliers, the task requires a more careful handling. Typical methods that have been already proposed to address the image denoising task include: (a) the wavelet-based image denoising methods, which have dominated the research in recent years [41, 42, 43], (b) methods based on Partial Differential Equations [44], (c) neighborhood filters and (d) methods of nonlinear modeling using local expansion approximation techniques, [45]. The majority of these methods assume a specific type of noise model and work based on this assumption. In fact, most of them require some sort of a priori knowledge of the noise distribution. In contrast to this approach, the more recently introduced denoising methods based on kernel ridge regression (KRR) make no assumptions on the underlying noise model and thus they can effectively treat more complex models, see [46]. Our contribution is the application of the KGARD algorithmic scheme to the image denoising task, for cases where the noise model includes outliers. In particular, two different denoising methods that deal with this type of noise are proposed. The first one is directly based on KGARD algorithmic scheme. The second method splits the denoising procedure into two parts: the identification and removal of the impulses, which is first carried out via the KGARD, and finally the removal of the remaining component from the intermediate output via a cutting edge wavelet based denoising method. The obtained denoising results (measured in PSNR) of the second method demonstrate the superior performance of the proposed scheme, which is based on the combination of the KGARD and a popular wavelet-based one.

1.4 Outline of the Thesis

The remaining chapters of the thesis are organized as follows.

In Chapter 2 a brief overview of the Least Squares (LS) method, with an emphasis on the associated problematic estimation in the presence of outliers, is given. Moreover, the challenges that arise in an attempt to detect a possible outlier via the LS residual are discussed. Finally, some classical methods on robust estimation are discussed.

Chapter 3 presents an overview of the basics of sparse modeling. Next, the two major paths, that lead to sparse solutions/representations, are discussed; that is, the greedy methods and the ℓ_1-norm minimization ones. In the context of sparse modeling, the respective tasks of robust linear regression are formulated and recently established methods are also presented.

In Chapter 4 the proposed robust scheme for linear regression, i.e., GARD, is introduced. An analysis of the scheme follows and additional tools for optimizing the method are also provided. Next, the method's theoretical study is provided, which is where the main results of this work are included. Moreover, extensive experiments that verify the overall advantages of the proposed scheme against other competitive methods are performed.

Chapter 5 departs from the linear regression task and proceeds to the nonlinear one. The study is performed in the framework of the Reproducing Kernel Hilbert Spaces (RKHS), the basics of which are discussed at the beginning of the chapter. Next, the classic Kernel Ridge Regression (KRR) task is reviewed and the problems that arise when performing an estimation in the presence of outliers are stated. Finally, recently established robust methods are also presented.

In Chapter 6 the novel robust scheme, i.e., Kernel Greedy Algorithm for Robust Denoising (KGARD), is introduced. The properties of the method are presented and a theoretical analysis is also provided, in terms of convergence and the method's capability in identifying the outliers, followed by an extensive set of experiments performed with synthetic data.

The objective of Chapter 7 is to present some applications of the proposed method, i.e., KGARD, in the context of image denoising. To this end, the method has been slightly modified and adapted to the task, so that no tuning parameters are involved; instead, the parameters are automatically tuned by the method. As a result, two new parameter-free methods are proposed for the task of robust denoising: a) a direct KGARD implementation that can perform the estimation and b) a KGARD scheme combined with a popular wavelet method, which performs first the identification and estimation of the outliers and then it removes the remaining of the noise.

Finally, Chapter 8 provides a summary and the conclusions of the research work in the context of the thesis and outlines some possible future research directions.

Chapter 2

Robust Methods for Linear Regression

2.1 Introduction

Our goal in this chapter is to introduce the concept of *robustness*. Initially, since the Least Squares (LS) estimator is not a robust one, we turn our attention to the LS residual as a means to identify outliers. From the provided analysis, it turns out that certain general assumptions should be imposed on the input data matrix. Moreover, the basic challenges associated with the task of robust estimation are discussed.

In this chapter, the basic robust regression method for the linear task is reviewed; that is, the core of the Maximum Likelihood type estimates (M-est), for various robust cost functions. Finally, since the M-est is primarily used for robust estimation with noise on the output data, other methods that deal with noise on the input data are also discussed. These are: the Generalized M-est (GM-est), the Least Median of Squares (LMedS), the Least Trimmed Squares and the Random Sample Consensus (RANSAC). However, due to their heavy computational requirements, there are only given indicatively.

2.2 Least Squares and the Quest for Robust Methods

In a typical linear regression task, we are interested in estimating the linear relation between two variables, $\boldsymbol{x} \in \mathbb{R}^M$ and $y \in \mathbb{R}$, i.e., $y = \boldsymbol{x}^T\boldsymbol{\theta}$, when several noisy instances are known. To this end, given a training set, $\mathcal{D} = \{(y_i, \boldsymbol{x}_i)\}_{i=1}^N$, we adopt the following regression modeling

$$y_i = \boldsymbol{x}_i^T\underline{\boldsymbol{\theta}} + \nu_i, \ i = 1, ..., N, \tag{2.1}$$

where ν_i is some observation noise. Hence, our goal is to estimate $\underline{\boldsymbol{\theta}} \in \mathbb{R}^M$ from the given training dataset of N observations. In matrix notation, (2.1) can be written as follows:

$$\boldsymbol{y} = \boldsymbol{X}\underline{\boldsymbol{\theta}} + \boldsymbol{\nu}, \tag{2.2}$$

where $\boldsymbol{y} = (y_1, \dots, y_N)^T$, $\boldsymbol{\nu} = (\nu_1, \dots, \nu_N)^T$ and $\boldsymbol{X} = [\boldsymbol{x}_1, \dots, \boldsymbol{x}_N]^T \in \mathbb{R}^{N \times M}$.

Applying a regression estimator to the given data set $\mathcal{D}$ yields

$$\hat{\boldsymbol{\theta}} = \begin{pmatrix} \hat{\theta}_1 \\ \vdots \\ \hat{\theta}_M \end{pmatrix}, \tag{2.3}$$

where the estimates $\hat{\theta}_i$ are called the *regression estimates*. Thus, the predicted/estimated value of the observation y_i is given by $\hat{y}_i = \boldsymbol{x}_i^T \hat{\boldsymbol{\theta}}$ and the *residual* value, between the observation and the estimate, is given by:

$$r_i = y_i - \hat{y}_i. \tag{2.4}$$

The most popular regression estimator is the Least Squares (LS) one (see [2], [1]) and corresponds to the estimate given by:

$$\hat{\boldsymbol{\theta}}_{LS} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{i=1}^{N} r_i^2 = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left\| \boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta} \right\|_2^2. \tag{2.5}$$

The respective LS regression estimate is obtained from

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\boldsymbol{\theta}}_{LS} = \boldsymbol{X}\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{y} = \boldsymbol{H}\boldsymbol{y}, \tag{2.6}$$

and the corresponding LS residual is

$$\boldsymbol{r} = \boldsymbol{y} - \hat{\boldsymbol{y}} = \boldsymbol{y}(\boldsymbol{I}_N - \boldsymbol{H}), \tag{2.7}$$

where

$$\boldsymbol{H} := \boldsymbol{X}\left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T. \tag{2.8}$$

As it is common in regression analysis, we consider that the number of observations exceeds the number of unknowns, i.e., $N > M$. Moreover, in order to obtain a unique solution, we should assume that $\boldsymbol{X}$ is a full rank matrix, i.e., $\operatorname{rank}(\boldsymbol{X}) = M$. For the case where $N < M$ (underdetermined system of linear equations[1]), an additional constraint/condition should be imposed, if one wishes to recover the vector of the unknowns.

The matrix $\boldsymbol{H}$ in (2.8) is often called "hat matrix" (since it puts a hat ($\hat{\boldsymbol{y}}$) on the vector of observations) and plays an important role to the task of robust estimation, as it will be demonstrated later. $\boldsymbol{H}$ depends on the design matrix, $\boldsymbol{X}$, and it is a symmetric $N \times N$ projection matrix, i.e., $\boldsymbol{H}^2 = \boldsymbol{H}$. Moreover, it has M eigenvalues equal to 1 and $N - M$ eigenvalues equal to 0. Thus,

$$\operatorname{tr}(\boldsymbol{H}) = \sum_{i=1}^{N} h_{ii} = M, \tag{2.9}$$

and hence the average value for h_{ii}'s is $\bar{h} = M/N$. We will return to the properties of the hat matrix, after a more in depth discussion on the properties of the LS estimator.

The LS estimator satisfies some important properties, under certain assumptions on the statistical nature of the noise samples ν_i. Assuming that the noise vector $\boldsymbol{\nu}$ is zero mean and independent on the input data leads to the fact that the LS estimator is unbiased. Furthermore, by assuming that the source generating the noise samples is white, i.e., $\mathbb{E}[\boldsymbol{\nu}\boldsymbol{\nu}^T] = \sigma_\nu^2 \boldsymbol{I}_N$, the following properties hold for the estimator:

[1] An infinite number of solutions exist.

- Its covariance matrix tends asymptotically to zero.
- It is the best linear unbiased estimator (BLUE).

Moreover, under the assumption of white Gaussian noise the estimator becomes minimum variance unbiased estimator (MVUE). This a strong result, which guarantees that, under the assumption of additive white Gaussian noise (AWGN), no other unbiased estimator will do better than the LS one, see [1]. However, this is not the case in the presence of outliers or when the noise distribution exhibits long tails. It is well known that even a single outlier can cause the LS estimator to collapse, see [19], [18], [16, 17]. Let us now provide a more in-depth investigation of this issue.

In the following a classic measure of robustness is presented.

Definition 2.1. *Consider a data set $\mathcal{D} = \{(y_i, \boldsymbol{x}_i)\}_{i=1}^N$ and let* T *be a regression estimator such that* $\mathrm{T}\left(\mathcal{D}\right) = \hat{\boldsymbol{\theta}}$. *Next, consider all possible corrupted samples $\mathcal{D}'$ that are obtained by replacing any K of the original N data points by arbitrary values and let*

$$\text{bias}(K; \mathrm{T}, \mathcal{D}) := \sup_{\mathcal{D}'} \left\|\mathrm{T}(\mathcal{D}') - \mathrm{T}(\mathcal{D})\right\|_2 .$$

The finite sample breakdown point of the estimator T *at the sample $\mathcal{D}$ is defined as*

$$\varepsilon_N(\mathrm{T}, \mathcal{D}) := \min\left\{\frac{K}{N} : \text{bias}(K; \mathrm{T}, \mathcal{D}) \textit{ is infinite}\right\}. \tag{2.10}$$

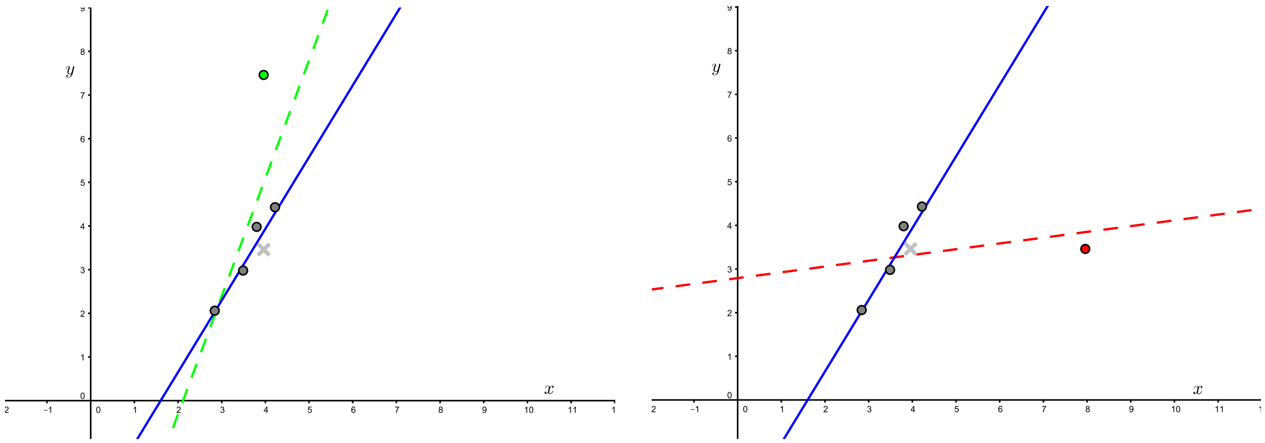
The Definition 2.1 states that if bias$(K; \mathrm{T}, \mathcal{D})$ is infinite, then K outliers can have an arbitrarily large effect on the estimator T, which is expressed by saying that the estimator “breaks down”. In other words, $\varepsilon_N(\mathrm{T}, \mathcal{D})$ is the smallest fraction of contamination that can cause the estimator T to take on values arbitrarily far from $\mathrm{T}(\mathcal{D})$. For the LS estimator the breakdown point is

$$\varepsilon_N(\mathrm{T}, \mathcal{D}) = \frac{1}{N}. \tag{2.11}$$

Thus, it can be said that the LS estimator is very sensitive to outliers and hence it is not advised to rely on estimations performed by the LS method whenever the the data set is suspicious to outliers. However, as it is shown in the following analysis, it seems that this holds mainly for the case where the input matrix has certain characteristics or if the input data is also contaminated with outliers. The sensitivity of the LS estimator to a single outlier, in either x or y direction, is demonstrated in Figure 2.1.

Since the LS estimator cannot provide any reliable solutions, the question raised is whether outliers can be identified by looking at the LS residuals in (2.7) or not. At first glance, this seems natural. One would expect that outliers in certain locations would produce large (positive or negative) residuals, so that they could easily be detected/diagnosed from (2.7). Unfortunately, this is not entirely true. The i-th element of the residual for the LS estimator can be viewed as

$$r_i = (1 - h_{ii})y_i - \sum_{\substack{j=1 \\ j\neq i}}^{N} h_{ij} y_j. \tag{2.12}$$



(a) (b)

Figure 2.1: Estimation performed for a data set of five points via the LS method. The blue line is the LS estimate for the outlier-free data set (gray points). A single outlier may cause the LS estimator to collapse. (a) The outlier in the y direction (green dot) affects the LS estimation (green dashed line), but not significantly. (b) The outlier in the x direction (red dot) greatly affects the estimation (red dashed line). The estimator is said to “break down” and thus the method is rendered unreliable.

From Equation (2.12), it is evident that the diagonal of the hat matrix contains extremely useful information ([18], [17]). More importantly, it characterizes whether or not an outlier in the observations is detectable via the LS residual. By employing some of the properties of the hat matrix, we also have

$$h_{ii} = h_{ii}^2 + \sum_{\substack{j=1 \\ j \neq i}}^{N} h_{ij}^2, \text{ for } i = 1, \dots, N. \tag{2.13}$$

Also observe in (2.13) that its diagonal elements, i.e., h_{ii}, satisfy

$$0 \leq h_{ii} \leq 1, \tag{2.14}$$

see [18]. Also, if $h_{ii} \to 0$ then $h_{ij} \to 0$ for all j. Thus, it follows directly from (2.12) that it is safe to decide on whether an observation y_i is contaminated by an outlier or not. On the other hand, if $h_{ii} \to 1$ notice that $h_{ij} \to 0$ for all $j \neq i$ in (2.13), which according to (2.12) leads to $r_i \to 0$. Hence, the evaluation can be misleading, due to the fact that the residual of the i-th observation is very small. Points with large h_{ii} are by definition *leverage points* and trigger effects known as *masking* and *swamping* of the outliers, see [18], [19], [20], [47]. However, the limits are not always indicative of when h_{ii} is considered large. Due to the established properties of the hat matrix, many authors suggest that a reasonable rule of thumb is if $h_{ii} > 2\bar{h} = 2M/N$, see [48]. However, this is only a rough estimate. Finally, the ratio $\bar{h} = M/N$ indicates that the dimension of the unknowns and the number of data should be distant. If the number of data (for a fixed dimension of unknown parameters) is insufficient, all of the elements in the diagonal of the hat matrix acquire large values (since $M \simeq N$) and the process of the estimation is meaningless.

Notice in Figure 2.1 (b) that the outlier in the x direction (red dot) produces a relatively small residual, whilst other samples (gray dots) that correspond to outlier-free data are now considered as outliers. Of course, this is something that could also occur with an outlier in the y direction for a different data sample. On the other hand, this effect is more dominant in the former case and requires a different handling. In the following chapters, only the case where outliers occur in the outputs (y values) is considered. Thus, for some methods (with minimal breakdown point) many authors assume that the diagonal elements of the hat matrix $\boldsymbol{H}$ are uniformly small, i.e.,

$$\max_{1\leq i\leq N} h_{ii} = h << 1. \tag{2.15}$$

However, the analysis based on (2.12) is not very informative. Since $h_{ii} \rightarrow 1$ is an extreme case, it does not provide any answers in the case, for example, $h_{ii} \simeq 0.5$. Is the identification still unreliable or can it be trusted depending on other factors too? As it will be shown in Chapter 4, the entire analysis on this issue can be established from a different point of view.

At this point, it should also be stated that, although the breakdown point provides a measure of robustness, it is a very rough one. For example, it does not provide us with any information on the number of outliers that a method can handle if no leverage points exist. To this end, the breakdown point is not employed in the following chapters.

Remark 2.1. *If outlier noise errors exist also in the inputs, i.e., $\boldsymbol{x}_i$'s, it is likely that the condition* (2.15) *is violated. Thus, in such case, a different handling/model is required. Hence, from here on, only the case where outliers occur in the outputs (y values) is considered. Besides, in an outlier-free data set, even if inlier noise errors occur in the inputs as well as the outputs, the Total Least Squares (TLS) is known to outperform the ordinary LS approach [1], [49].*

2.3 Robust Methods

We have already discussed that the LS estimator is not a robust one. Thus, an alternative path is to replace the squared-error loss function with a different cost function. A step forward towards another estimator came from Edgeworth in [50] back in 1887, where he proposed the Least Absolute Values (LAV) regression estimator, which is determined by

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} |r_i| \,.$$

This technique is often referred to as the ℓ_1-regression. However, the estimator is not unique and also, it does not profit a better breakdown point than the LS estimator. The replacement of the squared-loss function for the residuals by other robust costs is what finally led to the development of the entire field of Robust Statistics.

2.3.1 Maximum Likelihood Type Estimates (M-est)

The concept of using a robust function of the residuals in order to perform the minimization is known as the task of robustizing the LS approach and it is attributed to Huber, back in 1973. The Maximum Likelihood type estimates (M-est) are very flexible and they generalize in a straightforward way to multiparameter problems. In the following, an overview of the M-est is given and various types of robust loss functions are also listed.

Assuming a symmetric function ρ, i.e., $\rho(-t) = \rho(t)$ for all t with a unique minimum at zero, the M-est approach attempts to perform the following minimization:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \rho(r_i) = \sum_{i=1}^{N} \rho \left(y_i - \sum_{j=1}^{M} x_{ij}\theta_j \right), \tag{2.16}$$

or after taking derivatives, solves

$$\sum_{i=1}^{N} \psi(r_i)\boldsymbol{x}_i = \mathbf{0}, \tag{2.17}$$

where $\psi = \rho'$. The common approach is to assume that the ρ function is convex, so that the two approaches, i.e., (2.16) and (2.17), are equivalent. Moreover, a scaling parameter[2], $\hat{\sigma}$, is often used so that the residuals are standardized, see [19]. This is accomplished by defining the robust function as $\rho := \rho\left(r_i/\hat{\sigma}\right)$, which leads to the scaled version of M-est:

$$\sum_{i=1}^{N} \psi\left(\frac{r_i}{\hat{\sigma}}\right) \boldsymbol{x}_i = \mathbf{0}. \tag{2.18}$$

Next, by defining $\mathrm{w}(r) = \psi(r)/r$, and $w_i = \mathrm{w}\left(r_i/\hat{\sigma}\right)$, the set of normal equations is cast as

$$\sum_{i=1}^{N} w_i r_i \boldsymbol{x}_i = \mathbf{0}, \tag{2.19}$$

which is the basic version for the M-est. However, the method still remains vulnerable to leverage points. Besides, this is the reason that led Huber in [18] to the assumption of (2.15) for the M-est.

The robust cost function ρ in (2.16) can be selected from a list of various types of functions. The most significant ones are listed in Table 2.1 with their respective derivatives given. Moreover, in Table 2.2 the respective weight functions are also provided. Finally, in Figure 2.2, all the respective plots are depicted for the 2-dimensional case.

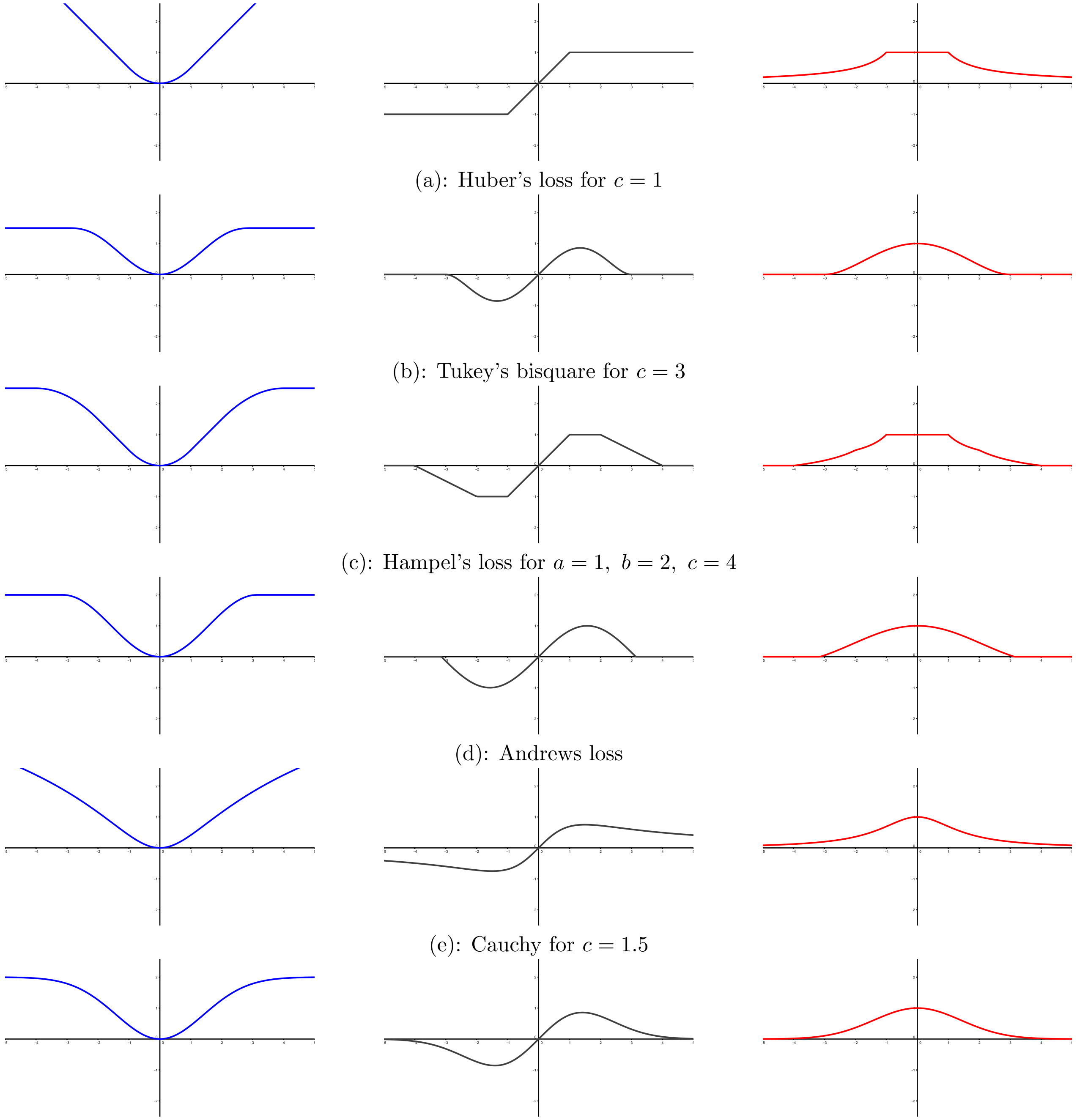
An alternative way to interpret the M-est in (2.19), is by solving a Weighted Least Squares (WLS) task, i.e.,

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} w_i r_i^2 \Leftrightarrow \min_{\boldsymbol{\theta}} \left\| \boldsymbol{W}^{1/2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}) \right\|_2^2, \tag{2.20}$$

where the diagonal weight matrix $\boldsymbol{W}$ assigns the weights, whose values depend on the robust selected function, e.g., in Table 2.2. In simple words, it penalizes large residuals in order to minimize their effect on the solution. However, the weighted LS task is more general, since other weights (not only robust weights) can be used too, depending on the application. Also, notice that for $\boldsymbol{W} = \boldsymbol{I}_N$ in (2.20), the scheme resorts to the ordinary Least Squares solution. The task in (2.20) is solved by the so called *Iteratively Reweighted Least Squares* (IRLS) algorithmic scheme, with updates given by:

$$\hat{\boldsymbol{\theta}}_{(k)} = \left(\boldsymbol{X}^T \boldsymbol{W}_{(k-1)} \boldsymbol{X}\right)^{-1} \boldsymbol{X}^T \boldsymbol{W}_{(k-1)} \boldsymbol{y}, \tag{2.21}$$

[2] The scale parameter corresponds to a robust function with respect to the values of the residual vector. Usually, the normalized mean absolute deviation is employed, i.e., $\mathrm{MADN}(\boldsymbol{x}) := \mathrm{Med}\left(|\boldsymbol{x} - \mathrm{Med}\left(\boldsymbol{x}\right)|\right)/0.675$. This dispersion estimate offers a measure similar to what the standard deviation is to the normal distribution.



(a): Huber's loss for $c = 1$

(b): Tukey's bisquare for $c = 3$

(c): Hampel's loss for $a = 1,\ b = 2,\ c = 4$

(d): Andrews loss

(e): Cauchy for $c = 1.5$

(f): Welsch for $c = 2$

Figure 2.2: Robust loss function graphs for various types. From left to right, the $\rho(r)$, the $\psi(r)$ and the $\mathrm{w}(r) = \psi(r)/r$ (weight) functions.

Table 2.1: Various types of *robust cost functions*, ρ, with their derivatives, $\psi = \rho'$, given.

Estimator type	Function $\rho(r)$	Function $\psi(r)$
Huber's	$\begin{cases} \frac{1}{2}r^2, & \text{for } \lvert r\rvert \le c \\ c\lvert r\rvert - \frac{1}{2}c^2, & \text{for } \lvert r\rvert > c \end{cases}$	$\begin{cases} r, & \text{for } \lvert r\rvert \le c \\ c\,\mathrm{sgn}(r), & \text{for } \lvert r\rvert > c \end{cases}$
Tukey's bisquare	$\begin{cases} (c^2/6)[1 - \left(1-(r/c)^2\right)^3], & \text{for } \lvert r\rvert \le c \\ c^2/6, & \text{for } \lvert r\rvert > c \end{cases}$	$\begin{cases} r[1-(r/c)^2]^2, & \text{for } \lvert r\rvert \le c \\ 0, & \text{for } \lvert r\rvert > c \end{cases}$
Hampel's	$\begin{cases} \frac{1}{2}r^2, & \text{for } \lvert r\rvert \le a \\ a\lvert r\rvert - a^2/2, & \text{for } a < \lvert r\rvert \le b \\ (a/2)[2b - a + (\lvert r\rvert - b)(1 + \frac{c-\lvert r\rvert}{c-b})], & \text{for } b < \lvert r\rvert \le c \\ \frac{a}{2}(b - a + c), & \text{for } \lvert r\rvert > c \end{cases}$	$\begin{cases} r, & \text{for } \lvert r\rvert \le a \\ a\,\mathrm{sgn}(r), & \text{for } a < \lvert r\rvert \le b \\ a\,\mathrm{sgn}(r)(c - \lvert r\rvert)/(c-b), & \text{for } b < \lvert r\rvert \le c \\ 0, & \text{for } \lvert r\rvert > c \end{cases}$
Andrews	$\begin{cases} 1 - \cos r, & \text{for } \lvert r\rvert \le \pi \\ 2, & \text{for } \lvert r\rvert > \pi \end{cases}$	$\begin{cases} \sin r, & \text{for } \lvert r\rvert \le \pi \\ 0, & \text{for } \lvert r\rvert > \pi \end{cases}$
Cauchy	$\frac{c^2}{2}\ln[1 + (r/c)^2]$	$\frac{r}{1+(r/c)^2}$
Welsch	$\frac{c^2}{2}\left(1 - e^{-(r/c)^2}\right)$	$re^{-(r/c)^2}$

Table 2.2: The respective weight functions.

Estimator type	Weight function $\mathrm{w}(r)$
Huber's	$\begin{cases} 1, & \text{for } \lvert r\rvert \le c \\ \frac{c}{\lvert r\rvert}, & \text{for } \lvert r\rvert > c \end{cases}$
Tukey's bisquare	$\begin{cases} [1-(r/c)^2]^2, & \text{for } \lvert r\rvert \le c \\ 0, & \text{for } \lvert r\rvert > c \end{cases}$
Hampel's	$\begin{cases} 1, & \text{for } \lvert r\rvert \le a \\ \frac{a}{\lvert r\rvert}, & \text{for } a < \lvert r\rvert \le b \\ \frac{a}{\lvert r\rvert}(c - \lvert r\rvert)/(c-b), & \text{for } b < \lvert r\rvert \le c \\ 0, & \text{for } \lvert r\rvert > c \end{cases}$
Andrews	$\begin{cases} \frac{\sin r}{r}, & \text{for } \lvert r\rvert \le \pi \\ 0, & \text{for } \lvert r\rvert > \pi \end{cases}$
Cauchy	$\frac{1}{1+(r/c)^2}$
Welsch	$e^{-(r/c)^2}$

where $w_{(0),i} = 1$.

Finally, it should be noted that the breakdown point of the M-est is not improved, thus it remains $1/N$. In order to overcome this issue, a different approach is required, as we will see in the following section.

2.3.2 Robust Estimators with Higher Breakdown Point

At this point, we have presented robust methods that are vulnerable to gross errors on the input data, therefore attaining a small breakdown point. Thus, the question raised is whether we can develop robust estimators with a higher breakdown point. The answer is to the affirmative; however, this is obtained at the expense of increased computational effort.

In this section, the basic methods that address this issue are discussed. However, due to the fact that their computational load is heavy, they are limited within theoretical interest only.

Generalized M-est (GM-est)

A few years after the development of the M-est, an improved model, that gained ground in terms of robustness and without the assumption of (2.15), was proposed. The authors in [51, 52], introduced the Generalized M-est (GM-est), by considering the weights w as a function of $\boldsymbol{x}_i$'s, i.e., replacing (2.18) by

$$\sum_{i=1}^{N} \mathrm{w}(\boldsymbol{x}_i)\psi(r_i/\hat{\sigma})\boldsymbol{x}_i = \mathbf{0}, \text{ or}$$

$$\sum_{i=1}^{N} \mathrm{w}(\boldsymbol{x}_i)\psi\left(\frac{r_i}{\mathrm{w}(\boldsymbol{x}_i)\hat{\sigma}}\right)\boldsymbol{x}_i = \mathbf{0}.$$

These estimators were proposed with the goal of bounding the influence of a single outlying observation, the effect of which can be measured by means of the so-called influence function. However, the breakdown point diminishes with an increasing dimension of unknown coefficients, i.e., M, see [20]. This fact is quite unsatisfactory, since in such dimensions there are more chances for outliers to occur. Moreover, not all of the estimators achieve the same breakdown point and some of them are not even defined for dimension of $M > 2$.

Least Median of Squares (LMedS)

The Least Median of Squares (LMedS) method estimates the parameters by solving the minimization problem:

$$\min_{\boldsymbol{\theta}} \operatorname*{Med}_{i} \, r_i^2.$$

That is, the estimator yields the smallest value for the median of squared residuals computed for the entire data set. It turns out that this method is very robust to false matches as well as to outliers due to bad localization. In particular, the attained breakdown point of this method is $(\lfloor N/2 \rfloor - M + 2)/N$; the best ratio, to be achieved, is only for dimension of the unknown $M = 2$ and corresponds to the value 50% (asymptotically), see [20].

Unfortunately, the LMedS perform poorly from a point of view of asymptotic efficiency. Unlike the M-est, the LMedS problem cannot be reduced to a weighted LS problem. In fact, it is probably impossible to write down a straightforward formula for the LMedS estimator. On the contrary, it must be solved by a search in the space of all possible estimates generated from the data. Since this space is too large, only a randomly chosen subset of data can be analyzed. Thus, the method is not applicable to any of the modern practical problems, where large data sizes are often the norm.

Least Trimmed Squares (LTS)

Another robust estimator that is very similar to the ordinary LS ones is the Least Trimmed Squares (LTS) estimator:

$$\min_{\boldsymbol{\theta}} \sum_{i=1}^{L} \{r_i^2\}_{1:N}, \tag{2.22}$$

where $\{r_i^2\}_{1:N}$ represents the ordered squared residuals (first squared then ordered). The main difference with respect to the ordinary LS method is that the largest squared residuals are not used in the summation. It should be noted that the so-called trimming constant, L, satisfies $N/2 < L \leq N$. It also determines the breakdown point of the LTS estimator, whose maximum breakdown point is $\left(\lfloor (N-M)/2 \rfloor + 1\right)/N$, attained for $L = \lfloor N/2 \rfloor + \lfloor (M+1)/2 \rfloor$, see [20]. Finally, it is evident that if $L = N$ the LTS corresponds to the LS estimator with breakdown point equal to $1/N$.

However, the disadvantage of the LTS is its computational complexity. Searching for a solution requires a combinatorial search for every subset, thus we have $\binom{N}{L}$ candidates for this estimator. Of course, since the exact solution is not easily computed, several approximation methods exist. However, there are no theoretical assurances on the quality of the resulting fit.

RANdom SAmple Consensus (RANSAC)

The RANdom SAmple Consensus (RANSAC) algorithm was proposed by Fischler and Bolles in [21] and was used in order to solve the Location Determination Problem (LDP). It is a non-deterministic algorithm in the sense that it produces a reasonable result only with a certain probability, which is increasing with respect to the number of the iterations that are allowed. This general parameter estimation approach is designed to cope with a large proportion of outliers in the input data. Unlike many of the common robust estimation techniques, such as the M-est or the LMedS that have been adopted by the computer vision community from the statistics literature, RANSAC was developed in the computer vision community.

RANSAC is a resampling technique that generates candidate solutions by using the minimum number of observations (data points) required to estimate the underlying model parameters. Unlike conventional sampling techniques, that use as much of the data as possible to obtain an initial solution and then proceed to prune outliers, RANSAC uses the smallest set possible and proceeds to enlarge this set with consistent data points. The basic algorithm is summarized as follows:

Algorithm 1 RANdom SAmple Consensus: RANSAC

```
1: Select randomly the minimum number of points required to determine the model parameters.
2: Solve for the parameters of the model.
3: Determine how many points from the set of all points fit with a predefined tolerance $\epsilon$.
4: if the fraction of the number of inliers over the total number points in the set exceeds a
   predefined threshold $\tau$ then
5:     re-estimate the model parameters using all the identified inliers and terminate.
6: else
7:     repeat steps 1 through 4 (maximum of $k$ times).
```

The number of iterations, k, is chosen to be high enough to ensure, with probability p (usually set to 0.99), that at least one of the sets of random samples does not include an outlier. Finally, it should be noted that this method attains an asymptotically breakdown point greater than 50%, which is the upper bound for many methods.

However, the gains of this method too do not appear without a trade-off. The shortcomings of the RANSAC method are:

- There is no upper bound on the time it takes to compute the parameters (except exhaustion); moreover, if the number of iterations computed is limited then the obtained solution may not be optimal.

- It requires the setting of problem-specific thresholds.

Chapter 3

Robust Linear Regression via Sparse Modeling: Greedy Methods and ℓ_1-norm Minimization

3.1 Introduction

The recent developments of methods in the spirit of robust analysis owes a lot to the emergence of *sparse* optimization methods.

In this chapter, the reader is introduced to the fundamentals of sparse optimization techniques. To this end, two basic paths are discussed: a) the task is formulated so that to minimize the ℓ_0(pseudo)-norm of a vector and b) the minimization is built around the ℓ_1-norm. The latter is the closest convex relaxation to the ℓ_0(pseudo)-norm, and it turns out that both approaches generate sparse solutions. Although the methods that employ the ℓ_0-norm are by nature NP-Hard (combinatorial), several variants have established suboptimal solutions to the task, bypassing its combinatorial nature. Moreover, under certain assumptions, such techniques guarantee the optimal solution. At the core of these methods lies the Orthogonal Matching Pursuit (OMP), which belongs to the *greedy* family of methods. The OMP is presented in detail, since it is the motivation for our proposed robust scheme for signal denoising, which is introduced in the next chapter.

Finally, the formulation of the robust regression task via sparse modeling is presented and the recently established methods are reviewed. This includes: a) the Second Order Cone Programming (SOCP) technique, b) the Alternating Direction Method of Multipliers (ADMM), c) the Sparse Bayesian Learning (SBL) robust scheme, which employs sparse Bayesian optimization techniques to deal with the task and d) the so-called Robust Orthogonal Matching Pursuit ROMP, which is based on both the OMP and the M-est.

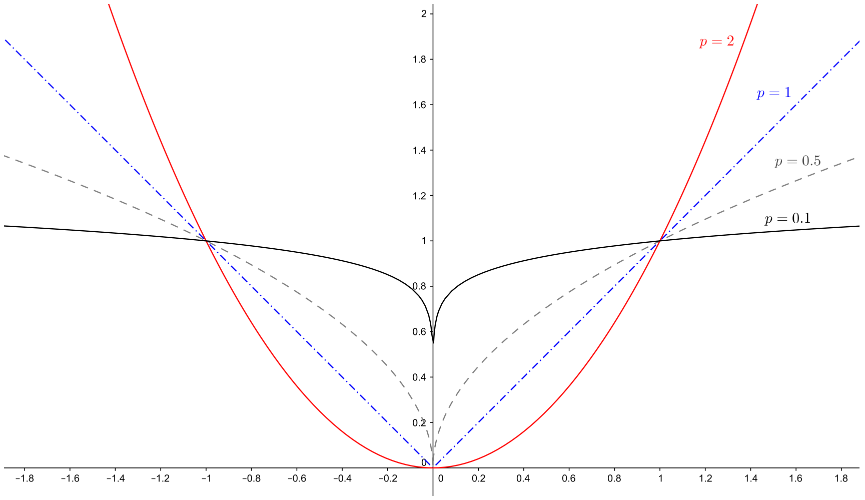


Figure 3.1: The function f$(s) = |s|^p$ for various values of p. As $p \to 0$, f approaches the indicator function, which is 0 for $s = 0$ and 1 elsewhere. Convexity is retained for $p \geq 1$, while for $p < 1$ the epigraph is non-convex.

3.2 Searching for Sparse Representations

Let $\boldsymbol{A} \in \mathbb{R}^{N \times M}$, with $N < M$, be a full-rank matrix, $\boldsymbol{b} \in \mathbb{R}^N$ and $\boldsymbol{s} \in \mathbb{R}^M$. Since $\boldsymbol{A}$ is an overcomplete dictionary, the linear system of equations $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{s}$ is known to have infinitely many solutions. Hence, if we wish to restrict the set of all possible solutions and narrow the choice to a well-defined one, an additional constraint is required. One way to accomplish this is by employing the corresponding ℓ_p-norm, i.e.,

$$\|\boldsymbol{s}\|_p := \left(\sum_{i=1}^{M} |s_i|^p \right)^{1/p}, \tag{3.1}$$

for $p \geq 1$. The most frequently used norm is defined for the choice of $p = 2$ and the respective minimization task becomes:

$$\min_{\boldsymbol{s}} \|\boldsymbol{s}\|_2^2, \text{ subject to } \boldsymbol{b} = \boldsymbol{A}\boldsymbol{s}, \tag{3.2}$$

which corresponds to the so-called minimum-norm solution. This norm defines a strictly convex function, which guarantees the uniqueness of the solution. However, the ℓ_2-norm is a measure of *energy* and it does not designate sparse representations; if we are interested in generating sparse solutions a different measure is required.

Although the definition of a norm implies that $1 \leq p \leq +\infty$, it turns out that the interesting range of p for imposing sparsity is $0 < p < 1$. However, if we let $0 < p < 1$ in equation (3.1), the resulting function does not define a norm (the triangular inequality is violated). An even more interesting case is for $p \to 0$, leading to the following definition for the ℓ_0(pseudo)-norm:

$$\|\boldsymbol{s}\|_0 := \lim_{p \to 0^+} \|\boldsymbol{s}\|_p^p = \lim_{p \to 0^+} \sum_{i=1}^{M} |s_i|^p = \# \{i : s_i \neq 0\}. \tag{3.3}$$

The definition in (3.3) does not imply an actual norm (the positive homogeneous property is not satisfied). In simple words, it represents the number of nonzero coefficients for a vector. In

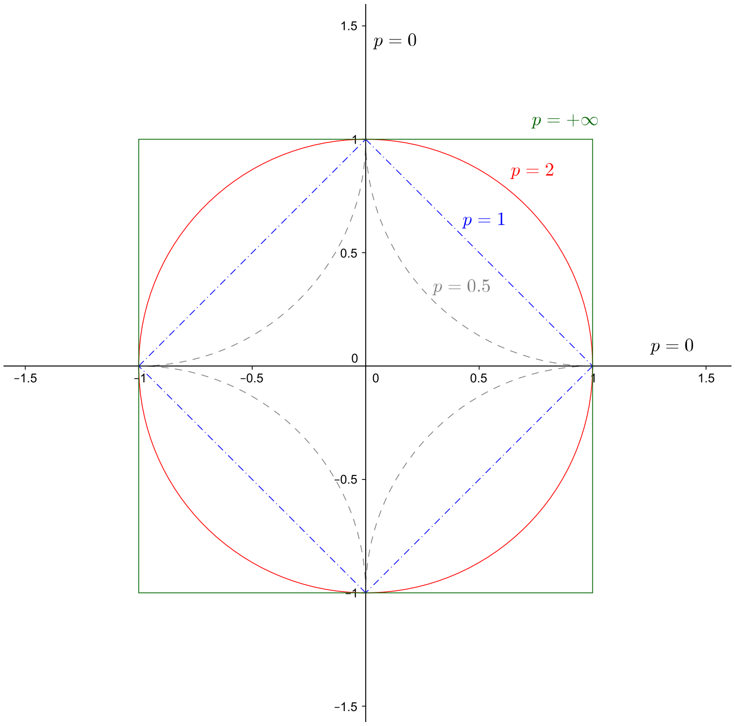


Figure 3.2: The 2-dimensional unit balls for the ℓ_p-norms: a) $p = +\infty$ (max-norm), b) $p = 2$, c) $p = 1$, d) $p = 0.5$ and e) $p = 0$ (ℓ_0-norm). As $p \to 0$ the unit sphere covers the axes with the exception of the center $(0, 0)$. Observe that for $p < 1$ the unit balls are non-convex.

Figure 3.1, the graph of $|\cdot|^p$ demonstrates the contribution of each component of a vector to the ℓ_p-norm, for various values of p. As $p \to 0$, the function approaches the indicator function. Although not truly a norm (the ℓ_0), in the sparsity-aware learning literature it is still referred to as a norm or a (pseudo)-norm.

The heart of the sparse modeling methods beats around the following optimization task:

$$\min_{\boldsymbol{s}} \|\boldsymbol{s}\|_0, \text{ subject to } \boldsymbol{b} = \boldsymbol{A}\boldsymbol{s}, \tag{3.4}$$

which seeks for the sparsest solution of the underdetermined system of linear equations[1]. Accordingly, if it is assumed that $\boldsymbol{b} = \boldsymbol{A}\boldsymbol{s} + \boldsymbol{\eta}$, where $\boldsymbol{\eta}$ is a noise vector of bounded energy, i.e., $\|\boldsymbol{\eta}\|_2 \leq \epsilon$, the task is formulated as:

$$\min_{\boldsymbol{s}} \|\boldsymbol{s}\|_0, \text{ subject to } \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{s}\|_2^2 \leq \varepsilon, \tag{3.5}$$

which is known as the task of *sparse denoising* in the respective literature [53], [1]. This is a task of major importance, since noise is also involved, something that almost always occurs in practice. It should be noted that, the value of the parameter $\epsilon > 0$ is used for controlling the noise level[2]. Also, notice that, if $\epsilon = 0$ the task in (3.5) resorts to (3.4).

Unfortunately, in [54], it was proven that the task in (3.5) is NP-Hard (combinatorial - not solvable in polynomial time). Moreover, the solution is not always unique. However, there exist methods that under reasonable assumptions manage to overcome the stated problems, see [53].

[1] Recall that the number of columns exceed the number of rows for matrix $\boldsymbol{A}$.

[2] It is obvious that $\varepsilon = \epsilon^2$ is considered.

3.2.1 Greedy Methods - Orthogonal Matching Pursuit (OMP)

One among the most popular algorithms that attempts to solve the task in (3.4) or (3.5) is the Orthogonal Matching Pursuit (OMP) and it belongs to the core of the greedy family of methods, see [55, 56, 27]. The iterative scheme is an offspring of the classic Matching Pursuit method, which was proposed for signal compression, see [57, 58, 29], although it was already known to the Statisticians. Under certain assumptions imposed, e.g., on the spark, the mutual coherence, the ERC in [59, 1, 56], the OMP algorithm is guaranteed to recover the sparsest representation for the noiseless case, in (3.4), or to provide sufficiently good sparse estimates for the noisy one, in (3.5). Apart from the OMP, other variants also exist [60, 30, 31, 61, 62], although they fall out of the scope of this dissertation.

The OMP algorithm is summarized in Algorithm 2. Initializing at the zero solution and the respective residual, the vector $\boldsymbol{b}$, the scheme sequentially selects columns from matrix $\boldsymbol{A}$ (called atoms) that correspond to nonzero elements for the sparse vector estimate $\hat{\boldsymbol{s}}$. In particular, at the selection step 6, the method identifies the column from matrix $\boldsymbol{A}$ which is more correlated to the residual up to this point. Next, the set of active columns, i.e., $\widetilde{\mathcal{S}}$ (associated with indices of previously selected columns), is augmented by the newly selected column. Finally, at step 8, the LS minimization task is performed by projecting on the subspace that originates from the columns $\boldsymbol{a}_j$ of $\boldsymbol{A}$ that belong to the set $\widetilde{\mathcal{S}}$ of active columns, i.e., the column vectors of matrix $\boldsymbol{A}_{\widetilde{\mathcal{S}}}$. In fact, the set $\widetilde{\mathcal{S}}$ forms the support for the sparse vector estimate, $\hat{\boldsymbol{s}}$.

The scheme also preserves an interesting geometric interpretation. Under the assumption of certain conditions, which guarantee that the columns of matrix $\boldsymbol{A}$ approximately form an orthogonal system, e.g., bounds on the Restricted Isometry Property (RIP) constant in [39], the spark or the mutual coherence in [53], the method successfully selects the atoms that are more informative to the representation of the signal vector, $\boldsymbol{b}$. Moreover, because of the orthogonalization, once an atom is selected, it can never be selected again in subsequent iterations.

Algorithm 2 Orthogonal Matching Pursuit: OMP

```
 1: procedure OMP($\boldsymbol{A}, \boldsymbol{b}, \epsilon$)
 2:     $\hat{\boldsymbol{s}} \leftarrow \boldsymbol{0}$
 3:     $\boldsymbol{r} \leftarrow \boldsymbol{b} - \boldsymbol{A}\hat{\boldsymbol{s}} = \boldsymbol{b}$
 4:     $\widetilde{\mathcal{S}} \leftarrow \emptyset$
 5:     while $\|\boldsymbol{r}\|_2 > \epsilon$ do
 6:         $j_k := \text{argmax}_j \frac{|\langle \boldsymbol{r}, \boldsymbol{a}_j \rangle|}{\|\boldsymbol{a}_j\|_2^2}$          ▷ Selection step.
 7:         $\widetilde{\mathcal{S}} \leftarrow \widetilde{\mathcal{S}} \cup \{j_k\}$
 8:         $\hat{\boldsymbol{s}} := \text{argmin}_{\boldsymbol{s}} \|\boldsymbol{b} - \boldsymbol{A}_{\widetilde{\mathcal{S}}}\boldsymbol{s}\|_2^2$          ▷ Least Squares solution step.
 9:         $\boldsymbol{r} \leftarrow \boldsymbol{b} - \boldsymbol{A}\hat{\boldsymbol{s}}$
10:     Output: a sparse vector $\hat{\boldsymbol{s}}$.
```

3.2.2 Convex Relaxation to the ℓ_1-norm

An alternative to the greedy selection methods is the relaxation of the ℓ_0-norm to its closest convex one, i.e., the ℓ_1-norm. The task of interest in (3.5) is reshaped to:

$$\min_{\boldsymbol{s}} \|\boldsymbol{s}\|_1, \text{ subject to } \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{s}\|_2^2 \leq \varepsilon, \tag{3.6}$$

or equivalently to its Lagrangian formulation

$$\min_{\boldsymbol{s}} \left\{ \frac{1}{2} \|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{s}\|_2^2 + \lambda \|\boldsymbol{s}\|_1 \right\}, \tag{3.7}$$

which is known as the *Basis Pursuit* (BP) in the Signal Processing literature and as the *Least Absolute Shrinkage and Selection Operator* (LASSO) in the Statistics literature.

3.3 Sparse Outlier Modeling for Robust Linear Regression

Now that the previous fundamental formulations of sparse modeling have been discussed, the direction taken is to apply the aforementioned techniques to the robust linear regression task.

While for the standard linear regression task it is assumed that the observations are generated via (2.1), for the robust task, the noise variable is expressed as a sum of two independent components, i.e., $\nu_i = \underline{u}_i + \eta_i$, and the model transforms into:

$$y_i = \boldsymbol{x}_i^T \underline{\boldsymbol{\theta}} + \underline{u}_i + \eta_i. \tag{3.8}$$

Moreover, since the compact form of the noise model decomposition is:

$$\boldsymbol{\nu} = \underline{\boldsymbol{u}} + \boldsymbol{\eta}, \tag{3.9}$$

sparsity constraints are imposed to $\underline{\boldsymbol{u}} \in \mathbb{R}^N$. Thus, if $\mathcal{S}$ is the support set for the sparse outlier noise vector, it is assumed that $|\mathcal{S}| \leq S << N$. Thus, (3.8) results in the following compact robust linear regression form:

$$\boldsymbol{y} = \boldsymbol{X}\underline{\boldsymbol{\theta}} + \underline{\boldsymbol{u}} + \boldsymbol{\eta}, \tag{3.10}$$

where $\underline{\boldsymbol{u}}$ is a sparse (unknown) vector. Furthermore, if we assume that $\boldsymbol{\eta}$ is a bounded vector of inlier noise, the respective robust minimization task, according to (3.5), is formulated as:

$$\min_{\boldsymbol{\theta}, \boldsymbol{u}} ||\boldsymbol{u}||_0, \text{ subject to } ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{u}||_2^2 \leq \varepsilon. \tag{3.11}$$

On the other hand, the respective robust minimization task, according to (3.5), is given via:

$$\min_{\boldsymbol{\theta}, \boldsymbol{u}} ||\boldsymbol{u}||_1, \text{ subject to } ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{u}||_2^2 \leq \varepsilon, \tag{3.12}$$

or in its equivalent formulation

$$\min_{\boldsymbol{\theta}, \boldsymbol{u}} \left\{ \frac{1}{2} ||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta} - \boldsymbol{u}||_2^2 + \lambda ||\boldsymbol{u}||_1 \right\}, \tag{3.13}$$

where λ is a tuning parameter (depending on the selection of ε) that controls the amount of regularization. It is clear that the task in (3.13) shares resemblance to the Ridge Regression (RR) task. However, although the latter has a solution obtained in closed form, this is not the case for the task in (3.13). Recall that the ℓ_1-norm is not a differentiable function; to this end, one has to resort to sub-differential techniques.

As expected, the task in (3.11) is also NP-Hard, in general, while its relaxation is a convex one, solvable with the use of a variety of methods. Moreover, if certain conditions/assumptions are considered, the solution is unique and a solution is readily obtained.

Finally, it should be noted that in the majority of the linear regression tasks, the vector of unknowns, $\underline{\boldsymbol{\theta}}$, is considered as a dense (without non-zero coefficients) vector. However, if one considers that the observations admit a sparse representation then both vectors $\underline{\boldsymbol{\theta}}$ and $\underline{\boldsymbol{u}}$ are sparse. Consequently, this leads to the original sparse denoising task presented in (3.5) for the observation vector $\boldsymbol{y}$, the augmented matrix $\boldsymbol{A} = [\boldsymbol{X}\ \boldsymbol{I}_N]$ and vector $\boldsymbol{s} = \begin{pmatrix} \underline{\boldsymbol{\theta}} \\ \underline{\boldsymbol{u}} \end{pmatrix}$, which is assumed to be sparse.

3.4 Related Works

The existing works that deal with the robust regression task are divided into three major categories. The first one consists of methods that employ convex optimization techniques in order to minimize the ℓ_1-norm of the sparse outlier vector. The methods that are presented here are based on a) the so-called Basis Pursuit (BP) formulation in (3.12) and b) the LASSO formulation in (3.13). The second approach is probabilistic and it is based on sparse Bayesian inferring techniques. The final one revolves around the core greedy scheme, i.e., the OMP, albeit via the weighted LS minimization path. All of these method are reviewed in the following section.

3.4.1 Robust Denoising via the Minimization of the ℓ_1-norm

Although several algorithms are established for the minimization tasks in (3.12) or (3.13), e.g., the Least Angle Regression (LARS), the forward stagewise, the pathwise coordinate descent e.t.c., we present two of the most commonly used in practice, i.e., the convex optimizer via the Second Order Cone and the Alternating Direction Method of Multipliers (ADMM), which have been employed in previously published research, related to the robust regression task.

Second Order Cone Programming (SOCP)

The task in (3.12) is also known as *Basis Pursuit for Robust Regression*-(BPRR) and it can be solved via quadratic optimization methods, read [63, 64]. In particular, by expressing it as a Second Order Cone Programming (SOCP) task, leads to:

$$\hat{\boldsymbol{c}} := \operatorname*{argmin}_{\boldsymbol{c}} \boldsymbol{g}^T \boldsymbol{c}, \text{ subject to } \boldsymbol{B}_c^T \boldsymbol{c} \geq \boldsymbol{0},\ \boldsymbol{y} - \boldsymbol{A}_c \boldsymbol{c} \in \mathcal{C}_\epsilon^{N+1} \tag{3.14}$$

where $\boldsymbol{c} = \begin{pmatrix} \boldsymbol{\theta} \\ \boldsymbol{u} \\ \boldsymbol{\omega} \end{pmatrix}$, $\boldsymbol{g} = \begin{pmatrix} 0 \\ 0 \\ \boldsymbol{1} \end{pmatrix} \in \mathbb{R}^{M+2N}$, $\boldsymbol{B}_c = \begin{bmatrix} \boldsymbol{O}_{M\times N} & \boldsymbol{O}_{M\times N} \\ -\boldsymbol{I}_N & \boldsymbol{I}_N \\ \boldsymbol{I}_N & \boldsymbol{I}_N \end{bmatrix}$, $\boldsymbol{A}_c = [\boldsymbol{X}\ \boldsymbol{I}_N\ \boldsymbol{O}_{N\times N}]$ and $\mathcal{C}_\epsilon^{N+1}$ is the unit second order (convex) cone of dimension $N+1$ and ϵ is the bound for the ℓ_2-norm of the noise vector.

Generalized LASSO and the Alternating Direction Method of Multipliers (ADMM)

The convex optimization task in (3.13) can also be expressed in its more general form; that is, the Generalized LASSO one. Its solution is given by

$$\hat{\boldsymbol{z}} := \operatorname*{argmin}_{\boldsymbol{z}} \{ \frac{1}{2} ||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{z}||_2^2 + \lambda ||\boldsymbol{G}_L \boldsymbol{z}||_1 \}, \tag{3.15}$$

where $\boldsymbol{z} = \begin{pmatrix} \boldsymbol{\theta} \\ \boldsymbol{u} \end{pmatrix}$, $\boldsymbol{A} = [\boldsymbol{X} \ \boldsymbol{I}_N]$ and $\boldsymbol{G}_L = [\boldsymbol{O}_{N \times M} \ \boldsymbol{I}_N]$. The $\boldsymbol{G}_L$ matrix is related to the formulated task, thus other choices are available too; for example, for $\boldsymbol{G}_L = \boldsymbol{I}_{N+M}$ one obtains the standard LASSO form.

The ADMM is a technique established for obtaining a solution to (3.15) for appropriate multiplier values $\lambda > 0$ and was studied in the 1970s and 1980s, as a good alternative to penalty methods, although it was originally established as a method to solve partial differential equations, [65, 66]. An application in the context of robust denoising was proposed in [67, 68, 69]. According to (3.15), the task is expressed via a linear constraint as

$$\hat{\boldsymbol{z}} := \operatorname*{argmin}_{\boldsymbol{z}} \{ \frac{1}{2} ||\boldsymbol{y} - \boldsymbol{A}\boldsymbol{z}||_2^2 + \lambda ||\boldsymbol{u}||_1 \}, \text{ subject to } \boldsymbol{G}_L \boldsymbol{z} = \boldsymbol{u}, \tag{3.16}$$

and its solution is given by Algorithm 3. The Soft-thresholding operator in 8-th row of Algorithm 3 is defined as: $\mathrm{S}_\mu(v_i) := \mathrm{sgn}(v_i)(|v_i| - \mu)_+$, where $(x)_+ := \max\{x, 0\}$. Finally, it should be noted that various additional termination criteria could also be adopted. For example, the iteration loop could also be terminated (prior to the maximum number of iterations) if the estimate is not significantly altered (in the ℓ_2-norm sense) from one iteration to the next.

Algorithm 3 Alternating Direction Methods of Multipliers: ADMM

```
1:  procedure ADMM($\boldsymbol{X}$, $\boldsymbol{y}$, $\lambda$, $\rho$, $n_{max}$)
2:      $n \leftarrow 0$
3:      $\boldsymbol{A} = [\boldsymbol{X} \ \boldsymbol{I}_N]$, $\boldsymbol{G}_L = [\boldsymbol{O}_{N \times M} \ \boldsymbol{I}_N]$
4:      $\hat{\boldsymbol{u}}_{(0)} \leftarrow$ randomly chosen, $\boldsymbol{o}_{(0)} \leftarrow \boldsymbol{0}$, $\rho_{(0)} \leftarrow \rho$
5:      while $n < n_{max}$ do
6:          $n \leftarrow n + 1$
7:          $\hat{\boldsymbol{z}}_{(n)} \leftarrow (\boldsymbol{A}^T\boldsymbol{A} + \rho_{(n-1)}\boldsymbol{G}_L^T\boldsymbol{G}_L)^{-1}(\boldsymbol{A}^T\boldsymbol{y} + \rho_{(n-1)}\boldsymbol{G}_L^T\hat{\boldsymbol{u}}_{(n-1)} - \boldsymbol{G}_L^T\boldsymbol{o}_{(n-1)})$
8:          $\hat{\boldsymbol{u}}_{(n)} \leftarrow \mathrm{S}_{\frac{\lambda}{\rho_{(n-1)}}}(\boldsymbol{G}_L\hat{\boldsymbol{z}}_{(n)} + \boldsymbol{o}_{(n-1)}/\rho_{(n-1)})$
9:          $\boldsymbol{o}_{(n)} \leftarrow \boldsymbol{o}_{(n-1)} + \rho_{(n-1)}(\boldsymbol{G}_L\hat{\boldsymbol{z}}_{(n)} - \hat{\boldsymbol{u}}_{(n)})$
10:         $\rho_{(n)} \leftarrow \min\{5, 1.1\rho_{(n-1)}\}$
11:     Output: $\hat{\boldsymbol{z}}_{(n_{max})} = (\hat{\boldsymbol{\theta}}_{(n_{max})}^T, \hat{\boldsymbol{u}}_{(n_{max})}^T)^T$.
```

3.4.2 A Probabilistic Approach for the Robust Denoising Task

Another path that has been exploited in the respective literature for the Robust denoising task is via Sparse Bayesian Learning (SBL) techniques [70, 71], [72, 73, 71, 1].

Algorithm 4 Sparse Bayesian Learning: SBL

```
procedure SBL($\boldsymbol{X}$, $\boldsymbol{y}$, $k_{max}$)
    $k \leftarrow 0$
    $\hat{\boldsymbol{\theta}}_{(0)}$, $\hat{\sigma}^2_{(0)}$ and $\boldsymbol{\gamma}_{(0),i}$ for $i = 1, \dots, N$ randomly chosen
    $\boldsymbol{\Gamma}_{(0)} \leftarrow \operatorname{diag}(\gamma_{(0),1}, \dots, \gamma_{(0),N})$
    while $k < k_{max}$ do
        $k \leftarrow k + 1$
        $\hat{\boldsymbol{u}}_{(k)} \leftarrow (\boldsymbol{I}_N + \sigma^2_{(k-1)}\boldsymbol{\Gamma}^{-1}_{(k-1)})^{-1}(\boldsymbol{y} - \boldsymbol{X}\hat{\boldsymbol{\theta}}_{(k-1)})$
        $\widehat{\boldsymbol{U}}'_{(k)} \leftarrow \hat{\boldsymbol{u}}_{(k)}\hat{\boldsymbol{u}}^T_{(k)} + (\sigma^{-2}_{(k-1)}\boldsymbol{I}_N + \boldsymbol{\Gamma}^{-1}_{(k-1)})^{-1}$
        $\gamma_{(k),i} \leftarrow \hat{u}'_{(k),ii}$
        $\sigma^2_{(k)} \leftarrow \frac{1}{N}||\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}_{(k-1)}||_2^2 + \frac{1}{N}\operatorname{tr}(\widehat{\boldsymbol{U}}'_{(k)}) - \frac{2}{N}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\theta}_{(k-1)})^T\hat{\boldsymbol{u}}_{(k)}$
        $\hat{\boldsymbol{\theta}}_{(k)} \leftarrow (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T(\boldsymbol{y} - \hat{\boldsymbol{u}}_{(k)})$
    Output: $\hat{\boldsymbol{\theta}}_{(k_{max})}$
```

Sparse Bayesian Learning (SBL)

The model is based on equation (3.10). The development, analysis and experimental study of the original SBL for sparse signal recovery has been extensively discussed in [72, 73, 74].

To this end, it is assumed that $\underline{u}_i$ is a random variable with prior distribution $\underline{u}_i \sim \mathcal{N}(0, \gamma_i)$, where γ_i is the hyperparameter that controls the variance of each $\underline{u}_i$ and has to be learnt. The hyperparameters are stored in a vector $\boldsymbol{\gamma}$ and the diagonal matrix $\boldsymbol{\Gamma} := \operatorname{diag}(\gamma_1, \dots, \gamma_N)$ is also involved. If $\gamma_i = 0$, then $\underline{u}_i = 0$, i.e., no outlier is identified. In contrast, a positive value of γ_i corresponds to an outlier in the i-th observation, y_i. The regression coefficients are estimated, by jointly searching for

$$(\hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\gamma}}, \hat{\sigma}^2) = \arg\max_{\boldsymbol{\theta}, \boldsymbol{\gamma}, \sigma^2} p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}, \boldsymbol{\gamma}, \sigma^2), \tag{3.17}$$

where $\boldsymbol{\gamma} := (\gamma_1, \dots, \gamma_N)^T$ and $\eta_i \sim \mathcal{N}(0, \sigma^2)$. The posterior estimation of $\underline{\boldsymbol{u}}$, follows, from:

$$\hat{\boldsymbol{u}} = \mathbb{E}[\underline{\boldsymbol{u}}|\boldsymbol{X}, \hat{\boldsymbol{\theta}}, \hat{\boldsymbol{\gamma}}, \hat{\sigma}^2]. \tag{3.18}$$

At each iteration, the method obtains the current estimate of the outlier components and then it performs an ordinary LS estimation on the corrected data, i.e., $\boldsymbol{y} - \hat{\boldsymbol{u}}$. The termination of the scheme could be set either when the number of iteration reaches a user-defined threshold or when successive iterations no longer offer significant gains in term of estimation. The scheme is described in Algorithm 4.

Finally, it should also be noted that, the authors in [72, 73] suggest that the hyperparameters, $\gamma_{(k),i}$, that are smaller than a predefined threshold, are pruned from future iterations.

3.4.3 A Combined OMP Selection-based and M-est Method

Robust Orthogonal Matching Pursuit (ROMP)

The Robust Orthogonal Matching Pursuit (ROMP) method, that the authors have developed in [75], is based on a combination of the popular OMP algorithm with the M-est. The key

Algorithm 5 Robust Orthogonal Matching Pursuit: ROMP

```
1:  procedure ROMP($\boldsymbol{X},\ \boldsymbol{y},\ \epsilon$)
2:      $k \leftarrow 0$
3:      $\mathcal{T}_0 \leftarrow \emptyset$, $\hat{\boldsymbol{\theta}}_{(0)} = \mathbf{0}$, $\boldsymbol{r}_{(0)} = \boldsymbol{y}$
4:      $\widetilde{\boldsymbol{X}} = \left[\boldsymbol{x}_1/\|\boldsymbol{x}_1\|_2, \ldots, \boldsymbol{x}_M/\|\boldsymbol{x}_M\|_2\right]$
5:      while $\|\boldsymbol{r}_{(k)}\|_2 > \epsilon$ do
6:          $k \leftarrow k + 1$
7:          $\hat{\sigma} \leftarrow \mathrm{MAD}(\boldsymbol{r}_{(k-1)})$, $\underset{\sim}{\boldsymbol{r}}_{(k-1)} \leftarrow \psi(\boldsymbol{r}_{(k-1)}/\hat{\sigma})$ and $\boldsymbol{a}_{(k-1)} \leftarrow \underset{\sim}{\boldsymbol{r}}^T_{(k-1)}\widetilde{\boldsymbol{X}}$
8:          $i_k \leftarrow \mathrm{argmax}_i \left|\boldsymbol{a}_{(k-1)}\right|$, $\mathcal{T}_k \leftarrow \mathcal{T}_{k-1} \cup i_k$
9:          $\hat{\boldsymbol{\theta}}_{(k)} \leftarrow \mathrm{argmin}_{\boldsymbol{\theta}} \, ||\boldsymbol{W}_r^{1/2}(\boldsymbol{y} - \boldsymbol{X}_{\mathcal{T}_k}\boldsymbol{\theta})||_2^2$
10:         $\boldsymbol{r}_{(k)} \leftarrow \boldsymbol{y} - \boldsymbol{X}_{\mathcal{T}_k}\hat{\boldsymbol{\theta}}_{(k)}$
11:     Output: $\hat{\boldsymbol{\theta}}_{(k)}$
```

aspect of the algorithm, which is also the feature that introduces robustness, is the execution of a weighted LS step (M-est), instead of an ordinary one, each time the support set is augmented by an atom.

As presented in Algorithm 5, the main procedure starts with the computation of the Median Absolute Deviation[3], $\hat{\sigma} = \mathrm{MAD}(\boldsymbol{r})$, and the residual pseudo-values $\underset{\sim}{\boldsymbol{r}}$; ψ is a robust function that could be selected from Table 2.1. At the selection step, the atom is chosen from matrix $\boldsymbol{X}$ based on the maximum correlation between its normalized columns and the residual pseudo-values. At the weighted Least Squares step, 9, the diagonal elements of matrix $\boldsymbol{W}_r$ (weights) are assigned according to the selected ψ function. The difference to (2.20) is that at each k-th step, $\boldsymbol{X}_{\mathcal{T}_k}$ includes only the columns of $\boldsymbol{X}$ that have been selected until the current step. Unfortunately, no theoretical justifications are established, either on the selection of the atom based on the residual pseudo-values or on the iterative employment of the M-est. Although a variety of termination criteria exist, we let the algorithm terminate, as soon as the length of the residual vector drops below a predefined threshold.

[3] $\mathrm{MAD}(\boldsymbol{x}) = \mathrm{Med}_i(|x_i - \mathrm{Med}_i(x_i)|)$.

Chapter 4

Greedy Algorithm for Robust Denoising

4.1 Introduction

The basic methods that deal with the linear regression task via sparse modeling have already been addressed in Chapter 3. The focus of this chapter is to introduce a novel robust scheme for the task of linear regression, which is based on the popular Orthogonal Matching Pursuit (OMP). The method combines the two approaches of regression and diagnostics, in order to perform a single-outlier detection step iteratively.

The path taken here is to split the noise into two components: a) the inlier noise and b) the outliers, which are explicitly modeled by employing sparsity arguments, according to (3.9). Based on this model, an efficient algorithm, i.e., the Greedy Algorithm for Robust Denoising (GARD), is derived. The method alternates between a Least Squares (LS) optimization criterion and an Orthogonal Matching Pursuit (OMP) selection step, that identifies the outliers. Furthermore, efficient implementations are proposed for the method. Next, the evaluation of the algorithm in terms of its convergence is studied, where it is proved that it converges in a finite number of iterations. The establishment of conditions/bounds, based on the Restricted Isometry Property, which guarantee the recovery of the sparse outlier vector's support, follow. The case where only outliers are present has been studied separately; it is derived that the recovery of the original signal via GARD is exact. Moreover, for the case of additional inlier bounded noise, results concerning the recovery of the sparse outlier vector's support and the error of the approximation are given. Finally, extensive experimentation demonstrates the comparative advantages of the new iterative scheme.

4.2 Greedy Algorithm for Robust Denoising (GARD)

The proposed algorithmic scheme attempts to solve problem (3.11) by using the split noise model of (3.9). It is built around the celebrated Orthogonal Matching Pursuit (OMP) rationale and alternates between a Least Squares minimization task and an OMP selection technique.

Algorithm 6 Greedy Algorithm for Robust Denoising: GARD

```
 1: procedure GARD($\boldsymbol{X}$, $\boldsymbol{y}$, $\epsilon$)
 2:     $k \leftarrow 0$
 3:     $\widetilde{\mathcal{S}}_0 \leftarrow \{1, ..., M\}$, $\mathcal{S}_0^c \leftarrow \{1, ..., N\}$, $\boldsymbol{A} = [\boldsymbol{X}\ \boldsymbol{I}_N]$
 4:     $\hat{\boldsymbol{z}}_{(0)} \leftarrow (\boldsymbol{A}_{\widetilde{\mathcal{S}}_0}^T \boldsymbol{A}_{\widetilde{\mathcal{S}}_0})^{-1} \boldsymbol{A}_{\widetilde{\mathcal{S}}_0}^T \boldsymbol{y}$        ▷ Initial LS solution step.
 5:     $\boldsymbol{r}_{(0)} \leftarrow \boldsymbol{y} - \boldsymbol{A}_{\widetilde{\mathcal{S}}_0} \hat{\boldsymbol{z}}_{(0)}$
 6:     while $\|\boldsymbol{r}_{(k)}\|_2 > \epsilon$ do
 7:         $k \leftarrow k + 1$
 8:         $j_k \leftarrow \text{argmax}_{j \in \mathcal{S}_{k-1}^c} |r_{(k-1),j}|$, $i_k = j_k + |\widetilde{\mathcal{S}}_0|$        ▷ Selection step.
 9:         $\widetilde{\mathcal{S}}_k \leftarrow \widetilde{\mathcal{S}}_{k-1} \cup \{i_k\}$, $\mathcal{S}_k^c \leftarrow \mathcal{S}_{k-1}^c \setminus \{j_k\}$
10:         $\hat{\boldsymbol{z}}_{(k)} \leftarrow (\boldsymbol{A}_{\widetilde{\mathcal{S}}_k}^T \boldsymbol{A}_{\widetilde{\mathcal{S}}_k})^{-1} \boldsymbol{A}_{\widetilde{\mathcal{S}}_k}^T \boldsymbol{y}$        ▷ LS solution step.
11:         $\boldsymbol{r}_{(k)} \leftarrow \boldsymbol{y} - \boldsymbol{A}_{\widetilde{\mathcal{S}}_k} \hat{\boldsymbol{z}}_{(k)}$
12:     Output: $\hat{\boldsymbol{z}}_{(k)} = (\hat{\boldsymbol{\theta}}_{(k)}^T, \hat{\boldsymbol{u}}_{(k)}^T)^T$ after $k$ iterations.
```

Thus, it generates sparse solutions in a greedy way, in line with the standard OMP.

The task in (3.11) can readily be expressed in the form:

$$\min_{\boldsymbol{\theta}, \boldsymbol{u}} ||\boldsymbol{u}||_0, \text{ subject to } \left\| \boldsymbol{y} - \boldsymbol{A} \begin{pmatrix} \boldsymbol{\theta} \\ \boldsymbol{u} \end{pmatrix} \right\|_2^2 \leq \varepsilon, \tag{4.1}$$

where $\boldsymbol{A} = [\boldsymbol{X}\ \boldsymbol{I}_N] \in \mathbb{R}^{N \times N+M}$ is the augmented matrix. The key feature of the proposed scheme is the restriction of the greedy-selection over atoms of the second half of matrix $\boldsymbol{A}$, i.e., matrix $\boldsymbol{I}_N = [\boldsymbol{e}_1\ \boldsymbol{e}_2\ ...\ \boldsymbol{e}_N]$, where $\boldsymbol{e}_i$ are the vectors of the standard Euclidean basis of $\mathbb{R}^N$. As it is demonstrated, the improved performance of the proposed scheme is due to the orthogonality between the columns of $\boldsymbol{I}_N$.

The method is described best, via the use of subsets, corresponding to a set of *active* and *inactive* columns of matrix $\boldsymbol{A}$. The active set, $\widetilde{\mathcal{S}}_k$, which contains the indices of the active columns from $\boldsymbol{A}$ at the k-th step, and the inactive set, $\widetilde{\mathcal{S}}_k^c$, which contains the remaining ones, i.e., those that do not participate in the representation. Moreover, the set of indices that refers to the selected columns of the identity matrix, $\boldsymbol{I}_N$, and with respect to the set $\widetilde{\mathcal{S}}$, is defined as:

$$\mathcal{S}_k := \begin{cases} \left\{ j - |\widetilde{\mathcal{S}}_0| \ : \ j \in \widetilde{\mathcal{S}}_k \setminus \widetilde{\mathcal{S}}_0 \right\} & \text{for } k = 1, 2, \dots \\ \emptyset & \text{for } k = 0 \end{cases}, \tag{4.2}$$

where $|\widetilde{\mathcal{S}}_0|$ denotes the cardinality of the set $\widetilde{\mathcal{S}}_0$ and $\widetilde{\mathcal{S}}_k \setminus \widetilde{\mathcal{S}}_0 := \{j :\ j \in \widetilde{\mathcal{S}}_k \text{ and } j \notin \widetilde{\mathcal{S}}_0\}$. The set $\mathcal{S}_k$ is of great importance, since it indicates the support for the sparse outlier estimate. Also note that, $\mathcal{S}_k^c$ is used for its complementary set. For example, at the initial step, $\mathcal{S}_0 = \emptyset$, the first selection of an index is performed over the set $\mathcal{S}_0^c = \{1, \dots, N\}$. Suppose now that the index $j_1 = 2$ is selected; according to the selection step 8 of GARD, the update for the active set is $\widetilde{\mathcal{S}}_1 = \{1, \dots, M, M + 2\}$, which results to $\mathcal{S}_1 = \{2\}$. At the second iteration of the algorithm, the next index is selected from the set $\mathcal{S}_1^c = \mathcal{S}_0^c \setminus \{2\}$. The algorithmic scheme is provided in Algorithm 6, see [35].

In simple words, following OMP's rationale, we have that:

- Initially, the method performs a LS minimization by projecting the measurement vector $\boldsymbol{y}$ onto the subspace formed by the columns of matrix $\boldsymbol{X}$ and computes the initial residual.

- At each subsequent step, it selects that atom of matrix $\boldsymbol{I}_N$, which is more correlated with the current residual (corresponding to an outlier identification) and augments the set of (active) columns that participate in the representation by the newly selected one. The correlation is measured with respect to the angle, which in turn leads to the maximization of $\left|\langle \boldsymbol{r}_{(k)}, \boldsymbol{e}_i \rangle\right| = \left|r_{(k),i}\right|$ for an index $i = 1, \ldots, N$.

- Finally, it performs a new LS minimization step over the current set of active columns and updates the residual. The procedure is repeated until the residual drops below a specific predefined threshold.

Remark 4.1. *In the following, in order to keep the notation as simple as possible, $\boldsymbol{A}_{(k)}$ is used for the restriction of active columns of matrix $\boldsymbol{A}$ over the set $\widetilde{\mathcal{S}}_k$, instead of using $\boldsymbol{A}_{\widetilde{\mathcal{S}}_k}$. The numerical index, (k), also refers to the current iteration of the algorithmic scheme.*

The complexity of the algorithm is $O\big((M+k)^3 + 2N(M+k)^2\big)$ at each k-th step; although $k << N$ (there are only few outliers compared to the number of data), it is considered barely adequate. Thus, an improvement should be considered for large dimensionality. Since at each step the method solves a standard LS task, the complexity could be further reduced by using: a) a *Cholesky* decomposition, b) a *QR* factorization or c) the *Matrix Inversion Lemma* (MIL). For details on those implementations of the classic OMP, see [76]. Playing with all schemes, the most efficient implementation was found to be the Cholesky decomposition, as described below:

- *Replace* the initial ($k := 0$) solution step 4 of Algorithm 6 with:
  Factorization step: $\boldsymbol{X}^T\boldsymbol{X} = \boldsymbol{L}_{(0)}\boldsymbol{L}_{(0)}^T$.

  Solve $\boldsymbol{L}_{(0)}\boldsymbol{L}_{(0)}^T\boldsymbol{z} = \boldsymbol{X}^T\boldsymbol{y}$ using:
  - forward substitution $\boldsymbol{L}_{(0)}\boldsymbol{q} = \boldsymbol{X}^T\boldsymbol{y}$
  - backward substitution $\boldsymbol{L}_{(0)}^T\hat{\boldsymbol{z}}_{(0)} = \boldsymbol{q}$.

- *Replace* the update ($k := k+1$) solution step 10 of Algorithm 6, with:
  Compute $\boldsymbol{v}$ such that: $\boldsymbol{L}_{(k-1)}\boldsymbol{v} = \boldsymbol{A}_{(k-1)}^T\boldsymbol{e}_{j_k}$
  Compute: $b = \sqrt{1 - ||\boldsymbol{v}||_2^2}$
  Matrix Update: $\boldsymbol{L}_{(k)} = \begin{pmatrix} \boldsymbol{L}_{(k-1)} & \boldsymbol{0} \\ \boldsymbol{v}^T & b \end{pmatrix}$
  Solve $\boldsymbol{L}_{(k)}\boldsymbol{L}_{(k)}^T\boldsymbol{z} = \boldsymbol{A}_{(k)}^T\boldsymbol{y}$ using:
  - forward substitution $\boldsymbol{L}_{(k)}\boldsymbol{p} = \boldsymbol{A}_{(k)}^T\boldsymbol{y}$
  - backward substitution $\boldsymbol{L}_{(k)}^T\hat{\boldsymbol{z}}_{(k)} = \boldsymbol{p}$.

This modification leads to a squared cost for the main iteration steps. Analytically, the cost at the initial factorization plus that of the forward and backward substitution is $O(M^3/3 + (N+1)M^2)$. At each subsequent step, neither inversion nor factorization is required. The lower triangular matrix $\boldsymbol{L}_{(k)}$ is updated, only with a minimal cost of square-dependence. Furthermore, the cost required for solving the linear system using forward and backward substitution at the k-th step is $O(3(M+k)^2/2 + N(M+k))$, for $k = 1, 2, \ldots$. Thus, the *total complexity* of the efficient GARD implementation via the Cholesky decomposition is

$$O\big(M^3/3 + k^3/2 + M^2(N + 1 + 3k/2) + k^2(N + 3M)/2 + kMN\big).$$

Remark 4.2. *The algorithm begins with a LS solution to obtain $\hat{\boldsymbol{z}}_{(0)}$. Thus, if no outliers exist, GARD solves the ordinary LS problem; it provides the Maximum Likelihood (ML) estimator, assuming that the noise is Gaussian.*

Remark 4.3. *Since a LS task is implemented at each step, the new residual, $\boldsymbol{r}_{(k)}$, is orthogonal to each column that participates in the representation. Specifically, for the inner product between the residual and the identity matrix (the second part of matrix $\boldsymbol{A}$) it holds that $\langle \boldsymbol{r}_{(k)}, \boldsymbol{e}_{j_k} \rangle = r_{(k),j_k} = 0$, for every $j_k \in \mathcal{S}_k$. Thus, the column vector $\boldsymbol{e}_{j_k}$ of matrix $\boldsymbol{I}_N$ cannot be selected in subsequent iterations.*

Remark 4.4. *Considering the complexity of the efficient implementation of GARD, the algorithm speeds up in cases where the fraction of the outliers is very low, i.e., the outlier vector is very sparse ($S << N$).*

Remark 4.5. *The proposed scheme should not be confused with other OMP-based schemes, such as Robust OMP in [75]; although both are OMP-based, they perform in a distinctive manner and for dissimilar purposes. As both the selection step as well as the minimization step work quite different, GARD selects a column, based on the residual and performs an ordinary LS procedure. On the other hand, ROMP selects a column based on the residual pseudo-values and then solves a weighted LS minimization task. However, the major drawback of the ROMP is that the scheme does not perform any kind of outlier identification.*

4.3 Theoretical Analysis

This section is devoted to the study of the basic properties that the novel iterative robust scheme, GARD, satisfies. First, the convergence properties of the proposed scheme are derived. In the sequel, it is shown that GARD recovers the exact solution (if this is unique), under the condition of a derived bound in terms of the Restricted Isometry Property (RIP) constant and in the presence of outlier noise only. Finally, for the case of both inlier and outlier noise, bounds on the recovery of the sparse outlier support and the reconstruction error are also presented.

4.3.1 General Results

Lemma 4.1. *At every $k \leq N - M$ step, GARD selects a column vector $\boldsymbol{e}_{j_k}$ from matrix $\boldsymbol{I}_N$, that is linearly independent of all the column vectors of matrix $\boldsymbol{A}_{(k-1)}$. Hence, $\boldsymbol{A}_{(k)}$ has full rank and the solution to the Least Squares task at each step is unique.*

Proof. The proof relies on mathematical induction. At the initial step, the matrix $\boldsymbol{A}_{(0)} = \boldsymbol{X}$ has been assumed to be full rank, hence the solution of the LS task is unique. Suppose that at $(k-1)$-th step ($k \in \mathbb{N}^*$), matrix $\boldsymbol{A}_{(k-1)}$ is full rank, hence let $\hat{\boldsymbol{z}}_{(k-1)}$ denote the unique solution of the LS task and $\boldsymbol{r}_{(k-1)} = \boldsymbol{y} - \boldsymbol{A}_{(k-1)}\hat{\boldsymbol{z}}_{(k-1)}$, the respective residual. Assume that at the k-th step, the j_k-th column of matrix $\boldsymbol{I}_N$ is selected from the set $\mathcal{S}^c_{k-1}$. It is readily seen that the columns of the augmented matrix at this step, i.e., the columns of matrix $\boldsymbol{A}_{(k)} = [\boldsymbol{A}_{(k-1)} \ \boldsymbol{e}_{j_k}]$, are linearly independent. Notice here that $r_{(k-1),j_k} \neq 0$, otherwise either the index would have not been selected or the residual vector would be equal to zero. Next, we assume that the columns of matrix $\boldsymbol{A}_{(k)}$ are linearly dependent, i.e., there exists $\boldsymbol{v} \neq \boldsymbol{0}$ such that $\boldsymbol{e}_{j_k} = \boldsymbol{A}_{(k-1)}\boldsymbol{v}$. Also let $\mathbf{z}_{(k-1)} = \hat{\boldsymbol{z}}_{(k-1)} + r_{(k-1),j_k}\boldsymbol{v}$. Thus, it is readily obtained that

$$\begin{aligned}||\mathbf{r}_{(k-1)}||_2 = ||\boldsymbol{y} - \boldsymbol{A}_{(k-1)}\mathbf{z}_{(k-1)}||_2 = \\ = ||\boldsymbol{y} - \boldsymbol{A}_{(k-1)}\hat{\boldsymbol{z}}_{(k-1)} - r_{(k-1),j_k}\boldsymbol{A}_{(k-1)}\boldsymbol{v}||_2 = \\ = ||\boldsymbol{r}_{(k-1)} - r_{(k-1),j_k}\boldsymbol{e}_{j_k}||_2 < ||\boldsymbol{r}_{(k-1)}||_2,\end{aligned}$$

which contradicts the fact that the residual of the LS solution attains the smallest norm. Thus, all the selected columns of matrix $\boldsymbol{A}_{(k)}$ are linearly independent. □

Theorem 4.1. *The norm of GARD's residual vector*

$$\boldsymbol{r}_{(k)} = \boldsymbol{y} - \boldsymbol{A}_{(k)}\hat{\boldsymbol{z}}_{(k)}$$

is strictly decreasing. Moreover, the algorithm will always converge.

Proof. Let $\hat{\boldsymbol{z}}_{(k-1)}$ denote the unique LS solution (Lemma 4.1) and $\boldsymbol{r}_{(k-1)} = \boldsymbol{y} - \boldsymbol{A}_{(k-1)}\hat{\boldsymbol{z}}_{(k-1)}$ the respective residual at the $(k-1)$-th step. At the next step, the algorithm selects the column j_k and augments matrix $\boldsymbol{A}_{(k-1)}$ by the column $\boldsymbol{e}_{j_k}$ in order to form matrix $\boldsymbol{A}_{(k)}$. Let $\hat{\boldsymbol{z}}_{(k)}$ denote the unique solution of the LS task at the k-th step (Lemma 4.1) and $\boldsymbol{r}_{(k)} = \boldsymbol{y} - \boldsymbol{A}_{(k)}\hat{\boldsymbol{z}}_{(k)}$ the respective residual. Moreover, let $\mathrm{P}_{(k)}(\boldsymbol{z}) = ||\boldsymbol{y} - \boldsymbol{A}_{(k)}\boldsymbol{z}||_2$, be a cost function defined for every vector $\boldsymbol{z} \in \mathbb{R}^{M+k}$ at the k-th step. Thus,

$$||\boldsymbol{r}_{(k)}||_2 = \mathrm{P}_{(k)}(\hat{\boldsymbol{z}}_{(k)}) \leq \mathrm{P}_{(k)}(\boldsymbol{z}), \tag{4.3}$$

holds for every $\boldsymbol{z} \in \mathbb{R}^{M+k}$. Finally, let $\mathbf{z}_{(k)} = (\hat{\boldsymbol{z}}_{(k-1)}^T, r_{(k-1),j_k})^T$, where $r_{(k-1),j_k}$ is the j_k-th coordinate of the residual $\boldsymbol{r}_{(k-1)}$. Hence, it is obtained that

$$\begin{aligned}\mathrm{P}_{(k)}(\mathbf{z}_{(k)}) = ||\boldsymbol{y} - \boldsymbol{A}_{(k)}\mathbf{z}_{(k)}||_2 = \\ = ||\boldsymbol{y} - \boldsymbol{A}_{(k-1)}\hat{\boldsymbol{z}}_{(k-1)} - r_{(k-1),j_k}\boldsymbol{e}_{j_k}||_2 = \\ = ||\boldsymbol{r}_{(k-1)} - r_{(k-1),j_k}\boldsymbol{e}_{j_k}||_2 < ||\boldsymbol{r}_{(k-1)}||_2.\end{aligned} \tag{4.4}$$

Combining (4.3) and (4.4) leads to

$$||\boldsymbol{r}_{(k)}||_2 < ||\boldsymbol{r}_{(k-1)}||_2. \tag{4.5}$$

Since $\boldsymbol{y} \in \mathbb{R}^N$, the residual equals zero as soon as $N-M$ columns are selected. However, since the noise bound is a positive value, the algorithm terminates at the first step $k < N-M$, where the residual's norm drops below ϵ. □

4.3.2 The Presence of Outliers Only

The scenario where the signal is corrupted only by outliers is treated separately. This leads to a simplification of the provided analysis and an attractive presentation of the derived properties, compared to the noisy case. Moreover, the established results, pave the way for the study of the more complex case where inlier noise is also present.

In order to simplify the notation and reduce the size of the subsequent proofs, we orthonormalize $\boldsymbol{X}$ by the reduced QR decomposition, i.e., $\boldsymbol{X} = \boldsymbol{QR}$, where $\boldsymbol{Q}$ is a $N \times M$ matrix, whose columns form an orthonormal basis of the column space of $\boldsymbol{X}$, i.e., span($\boldsymbol{X}$), and $\boldsymbol{R}$ is a $M \times M$ upper triangular matrix. Since $\boldsymbol{X}$ has full column rank, the decomposition is unique; moreover, the matrix $\boldsymbol{R}$ is invertible. By using this decomposition, the split noise modeling described in equation (3.10) for $\boldsymbol{\eta} = \boldsymbol{0}$ (noiseless case) is expressed as

$$\boldsymbol{y} = \boldsymbol{Q}\underline{\boldsymbol{w}} + \underline{\boldsymbol{u}} = [\boldsymbol{Q}\ \boldsymbol{I}_N] \begin{pmatrix} \underline{\boldsymbol{w}} \\ \underline{\boldsymbol{u}} \end{pmatrix}, \tag{4.6}$$

where $\underline{\boldsymbol{w}} = \boldsymbol{R}\underline{\boldsymbol{\theta}}$. If $\underline{\boldsymbol{w}}$ is recovered, the unknown vector $\underline{\boldsymbol{\theta}}$ can also be recovered from $\underline{\boldsymbol{\theta}} = \boldsymbol{R}^{-1}\underline{\boldsymbol{w}}$. Also, by defining

$$\boldsymbol{\Phi} = [\boldsymbol{Q}\ \boldsymbol{I}_N], \tag{4.7}$$

Equation (4.6) can be cast as $\boldsymbol{y} = \boldsymbol{\Phi}\underline{\mathbf{z}}$, where $\underline{\mathbf{z}} = \begin{pmatrix} \underline{\boldsymbol{w}} \\ \underline{\boldsymbol{u}} \end{pmatrix}$.

In this section, the goal is to solve the following ℓ_0-norm minimization task with equality constraint:

$$\min_{\boldsymbol{w},\boldsymbol{u}} ||\boldsymbol{u}||_0, \text{ subject to } \ \boldsymbol{y} = \boldsymbol{Q}\boldsymbol{w} + \boldsymbol{u}. \tag{4.8}$$

It is further assumed that the vector $\underline{\boldsymbol{u}}$ is sparse over the support subset $\mathcal{S} \subset \mathcal{J} := \{1, \dots, N\}$, with $|\mathcal{S}| = S << N$; that is, $\underline{u}_i \neq 0$ for $i \in \mathcal{S}$ and $\underline{u}_i = 0$, for all $i \notin \mathcal{S}$. Moreover, $S < N/2$ should also be satisfied; if $S \geq N/2$, it seems unlikely that the task is solvable, see [32]. Also, let

$$\boldsymbol{\Phi}_{:|\mathcal{S}} = [\boldsymbol{Q}\ \boldsymbol{I}_{\mathcal{S}}], \tag{4.9}$$

denote the restriction of columns from the augmented matrix $\boldsymbol{\Phi}$ over its second part only, i.e., the identity matrix, over the set $\mathcal{S}$. Obviously, Equation (4.6) could also be written as

$$\boldsymbol{y} = \boldsymbol{\Phi}_{:|\mathcal{S}}\underline{\mathbf{z}}_{:|\mathcal{S}}, \tag{4.10}$$

where

$$\underline{\mathbf{z}}_{:|\mathcal{S}} = \begin{pmatrix} \underline{\boldsymbol{w}} \\ \underline{\boldsymbol{u}}_{\mathcal{S}} \end{pmatrix}. \tag{4.11}$$

Equation (4.10) and in particular matrix $\boldsymbol{\Phi}_{:|\mathcal{S}}$ reveals the hidden geometrical structure of the robust regression task. It should also be noted that, while $\underline{\boldsymbol{u}} \in \mathbb{R}^N$ has $N - S$ zero elements, the vector $\underline{\boldsymbol{u}}_{\mathcal{S}} \in \mathbb{R}^S$ and has no zero entries.

In the following, the notion of the *smallest principal angle* between subspaces is employed. Given the information concerning the index subset $\mathcal{S}$ (i.e., assuming that the support of the outlier vector is known), $\underline{\boldsymbol{w}}$ can be recovered if and only if the matrix $\boldsymbol{\Phi}_{:|\mathcal{S}}$ has full rank. The latter assumption can also be expressed in terms of the *smallest principal angle*, $\overset{\diamond}{\omega}_S$, between the subspace spanned by the columns of the regression matrix, i.e., span($\boldsymbol{Q}$) and the subspace spanned by the columns of $\boldsymbol{I}_S$, i.e., span($\boldsymbol{I}_S$).

Definition 4.1. *Let $\overset{\diamond}{\delta}_S$ be the smallest number that satisfies the inequality*

$$|\langle \boldsymbol{w}, \boldsymbol{u} \rangle| \leq \overset{\diamond}{\delta}_S ||\boldsymbol{w}||_2 ||\boldsymbol{u}||_2,$$

for all $\boldsymbol{w} \in$ span($\boldsymbol{Q}$) and $\boldsymbol{u} \in$ span($\boldsymbol{I}_S$). Then $\overset{\diamond}{\omega}_S = \arccos(\overset{\diamond}{\delta}_S)$ is the smallest principle angle between the spaces span($\boldsymbol{Q}$) *and* span($\boldsymbol{I}_S$).

However, since the support set of the outlier vector, $\underline{\boldsymbol{u}}$, is in general unknown to us, we resort to a generalized form of Definition 4.1, i.e.,

$$\delta_S = \max_{\mathcal{S}} \left\{ \overset{\diamond}{\delta}_S, \text{ for } \mathcal{S} \in \mathcal{T}_K^{\mathcal{J}}, \ K \leq S \right\}, \tag{4.12}$$

where the $\mathcal{T}_K^{\mathcal{J}}$ denotes the set of all possible K-th cardinality combinations of subsets over $\mathcal{J} = \{1, \ldots, N\}$ for $K = 1, \ldots, S$. Thus, the smallest principal angle between the regression subspace, span($\boldsymbol{Q}$), and all the at most S-dimensional outlier subspaces, span($\boldsymbol{I}_{\mathcal{S}}$) for all possible combinations of $\mathcal{S}$ such that $|\mathcal{S}| \leq S$, is defined as follows:

$$\omega_S = \arccos(\delta_S). \tag{4.13}$$

It can readily be seen that δ_S can be defined by employing only the value $K = S$ instead of all $K \leq S$. Furthermore, for any $\boldsymbol{w} \in \text{span}(\boldsymbol{Q})$ and any at most S-sparse vector $\boldsymbol{u}$ (regardless of its support) the important property holds:

$$|\langle \boldsymbol{w}, \boldsymbol{u} \rangle| \leq \delta_S ||\boldsymbol{w}||_2 ||\boldsymbol{u}||_2. \tag{4.14}$$

Remark 4.6. *The quantity $\delta_S \in [0, 1]$ (or equivalently $\omega_S \in [0°, 90°]$) is a measure of how well separated the regression subspace is from all the S-dimensional outlier subspaces.*

The following condition, which is also known as the Restricted Isometry Property (RIP), plays a central role in sparse optimization methods, [39]. Although the definition can be expressed with a more general matrix we present it with a specific one.

Definition 4.2 (Restricted Isometry Property-RIP)**.** *For an orthonormal matrix $\boldsymbol{Q}$ the constant $\mu_S > 0$, $S = 1, 2, ..., N$, is defined as the smallest number such that for the matrix defined in (4.9)*

$$(1 - \mu_S)||\boldsymbol{\alpha}_{:|\mathcal{S}}||_2^2 \leq ||\boldsymbol{\Phi}_{:|\mathcal{S}} \boldsymbol{\alpha}_{:|\mathcal{S}}||_2^2 \leq (1 + \mu_S)||\boldsymbol{\alpha}_{:|\mathcal{S}}||_2^2, \tag{4.15}$$

for all vectors $\boldsymbol{\alpha}$ and every set $\mathcal{S}$ with cardinality at most S.

Simply stated, the condition ensures that the matrix involved, is approximately an isometry. In [40] (Lemma III.1), it has been proved that for a matrix of the form in (4.9) and $\boldsymbol{Q}$ orthonormal, the smallest principal angle constant coincides with the one defined in the RIP condition, i.e., $\delta_S = \mu_S$, $S = 1, 2, ..., N$. Finally, the following theorem guarantees the uniqueness of the decomposition, see [32, 40].

Theorem 4.2. *Assume that the vector $\boldsymbol{y} \in \mathbb{R}^N$ can be decomposed as follows:*

$$\boldsymbol{y} = \boldsymbol{Q}\underline{\boldsymbol{w}} + \underline{\boldsymbol{u}},$$

where $\underline{\boldsymbol{w}} \in \mathbb{R}^M$ and $\underline{\boldsymbol{u}}$ is an at most S-sparse vector. If $\delta_{2S} < 1$ the decomposition is unique and the ℓ_0-norm minimization task has a unique solution.

One of the main theoretical results, established in this work is the following theorem, which guarantees the recovery of the support for the sparse outlier vector for the noiseless case. Moreover, it turns out that both the vector of unknowns and the outliers are recovered with zero error.

Theorem 4.3. *Let $\boldsymbol{X}$ be a full column rank matrix and assume that the vector of observations has a unique decomposition $\boldsymbol{y} = \boldsymbol{X}\underline{\boldsymbol{\theta}} + \underline{\boldsymbol{u}}$, such that $||\underline{\boldsymbol{u}}||_0 \leq S$ (at most S outliers exist in the $\boldsymbol{y}$ variable). If*

$$\delta_S < \sqrt{\frac{\min|\underline{u}|}{2||\underline{\boldsymbol{u}}||_2}}, \tag{4.16}$$

where $\min|\underline{u}|$ is the smallest absolute value of the sparse vector $\underline{\boldsymbol{u}}$ over the nonzero coordinates. Then, GARD guarantees that the unknown vector $\underline{\boldsymbol{\theta}}$ and the sparse outlier vector $\underline{\boldsymbol{u}}$ are recovered exactly, with no error.

Remark 4.7. *Since matrix $\boldsymbol{X}$ is assumed to be full rank, equation $\boldsymbol{y} = \boldsymbol{X}\underline{\boldsymbol{\theta}} + \underline{\boldsymbol{u}}$ could be transformed into* (4.6)*. Thus, the smallest principal angle defined in* (4.13) *is now involved.*

Remark 4.8. *The condition under which the measurement vector $\boldsymbol{y}$ can be uniquely decomposed into parts $\boldsymbol{Q}\underline{\boldsymbol{w}}$ plus $\underline{\boldsymbol{u}}$, is given in Theorem 4.2 (see also [32, 40]).*

Remark 4.9. *The bound that appears in* (4.16) *has also an interesting geometrical interpretation. The ratio, $\min|\underline{u}|/||\underline{\boldsymbol{u}}||_2$, corresponds to the cosine of the largest direction angle of vector $\underline{\boldsymbol{u}}$. Moreover, it can be readily seen that this ratio is no greater than* 1 *(attained only for* 1*-sparse vectors), which leads to the fact that the right hand side of* (4.16) *is bounded by $\sqrt{2}/2$. In other words, the condition of Theorem 4.3 forces ω_S to lie within the interval* $(45°, 90°]$.

Rather than delving into the main arguments of the proof, it is first required to establish the following proposition and lemmas.

Proposition 4.1. *Let $\boldsymbol{Q}$ be the orthonormal matrix of the reduced QR decomposition of the full rank matrix $\boldsymbol{X}$ and δ_S the smallest principal angle constant between the subspace spanned by* $\text{span}(\boldsymbol{Q})$ *and the subspace spanned by all the S-dimensional outlier subspaces. Then,*

$$||\boldsymbol{Q}^T\boldsymbol{v}||_2 \leq \delta_S||\boldsymbol{v}||_2 \tag{4.17}$$

holds for every vector $\boldsymbol{v} \in \mathbb{R}^N$ with $||\boldsymbol{v}||_0 \leq S$.

Proof. The proof is straightforward by the definition of δ_S and (4.14):

$$||\boldsymbol{Q}^T\boldsymbol{v}||_2^2 = |\langle\boldsymbol{v}, \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{v}\rangle| \leq \delta_S\|\boldsymbol{v}\|_2\|\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{v}\|_2 \leq \delta_S\|\boldsymbol{v}\|_2\|\boldsymbol{Q}\|_2\|\boldsymbol{Q}^T\boldsymbol{v}\|_2 = \delta_S\|\boldsymbol{v}\|_2\|\boldsymbol{Q}^T\boldsymbol{v}\|_2,$$

which leads to (4.17). □

Lemma 4.2. *Let the assumptions of Proposition 4.1 be satisfied and $\mathcal{S}$ be any non-empty subset of $\mathcal{J} = \{1, \ldots, N\}$ with cardinality $|\mathcal{S}| = K \leq S < N$. Then*

$$||\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}}||_2 \leq \delta_S \tag{4.18}$$

holds for every such set $\mathcal{S}$.

Proof. Let $\underset{\sim}{\boldsymbol{v}} \neq \boldsymbol{0}$ be a vector of $\mathbb{R}^K$, $K \leq S$. It is clear that $\boldsymbol{I}_{\mathcal{S}}\underset{\sim}{\boldsymbol{v}} = \boldsymbol{v} \in \mathbb{R}^N$, with $||\boldsymbol{v}||_0 \leq S$ and $||\underset{\sim}{\boldsymbol{v}}||_2 = ||\boldsymbol{v}||_2$. Hence, it holds that

$$||\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}}\underset{\sim}{\boldsymbol{v}}||_2 = ||\boldsymbol{Q}^T\boldsymbol{v}||_2 \leq \delta_S||\underset{\sim}{\boldsymbol{v}}||_2,$$

due to Proposition 4.1. Since the matrix 2-norm is subordinate, by definition we have

$$||\boldsymbol{Q}^T\boldsymbol{I}_S||_2 = \max_{\underset{\sim}{\boldsymbol{v}}\neq\mathbf{0}} \frac{||\boldsymbol{Q}^T\boldsymbol{I}_S\underset{\sim}{\boldsymbol{v}}||_2}{||\underset{\sim}{\boldsymbol{v}}||_2}.$$

Since it is proved that $||\boldsymbol{Q}^T\boldsymbol{I}_S\underset{\sim}{\boldsymbol{v}}||_2/||\underset{\sim}{\boldsymbol{v}}||_2 \leq \delta_s$ for every $\underset{\sim}{\boldsymbol{v}} \neq \mathbf{0}$ inequality (4.18) directly follows. □

The importance of Lemma 4.2 is twofold. First of all, it is a bound on the 2-norm of the matrix $\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}}$. Moreover, since $||\boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}}||_2 = ||\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}}||_2^2$ and assuming that (4.16) is satisfied, we have that

$$||\boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}}||_2 \leq \delta_S^2 < 1/2, \tag{4.19}$$

which leads to the fact that the matrix

$$\boldsymbol{W}_{(\mathcal{S})} = \boldsymbol{I}_K - \boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}} \tag{4.20}$$

is invertible (see Appendix A). The matrix $\boldsymbol{W}_{(\mathcal{S})}$ is specified by the set, $\mathcal{S}$, on which the columns of the identity matrix are restricted, and its cardinality, $|\mathcal{S}| = K \leq S$. The matrix notation with the index of the set in parenthesis is adopted in order to distinguish from the notation of the restriction of its columns over the set. Furthermore, the following bound is obtained

$$||\boldsymbol{W}_{(\mathcal{S})}^{-1}||_2 \leq (1 - ||\boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}}||_2)^{-1} < 2, \tag{4.21}$$

due to a very popular lemma of linear algebra (see Appendix A).

Lemma 4.3. *Let the assumptions of Lemma 4.2 be satisfied. Then*

$$||\boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{v}||_2 \leq \delta_S^2||\boldsymbol{v}||_2 \tag{4.22}$$

holds for every vector $\boldsymbol{v} \in \mathbb{R}^N$, with $||\boldsymbol{v}||_0 \leq S$.

Proof. Let $\mathcal{S}'$ denote the support set of the vector $\boldsymbol{v}$, with $|\mathcal{S}'| = K \leq S$. The tricky part of the proof is that the support of the K-sparse vector $\boldsymbol{v} \in \mathbb{R}^N$ does not necessarily coincide with the set $\mathcal{S}$; however, both sets, $\mathcal{S}, \mathcal{S}'$, are S-sparse at most. Thus, by using $\boldsymbol{v}_{\mathcal{S}'} \in \mathbb{R}^K$ to denote the non-sparse vector we have $\boldsymbol{v} = \boldsymbol{I}_{\mathcal{S}'}\boldsymbol{v}_{\mathcal{S}'}$ (notice that $\|\boldsymbol{v}\|_2 = \|\boldsymbol{v}_{\mathcal{S}'}\|_2$). Hence, due to the sub-multiplicative property of the matrix 2-norm, we have

$$||\boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{v}||_2 = ||\boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}'}\boldsymbol{v}_{\mathcal{S}'}||_2 \leq ||\boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{Q}||_2||\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}'}||_2||\boldsymbol{v}_{\mathcal{S}'}||_2 \leq \delta_S^2||\boldsymbol{v}||_2,$$

where we have used $||\boldsymbol{I}_{\mathcal{S}}^T\boldsymbol{Q}||_2 = ||\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}}||_2$ and both the results of Proposition 4.1 and Lemma 4.2. □

Remark 4.10. *For the simplification of the calculations in the following proofs, we make use of an equivalent GARD implementation. Instead of taking into account equation* (3.10) *and letting* $\mathrm{GARD}(\boldsymbol{X}, \boldsymbol{y}, \epsilon)$ *run, we employ the QR decomposition of* $\boldsymbol{X}$, *i.e.,* $\boldsymbol{X} = \boldsymbol{Q}\boldsymbol{R}$, *so that* $\boldsymbol{y} = \boldsymbol{Q}\underline{\boldsymbol{w}} + \underline{\boldsymbol{u}} + \boldsymbol{\eta}$ *and let* $\mathrm{GARD}(\boldsymbol{Q}, \boldsymbol{y}, \epsilon)$ *run. Since for the two implementations,* $\boldsymbol{H} = \boldsymbol{Q}\boldsymbol{Q}^T$, *the respective residuals at each step are equal; however, their solutions are not, albeit related. Thus, running GARD with the regression matrix* $\boldsymbol{X}$ *and solution* $\hat{\boldsymbol{z}}_{(k)} := \begin{pmatrix} \hat{\boldsymbol{\theta}}_{(k)} \\ \hat{\boldsymbol{u}}_{(k)} \end{pmatrix}$ *is equivalent to running GARD with the regression matrix* $\boldsymbol{Q}$ *and respective solution* $\hat{\mathbf{z}}_{(k)} := \begin{pmatrix} \hat{\boldsymbol{w}}_{(k)} \\ \hat{\boldsymbol{u}}_{(k)} \end{pmatrix}$, *taking into account that at each step,* $\hat{\boldsymbol{\theta}}_{(k)} = \boldsymbol{R}^{-1}\hat{\boldsymbol{w}}_{(k)}$.

Proof of the main Theorem 4.3

Proof. Since the matrix $\boldsymbol{X}$ is full rank, the observation vector, $\boldsymbol{y}$, can be uniquely decomposed as in (4.6) via the QR decomposition, $\boldsymbol{X} = \boldsymbol{QR}$. Suppose we let GARD run with $\epsilon = 0$ and the replacement of matrix $\boldsymbol{X}$ by matrix $\boldsymbol{Q}$ ($\boldsymbol{\Phi}$ is also used instead of $\boldsymbol{A}$), which serves the purposes of the proof. Moreover, let $\mathcal{S} = \text{supp}(\underline{\boldsymbol{u}}) \subset \mathcal{J}$ denote the support set for the sparse outlier vector, with $|\mathcal{S}| \leq S$.

At the initial step of GARD, the LS solution, $\hat{\boldsymbol{w}}_{(0)}$, is computed over the columns of matrix $\boldsymbol{Q}$. The corresponding residual is $\boldsymbol{r}_{(0)} = \boldsymbol{y} - \boldsymbol{Q}\hat{\boldsymbol{w}}_{(0)} = \boldsymbol{y} - \boldsymbol{QQ}^T\boldsymbol{y}$. The matrix $\boldsymbol{P_Q} := \boldsymbol{QQ}^T$ is the projection matrix onto the range of matrix $\boldsymbol{Q}$. Thus, taking into account (4.6), the residual is expressed as

$$\boldsymbol{r}_{(0)} = (\boldsymbol{I}_N - \boldsymbol{QQ}^T)\underline{\boldsymbol{u}},$$

due to the fact that $(\boldsymbol{I}_N - \boldsymbol{P_Q})\boldsymbol{Q}\underline{\boldsymbol{w}} = \boldsymbol{0}$.

At the first step, in order to ensure a selection from the correct subset, $\mathcal{S}$, we impose

$$|r_{(0),i}| > |r_{(0),j}|, \ \forall \ i \in \mathcal{S} \text{ and } j \in \mathcal{S}^c. \tag{4.23}$$

The basic concept of the proof is to obtain lower and upper bounds for the left and right part of equation (4.23). Employing Lemma 4.3, the left part is bounded below by

$$\begin{aligned} |r_{(0),i}| &= |\langle \boldsymbol{r}_{(0)}, \boldsymbol{e}_i \rangle| = |\langle \underline{\boldsymbol{u}} - \boldsymbol{QQ}^T\underline{\boldsymbol{u}}, \boldsymbol{e}_i \rangle| \geq \\ &\geq |\underline{u}_i| - |\langle \boldsymbol{QQ}^T\underline{\boldsymbol{u}}, \boldsymbol{e}_i \rangle| = |\underline{u}_i| - |\boldsymbol{e}_i^T\boldsymbol{QQ}^T\underline{\boldsymbol{u}}| \geq \\ &\geq \min|\underline{u}| - \delta_S^2||\underline{\boldsymbol{u}}||_2. \end{aligned} \tag{4.24}$$

Following similar steps, the right part is upper bounded by

$$\begin{aligned} |r_{(0),j}| &= |\langle \boldsymbol{r}_{(0)}, \boldsymbol{e}_j \rangle| = |\langle \underline{\boldsymbol{u}} - \boldsymbol{QQ}^T\underline{\boldsymbol{u}}, \boldsymbol{e}_j \rangle| = \\ &= |\boldsymbol{e}_j^T\boldsymbol{QQ}^T\underline{\boldsymbol{u}}| \leq \delta_S^2||\underline{\boldsymbol{u}}||_2, \end{aligned} \tag{4.25}$$

using that $\langle \underline{\boldsymbol{u}}, \boldsymbol{e}_j \rangle = 0$, since $j \in \mathcal{S}^c$.

Hence, by imposing

$$\min|\underline{u}| - \delta_S^2||\underline{\boldsymbol{u}}||_2 > \delta_S^2||\underline{\boldsymbol{u}}||_2,$$

condition (4.23) is guaranteed and one of the correct columns, i.e., j_1, is bound to be selected at the first step (note that the selection does not necessarily correspond to the largest valued outlier).

Considering $\mathcal{S}_1 = \{j_1\} \subset \mathcal{S}$, the matrix of active columns, $\boldsymbol{\Phi}_{(1)} = [\boldsymbol{Q} \ \boldsymbol{e}_{j_1}]$, is augmented and the new residual is computed with the requirement of the inversion of

$$\boldsymbol{\Phi}_{(1)}^T\boldsymbol{\Phi}_{(1)} = \begin{bmatrix} \boldsymbol{I}_M & \boldsymbol{Q}^T\boldsymbol{e}_{j_1} \\ \boldsymbol{e}_{j_1}^T\boldsymbol{Q} & 1 \end{bmatrix}.$$

Taking into account that $\boldsymbol{I}_M$ is invertible and $\beta = 1 - ||\boldsymbol{Q}^T\boldsymbol{e}_{j_1}||_2^2 > 1/2$ (inequality (4.19) for $|\mathcal{S}| = 1$) and using the *Matrix Inversion Lemma* (MIL) in block form (see Appendix B), we obtain:

$$(\boldsymbol{\Phi}_{(1)}^T\boldsymbol{\Phi}_{(1)})^{-1} = \begin{bmatrix} \boldsymbol{I}_M + \boldsymbol{Q}^T\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T\boldsymbol{Q}/\beta & -\boldsymbol{Q}^T\boldsymbol{e}_{j_1}/\beta \\ -\boldsymbol{e}_{j_1}^T\boldsymbol{Q}/\beta & 1/\beta \end{bmatrix}.$$

After a few elementary algebra calculations

$$\boldsymbol{\Phi}_{(1)}(\boldsymbol{\Phi}_{(1)}^T\boldsymbol{\Phi}_{(1)})^{-1}\boldsymbol{\Phi}_{(1)}^T = \boldsymbol{Q}\boldsymbol{Q}^T + \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{Q}^T/\beta - \boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{Q}^T/\beta - \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T/\beta + \boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T/\beta.$$

Hence, the new residual, $\boldsymbol{r}_{(1)} = \boldsymbol{y} - \boldsymbol{\Phi}_{(1)}(\boldsymbol{\Phi}_{(1)}^T\boldsymbol{\Phi}_{(1)})^{-1}\boldsymbol{\Phi}_{(1)}^T\boldsymbol{y}$, is cast as

$$\boldsymbol{r}_{(1)} = (\boldsymbol{I}_N - \boldsymbol{Q}\boldsymbol{Q}^T - \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{Q}^T/\beta + \boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{Q}^T/\beta + \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T/\beta - \boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T/\beta)\underline{\boldsymbol{u}}. \tag{4.26}$$

The relation in (4.26) is further simplified by the use of the following decomposition for the outlier vector:

$$\underline{\boldsymbol{u}} = \mathrm{F}_{\mathcal{S}_1}(\underline{\boldsymbol{u}}) + \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}}), \tag{4.27}$$

where $\mathrm{F}_{\mathcal{S}_1}(\underline{\boldsymbol{u}}) = \underline{u}_{j_1}\boldsymbol{e}_{j_1}$ and $\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}})$ is the vector which has the same elements as $\underline{\boldsymbol{u}}$ over the set $\mathcal{S}\setminus\mathcal{S}_1$ and zero at its j_1-th coordinate. Obviously, the second term in the right hand side of (4.27) is an $(S-1)$-sparse vector at most and its support is a subset of $\mathcal{S}$. Thus, we have:

$$\boldsymbol{r}_{(1)} = (\boldsymbol{I}_N - \boldsymbol{Q}\boldsymbol{Q}^T - \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{Q}^T/\beta + \boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{Q}^T/\beta)\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}}) = \boldsymbol{u}_{(1)} - \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{u}_{(1)}, \tag{4.28}$$

where

$$\boldsymbol{u}_{(1)} = \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}}) + \frac{1}{\beta}\left(\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{Q}^T\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}})\right)\cdot\boldsymbol{e}_{j_1}, \tag{4.29}$$

It should also be noted that $\mathrm{supp}(\boldsymbol{u}_{(1)}) = \mathrm{supp}(\underline{\boldsymbol{u}}) = \mathcal{S}$; however, their values at the j_1-th coordinate are not equal. Following a similar rational, for the next step, we impose $|r_{(1),i}| > |r_{(1),j}|$ for all $i \in \mathcal{S}\setminus\mathcal{S}_1$ and $j \in \mathcal{S}^c$. Hence, using lower and upper bounds leads to

$$\begin{aligned}|r_{(1),i}| &= |\langle\boldsymbol{r}_{(1)},\boldsymbol{e}_i\rangle| = |\langle\boldsymbol{u}_{(1)} - \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{u}_{(1)},\boldsymbol{e}_i\rangle| \geq \\ &\geq |\underline{u}_i| - |\boldsymbol{e}_i^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{u}_{(1)}| \geq \min|\underline{u}| - \delta_S^2||\boldsymbol{u}_{(1)}||_2,\end{aligned} \tag{4.30}$$

where it is employed that $\langle\boldsymbol{e}_{j_1},\boldsymbol{e}_i\rangle = 0$ for every $i \in \mathcal{S}\setminus\mathcal{S}_1$. Moreover

$$\begin{aligned}|r_{(1),j}| &= |\langle\boldsymbol{r}_{(1)},\boldsymbol{e}_j\rangle| = |\langle\boldsymbol{u}_{(1)} - \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{u}_{(1)},\boldsymbol{e}_j\rangle| = \\ &= |\boldsymbol{e}_j^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{u}_{(1)}| \leq \delta_S^2||\boldsymbol{u}_{(1)}||_2,\end{aligned} \tag{4.31}$$

where the relationship $\langle\boldsymbol{u}_{(1)},\boldsymbol{e}_j\rangle = 0$ is used, for every $j \in \mathcal{S}^c$, as well as Lemma 4.3. By imposing $\min|\underline{u}| - \delta_S^2||\boldsymbol{u}_{(1)}||_2 > \delta_S^2||\boldsymbol{u}_{(1)}||_2$, leads equivalently to

$$\delta_S < \sqrt{\frac{\min|\underline{u}|}{2||\boldsymbol{u}_{(1)}||_2}}. \tag{4.32}$$

Although (4.32), seems inadequate, it can be proved indeed that it always holds true, provided (4.16) is satisfied. One needs to prove that $||\underline{\boldsymbol{u}}||_2 > ||\boldsymbol{u}_{(1)}||_2$, which is equivalent to showing that $|\left(\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{Q}^T\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}})\right)/\beta| < |\underline{u}_{j_1}|$, using the aforementioned decompositions of $\underline{\boldsymbol{u}}$, $\boldsymbol{u}_{(1)}$ and the Pythagorean Theorem. Thus, it is obtained that

$$|\left(\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{Q}^T\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}})\right)/\beta| \leq 2\delta_S^2||\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}})||_2 < \min|\underline{u}| \leq |\underline{u}_{j_1}|,$$

due to $\beta > 1/2$, (4.22), (4.16) and the fact that the inequality $||\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}})||_2 < ||\underline{\boldsymbol{u}}||_2$ holds for any non-empty set $\mathcal{S}_1$. Hence, it is guaranteed that a second index which belongs to the support set $\mathcal{S}$ is selected.

At the k-th step $\mathcal{S}_k = \{j_1, j_2, ..., j_k\} \subset \mathcal{S}$ and the matrix that corresponds to the set, $\widetilde{\mathcal{S}}_k$, of active columns is $\boldsymbol{\Phi}_{(k)} = [\boldsymbol{Q}\ \boldsymbol{I}_{\mathcal{S}_k}]$. Once again, by the use of the MIL for the inversion of $\boldsymbol{\Phi}_{(k)}^T \boldsymbol{\Phi}_{(k)}$ the new residual is expressed as follows:

$$\begin{aligned} \boldsymbol{r}_{(k)} &= \left(\boldsymbol{I}_N - \boldsymbol{Q}\boldsymbol{Q}^T - \boldsymbol{Q}\boldsymbol{Q}^T \boldsymbol{I}_{\mathcal{S}_k} \boldsymbol{W}_{(\mathcal{S}_k)}^{-1} \boldsymbol{I}_{\mathcal{S}_k}^T \boldsymbol{Q}\boldsymbol{Q}^T + \boldsymbol{I}_{\mathcal{S}_k} \boldsymbol{W}_{(\mathcal{S}_k)}^{-1} \boldsymbol{I}_{\mathcal{S}_k}^T \boldsymbol{Q}\boldsymbol{Q}^T \right) \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_k}(\underline{\boldsymbol{u}}) = \\ &= \boldsymbol{u}_{(k)} - \boldsymbol{Q}\boldsymbol{Q}^T \boldsymbol{u}_{(k)}, \end{aligned} \tag{4.33}$$

where (4.20) and the following identities are employed:

$$\underline{\boldsymbol{u}} = \mathrm{F}_{\mathcal{S}_k}(\underline{\boldsymbol{u}}) + \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_k}(\underline{\boldsymbol{u}}), \tag{4.34}$$

$$\boldsymbol{u}_{(k)} = \boldsymbol{I}_{\mathcal{S}_k} \boldsymbol{W}_{(\mathcal{S}_k)}^{-1} \boldsymbol{I}_{\mathcal{S}_k}^T \boldsymbol{Q}\boldsymbol{Q}^T \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_k}(\underline{\boldsymbol{u}}) + \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_k}(\underline{\boldsymbol{u}}). \tag{4.35}$$

It is readily seen that $\mathrm{supp}(\boldsymbol{u}_{(k)}) = \mathrm{supp}(\underline{\boldsymbol{u}}) = \mathcal{S}$ still holds. For a correct outlier index selection from the set $\mathcal{S}$, at the $(k+1)$-th step, one needs to impose $|r_{(k),i}| > |r_{(k),j}|$ for all $i \in \mathcal{S} \setminus \mathcal{S}_k$ and $j \in \mathcal{S}^c$. Using lower and upper bounds on the inner products, one obtains relations similar to (4.30), (4.31) with $\boldsymbol{u}_{(k)}$ instead of $\boldsymbol{u}_{(1)}$, which leads to

$$\delta_S < \sqrt{\frac{\min |\underline{u}|}{2||\boldsymbol{u}_{(k)}||_2}}. \tag{4.36}$$

The proof ends, by showing that the last bound is looser than that of inequality (4.16), simply by proving that $||\boldsymbol{u}_{(k)}||_2 < ||\underline{\boldsymbol{u}}||_2$ for all $k = 1, 2, ..., S-1$. Using the decompositions of these vectors (4.34), (4.35) and the Pythagorean Theorem, it suffices to show that $||\boldsymbol{W}_{(\mathcal{S}_k)}^{-1} \boldsymbol{I}_{\mathcal{S}_k}^T \boldsymbol{Q}\boldsymbol{Q}^T \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_k}(\underline{\boldsymbol{u}})||_2 < ||\mathrm{F}_{\mathcal{S}_k}(\underline{\boldsymbol{u}})||_2$, which follows from the fact that

$$||\boldsymbol{W}_{(\mathcal{S}_k)}^{-1} \boldsymbol{I}_{\mathcal{S}_k}^T \boldsymbol{Q}\boldsymbol{Q}^T \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_k}(\underline{\boldsymbol{u}})||_2 \leq ||\boldsymbol{W}_{(\mathcal{S}_k)}^{-1}||_2 ||\boldsymbol{I}_{\mathcal{S}_k}^T \boldsymbol{Q}\boldsymbol{Q}^T \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_k}(\underline{\boldsymbol{u}})||_2 < \min |\underline{u}| \leq ||\mathrm{F}_{\mathcal{S}_k}(\underline{\boldsymbol{u}})||_2, \tag{4.37}$$

where we employed the sub-multiplicative property of the matrix 2-norm, inequality (4.21), Lemma 4.3 and (4.16).

Thus, at the final selection step, $k+1 = S$, the final index, j_{k+1}, that belongs to the set $\mathcal{S}$ is selected and the correct support is recovered; that is, $\mathcal{S}_{k+1} = \mathcal{S}$. Hence, the linear subspace, onto which the measurement vector $\boldsymbol{y}$ lies, is formed. In turn, this results to a LS solution of zero error for GARD, i.e., $\hat{\mathbf{z}}_{(k+1)} = \underline{\mathbf{z}}_{:|\mathcal{S}}$; hence, it follows that $\hat{\boldsymbol{\theta}}_{(k+1)} = \boldsymbol{R}^{-1}\underline{\boldsymbol{w}} = \underline{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{u}}_{(k+1)} = \underline{\boldsymbol{u}}_{\mathcal{S}}$. □

4.3.3 The Presence of Both Inlier and Outlier Noise

In the following section, the theoretical results regarding the performance of GARD for the case both inlier bounded noise and outliers exist are provided.

Theorem 4.4. *Let $\boldsymbol{X}$ be a full column rank matrix and assume that $\boldsymbol{y} = \boldsymbol{X}\underline{\boldsymbol{\theta}} + \underline{\boldsymbol{u}} + \boldsymbol{\eta}$, such that $||\underline{\boldsymbol{u}}||_0 \leq S$ (at most S outliers exist in the $\boldsymbol{y}$ variable) and $\|\boldsymbol{\eta}\|_2 \leq \epsilon$. If*

$$\delta_S < \sqrt{\frac{\min |\underline{u}| - (2+\sqrt{6})\epsilon}{2||\underline{\boldsymbol{u}}||_2}}, \tag{4.38}$$

where $\min |\underline{u}|$ is the smallest absolute value of the sparse vector $\underline{\boldsymbol{u}}$ over the nonzero coordinates. Then, GARD guarantees that the support of the sparse outlier vector $\underline{\boldsymbol{u}}$ is recovered.

Proof of Theorem 4.4

Since Theorem 4.4 is the generalization of Theorem 4.3 (notice that if $\epsilon = 0$ (4.38) resorts to (4.16)), some intermediate results regarding the proof presented in Section 4.3.2 are also used here. Moreover, once again, for the simplification of the following proof, we let GARD run with matrix $\boldsymbol{Q}$, from the QR decomposition of $\boldsymbol{X}$. On the other hand, the technical parts that share obvious similarities are omitted.

Proof. Due to the QR decomposition of matrix $\boldsymbol{X}$ the equation (3.10) is expressed as:

$$\boldsymbol{y} = \boldsymbol{Q}\underline{\boldsymbol{w}} + \underline{\boldsymbol{u}} + \boldsymbol{\eta}, \tag{4.39}$$

where $\underline{\boldsymbol{w}} = \boldsymbol{R}\underline{\boldsymbol{\theta}}$. Since GARD initially performs a LS step, where the columns that participate in the representation are only those of matrix $\boldsymbol{Q}$, the obtained residual is $\boldsymbol{r}_{(0)} = (\boldsymbol{I}_N - \boldsymbol{Q}\boldsymbol{Q}^T)\boldsymbol{y}$. However, taking into account (4.39), we have the following expression for the initial residual:

$$\boldsymbol{r}_{(0)} = \underline{\boldsymbol{u}} + \boldsymbol{\eta} - \boldsymbol{Q}\boldsymbol{Q}^T\underline{\boldsymbol{u}} - \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{\eta}, \tag{4.40}$$

where the extra terms are due to the existence of the noise vector $\boldsymbol{\eta}$. Once again, we should impose (4.23). Also, recall in Theorem 4.3, that[1] $\delta_s < \sqrt{2}/2$. Thus, we have:

$$|r_{(0),i}| \geq |\underline{u}_i| - |\langle \boldsymbol{Q}\boldsymbol{Q}^T\underline{\boldsymbol{u}}, \boldsymbol{e}_i\rangle| - |\langle \boldsymbol{\eta}, \boldsymbol{e}_i\rangle| - |\langle \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{\eta}, \boldsymbol{e}_i\rangle| \geq \min|\underline{u}| - \delta_S^2||\underline{\boldsymbol{u}}||_2 - \epsilon - \epsilon\delta_S >$$
$$> \min|\underline{u}| - \delta_S^2||\underline{\boldsymbol{u}}||_2 - \epsilon - \frac{\epsilon}{\sqrt{2}} > \min|\underline{u}| - \delta_S^2||\underline{\boldsymbol{u}}||_2 - \epsilon - \epsilon\sqrt{\frac{3}{2}}$$

and

$$|r_{(0),j}| \leq \epsilon + \delta_S^2||\underline{\boldsymbol{u}}||_2 + \epsilon\delta_S < \epsilon + \delta_S^2||\underline{\boldsymbol{u}}||_2 + \frac{\epsilon}{\sqrt{2}} < \epsilon + \delta_S^2||\underline{\boldsymbol{u}}||_2 + \epsilon\sqrt{\frac{3}{2}},$$

for every $i \in \mathcal{S}$ and $j \in \mathcal{S}^c$, respectively. Thus, inequality (4.38) follows for the initial step. From this point, we proceed with the general k-th selection step. The second one is omitted, since it could be viewed as a special case of the general step; it was included in the proof of Theorem 4.3 for comprehension reasons only. It should also be noted, that the matrices augmented and inverted at each step, are those presented in the proof of Theorem 4.3. However, this is not the case for the solution and the residual, which is of our greatest interest.

The condition in (4.38) guarantees that at each selection step the support of our sparse outlier estimate is a subset of the sparse outlier vector, $\underline{\boldsymbol{u}}$. Simply stated $\mathcal{S}_k \subset \mathcal{S}$ and $\boldsymbol{\Phi}_{(k)} = [\boldsymbol{Q}\ \boldsymbol{I}_{\mathcal{S}_k}]$ is the matrix that corresponds to the set of active columns. Employing familiar techniques, we have the expression for the residual after the LS solution of the k-th step:

$$\boldsymbol{r}_{(k)} = \boldsymbol{u}_{(k)} + \boldsymbol{\eta}_{(k)} - \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{u}_{(k)} - \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{\eta}_{(k)}, \tag{4.41}$$

where $\boldsymbol{u}_{(k)}$ is the vector defined in (4.35) and

$$\boldsymbol{\eta}_{(k)} = \boldsymbol{I}_{\mathcal{S}_k}\boldsymbol{W}_{(\mathcal{S}_k)}^{-1}\boldsymbol{I}_{\mathcal{S}_k}^T\boldsymbol{Q}\boldsymbol{Q}^T\mathrm{F}_{\mathcal{J}\setminus\mathcal{S}_k}(\boldsymbol{\eta}) + \mathrm{F}_{\mathcal{J}\setminus\mathcal{S}_k}(\boldsymbol{\eta}), \tag{4.42}$$

[1] In the noiseless case, $\sqrt{2}/2$ is the upper bound for the left hand part of (4.16), which is achieved only for 1-sparse outlier vectors. Thus, if δ_S exceeds this bound, GARD has little chance in recovering the correct support, even in the presence of outlier noise only.

where $\mathcal{J} = \{1, 2, \ldots, N\}$; simply stated, the vector is decomposed into two disjoint subsets (recall that the noise vector $\boldsymbol{\eta}$ is not sparse). From (4.42), it is clear that the only difference between $\boldsymbol{\eta}_{(k)}$ and $\boldsymbol{\eta}$ lies solely on the elements indexed as $j \in \mathcal{S}_k$, i.e, indices that GARD has selected as outliers. Prior to completing the proof, it is required to establish appropriate bounds for the inner products $\left|\langle \boldsymbol{e}_i, \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{\eta}_{(k)}\rangle\right|$ and $\left|\langle \boldsymbol{e}_i, \boldsymbol{\eta}_{(k)}\rangle\right|$. Due to the Pythagorean Theorem, (4.18) and (4.21)

$$\left\|\boldsymbol{\eta}_{(k)}\right\|_2^2 = \left\|\mathrm{F}_{\mathcal{J}\setminus\mathcal{S}_k}(\boldsymbol{\eta})\right\|_2^2 + \left\|\boldsymbol{W}_{(\mathcal{S}_k)}^{-1}\boldsymbol{I}_{\mathcal{S}_k}^T\boldsymbol{Q}\boldsymbol{Q}^T\mathrm{F}_{\mathcal{J}\setminus\mathcal{S}_k}(\boldsymbol{\eta})\right\|_2^2 \leq \epsilon^2 + 2\epsilon^2 = 3\epsilon^2.$$

Hence,

$$\left|\boldsymbol{e}_i^T\boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{\eta}_{(k)}\right| \leq \delta_S \left\|\boldsymbol{\eta}_{(k)}\right\|_2 \leq \delta_S\epsilon\sqrt{3} < \epsilon\sqrt{\frac{3}{2}},$$

where we have also used the maximum bound for the smallest principal angle, i.e., that $\delta_S < \sqrt{2}/2$. Also, for all $i \in \mathcal{J} \setminus \mathcal{S}_k$, it holds that $\left|\langle \boldsymbol{e}_i, \boldsymbol{\eta}_{(k)}\rangle\right| = \left|\langle \boldsymbol{e}_i, \mathrm{F}_{\mathcal{J}\setminus\mathcal{S}_k}(\boldsymbol{\eta})\rangle\right| \leq \epsilon$. Thus, adopting bounds on the absolute value of the inner products we obtain

$$\begin{aligned} |r_{(k),i}| \geq |\underline{u}_i| - |\langle \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{u}_{(k)}, \boldsymbol{e}_i\rangle| - |\langle \boldsymbol{\eta}_{(k)}, \boldsymbol{e}_i\rangle| - |\langle \boldsymbol{Q}\boldsymbol{Q}^T\boldsymbol{\eta}_{(k)}, \boldsymbol{e}_i\rangle| > \\ > \min|\underline{u}| - \delta_S^2||\boldsymbol{u}_{(k)}||_2 - \epsilon - \epsilon\sqrt{\frac{3}{2}} \end{aligned}$$

and

$$|r_{(k),j}| < \delta_S^2||\boldsymbol{u}_{(k)}||_2 + \epsilon + \epsilon\sqrt{\frac{3}{2}},$$

for $i \in \mathcal{S} \setminus \mathcal{S}_k$ and $j \in \mathcal{S}^c$, respectively. Thus, imposing $|r_{(k),i}| > |r_{(k),j}|$, leads to

$$\delta_S < \sqrt{\frac{\min|\underline{u}| - (2+\sqrt{6})\epsilon}{2||\boldsymbol{u}_{(k)}||_2}},$$

which is satisfied, supposing (4.38) holds true. This is due to the fact that $\left\|\boldsymbol{u}_{(k)}\right\|_2 < \left\|\underline{\boldsymbol{u}}\right\|_2$ for all $k = 1, 2, ..., S-1$. Thus, the selection of the final index, j_{k+1}, that belongs to the set $\mathcal{S}$, is guaranteed. The procedure ends with the projection of the measurements' vector $\boldsymbol{y}$ onto the subspace originating from the columns of matrix $\boldsymbol{\Phi}_{:|\mathcal{S}}$, which produces an error $\|\boldsymbol{r}_{(k+1)}\|_2 \leq \epsilon$. □

Now that the bound on the support for the sparse outlier estimate is established the goal is to evaluate the error of the approximation.

Lemma 4.4. *Assume that there exists* $0 \leq \delta_S < 1$, *such that the RIP condition holds. It stems directly that the smallest singular value* σ_{min} *of the matrix* $\boldsymbol{\Phi}_{:|\mathcal{S}} = [\boldsymbol{Q}\ \boldsymbol{I}_{\mathcal{S}}]$ *is lower bounded by*

$$\sigma_{min} \geq \sqrt{1-\delta_S}. \tag{4.43}$$

Proof. Let $\boldsymbol{v}_{min}$ be the eigenvector which is associated with the smallest singular value of $\boldsymbol{\Phi}_{:|\mathcal{S}}$, then

$$\left\|\boldsymbol{\Phi}_{:|\mathcal{S}}\boldsymbol{v}_{min}\right\|_2^2 = \sigma_{min}^2\left\|\boldsymbol{v}_{min}\right\|_2^2.$$

Since (4.15) holds for every vector, (4.43) follows. □

Theorem 4.5. *If the conditions of Theorem 4.4 are satisfied, GARD approximates $\underline{\boldsymbol{\theta}}$, with estimate $\hat{\boldsymbol{\theta}}$, acquiring an error*

$$||\hat{\boldsymbol{\theta}} - \underline{\boldsymbol{\theta}}||_2 \leq \frac{\epsilon}{\tau\sqrt{1-\delta_S}}. \tag{4.44}$$

where τ is the smallest singular value of matrix $\boldsymbol{X}$.

Proof. The proof follows the same concepts as the stability result of Theorem 5.1 in [77]. Theorem 4.4 guarantees that the support is recovered at the last iteration step of GARD. Thus, the corresponding solution of GARD $(\boldsymbol{Q}, \boldsymbol{y}, \epsilon)$ is[2]:

$$\hat{\mathbf{z}} = \begin{pmatrix} \hat{\boldsymbol{w}} \\ \hat{\boldsymbol{u}}_{\mathcal{S}} \end{pmatrix} := \underset{\mathbf{z}}{\operatorname{argmin}} \, ||\boldsymbol{y} - \boldsymbol{\Phi}_{:|\mathcal{S}}\mathbf{z}||_2^2 = \boldsymbol{\Phi}^{\dagger}_{:|\mathcal{S}}\boldsymbol{y},$$

where $\boldsymbol{\Phi}^{\dagger}_{:|\mathcal{S}}$ denotes the Moore-Penrose pseudoinverse of matrix $\boldsymbol{\Phi}_{:|\mathcal{S}}$ in (4.9). Thus, according to (4.11), $\boldsymbol{y} = \boldsymbol{\Phi}_{:|\mathcal{S}}\underline{\mathbf{z}}_{:|\mathcal{S}} + \boldsymbol{\eta}$ and the estimated solution is expressed as:

$$\hat{\mathbf{z}} = \boldsymbol{\Phi}^{\dagger}_{:|\mathcal{S}}\boldsymbol{y} = \underline{\mathbf{z}}_{:|\mathcal{S}} + \boldsymbol{\Phi}^{\dagger}_{:|\mathcal{S}}\boldsymbol{\eta}.$$

Finally,

$$\begin{aligned} ||\hat{\mathbf{z}} - \underline{\mathbf{z}}_{:|\mathcal{S}}||_2 &\leq ||\boldsymbol{\Phi}^{\dagger}_{:|\mathcal{S}}\boldsymbol{\eta}||_2 \leq ||\boldsymbol{\Phi}^{\dagger}_{:|\mathcal{S}}||_2 \cdot ||\boldsymbol{\eta}||_2 \\ &\leq \sigma_{min}^{-1}\epsilon \leq \epsilon/\sqrt{1-\delta_S}, \end{aligned} \tag{4.45}$$

where we have also used that $||\boldsymbol{\Phi}^{\dagger}_{:|\mathcal{S}}||_2$ is bounded, engaging the smaller singular value σ_{min} of matrix $\boldsymbol{\Phi}_{:|\mathcal{S}}$, as well as (4.43). However, the original approximation process is performed with the regression matrix $\boldsymbol{X}$, instead of the orthonormal $\boldsymbol{Q}$, which was used for the simplification of the calculations. Hence, the result of the theorem follows from the fact that

$$\|\hat{\boldsymbol{\theta}} - \underline{\boldsymbol{\theta}}\|_2 \leq \|\boldsymbol{R}^{-1}\|_2 \|\hat{\boldsymbol{w}} - \underline{\boldsymbol{w}}\|_2 \leq \|\boldsymbol{R}^{-1}\|_2 \|\hat{\mathbf{z}} - \underline{\mathbf{z}}_{:|\mathcal{S}}\|_2,$$

where $\|\boldsymbol{R}^{-1}\|_2$ is the spectral norm of $\boldsymbol{R}^{-1}$ equal to $\sigma_{\min}(\boldsymbol{R})^{-1}$. Since $\boldsymbol{X} = \boldsymbol{QR}$, the smallest singular value of $\boldsymbol{R}$ equals[3] the smallest singular value, $\tau = \sigma_{\min}(\boldsymbol{X})$, of $\boldsymbol{X}$, thus the proof is complete. □

At this point, it is interesting to recall the discussion in Chapter 2, related to the leverage points. It is evident from (4.40) that the relation similar to (2.12) is:

$$r_i = (1 - h_{ii})\nu_i - \sum_{\substack{j=1 \\ j \neq i}}^{N} h_{ij}\nu_j, \tag{4.46}$$

where $\nu_i = \underline{u}_i + \eta_i$. This is the general case for the initial residual, while specifically for the noiseless case it is considered that $\eta_i = 0$. At each subsequent iteration, a similar condition holds and it is obtained with the replacement of ν_i by $\nu_{(k),i}$. Although the general feeling is that the LS residual is an unreliable source for the detection of the outliers, this is not entirely true. In order to justify our claim, the following analysis is provided.

[2] The indices referred to the iterative process are omitted.

[3] Matrices $\boldsymbol{X}$ and $\boldsymbol{R}$ share the same singular values.

First of all, there exist other values too, e.g., $h_{ii} \simeq 0.5$, for which the analysis based on (4.46) is blur. Secondly, the fact that the detection also depends on other aspects too, e.g., the dimension of the unknowns versus the number of observations, should also be considered. On the other hand, in the analysis provided via the perspective of sparse modeling and optimization, the whole issue is investigated from another point of view. The bounds derived in (4.16) and (4.38), where the values of the outliers are also taken into consideration, are greater guarantees for a safe detection via the residual. Thus, if such conditions are satisfied, the detection of an outlier is valid, despite the fact that leverage points may exist (possibly not in extreme sense of $h_{ii} = 1$). However, our humble opinion is that, if such abnormalities exist, it is rather unlikely that the derived conditions/bounds are satisfied.

Remark 4.11. *Let*

$$c = \sqrt{\frac{\min|\underline{u}| - (2+\sqrt{6})\epsilon}{2||\underline{\boldsymbol{u}}||_2}}.$$

Although c is readily computed, recall that δ_S is not, since it inherits the combinatorial nature of the problem for all the possible subsets of cardinality at most S. As a consequence, inequalities (4.16)*,* (4.38)*,* (4.43) *and* (4.44)*, cannot be verified in practice; nonetheless, they all serve significant theoretical purposes.*

Remark 4.12. *Combining* (4.44) *with* (4.38)*, we also have the following bound for the approximation of $\underline{\boldsymbol{\theta}}$:*

$$\left\|\hat{\boldsymbol{\theta}} - \underline{\boldsymbol{\theta}}\right\|_2 \leq \frac{\epsilon}{\tau\sqrt{1-c}}, \tag{4.47}$$

which due to its immediacy will be tested and verified later, in section 6.4. However, it is looser than that of (4.44)*.*

Remark 4.13. *The bound c in* (4.38) *clearly depends on the sparsity level, the values of the outlier vector and finally the level, ϵ, of the inlier noise. Also notice that, $\epsilon = 0$ leads to the bound of δ_S for the noiseless case, i.e., in* (4.16)*. Since in* (4.38) *more terms affect the bound, we cannot expect to recover the support perfectly in the case of both dense outlier noise and heavy inlier noise. Such a scenario would imply the bound on δ_S to be extremely tight, thus it is likely not satisfied. Finally, notice that $\min|\underline{u}|$ should be greater than $(2+\sqrt{6})\epsilon$, if we would like* (4.38) *to be valid.*

4.4 Experiments

In this section, GARD is directly compared against its competitors in various experiments. The set-up for each one of the methods established prior to GARD, is the following:

- M-est: The Tukey's biweight (or bisquare) robust (but nonconvex) function is employed, see Tables 2.1, 2.2 and Figure 2.2. This option is included in the MATLAB function "robustfit"; for $\hat{\sigma}$, we have used the default parameter value (unless otherwise stated), see [20, 19].
- SOCP: For the (SOCP) formulation the MATLAB function "SeDuMi" is employed; this is included in the optimization package "CVX" of Stanford University, (CVX RESEARCH:

Table 4.1: Computational costs for the methods that deal with the linear regression task. For GARD, k is the number of times the algorithm identifies an outlier. For GARD and SOCP total complexity is given, while for the rest of the methods the total complexity depends on the number of iterations for convergence. For ROMP, many parameters are involved, thus its complexity is not readily computable in closed form.

Algorithm	Complexity
GARD	$O\big(M^3/3 + k^3/2 + M^2(N+1+3k/2) + k^2(N+3M)/2 + kMN\big),\ k << N$
M-est	$O\big(M^3/3 + NM^2\big)/\text{step}$
SOCP	$O\big((N+M)^{2.5}N\big)$
ADMM	$O\big((N+M)^3/3 + N(N+M)^2\big)/\text{step}$
SBL	$O\big(M^3/3 + NM^2\big)/\text{step}$
ROMP	-

http://cvxr.com/ (6/02/2016)). The input parameter for SeDuMi is the bound of the inlier noise that is used for the definition of the second order cone.

- ADMM: For this method, the parameter λ is given for each experiment. Furthermore, the parameter, ρ, that is used for the soft- thresholding operator is also given initially (low) at $\rho = 10^{-4}$ and adapts at each step via $\rho_{(n)} = \min\{5, 1.1\rho_{(n-1)}\}$. Finally, a termination criterion is employed, when the norm of the estimate undergoes changes from one step to the next, less than the predefined threshold of 10^{-4}.

- SBL: The input parameters, $\sigma^2_{(0)}$, $\boldsymbol{\theta}_{(0)}$ and $\gamma_{(0),i}$ are initialized. Following [72, 73], we have also pruned the hyperparameters $\gamma_{(k),i}$ from future iterations, if they become smaller than a predefined threshold (set low to 10^{-5}). Although the computational cost for SBL is $O(M^3/3 + NM^2)$ per step, the total cost depends on other variables too; such are the number of hyperparameters that are pruned from future iterations, as well as the number of iterations until convergence. This is also the case for other methods, too.

- ROMP: Since the method employs an IRLS algorithm at each step, its complexity is not given in closed form. The Tukey's biweight function "robustfit" (with the default parameter settings unless stated) is used, once again, as in the M-est. The algorithm is chosen to terminate once the residual error drops below the bound of the inlier noise ϵ.

In the current section, we have tested and analyzed the performance of each related algorithm. The computational cost for each method is depicted in Table 4.1; it is observed that the total cost is computable in closed form only for SOCP and GARD. The experimental set-up parallels that of [40]. Our data $(y_i, \boldsymbol{x}_i),\ i = 1, 2, ..., N,\ \boldsymbol{x}_i \in \mathbb{R}^M$ is generated via equation (3.8); for the case where no inlier noise exists, we have set $\eta_i = 0$. The $\boldsymbol{x}_i$'s, i.e., the rows of matrix $\boldsymbol{X}$, are obtained by uniformly sampling an M-dimensional Latin hypercube centered around the origin. Finally, $\underline{\boldsymbol{\theta}} \in \mathbb{R}^M$ are random vectors with values chosen from the normal distribution with mean value 0 and standard deviation set to 5.

4.4.1 Mean-Square-Error (MSE) Estimation

In the first experiment, all methods are compared with respect to the mean-square-error (MSE), which is computed as the average, over 100 independent realizations at each outlier vector density, of the squared norm of the difference between the estimated vector $\hat{\boldsymbol{\theta}}$ and the unknown vector $\underline{\boldsymbol{\theta}}$. In parallel, the Mean Implementation Time (MIT) (in sec), that is required for each method in order to complete the estimation task, is measured for each outlier density. Aiming for detail, the given plots of the results are presented in a logarithmic scale, for each dimension, $M = 50, 100, 170$, of the unknown vector/signal $\underline{\boldsymbol{\theta}}$.

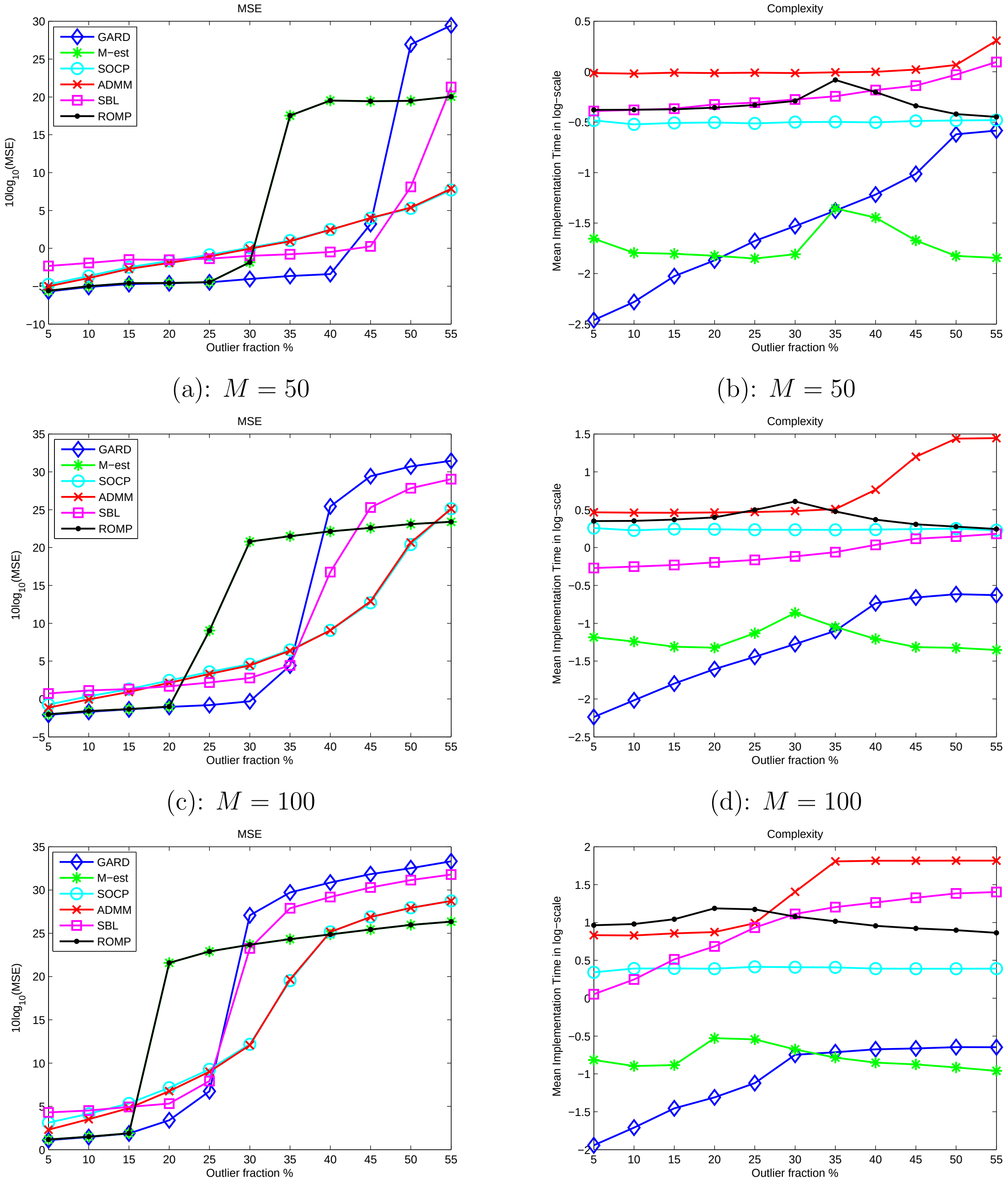
The values of the outliers are equal to ± 25, in S indices, uniformly sampled over N coordinates ($S < N$). Although outlier vectors are in general considered sufficiently sparse, in some experiments the density level is extended, so that each method is tested to its limits. The inlier noise vector has elements drawn from the standard Gaussian distribution, with $\sigma = 1$ and inlier noise bound ϵ.

The input parameter for GARD, SOCP and ROMP is the inlier noise bound ϵ. For ADMM, the regularization parameter is set to $\lambda = 1.2$. Note that all methods are carefully tuned so that their performance is optimized. For the SBL, a major drawback is its sensitivity to the choice of the initial values. Recall that, this is a non-convex method, which cannot guarantee that the global minimum is attained for each dimension M, while the time required for each implementation cannot be assured, since the number of iterations until convergence strongly depends on those parameters. Hence, for this method, random initialization is performed a number of times and the best solution is selected. Finally, it should be noted that the M-est does not require any predefined parameters.

In Figure 4.1 (a), (c) and (e), the MSE (in dBs) versus the fraction of the sparse outlier vector is depicted, for various dimensions of the unknown vector $\underline{\boldsymbol{\theta}}$. The Mean Implementation Time (MIT) is also plotted in logarithmic scale (right column) in Figure 4.1, for each dimensions of the unknown vector. Although the complexity for each method is already discussed in Table 4.1, in certain algorithms, the number of iterations until convergence greatly influences the required total implementation time. Observe that GARD attains the lowest MSE among the competitive methods for outlier fraction lower than 40%, 35% and 25% for dimensionality $M = 50, 100, 170$, respectively. The performance of M-est and ROMP is also notable, since both methods also attain a low MSE. However, this is only possible for outlier fraction of less than 25%, 20% and 15% (MSE equal to that of GARD). In particular, it appears that M-est and ROMP have identical performance, albeit ROMP combines two methods, which results to a higher computational cost.

It should also be noted that, in Figure 4.1 (c) and (e), the performance of GARD's competitors deteriorates for lower outlier fractions. However, the interesting *zone* of outlier vector density, in practice, is between 0% and 20% of the sample set size. Hence, GARD attains the lowest MSE within this sensitive zone. Finally, the experiments show that ADMM and SOCP attain similar performance, as expected, due to the fact that both address the same task.

Besides its superior performance with respect to the approximation error, GARD's computational requirements remain low. As shown in Figure 4.1 (b), (d) and (f), GARD appears to operate with the lower computational effort among its competitors, for outlier fraction less than 20%.



(a): $M = 50$

(b): $M = 50$

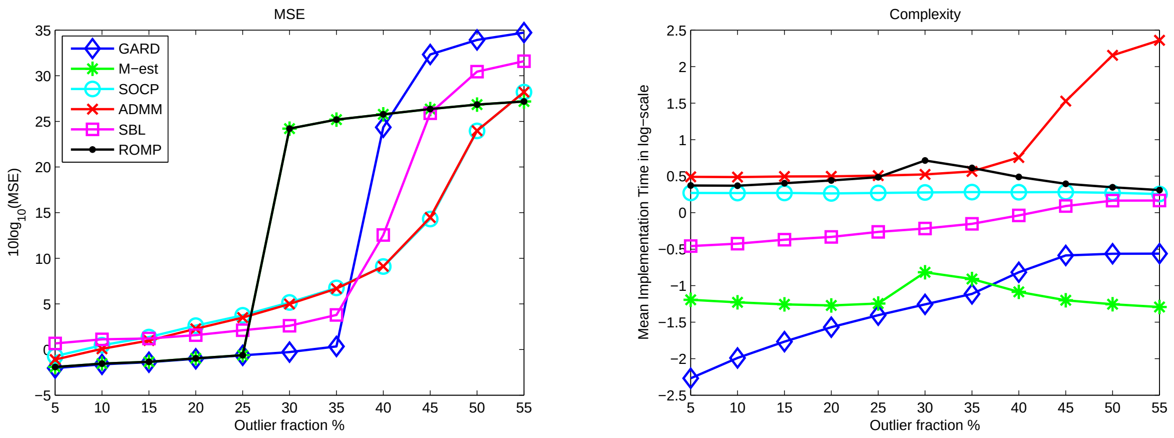
(c): $M = 100$

(d): $M = 100$

(e): $M = 170$

(f): $M = 170$

Figure 4.1: (a), (c) and (e): The attained MSE versus the outlier fraction, for various dimensions of the unknown vector $\underline{\boldsymbol{\theta}}$. (b), (d) and (f): Log-scale of the Mean Implementation Time (MIT) versus the outlier fraction for each dimension of the unknown vector. For all dimensions, $N = 600$ observations are used.



(a): $M = 100$ (b): $M = 100$

Figure 4.2: (a): The attained MSE for the dimension $M = 100$ of the unknown vector, $\underline{\boldsymbol{\theta}}$, versus the fraction of the outliers, with their values drawn randomly from the set $[-50, -25] \cup [25, 50]$. (b): The respective Mean Implementation Time (MIT) in logarithmic scale. The number of data is $N = 600$.

Finally, we have performed a similar experiment (as described above), for the dimension $M = 100$ of the unknown vector, measuring the MSE versus fraction of the outliers, with their values drawn uniformly at random from the set $[-50, -25] \cup [25, 50]$. We have chosen not to include values within the interval $(-25, 25)$ in order to distinguish the outlier from the inlier noise; thus, the percentage at the x-axis corresponds to true fractions of outliers (otherwise the true fraction could not be determined). In Figure 4.2 (a), the MSE versus the fraction of the outliers is depicted, for each method. In parallel, in Figure 4.2 (b), we have measured the Mean Implementation Time (MIT) for each method. Compared to Figure 4.1 (c), it is observed that GARD achieves enhanced robustness, since it attains the lowest MSE for the fraction of outliers up to 35%.

4.4.2 Complexity Evaluation for Large Data Sets

In the current section, the evaluation of the Mean Implementation Time (MIT) for the most computationally efficient methods is carried out. The comparison is performed for all methods except for ADMM and ROMP, for the case where the number of generated data grows significantly compared to the dimension of the unknown vector $\underline{\boldsymbol{\theta}}$. As presented in Table 4.1, the ADMM algorithm does not handle efficiently large numbers of samples. On the other hand, although ROMP performs exactly as M-est, this comes at a higher computational cost, therefore it seems impractical to put it to test.

Once again, equation (3.8) is used for generating our data. The dimension of $\underline{\boldsymbol{\theta}}$ is set at $M = 100$ and the density of the outlier noise vector at 10% with values ±25 spread uniformly over N coordinates. Finally the inlier noise vector has elements drawn from the standard Gaussian distribution, with $\sigma = 1$ and inlier noise bound ϵ. For each number of observations, N, 100 independent experiments have been performed and the results have been averaged.

In Figure 4.3, the Mean Implementation Time (MIT) (in logarithmic scale) is evaluated

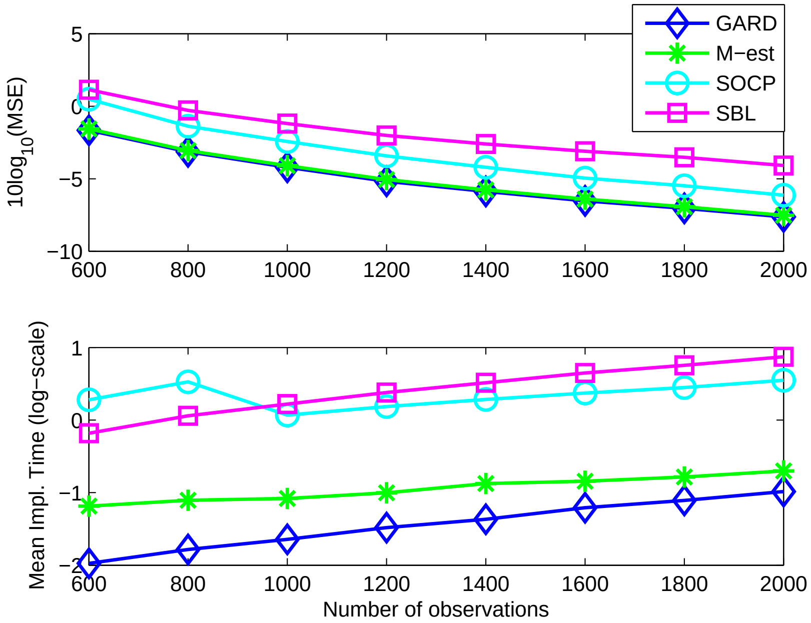


Figure 4.3: Large scale complexity test for dimension of the unknown vector set at $M = 100$. While varying the number of observations, the MSE (top) and the Mean Implementation Time (MIT) in log-scale (bottom), is shown for each method. It is clear that GARD attains the lowest MSE, whilst being the most efficient.

for each method (bottom), while the total MSE is measured in parallel (top), for each varying number of data, N. It is clear that, even for significantly large values of N, GARD excels. Whilst attaining the lowest MSE, its convergence rate is very fast.

4.4.3 Support Recovery Test

The goal of this section is to bridge the gap between the theoretical properties of Section 4.3 and the experimental performance of GARD. The results of Section 4.4.1, showcase the performance of GARD. However, it would be premature to conclude that the support of the sparse outlier vector is correctly identified in cases where the algorithm attains a low MSE, a matter that we would like to address here. Although the recovery of the sparse outlier support is desirable, since it guarantees the smallest MSE possible, it should be noted that GARD performs well (with respect to the MSE), even in cases where the recovery of the support is not exact; e.g., one of the most common cases is to identify a few extra indices (that do not belong to the support of $\underline{\boldsymbol{u}}$) as outlying elements.

For all of the support recovery simulations, the dimension of the unknown vector $\underline{\boldsymbol{\theta}}$ is set at $M = 100$ and the original data is corrupted by outliers in $S < N$ indices, uniformly sampled over $N = 600$ measurements. Also, for each fraction of outliers, i.e., $(S/N) \cdot 100\%$, we have performed 10000 Monte-Carlo runs.

According to the theoretical analysis, $\mathcal{S}_k$ denotes the support set of the sparse estimate $\hat{\boldsymbol{u}}$ and $\mathcal{S}$ the support set of the sparse outlier vector $\underline{\boldsymbol{u}}$. In the following figures, the green line (pointing up) corresponds to the percentage of correct indices that the proposed scheme has

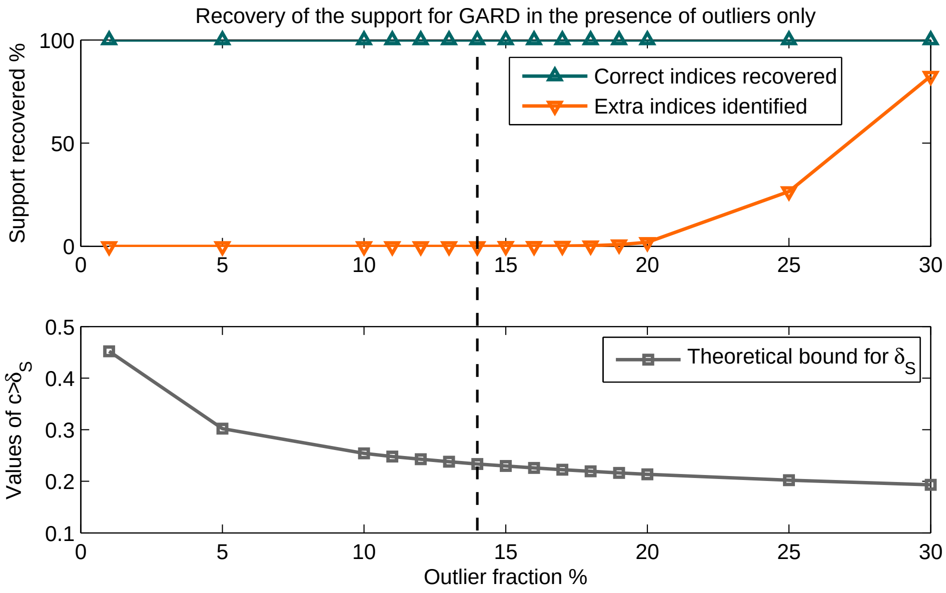


Figure 4.4: Recovery of the support and relation to the bound of δ_S, for the noiseless case. For outlier fraction of less than 14%, the bound for δ_S in (4.16), is guaranteed, hence the recovery is exact.

recovered, i.e., indices $i \in \mathcal{S}_k \subseteq \mathcal{S}$, while the orange line (pointing down) corresponds to the extra indices that the method has incorrectly identified as outliers, i.e., indices $j \in \mathcal{J} \setminus \mathcal{S}$. In addition, since the constant of the smallest principal angle cannot be computed directly, the bound of $c > \delta_S$ is tracked for the evaluation of the theoretical results reported in Section 4.3.2. The vertical line corresponds to the largest outlier fraction, that the proposed scheme succeeds in recovering the sparse outlier vector support, one to one element.

The presence of outliers only

The scenario in which our original data is corrupted by outlier values only, is treated separately. Our data are generated via equation (3.8), for $\eta_i = 0$ and outlier values[4] ± 25, in S indices, uniformly sampled over N coordinates. In Figure 4.4, the recovery of the exact support versus the fraction of the outliers is demonstrated. It is clear that for fraction of less than 14%, the bound for δ_S, as Theorem 4.3 suggests, is guaranteed, thus the recovery of the support is exact and also the approximation of $\underline{\boldsymbol{\theta}}$ is of zero error. It should also be pointed out that, the approximation error is also very small, in cases where only a few extra indices, that belong to $\mathcal{S}^c$ are imported into the support set $\mathcal{S}_k$.

The presence of both inlier and outlier noise

In the current section, the focus is turned on the empirical validation of (4.38) and (4.47), where two separate tests have been performed.

[4] In the noiseless case, arbitrarily small outlier values, are always identified; thus the performance of GARD is not affected by a particular selection of those values.

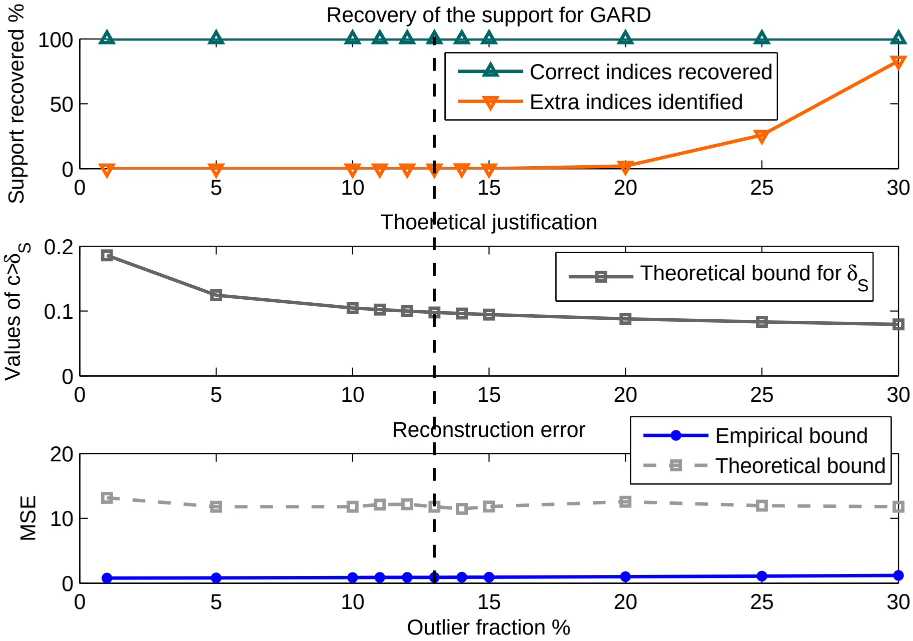


Figure 4.5: Recovery of the support and relation to the bound of δ_S, for the case inlier and outlier noise coexist. For an outlier fraction of less than 13%, the bound for δ_S (4.16) is guaranteed, hence the recovery of the support is exact, while the computed MSE is valid under the bound that inequality (4.47) suggests.

In the first test, the maximum bound for the norm of the inlier noise vector is fixed at $\epsilon = 28$, while the fraction of outliers varies. In order to achieve this, the MATLAB's random generator for the Gaussian distribution, with standard deviation depending on ϵ, is used, while the largest elements (in the absolute sense) are cut off if required, so that the norm of the inlier noise vector always remains bounded by ϵ. Also, recall on Remark 4.13, that the minimum element of the absolute value of the outlier vector should be larger than $(2 + \sqrt{6})\epsilon$, in order (4.38) to be valid. Thus, the outlier values have been set at ±150, while the values of the original outputs, $\boldsymbol{X}\underline{\boldsymbol{\theta}}$, range at $170 - 180$. In Figure 4.5, we have plotted the recovery of the support for GARD and its relation to the bound c of the smallest principal angle (or RIP) constant δ_S, for each outlier fraction. As one could observe, for fraction of outliers less than 13%, the bound for δ_S, as Theorem 4.4 proposes, is guaranteed, thus the recovery of the support is exact. In parallel, we have computed the MSE between $\underline{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\theta}}$ and tracked the relation to the theoretical bound[5] of (4.47).

In the second test, the capability of GARD to deal with heavy noise is demonstrated. The outlier values were set at ±150 and the bound of ϵ was increased, so that the inlier noise corresponds to a noise level of 20 dB. In such a case, the bounds established in (4.38) and (4.47) are violated, however GARD manages to cope with. In Figure 4.6, the recovery of the support versus the outlier fraction is demonstrated. We conclude that, although the method does not succeed to recover the sparse outlier support 100%, the MSE is relatively low, at least for low fraction of outliers, i.e., below 10%.

[5] Since the MSE is a squared norm between $\underline{\boldsymbol{\theta}}$ and $\hat{\boldsymbol{\theta}}$, the bound is the squared right hand side of (4.47).

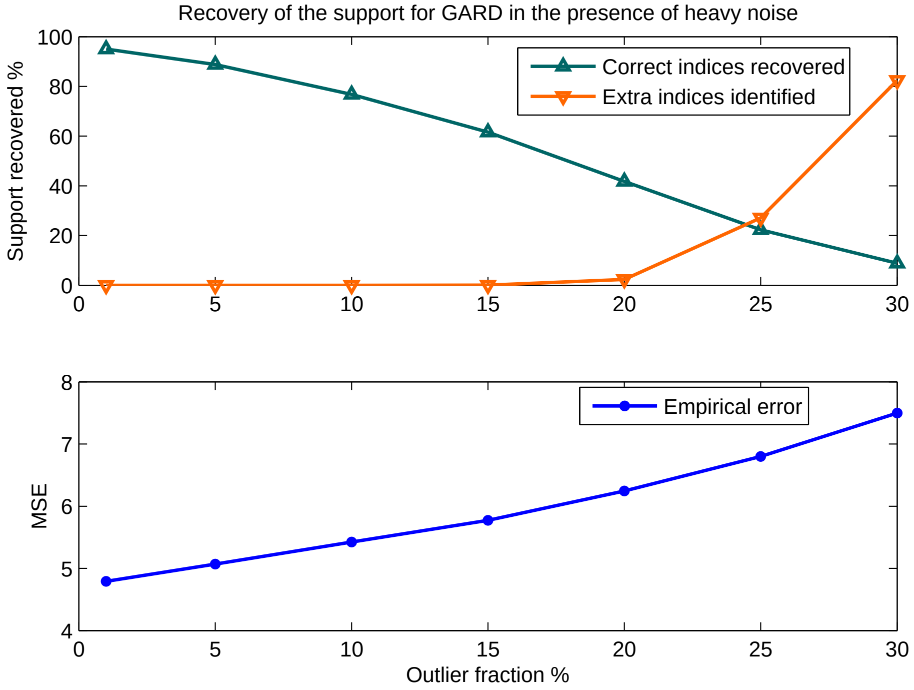


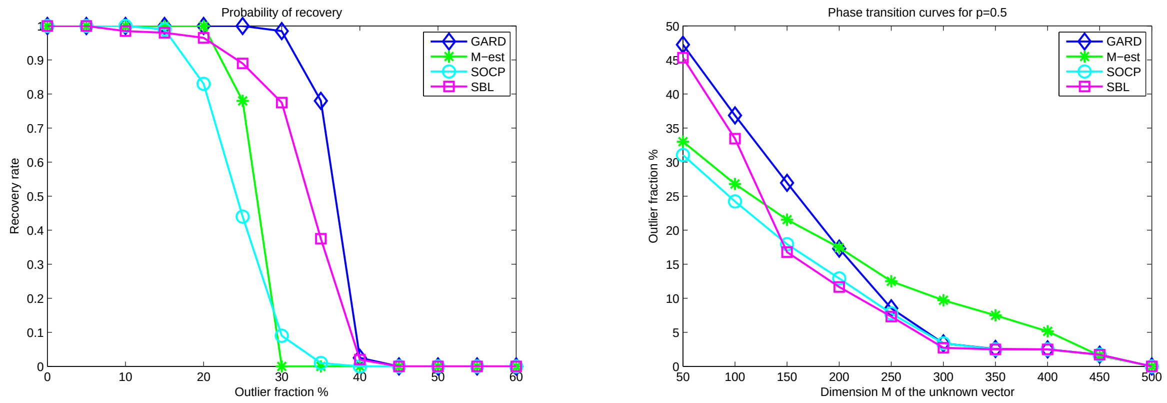
Figure 4.6: Recovery of the support for GARD, in the case where outlier and heavy inlier noise of approx. 20 dB coexist. Although, the support is not entirely recovered, the MSE is relatively low.

4.4.4 Evaluating the Probability of an Accurate Estimation

In this experiment, the probability of successful estimation is evaluated. Since ADMM and ROMP require higher computational loads and also perform very similar to other methods, we limited our efforts to test the rest of the methods, i.e., GARD, SOCP, M-est and SBL. Two sets of simulations are performed for a fixed number of data at $N = 600$. The noise comprises inlier AWGN with $\sigma = 1$ and outliers with values equal to 25 or -25, in S indices, uniformly sampled over N coordinates ($S < N$).

Figure 4.7 (a) demonstrates the probability of recovery for each method tested, while varying the fraction of outliers. The dimension of the unknown vector $\underline{\boldsymbol{\theta}}$ is fixed at $M = 100$. For each density of the sparse outlier vector, we have computed the probability over 200 Monte-Carlo runs. For each method, we have assumed that the solution is obtained, if $||\hat{\underline{\boldsymbol{\theta}}} - \underline{\boldsymbol{\theta}}||_2 / ||\underline{\boldsymbol{\theta}}||_2 \leq 0.03$. The major result is that, for fraction of outliers under 25%, GARD succeeds in recovering the solution, with probability $p = 1$. For M-est, the percentage drops below 20%, while for the rest of the methods the percentage is even lower. For SBL, the probability to recover the solution is not guaranteed, even for the lowest fractions of outliers.

Figure 4.7 (b) demonstrates the phase transition curves for each method. For each dimension of the unknown vector $\underline{\boldsymbol{\theta}}$, we have computed the fraction of outliers for which the method transits from success to failure with probability $p = 0.5$. Experiments were carried over 200 Monte-Carlo runs. Once again, we have assumed that the solution is obtained, using the criterion as in Figure 4.7 (a). We observe that for each fixed dimension of the unknown vector, the probability for each method to recover the solution (always within a given tolerance) increases



(a): Probability of recovery for $M = 100$

(b): Transition from success to failure.

Figure 4.7: (a): The probability of recovery while varying the fraction of outliers, for the the dimension $M = 100$ of the unknown vector, $\underline{\boldsymbol{\theta}}$, and $N = 600$ observations. As the fraction of the outliers increases, the probability for an accurate estimation drops. (b): Transition from success to failure with probability $p = 0.5$. A vertical line at $M = 100$ indicates the percentage of outliers (for each method respectively) that correspond to the values of the x-axis for probability $p = 0.5$, in (a).

for fractions of the outliers below each phase transition curve (where the fraction of outliers decreases). Contrariwise, the probability decreases as we move above the phase transition curves. Here, it is clear that up to $M = 200$, GARD succeeds to recover the solution with the highest probability from the rest of the methods. However, for larger dimensionality values, the number of data (here $N = 600$) seems pretty "poor" to allow GARD to preserve its good performance (in the sense that more data is required), although it is not worse than that of the ℓ_1-norm minimization techniques.

4.4.5 Experiments with Noise of More General Forms

In the current section, we have performed a set of experiments involving more general noise models for the described methods. To this end, simulations with various noise forms are carried out and the MSE, over an average of 100 Monte-Carlo runs, is measured. Equation (2.2) describes our model, where we produce ($N = 600$) measurements corrupted by different types of noise. The dimension of the unknown vector, $\underline{\boldsymbol{\theta}}$, is $M = 100$. For all tests, the ADMM was excluded from the last set of experiments, since the method proved weak to handle different orders of noise values simultaneously, thus failed to converge for all tests.

- **Tests A, B and C**. The noise vector is drawn from the Lévy alpha-stable distribution, $\mathcal{S}(\alpha, \beta, \gamma, \delta)$, with pdf expressed in closed form only for special cases of the involved parameters. The distribution's parameters β and δ, that control symmetry were set to zero (results to a symmetric distribution without skewness) for all three experiments. For test A, the distribution's parameters were set to $\alpha = 0.45$ and $\gamma = 0.3$; the parameters for each method were set to $\epsilon = 3$ for GARD and SOCP, $\hat{\sigma} = 1.2$ for M-est and ROMP (we have altered the parameter value of "robustfit"), while the hyperparameters for SBL were initialized to 10^{-4}. In Table 4.2, it is observed that almost all methods perform quite well

Table 4.2: Computed MSE, for various experiments. In tests A, B and C, the noise is drawn from the heavy-tailed distribution alpha-stable of Lévy distribution. In test D, noise consists of a sum of two vectors, drawn from 2 independent Gaussian distributions with different variance, plus an outlier noise vector of impulsive noise.

Algorithm	Test A	Test B	Test C	Test D
GARD	0.1772	0.0180	0.0586	0.690
M-est	0.2248	0.2859	1.844e+06	0.704
SOCP	0.4990	0.3502	5.852e+05	1.011
SBL	0.9859	58.3489	2.165e+06	1.292
ROMP	0.2248	0.2859	1.844e+06	0.704

(low MSE), with GARD appearing to perform better. For test B, $\alpha = 0.4$, $\gamma = 0.1$; for GARD $\epsilon = 3$, for SOCP $\epsilon = 2$, for M-est and ROMP $\hat{\sigma} = 1$ ("robustfit" parameter), while for SBL the hyperparameters were initialized at random (Gaussian) with variance equal to 10^{-5}, although fails to converge, for all values of the paramaters tested. Once again, it can be readily seen that GARD attains the lowest MSE. Finally, for the experiment C, $\alpha = 0.3$, $\gamma = 0.1$, resulting to a greater frequency of large values of noise; for GARD $\epsilon = 3$, for SOCP $\epsilon = 2$, for M-est and ROMP $\hat{\sigma} = 1$, while for SBL the hyperparameters were initialized at random (Gaussian) with variance equal to 10^{-6}. The attained MSE for GARD is significantly lower than in tests A and B; thus, we conclude that the method manages to handle better large values of outlier noise with respect to the other methods.

- **Test D**. The noise consists of a sum of two vectors, drawn from two independent Gaussian distributions $\mathcal{N}(0, 0.6^2)$ and $\mathcal{N}(0, 0.8^2)$, plus an outlier noise vector of 10% density (indices chosen uniformly at each repetition) with values ±25. The parameters required for each method are: the default tuning parameter for both M-est and ROMP; for GARD and SOCP $\max\{\epsilon_1, \epsilon_2\}$ is required, where ϵ_1, ϵ_2 are the bounds of each inlier noise vector, while for SBL an initialization at random with variance of 10^{-6} was performed. The model of the noise is now more complicated, hence the task mode complex to be solved for all of the methods. Once again, it is clear that GARD copes with this mixed type of noise too.

Chapter 5

Nonlinear Regression in RKHS and the Pursuit for Robustness

5.1 Introduction

In this chapter, an overview of the basic definitions and theorems concerning Reproducing Kernel Hilbert Spaces (RKHS) is given. RKHS play a central role in the task of learning nonlinear models. The approach consists of mapping the input variables of the original lower dimensional space to a higher dimensional one, such that the nonlinear task is transformed into a linear one. The main advantage of such spaces is that inner product operations are performed in a very efficient way, with complexity independent of the dimensionality of the respective RKHS.

The task of robust learning of nonlinear models in RKHS is also introduced. Robust methods in the context of RKHS have already been proposed for the nonlinear regression task and include: a) a Bayesian probabilistic approach and b) a deterministic sparsity-aware learning technique based on the minimization of the ℓ_1-norm. Both of these formulations are presented and discussed. Finally, the ℓ_0-norm formulation, on which our method is based for the respective robust estimation task, is also introduced.

5.2 Reproducing Kernel Hilbert Spaces (RKHS)

Consider a linear space $\mathcal{H}$, of real-valued functions defined on a set of points $\mathcal{X}$. Typically, $\mathcal{X}$ is a compact subset of $\mathbb{R}^K$, with $K \in \mathbb{N}^*$. Furthermore, suppose that $\mathcal{H}$ is a Hilbert space, i.e., a space equipped with a dot product operation $\langle \cdot, \cdot \rangle_{\mathcal{H}}$, that defines a corresponding norm $\| \cdot \|_{\mathcal{H}} := \sqrt{\langle \cdot, \cdot \rangle_{\mathcal{H}}}$ and $\mathcal{H}$ is complete with respect to this norm.

Definition 5.1. *A Hilbert space, $\mathcal{H}$, is called Reproducing Kernel Hilbert Space (RKHS), if there exists a function $\kappa : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$, with the following properties:*

- *For every $\boldsymbol{x} \in \mathcal{X}$, $\kappa(\cdot, \boldsymbol{x})$ belongs to $\mathcal{H}$.*
- *$\kappa(\cdot, \cdot)$ has the so-called reproducing property, that is,*

$$\mathrm{f}(\boldsymbol{x}) = \langle \mathrm{f}, \kappa(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}}, \ \textit{for every } \mathrm{f} \in \mathcal{H} \textit{ and every } \boldsymbol{x} \in \mathcal{X}. \tag{5.1}$$

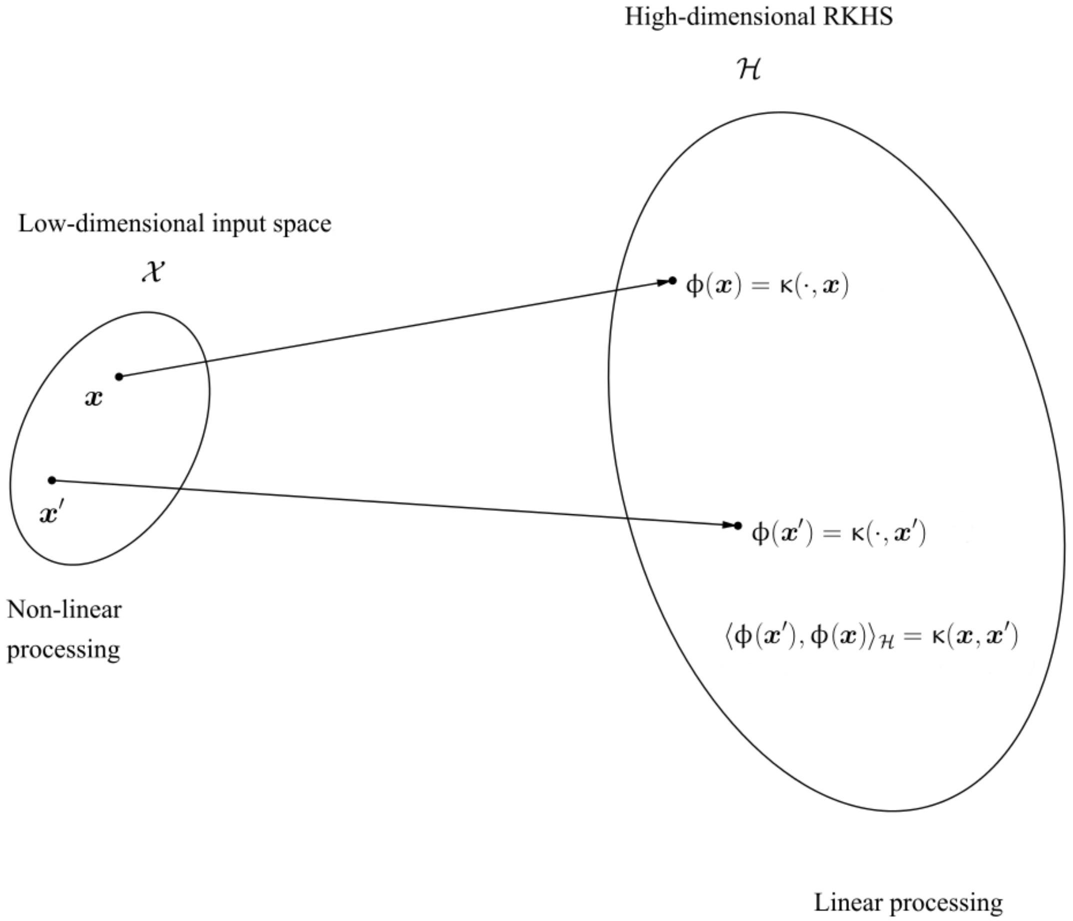

High-dimensional RKHS
$\mathcal{H}$
Low-dimensional input space
$\mathcal{X}$
$\boldsymbol{x}$
$\boldsymbol{x}'$
$\phi(\boldsymbol{x}) = \kappa(\cdot, \boldsymbol{x})$
$\phi(\boldsymbol{x}') = \kappa(\cdot, \boldsymbol{x}')$
$\langle \phi(\boldsymbol{x}'), \phi(\boldsymbol{x}) \rangle_{\mathcal{H}} = \kappa(\boldsymbol{x}, \boldsymbol{x}')$
Non-linear processing
Linear processing

Figure 5.1: The mapping from the original low-dimensional input space $\mathcal{X}$ to a linear one in the high-dimensional RKHS $\mathcal{H}$. Employing the kernel trick, inner product operations are efficiently performed via function evaluations on the original low-dimensional space $\mathcal{X}$.

Definition 5.2. *Let $\mathcal{H}$ be an RKHS, associated with a kernel function $\kappa(\cdot, \cdot)$ and $\mathcal{X}$ a set of elements. Then, for every $\boldsymbol{x} \in \mathcal{X}$ the mapping*

$$\boldsymbol{x} \mapsto \phi(\boldsymbol{x}) := \kappa(\cdot, \boldsymbol{x}) \in \mathcal{H}, \tag{5.2}$$

is known as the feature map and the space, $\mathcal{H}$, the feature space.

In other words, if $\mathcal{X}$ is a set of vectors, the *feature mapping* maps each vector from the original space to a high-dimensional RKHS $\mathcal{H}$, as demonstrated in Figure 5.1. Note that, in general, $\mathcal{H}$ can be of infinite dimension and that its elements could also be functions. In special cases only, where $\mathcal{H}$ becomes a (finite dimensional) Euclidean space, for example $\mathbb{R}^L$, the image is a vector $\boldsymbol{\phi}(\boldsymbol{x}) \in \mathbb{R}^L$.

As a direct consequence of Definitions 5.1 and 5.2, the inner product of the respective mappings of two points $\boldsymbol{x}, \boldsymbol{x}' \in \mathcal{X}$, results to

$$\langle \phi(\boldsymbol{x}'), \phi(\boldsymbol{x}) \rangle_{\mathcal{H}} = \langle \kappa(\cdot, \boldsymbol{x}'), \kappa(\cdot, \boldsymbol{x}) \rangle_{\mathcal{H}} = \kappa(\boldsymbol{x}, \boldsymbol{x}') \ : \ \textbf{Kernel Trick.} \tag{5.3}$$

Simply expressed, by employing this mapping, we can perform inner product operations, via function evaluations performed in the original low-dimensional space. This property, is known as the *kernel trick* and it greatly simplifies the involved computations. As a result, such computations promote their usage in algorithmic procedures.

5.3 Properties of RKHS

In this section, an overview of some of the basic properties of the RKHS is provided. For more details, also read [78, 79, 80, 1, 81].

Definition 5.3. *Given a function* $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ *and* $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathcal{X}$*, the square matrix* $\boldsymbol{K}$*, with elements* $\kappa_{nm} = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m)$ *for* $n, m = 1, \ldots, N$*, is called Gram matrix or kernel matrix of the function* κ *with respect to* $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$*.*

Definition 5.4. *The function* κ *is called a positive definite kernel, if*

$$\boldsymbol{\alpha}^T \boldsymbol{K} \boldsymbol{\alpha} = \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) \geq 0 \ : \ \textit{\textbf{Positive Definite Kernel}}, \tag{5.4}$$

for all $\boldsymbol{\alpha} \in \mathbb{R}^N$*, points* $\boldsymbol{x}_n, \boldsymbol{x}_m \in \mathcal{X}$ *and any* $N \in \mathbb{N}^*$*.*

At this point, it should be noted that, although (5.4) is the definition for a positive semidefinite matrix in the linear algebra literature, historically, the positive definite kernels in (5.4) were originally introduced by Mercer [82], in the context of integral equations. The connection to RKHS was developed later on. To this end, the term positive definite is adopted for (5.4) and should be distinguished from the positive definite matrix in the context of matrix analysis, which is a strict inequality.

Remark 5.1. *Generally, for a reproducing kernel, the respective Gram matrix is strictly positive definite. However, if not, there exists a non-zero vector* $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_N]^T$*, such that* $\left\| \sum_{n=1}^{N} \alpha_n \kappa(\cdot, \boldsymbol{x}_n) \right\|_{\mathcal{H}}^2 = 0$*. Hence, for every* $\mathrm{f} \in \mathcal{H}$*, we have that*

$$\sum_{n=1}^{N} \alpha_n \mathrm{f}(\boldsymbol{x}_n) = \left\langle \mathrm{f}, \sum_{n=1}^{N} \alpha_n \kappa(\cdot, \boldsymbol{x}_n) \right\rangle_{\mathcal{H}} = 0,$$

which results to the existence of an equation with linear dependence between the values of every function in $\mathcal{H}$ *at some finite set of points. Although such examples exist (e.g., Sobolev spaces), this is not a standard case. In most cases, the reproducing kernels define Gram matrices that are always strictly positive definite and thus invertible!*

Proposition 5.1. *The reproducing kernel of* $\mathcal{H}$*, is symmetric, i.e.,*

$$\kappa(\boldsymbol{x}', \boldsymbol{x}) = \kappa(\boldsymbol{x}, \boldsymbol{x}').$$

Lemma 5.1. *The reproducing kernel, associated with* $\mathcal{H}$*, is a positive definite kernel.*

The proofs of Proposition 5.1 and Lemma 5.1 can be found in [83, 1].

Definition 5.5. *Given a linear subspace $\mathcal{S}$ of a Hilbert space $\mathcal{H}$, let*

$$\mathcal{S}^{\perp} = \{w \in \mathcal{H} | \langle w, v\rangle_{\mathcal{H}} = 0 \text{ for all } v \in \mathcal{S}\}.$$

$\mathcal{S}^{\perp}$ is called the orthogonal complement of $\mathcal{S}$.

It is clear from the above definition and the continuity of the inner product that the orthogonal complement is always a *closed linear subspace*, due to the continuity of the inner product. Next, we present one of the most important theorems in Hilbert spaces.

Theorem 5.1 (Projection Theorem)**.** *Let $\mathcal{H}$ be a Hilbert space, $u \in \mathcal{H}$ and $\mathcal{S}$ a closed subspace of $\mathcal{H}$. Then*

1. *There exists a unique element $u_* \in \mathcal{S}$ (called the projection of u onto $\mathcal{S}$), such that*

$$\|u - u_*\|_{\mathcal{H}} = \inf_{v \in \mathcal{S}} \|u - v\|_{\mathcal{H}}$$

2. *u_* is uniquely characterized by*

$$(u - u_*) \in \mathcal{S}^{\perp}.$$

Theorem 5.1 obviously holds for an RKHS too, however it has a wider usage in more general Hilbert spaces, see [84].

Theorem 5.2. *Let $\mathcal{S}$ be a closed linear subspace of $\mathcal{H}$. Then*

$$\mathcal{H} = \mathcal{S} \oplus \mathcal{S}^{\perp},$$

where $\oplus$ denotes the direct sum.

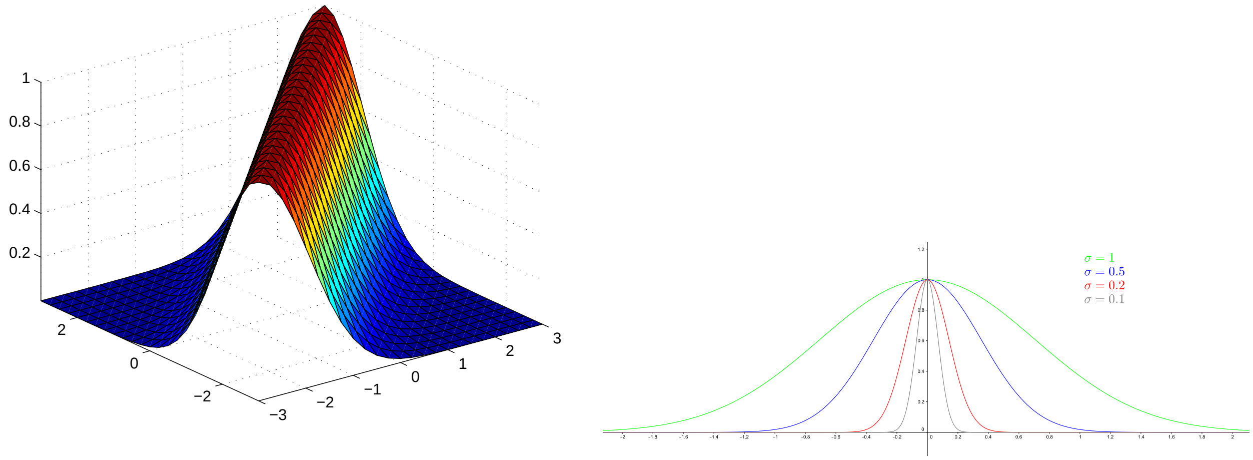
Lemma 5.2. *Let $\mathcal{H}$ be a RKHS on the set $\mathcal{X}$ with reproducing kernel $\kappa(\cdot, \cdot)$. Then the linear span of the function $\kappa(\cdot, \boldsymbol{x})$, $\boldsymbol{x} \in \mathcal{X}$ is dense in $\mathcal{H}$, that is,*

$$\mathcal{H} = \overline{\operatorname{span}\{\kappa(\cdot, \boldsymbol{x}), \boldsymbol{x} \in \mathcal{X}\}}. \tag{5.5}$$

Theorem 5.2 is a direct consequence of the Projection Theorem. In Lemma 5.2 it is stated that, $\mathcal{H}$ can be constructed by all possible linear combinations of the kernel function computed in $\mathcal{X}$, as well as the limit points of the set. In other words, $\mathcal{H}$ can be fully generated from the knowledge of the kernel κ. For a more in depth view, also read [83, 1, 80].

5.3.1 Examples of Kernel Functions

In this subsection, we present the most commonly (application-wise) used kernel functions, defined on $\mathcal{X} \times \mathcal{X}$, $\mathcal{X} \subseteq \mathbb{R}^K$, as:



(a)

(b)

Figure 5.2: (a) The Gaussian kernel for $\mathcal{X} = \mathbb{R}$ and $\sigma = 1.5$. (b) The element $\phi(0) = \kappa(\cdot, 0)$ for different values of σ.

- The *Gaussian* Radial Basis Function (RBF)

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2}{\sigma^2}\right), \tag{5.6}$$

  where $\sigma > 0$ is the kernel's parameter. In Figures 5.2 and 5.3, the shape of the Gaussian RBF is shown, for various $\mathcal{X}$ and values of σ. The dimensionality of the RKHS associated with the Gaussian kernel is *infinite*, see [85].

- The *Laplacian* RBF

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \exp\left(-t\|\boldsymbol{x} - \boldsymbol{x}'\|_2\right), \tag{5.7}$$

  where $t > 0$ is a parameter. In Figures 5.4 and 5.6, the shape of the Laplacian RBF is shown, for various $\mathcal{X}$ and values of t. The dimensionality of the RKHS associated with the Laplacian kernel is also infinite.

- The *inhomogeneous polynomial* kernel is given by

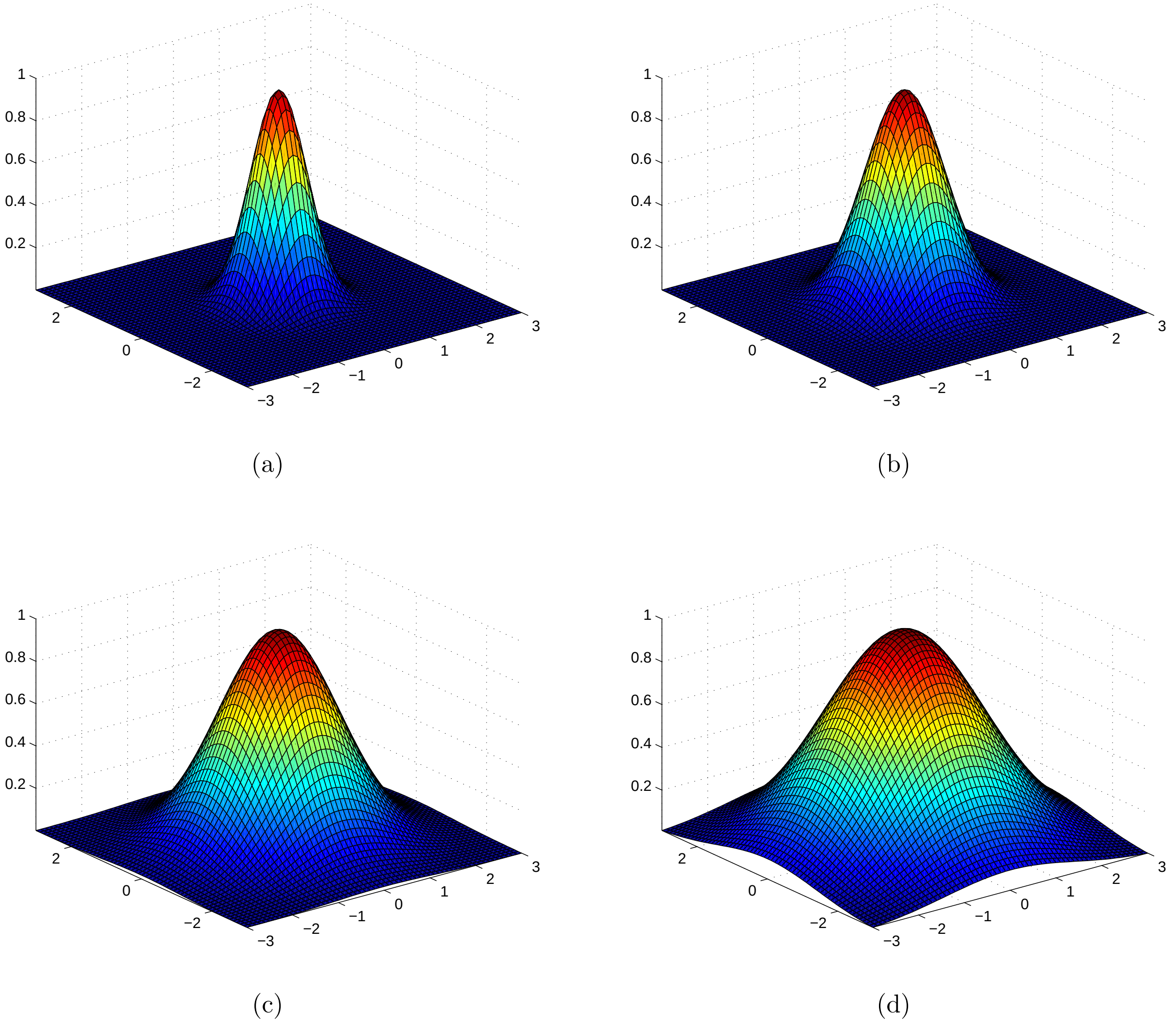
$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \left(\boldsymbol{x}^T\boldsymbol{x}' + c\right)^d, \tag{5.8}$$

  where $c \geq 0$ and d are parameters. For $c = 0$, the *homogeneous polynomial* kernel follows. In Figure 5.5, the shape of the polynomial kernel function is shown, for $\mathcal{X} = \mathbb{R}$ and different values of c, d. Finally, it should be stated, that the dimensionality of the polynomial kernels is finite.

- The *spline* kernel is given by

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{B}_{2p+1}\left(\|\boldsymbol{x} - \boldsymbol{x}'\|_2^2\right), \tag{5.9}$$

  where the B_n spline is defined via the $n+1$ convolutions of the unit interval $[-\frac{1}{2}, \frac{1}{2}]$, that is, $\mathrm{B}_n := \otimes_{i=1}^{n+1} \mathrm{I}_{[-\frac{1}{2},\frac{1}{2}]}(\cdot)$, where $\mathrm{I}_{[-\frac{1}{2},\frac{1}{2}]}(\cdot)$ is the characteristic function on the respective interval, i.e., equal to one if the variable belongs to the interval and zero otherwise.



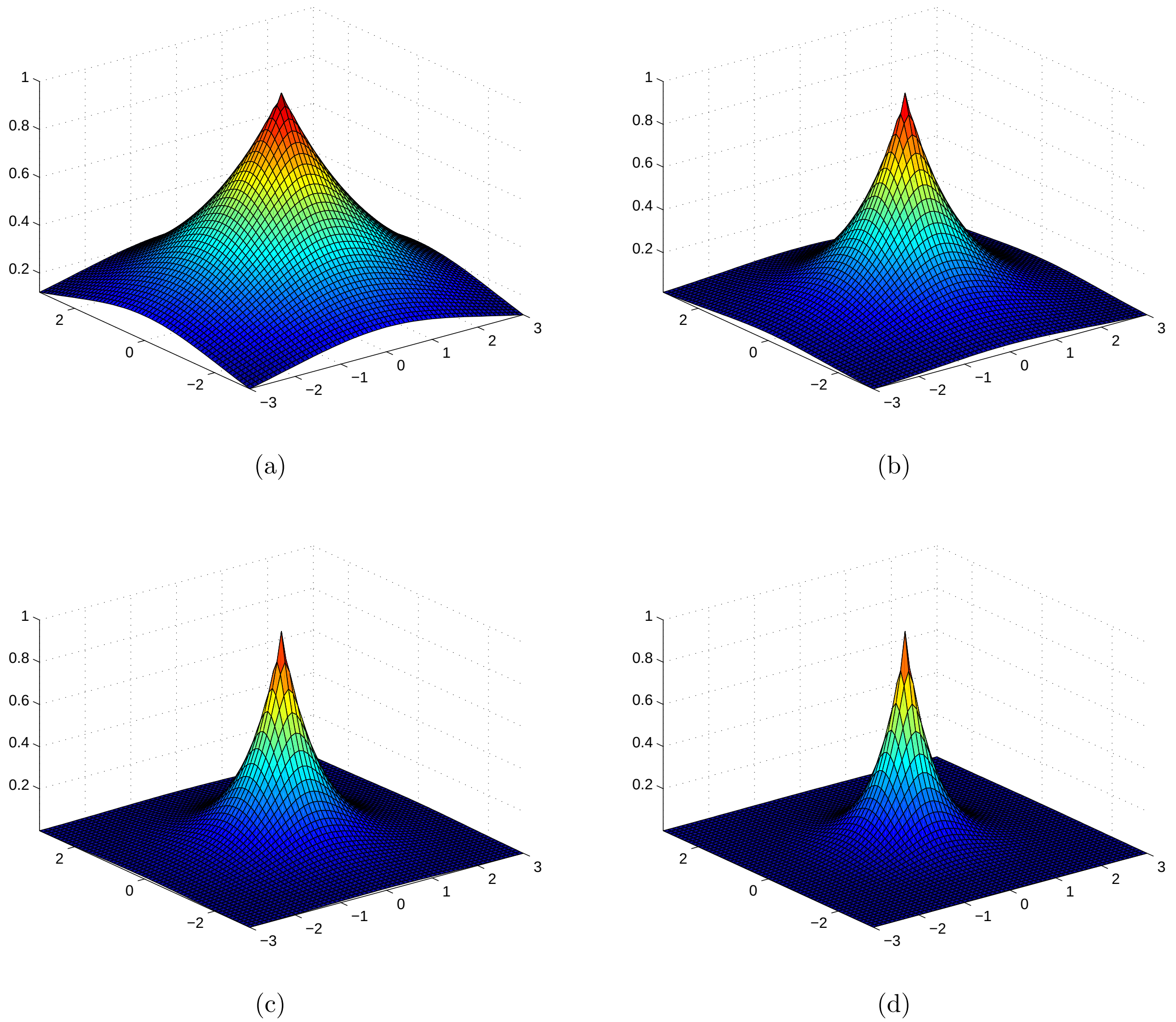
(a)

(b)

(c)

(d)

Figure 5.3: The element $\phi(0) = \kappa(\cdot, 0)$ of the Gaussian kernel for $\mathcal{X} = \mathbb{R}^2$ and for various values of the parameter σ. (a) $\sigma = 0.7$, (b) $\sigma = 1$, (c) $\sigma = 1.5$, (d) $\sigma = 2$.



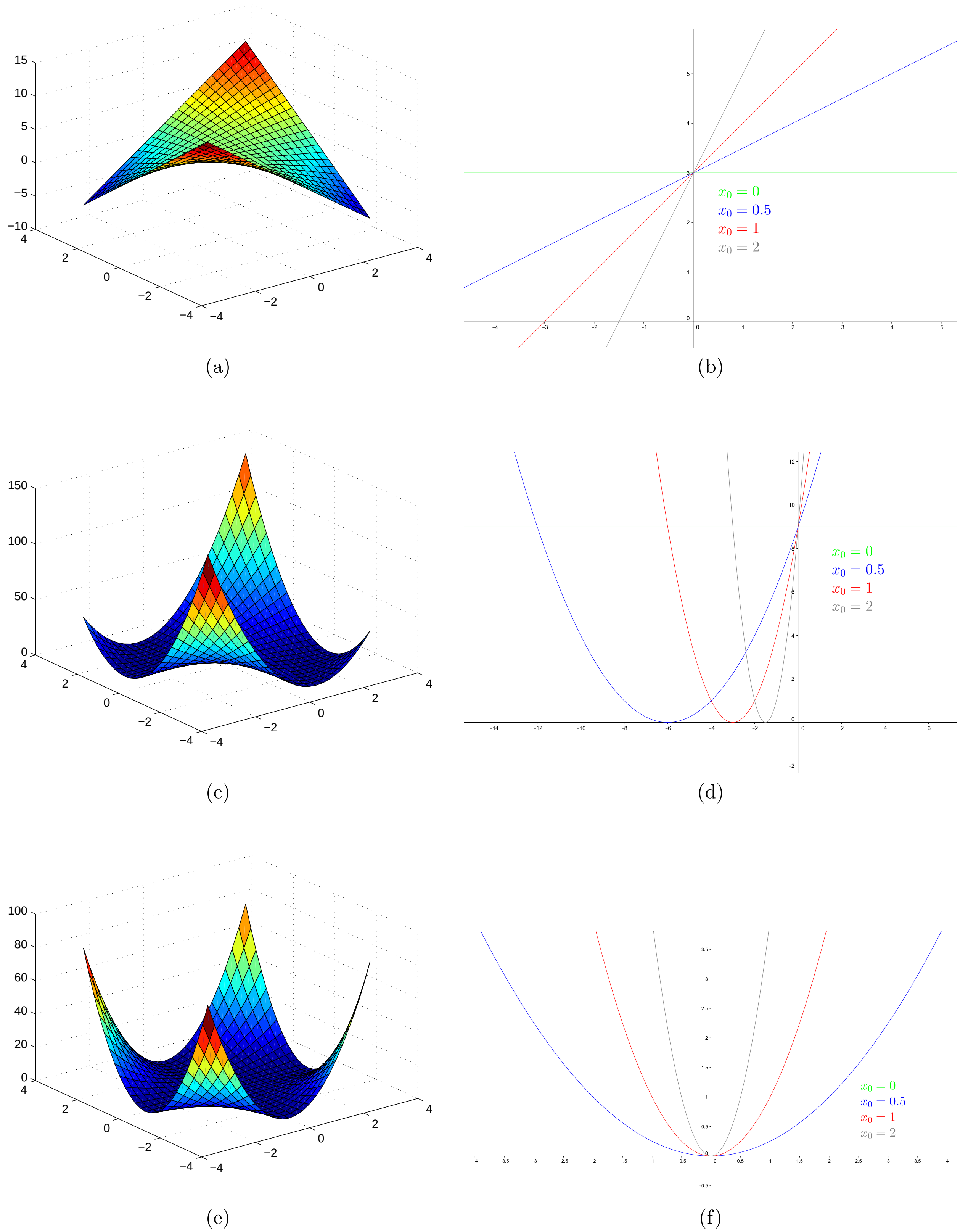
(a)

(b)

(c)

(d)

Figure 5.4: The element $\phi(0) = \kappa(\cdot, 0)$ of the Laplacian kernel for $\mathcal{X} = \mathbb{R}^2$ and for various values of the parameter t. (a) $t = 0.5$, (b) $t = 1$, (c) $t = 1.5$, (d) $t = 2$.



(a)

(b)

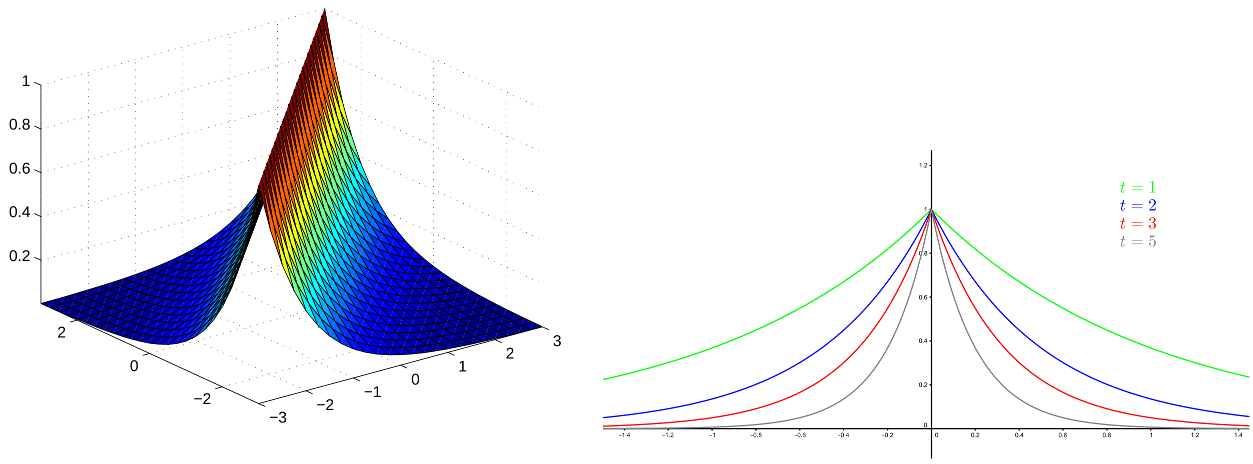
(c)

(d)

(e)

(f)

Figure 5.5: The polynomial kernel for $\mathcal{X} = \mathbb{R}$: (a) inhomogeneous $d = 1, c = 3$, (c) inhomogeneous $d = 2, c = 3$ and (e) homogeneous $d = 2, c = 0$. (b), (d), (f) The respective element $\phi(x_0) = \kappa(\cdot, x_0)$ for various values of x_0.



(a) (b)

Figure 5.6: (a) The Laplacian kernel for $\mathcal{X} = \mathbb{R}$ and $t = 1$. (b) The element $\phi(0) = \kappa(\cdot, 0)$ of for different values of t.

Although a large variety from which to choose a kernel function exists, in many applications the *Gaussian* RBF in (5.6) for $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^K$ is preferred, due to its desired properties. The most important property is provided in the following theorem, see [80], [1].

Theorem 5.3 (Full Rank of Gaussian RBF Gram Matrix)**.** *Suppose that* $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_N \subset \mathcal{X}$ *where* $\mathcal{X} \subseteq \mathbb{R}^K$ *are distinct points and* $\sigma > 0$. *The matrix* $\boldsymbol{K}$*, given by*

$$\kappa_{ij} := exp(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||_2^2}{\sigma^2}),$$

$i, j = 1, \ldots, N$ *has full rank.*

The significance of the theorem is that the points $\kappa(\cdot, \boldsymbol{x}_1), ..., \kappa(\cdot, \boldsymbol{x}_N) \in \mathcal{H}$ are linearly independent, i.e. span the N-dimensional subspace of $\mathcal{H}$, see [79]. In the following, κ is adopted to denote the Gaussian RBF.

5.3.2 Basic Theorems

The following theorem is of major importance and it allows us to perform empirical loss function optimization, based on a finite set of training points, in a very efficient way, even if the function to be estimated belongs to a very high dimensional space $\mathcal{H}$. For more details, also see [80], [1].

Theorem 5.4 (Representer Theorem)**.** *Let* $\Omega : [0, +\infty) \rightarrow \mathbb{R}$ *be a strictly monotonic increasing function,* $\mathcal{X}$ *a nonempty set and* $\mathrm{L} : \mathbb{R}^2 \rightarrow \mathbb{R} \cup \{\infty\}$ *an arbitrary loss function. Then, each minimizer* $\mathrm{f} \in \mathcal{H}$ *of the regularized minimization problem:*

$$\min_{\mathrm{f} \in \mathcal{H}} \left\{ \mathrm{L}\left(\{(y_n, \mathrm{f}(\boldsymbol{x}_n))\}_{n=1}^N \right) + \lambda \Omega\left(||\mathrm{f}||_{\mathcal{H}}^2 \right) \right\}, \tag{5.10}$$

admits a representation of the form

$$\mathrm{f} = \sum_{n=1}^{N} \alpha_n \kappa(\cdot, \boldsymbol{x}_n), \tag{5.11}$$

with $\alpha_n \in \mathbb{R}$ *for all* $n = 1, \dots, N$*.*

In simple words, Thoerem 5.4 states, that the solution of any regularized ridge regression optimization task lies in the span of N particular kernels. However, in many applications, a *bias* term c is often included to the aforementioned expansion; in other words, we assume that the desired solution admits the following representation:

$$\mathrm{f} = \sum_{n=1}^{N} \alpha_n \kappa(\cdot, \boldsymbol{x}_n) + c. \tag{5.12}$$

In practice, the use of a bias term turns out to improve performance. First, it enlarges the class of functions in which we seek for a solution and second, the minimizer forces the values, which the function takes at the training points, to smaller ones, due to the penalization imposed by the regularization term. Finally, the use of the bias factor is theoretically justified by the Semi-parametric Representer Theorem [80, 1].

Theorem 5.5 (Semi-Parametric Representer Theorem)**.** *Let us assume that in addition to the assumptions adopted in Theorem 5.4, we are given a set of real-valued functions*

$$\psi_m : \mathcal{X} \mapsto \mathbb{R},\ m = 1, \dots M,$$

with the property that the $N \times M$ *matrix with elements* $(\psi_m(\boldsymbol{x}_n))_{nm}$*,* $n = 1, \dots, N,\ m = 1, \dots, M$*, has rank* M*. Then, any*

$$\mathrm{g} = \mathrm{f} + \mathrm{h},\ \mathrm{f} \in \mathcal{H},\ \mathrm{h} \in \mathrm{span}\{\psi_m,\ m = 1, \dots, M\},$$

solving the minimization task

$$\min_{\mathrm{g}} \left\{ \mathrm{L}\left(\{(y_n, \mathrm{g}(\boldsymbol{x}_n))\}_{n=1}^{N} \right) + \lambda \Omega\left(||\mathrm{f}||_{\mathcal{H}}^2 \right) \right\}, \tag{5.13}$$

admits the following representation:

$$\mathrm{g} = \sum_{n=1}^{N} \alpha_n \kappa(\cdot, \boldsymbol{x}_n) + \sum_{m=1}^{M} b_m \psi_m(\cdot), \tag{5.14}$$

with $\alpha_n,\ b_m \in \mathbb{R}$ *for all* $n = 1, \dots, N,\ m = 1, \dots, M$*.*

Obviously, the use of a bias term is a special case of the expansion in (5.14). An example of application of this theorem was demonstrated in [46].

5.4 Kernel Ridge Regression (KRR)

The regression task has already been discussed in Chapter 2 for the linear case. Here, the task is stated in its more general form, i.e., in an RKHS. Note that searching for a model function in an RKHS is a typical task of *nonparametric* modeling; that is, the minimization is performed with respect to functions that are constrained to belong to a specific space, which is infinite dimensional. In the following, we state the regression task, which has been already developed for linear models, to the more general nonlinear form in the RKHS case.

The task of nonlinear regression is typically described as follows: given a data set of the form $\mathcal{D} = \{(y_i, \boldsymbol{x}_i)\}_{i=1}^N$, we aim to estimate the input-output relation between $\boldsymbol{x}_i$ and y_i, i.e., a function f, such that $\mathrm{f}(\boldsymbol{x}_i)$ is "close" to y_i, for all i. This is usually achieved by employing a *loss function*, i.e., a function $\mathrm{L}(y_i, \mathrm{f}(\boldsymbol{x}_i))$, that measures the difference between the observed values, y_i, and the predicted values, $\mathrm{f}(\boldsymbol{x}_i)$, and minimizing the so called *empirical risk*, i.e. $\sum_{i=1}^N \mathrm{L}(y_i, \mathrm{f}(\boldsymbol{x}_i))$. For example, in the least squares regression, one adopts the squared error, i.e., $\mathrm{L}_{\ell_2}(y_i, \mathrm{f}(\boldsymbol{x}_i)) := (y_i - \mathrm{f}(\boldsymbol{x}_i))^2$ and minimizes a quadratic function. Moreover, in order to avoid a solution that overfits the data, we usually attempt to minimize a regularized version, i.e.,

$$\min_{\mathrm{f}} \left\{ \sum_{i=1}^N \mathrm{L}(y_i, \mathrm{f}(\boldsymbol{x}_i)) + \lambda \rho(\mathrm{f}) \right\}, \tag{5.15}$$

where L can be any loss function, e.g., the quadratic, the absolute value, the Vapnik's ϵ-sensitive loss, e.t.c. and ρ an appropriately chosen regularization functional (also see [86]).

In the classic regression task, we assume that the generation mechanism of the data, represented by the training set $\mathcal{D}$, is modeled via the nonlinear mechanism

$$y_i = \underline{\mathrm{f}}(\boldsymbol{x}_i) + \nu_i, \ i = 1, ..., N, \tag{5.16}$$

where $\underline{\mathrm{f}}$ is the original function that generates the uncorrupted data and ν_i's are random noise variables.

Naturally, the choice for the estimate of $\underline{\mathrm{f}}$, strongly depends on the underlying true model. Assuming that this function belongs to an RKHS and motivated by the Representer Theorem, we adopt the linear expansion in (5.11) for the desired solution. According to the kernel ridge regression (KRR) approach, the unknown coefficients are estimated by solving the following (convex) optimization task

$$\begin{aligned} \hat{\boldsymbol{\alpha}} &= \operatorname*{argmin}_{\boldsymbol{\alpha}} \mathrm{J}(\boldsymbol{\alpha}), \\ \mathrm{J}(\boldsymbol{\alpha}) &:= \sum_{n=1}^N \left(y_n - \sum_{m=1}^N \alpha_m \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) \right)^2 + \lambda \|\mathrm{f}\|_{\mathcal{H}}^2, \end{aligned} \tag{5.17}$$

where λ is the regularization parameter and f is given in (5.11). The cost function J of (5.17), can be equivalently written as

$$\mathrm{J}(\boldsymbol{\alpha}) = (\boldsymbol{y} - \boldsymbol{K}\boldsymbol{\alpha})^T (\boldsymbol{y} - \boldsymbol{K}\boldsymbol{\alpha}) + \lambda \boldsymbol{\alpha}^T \boldsymbol{K}^T \boldsymbol{\alpha}, \tag{5.18}$$

where $\boldsymbol{K}$ is the kernel Gram matrix (Definition 5.3) and $\boldsymbol{y} = [y_1, \dots, y_N]^T, \boldsymbol{\alpha} = [\alpha_1, \dots, \alpha_N]^T$. Next, minimization of J with respect to $\boldsymbol{\alpha}$, leads to

$$\left(\boldsymbol{K}^T \boldsymbol{K} + \lambda \boldsymbol{K}^T\right) \hat{\boldsymbol{\alpha}} = \boldsymbol{K}^T \boldsymbol{y},$$

where $\hat{\boldsymbol{\alpha}}$ is the vector of estimates or

$$(\boldsymbol{K} + \lambda \boldsymbol{I}_N) \hat{\boldsymbol{\alpha}} = \boldsymbol{y}, \tag{5.19}$$

where $\boldsymbol{K}^T = \boldsymbol{K}$ has been assumed invertible[1].

Next, by solving the linear system of equations (5.19), the corresponding prediction value of the dependent variable is given by

$$\hat{y}_i = \sum_{n=1}^{N} \hat{\alpha}_n \kappa(\boldsymbol{x}_i, \boldsymbol{x}_n) = \hat{\boldsymbol{\alpha}}^T \boldsymbol{\kappa}(\boldsymbol{x}_i), \ i = 1, \ldots, N, \tag{5.20}$$

where $\boldsymbol{\kappa}(\boldsymbol{x}_i) = [\kappa(\boldsymbol{x}_i, \boldsymbol{x}_1), \ldots, \kappa(\boldsymbol{x}_i, \boldsymbol{x}_N)]^T$ is the vectorised i-th row of the kernel matrix $\boldsymbol{K}$. Finally, combining (5.20) with (5.19), we obtain

$$\hat{y}_i = \boldsymbol{y}^T \left(\boldsymbol{K} + \lambda \boldsymbol{I}_N\right)^{-1} \boldsymbol{\kappa}(\boldsymbol{x}_i), \ i = 1, \ldots, N. \tag{5.21}$$

Alternatively, if we wish to adopt the method with the use of the bias term as the Semi-Representer Theorem suggests, i.e., in (5.12), the cost function is now expressed as

$$\mathrm{J}_b(\boldsymbol{\alpha}) := (\boldsymbol{y} - \boldsymbol{K}\boldsymbol{\alpha} - c\mathbf{1})^T (\boldsymbol{y} - \boldsymbol{K}\boldsymbol{\alpha} - c\mathbf{1}) + \lambda \boldsymbol{\alpha}^T \boldsymbol{K}^T \boldsymbol{\alpha}, \tag{5.22}$$

and minimizing accordingly with respect to both $\boldsymbol{\alpha}$ and c, leads to

$$\begin{bmatrix} \boldsymbol{K} + \lambda \boldsymbol{I}_N & \mathbf{1} \\ \mathbf{1}^T \boldsymbol{K} & N \end{bmatrix} \hat{\boldsymbol{\theta}} = \begin{pmatrix} \boldsymbol{y} \\ \mathbf{1}^T \boldsymbol{y} \end{pmatrix}, \tag{5.23}$$

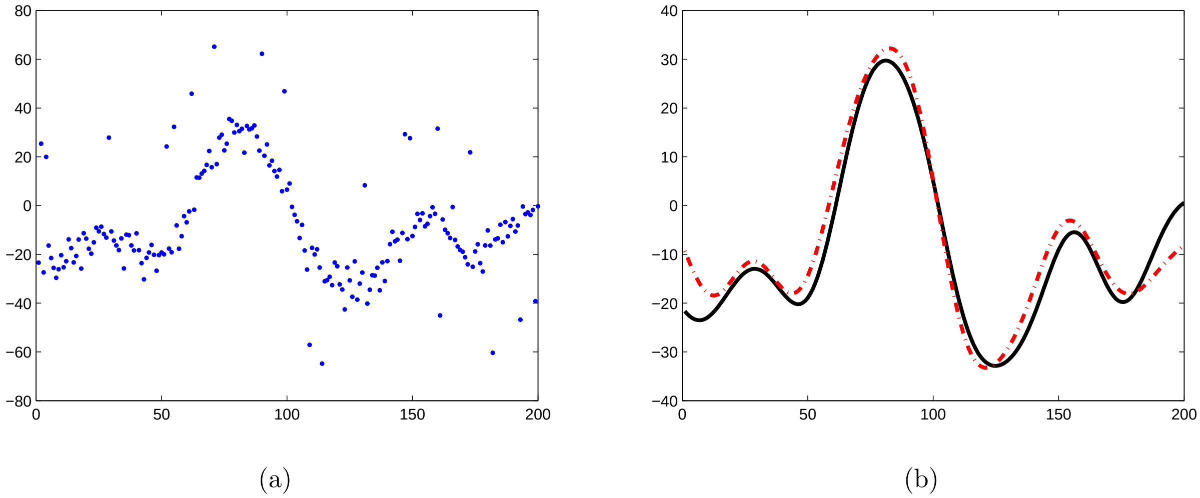
where $\hat{\boldsymbol{\theta}} = \begin{pmatrix} \hat{\boldsymbol{\alpha}} \\ \hat{c} \end{pmatrix}$ is the vector of the estimated kernel coefficient estimates augmented by the estimated bias term, [1].

5.5 The Pursuit for Robustness - Robust Kernel Ridge Regression (RKRR)

The importance of robustness has already been addressed for the linear regression task. In this section, our goal is to pave the way in order to extend our proposed method, i.e., the GARD scheme, to the nonlinear case by employing kernels. This specific modeling lies in the framework of Robust Kernel Ridge Regression (RKRR). Although, initially, it seems that the two tasks (linear and nonlinear) share similarities, there is a major difference that should be emphasized. Searching for a (nonlinear) solution in an RKHS is a typical *nonparametric* task, as opposed to the linear case. Additionally, the regularization term which should be included in the optimization function for the nonlinear case, is due to the fact that data overfitting issues occur, since any function that interpolates the data is a solution. However, the purpose served is twofold, as it is demonstrated in the next chapter, where the proposed method is introduced.

Undeniably, the KRR method seems a logical choice, for a model in the presence of white Gaussian noise, see [80], [1]. However, when outliers are present or when the noise distribution exhibits long tails, the aforementioned method fails. It should be noted that, in principle, both *Support Vector Regression* (SVR) and KRR can be employed to address the KRR task.

[1] This is true, since the Gaussian kernel has been adopted.



(a)

(b)

Figure 5.7: (a) Noisy data of 17 dB inlier noise and 10% outliers. (b) Black line: the uncorrupted data and red dashed line: the estimated data. Estimation is performed with the classic KRR method. It is clear that the performance of the estimation is greatly affected by the presence of outliers (MSE = 10.79).

However, the presence of outliers reduces significantly their performance due to overfitting, [46, 87], even for the case where no inlier noise exists (outliers only). Of course, in SVR, this effect is not as dominant as in the standard KRR, due to the ℓ_1-loss, that it is employed; however its performance remains rather poor and it falls short of the expectations. The challenge of robustizing the KRR task has been mainly studied over the last few years.

The problematic estimation in the presence of outliers via the KRR is demonstrated in Figure 5.7. The original data (black line) in Figure 5.7(b) is contaminated by white Gaussian noise of 17 dB and 10 % outliers, uniformly distributed with values equal to 40 or −40. The noisy observations are shown as blue dots in Figure 5.7(a). The attained MSE is 10.79 (measured over 1000 independent runs), which clearly demonstrates the poor performance of the estimator. Hence, if we wish to improve the estimation, an alternative treatment is required.

As already discussed in Chapter 2, sparsity is the key feature that characterizes the outliers. In other words, it is assumed that the outlier noise contaminates only a small fraction of the output data. To this end, the random noise variable is decomposed into two parts and the model equation in (5.16) is cast as

$$y_i = \underline{\mathrm{f}}(\boldsymbol{x}_i) + \underline{u}_i + \eta_i,\ i = 1, \dots, N, \tag{5.24}$$

where $\underline{\mathrm{f}} \in \mathcal{H}$, $\underline{u}_i$ represents a possible outlier and η_i a noise component. In a more compact form, this can be cast as $\boldsymbol{y} = \underline{\mathbf{f}} + \underline{\boldsymbol{u}} + \boldsymbol{\eta}$, where $\underline{\mathbf{f}} = [\underline{\mathrm{f}}(\boldsymbol{x}_1), \dots, \underline{\mathrm{f}}(\boldsymbol{x}_N)]^T$ and $\underline{\boldsymbol{u}} = [\underline{u}_1, \dots, \underline{u}_N]^T$ is the sparse outlier vector. The decomposition was introduced in the framework of RKRR in [70]. Since $\underline{u}_i$'s are adopted to denote the outliers, most of its values equal zero, except for a few indices. Let $\mathcal{J} = \{1, \dots, N\}$ be the set of coordinates for a vector in $\mathbb{R}^N$. Assuming that the outlier vector $\underline{\boldsymbol{u}} \in \mathbb{R}^N$ is sparse, the support set of $\underline{\boldsymbol{u}}$ is denoted as $\mathcal{S} \subset \mathcal{J}$, with cardinality $|\mathcal{S}| = S << N$. Hence, the fraction of outliers equals S/N.

Our goal is to estimate the input-output relation $\underline{\mathrm{f}}$ from the noisy observations of the data set $\mathcal{D}$. This can be interpreted as the task of of simultaneously estimating both a sparse vector $\boldsymbol{u}$ and as well as a function $\mathrm{f} \in \mathcal{H}$, that maintains a low squared error for $\mathrm{L}(\mathcal{D}, \mathrm{f}, \boldsymbol{u}) = \sum_{i=1}^{N} \left(y_i - \mathrm{f}(\boldsymbol{x}_i) - u_i\right)^2$. Hence, the respective task for the Robust Kernel Ridge Regression (RKRR) task can be cast as:

$$\begin{array}{cc} \min\limits_{\boldsymbol{u}, \mathrm{f} \in \mathcal{H}} & \|\boldsymbol{u}\|_0 \\ \text{subject to} & \sum_{i=1}^{N} \left(y_i - \mathrm{f}(\boldsymbol{x}_i) - u_i\right)^2 + \lambda \|\mathrm{f}\|_{\mathcal{H}}^2 \leq \varepsilon, \end{array} \tag{5.25}$$

for some user-defined parameters λ and ε. Moreover, adopting the linear expansion in (5.11), the optimization task in (5.25) can be written in a more compact form as:

$$\begin{array}{cc} \min\limits_{\boldsymbol{u}, \boldsymbol{a} \in \mathbb{R}^N} & \|\boldsymbol{u}\|_0 \\ \text{subject to} & \|\boldsymbol{y} - \boldsymbol{K}\boldsymbol{a} - \boldsymbol{u}\|_2^2 + \lambda \boldsymbol{a}^T \boldsymbol{K}^T \boldsymbol{a} \leq \varepsilon, \end{array} \tag{5.26}$$

where $\boldsymbol{K}$ is the corresponding kernel Gram matrix. Although our task is now formulated, we are now faced with another challenge. The optimization task in (5.26) is not only non-convex, but combinatorial, due to the nature of the employed ℓ_0(pseudo)-norm. Thus, in order to overcome such an obstacle, recently established works seek for an alternative path.

Despite the fact that several methods exist for the linear method, only two methods have been recently established for the nonlinear kernel-based regression task. Both methods adopt the decomposition of the noise variable into two parts.

5.6 Related Works

5.6.1 Convex Relaxation: Refined Alternating Directions Method of Multipliers (RAM)

In order to achieve stable solutions and mobilize the rich toolbox of convex optimization, many authors prefer to consider the convex relaxation technique of the ℓ_0-norm. The method that was introduced in [69] relies on the substitution of the ℓ_0-norm of the sparse outlier vector $\boldsymbol{u}$ by its closest convex norm, i.e., the ℓ_1-norm, see [67, 68]. Such a relaxation is closely related to the original minimization problem (5.25), since the ℓ_1-norm also preserves parsimonious representations. Thus, the task in (5.25) leads to the following alternative convex formulation:

$$\begin{array}{cc} \min\limits_{\boldsymbol{u}, \mathrm{f} \in \mathcal{H}} & \|\boldsymbol{u}\|_1 \\ \text{subject to} & \sum_{i=1}^{N} \left(y_i - \mathrm{f}(\boldsymbol{x}_i) - u_i\right)^2 + \lambda \|\mathrm{f}\|_{\mathcal{H}}^2 \leq \varepsilon, \end{array} \tag{5.27}$$

for $\varepsilon, \lambda > 0$, . Considering the linear representation (5.11) (no bias term c proposed by the authors), the constraint task in (5.27) is equivalent to

$$\min_{\boldsymbol{\alpha}, \boldsymbol{u} \in \mathbb{R}^N} \left\{ \|\boldsymbol{y} - \boldsymbol{K}\boldsymbol{\alpha} - \boldsymbol{u}\|_2^2 + \lambda \boldsymbol{\alpha}^T \boldsymbol{K}^T \boldsymbol{\alpha} + \mu ||\boldsymbol{u}||_1 \right\}, \tag{5.28}$$

for values of $\mu > 0$, that correspond to the values of $\varepsilon > 0$. The convex minimization form in (5.28) is known as the (generalized) LASSO task [24, 33], which is solvable by a large variety of methods, e.g., using the Alternating Direction Method of Multipliers (ADMM) or its efficient

Algorithm 7 Weighted Alternating directions solver: WAM

```
procedure WAM($\boldsymbol{K}$, $\boldsymbol{y}$, $\lambda$, $\mu$, $\boldsymbol{w}$)
    $\hat{\boldsymbol{u}}_{(0)} \leftarrow \mathbf{0}$
    for $k = 1, 2, ...$ do
        $\hat{\boldsymbol{\alpha}}_{(k)} \leftarrow \left[\boldsymbol{K} + \lambda \boldsymbol{I}_N\right]^{-1} \left(\boldsymbol{y} - \hat{\boldsymbol{u}}_{(k-1)}\right)$
        $\boldsymbol{r}_{(k)} \leftarrow \boldsymbol{y} - \boldsymbol{K}\hat{\boldsymbol{\alpha}}_{(k)},\ \hat{u}_{(k),i} \leftarrow \mathrm{S}\left(r_{(k),i}, \frac{w_i \mu}{2}\right), i = 1, ...N$
    Output: $\hat{\boldsymbol{\alpha}}_{(k)}$ and $\hat{\boldsymbol{u}}_{(k)}$ after $k$ iterations.
```

Algorithm 8 Refined AM solver: RAM

```
procedure RAM($\boldsymbol{K}$, $\boldsymbol{y}$, $\lambda$, $\mu$, $\delta$)
    $\left[\hat{\boldsymbol{\alpha}}_{(0)}, \hat{\boldsymbol{u}}_{(0)}\right] \leftarrow \mathrm{WAM}(\boldsymbol{K}, \boldsymbol{y}, \lambda, \mu, \mathbf{1})$
    for $k = 1, 2, ...$ do
        $w_{(k),i} = (|\hat{u}_{(k-1),i}| + \delta)^{-1},\ i = 1, ..., N,$
        $\left[\hat{\boldsymbol{\alpha}}_{(k)}, \hat{\boldsymbol{u}}_{(k)}\right] \leftarrow \mathrm{WAM}(\boldsymbol{K}, \boldsymbol{y}, \lambda, \mu, \boldsymbol{w}_{(k)})$
    Output: $\hat{\boldsymbol{\alpha}}_{(k)}$ and $\hat{\boldsymbol{u}}_{(k)}$ after $k$ iterations.
```

implementation, i.e., the so-called *AM solver*, as proposed in [69]. Although existing works on based on the ℓ_1-norm minimization techniques provide guarantees of a fairly good approximation, in practice, the relaxation certainly compromises for something less, at least in terms of the achieved MSE.

Additionally, inspired by the work in [88], the authors have proposed in [69] an improvement of the optimization task (5.28). This has been achieved by using a non-convex relaxation technique of the task in (5.26), that attempts to solve

$$\min_{\boldsymbol{\alpha}, \boldsymbol{u} \in \mathbb{R}^N} \left\{ \|\boldsymbol{y} - \boldsymbol{K}\boldsymbol{\alpha} - \boldsymbol{u}\|_2^2 + \lambda \boldsymbol{\alpha}^T \boldsymbol{K}^T \boldsymbol{\alpha} + \mu \sum_{i=1}^{N} \log\left(|u_i| + \delta\right) \right\}, \tag{5.29}$$

for $\delta > 0$ sufficiently small in order to avoid numerical instability. Since the additional regularization term is now concave, the overall problem is non-convex. However, the last term in (5.29) could be replaced by the local linear approximation of the logarithmic function via the use of the reweighted ℓ_1-norm minimization technique proposed in [88], leading to the following iteration for $k = 0, 1, \ldots$

$$[\hat{\boldsymbol{a}}_{(k)}, \hat{\boldsymbol{u}}_{(k)}] := \underset{\boldsymbol{\alpha}, \boldsymbol{u}}{\mathrm{argmin}} \left\{ \|\boldsymbol{y} - \boldsymbol{K}\boldsymbol{\alpha} - \boldsymbol{u}\|_2^2 + \lambda \boldsymbol{\alpha}^T \boldsymbol{K}^T \boldsymbol{\alpha} + \mu \sum_{i=1}^{N} w_{(k),i} |u_i| \right\}, \tag{5.30}$$

where the elements of the vector of weights, $\boldsymbol{w}_{(k)}$, are given by:

$$w_{(k),i} := \left(\left|u_{(k-1),i}\right| + \delta\right)^{-1},\ i = 1, ..., N. \tag{5.31}$$

The procedure could also be viewed as a refinement step of the AM solver (RAM solver) and the scheme is summarized in Algorithms 8 and 7, where S denotes the soft-thresholding operator[2]. It should also be noted that, the original AM solver (an improved implementation of

[2] Soft-thresholding operator: $\mathrm{S}(z, \gamma) := \mathrm{sgn}(z) \cdot \max\{0, |z| - \gamma\}$.

ADMM), could be obtained from Algorithm 7, for weights equal to one, i.e., by setting $\boldsymbol{w} = \boldsymbol{1}$; the WAM solver is a more general scheme. For example, if for some reason the weights are set not equal to one, but to other values, one may establish a better performance scheme for certain types of problems. Notable is also the fact that, the scheme could be implemented more efficiently, by applying the Cholesky factorization (with cost $O(N^2)$ after the factorization) instead of an inversion, since matrix $[\boldsymbol{K} + \lambda \boldsymbol{I}_N]$ remains unchanged. The aforementioned refinement step improves the performance of the original AM solver [69], significantly. Moreover, it should be noted that, in practice, more than two iterations do not offer significant improvements on its performance. Furthermore, we should emphasize that the optimum parameters (λ_*, μ_*) to be used with RAM (in terms of MSE), are not identical to the parameters (λ, μ) of AM solver in (5.28) (WAM with $\boldsymbol{w} = \boldsymbol{1}$). Thus, for $\mu_* > \mu$ the convergence speed of the RAM scheme is also improved. Finally, theoretical properties of the method indicate that for small values of $\delta > 0$, the method attempts to approximate the ℓ_0-norm of the sparse outlier vector $\boldsymbol{u}$.

5.6.2 Sparse Bayesian learning: Robust Relevance Vector Machine (RB-RVM)

Relevance vector machines (RVM) have recently attracted much interest in the research community, because they provide a number of advantages. They are based on a Bayesian formulation of a linear model with an appropriate prior that results in a sparse representation. As a consequence, they can generalize well and provide inferences at low computational cost. The Sparse Bayesian Learning (SBL) scheme has already been presented in Chapter 2. As an extension of this work for the kernel-based nonlinear regression task, the Robust Bayesian-RVM (RB-RVM) is an RVM modified scheme that employees the use of hyperparameters to impose sparsity on the outlier estimates [70, 72, 1].

Assuming that f admits the linear representation in (5.12), the authors suggest the input-output relation of the form:

$$\boldsymbol{y} = \boldsymbol{K}\underline{\boldsymbol{\alpha}} + \underline{c}\boldsymbol{1} + \underline{\boldsymbol{u}} + \boldsymbol{\eta} = \boldsymbol{A}\underline{\boldsymbol{z}} + \boldsymbol{\eta}, \tag{5.32}$$

where $\boldsymbol{A} = [\boldsymbol{K}\ \boldsymbol{1}\ \boldsymbol{I}_N]$, $\underline{\boldsymbol{z}} = \begin{pmatrix} \underline{\boldsymbol{\theta}} \\ \underline{\boldsymbol{u}} \end{pmatrix}$ and $\underline{\boldsymbol{\theta}} = \begin{pmatrix} \underline{\boldsymbol{\alpha}} \\ \underline{c} \end{pmatrix}$, which is the vector of the unknown coefficients augmented by the bias term. Adopting the Gaussian assumption for the inlier noise, the joint posterior distribution of $\underline{\boldsymbol{\theta}}$ and $\underline{\boldsymbol{u}}$ (assumed to be independent) is estimated via:

$$p(\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}}|\boldsymbol{y}) = \frac{p(\underline{\boldsymbol{\theta}})p(\underline{\boldsymbol{u}})p(\boldsymbol{y}|\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}})}{p(\boldsymbol{y})},$$

where the likelihood term is given by

$$p(\boldsymbol{y}|\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}}) = \mathcal{N}(\boldsymbol{A}\underline{\boldsymbol{z}}, \sigma^2 \boldsymbol{I}_N), \tag{5.33}$$

where σ^2 is the inlier Gaussian noise variance. Next, priors which 'promote sparsity' are assigned to the vectors $\underline{\boldsymbol{\theta}}$ and $\underline{\boldsymbol{u}}$. To this end,

$$p(\boldsymbol{v}|\boldsymbol{h}) = \prod_{i=0}^{N} \mathcal{N}(v_i|0, h^{-1}) \tag{5.34}$$

holds for vectors $\underline{\boldsymbol{\theta}}$ and $\underline{\boldsymbol{u}}$, with hyperparameters $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_{N+1}]^T$ and $\boldsymbol{\delta} = [\delta_1, \ldots, \delta_N]^T$, respectively, where each of the hyperparameters follows a uniform distribution. Next, follows the *inference* stage.

Following the RVM inference rationale, we first find point-estimates for the hyperparameters $\boldsymbol{\beta}$, $\boldsymbol{\delta}$ and the inlier noise variance σ^2, by maximizing

$$p\left(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{\delta}, \sigma^2\right) = \int p\left(\boldsymbol{y}|\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}}, \sigma^2\right) p(\underline{\boldsymbol{\theta}}|\boldsymbol{\beta}) p(\underline{\boldsymbol{u}}|\boldsymbol{\delta}) d\underline{\boldsymbol{\theta}} d\underline{\boldsymbol{u}}.$$

Since, all the distributions in the right hand side are Gaussian with zero mean, it can be shown that $p(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{\delta}, \sigma^2)$ is a zero-mean Gaussian distribution with covariance matrix $\sigma^2 \boldsymbol{I}_N + \boldsymbol{A}^T \boldsymbol{C} \boldsymbol{A}$, where $\boldsymbol{C} := \operatorname{diag}\left(\boldsymbol{\beta}^T, \boldsymbol{\delta}^T\right)$. The maximization of $p(\boldsymbol{y}|\boldsymbol{\beta}, \boldsymbol{\delta}, \sigma^2)$ is performed by the EM algorithm[3] and the parameters $\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\delta}}$ and $\hat{\sigma}^2$ are estimated. With this point estimation of the hyperparameters and the noise variance, the (conditional) posterior distribution is given by

$$p(\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}}|\boldsymbol{y}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\delta}}, \hat{\sigma}^2) = \frac{p(\boldsymbol{y}|\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}}, \hat{\sigma}^2) p(\underline{\boldsymbol{\theta}}|\hat{\boldsymbol{\beta}}) p(\underline{\boldsymbol{u}}|\hat{\boldsymbol{\delta}})}{p(\boldsymbol{y}|\hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\delta}}, \hat{\sigma}^2)}.$$

Since, all the terms in the numerators are Gaussian, it can be shown that this is again a Gaussian distribution with covariance and mean given by

$$\widehat{\boldsymbol{\Sigma}} = \left(\sigma^{-2} \boldsymbol{A}^T \boldsymbol{A} + \widehat{\boldsymbol{C}}\right)^{-1} \text{ and } \widehat{\boldsymbol{\mu}} = \sigma^{-2} \widehat{\boldsymbol{\Sigma}} \boldsymbol{A}^T \boldsymbol{y}, \tag{5.35}$$

where $\widehat{\boldsymbol{C}} := \operatorname{diag}\left(\hat{\beta}_1, ..., \hat{\beta}_{N+1}, \hat{\delta}_1, ..., \hat{\delta}_N\right)$. To obtain the posterior distribution $p(\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}}|\boldsymbol{y})$, an approximation is used and thus we obtain

$$p(\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}}|\boldsymbol{y}) = p(\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}}|\boldsymbol{y}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\delta}}, \hat{\sigma}^2).$$

The desired posterior distribution of $\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}}$ is a Gaussian distribution with the posterior covariance and mean given by (5.35).

Finally, *prediction* is accomplished by using (5.12) as well as the covariance and mean of the posterior distribution in (5.35). The predictive distribution of $\hat{y}$ is given by

$$p(\hat{y}|\boldsymbol{y}, \hat{\boldsymbol{\beta}}, \hat{\sigma}^2) = \int p(\hat{y}|\underline{\boldsymbol{\theta}}, \hat{\sigma}^2) p(\underline{\boldsymbol{\theta}}|\boldsymbol{y}, \hat{\boldsymbol{\beta}}) d\underline{\boldsymbol{\theta}}, \tag{5.36}$$

where the posterior distribution of $\underline{\boldsymbol{\theta}}$, i.e., $p(\underline{\boldsymbol{\theta}}|\boldsymbol{y}, \hat{\boldsymbol{\beta}})$, can be obtained from the joint posterior distribution $p(\underline{\boldsymbol{\theta}}, \underline{\boldsymbol{u}}|\boldsymbol{y}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\delta}}, \hat{\sigma}^2)$. This is a Gaussian distribution with mean and covariance corresponding to the first part, $\underline{\boldsymbol{\theta}}$, of the $\underline{\boldsymbol{z}}$ vector, i.e.,

$$\widehat{\boldsymbol{\Sigma}}_{\underline{\boldsymbol{\theta}}} = \widehat{\boldsymbol{\Sigma}}_{\mathcal{J},\mathcal{J}} \text{ and } \widehat{\boldsymbol{\mu}}_{\underline{\boldsymbol{\theta}}} = \widehat{\boldsymbol{\mu}}_{\mathcal{J}}. \tag{5.37}$$

where $\mathcal{J} = \{1, \ldots, N+1\}$. The difference of the scheme, with respect to the classical RVM formulation is that instead of inferring just the parameter vector $\underline{\boldsymbol{\theta}}$ to the RVM algorithm, it also infers the joint parameter-outlier vector $\underline{\boldsymbol{z}}$. This is accomplished by replacing the matrix $[\boldsymbol{K}\ \boldsymbol{1}]$ by the matrix $[\boldsymbol{K}\ \boldsymbol{1}\ \boldsymbol{I}_N]$. The (MATLAB) code for the method can be found in:
`http://www.vectoranomaly.com/downloads/downloads.htm`.

[3] EM stands for *Expectation Maximization* methods.

Chapter 6

Robust Nonlinear Regression: A Greedy Approach Employing Kernels

6.1 Introduction

Our main goal in this chapter is to extend the established work for the robust linear regression task to the more general case of nonlinear models in RKHS. To this end, we resort to the original problem formulation in (5.25) and attempt to solve the task via a modified Orthogonal Matching Pursuit (OMP) scheme.

Moreover, theoretical properties of the proposed scheme regarding the identification of the outliers for the case only outlier noise exists are established. Finally, an extended set of experiments is reported, in which our theoretical findings are verified. The new scheme is tested against its competitors, i.e., the RB-RVM and RAM methods, which have already been described in Chapter 5.

6.2 Kernel Greedy Algorithm for Robust Denoising (KGARD)

Our focus here is turned on attempting to solve the RKRR task via a *greedy* optimization technique, which is inspired by the Orthogonal Matching Pursuit (OMP).

Although our motivation was originally driven by (5.26), we have also introduced and worked with a variant, concerning the regularization term. This is because, we have found that, in practice, this alternative leads to an enhanced performance. In the first of the two versions the regularization is performed with the use of the $\mathcal{H}$-norm. Additionally, a bias term (see the linear expansion in (5.12)) is also included:

$$\begin{aligned} \min_{\boldsymbol{u},\boldsymbol{a}\in\mathbb{R}^N, c\in\mathbb{R}} \quad & \|\boldsymbol{u}\|_0 \\ \text{subject to} \quad & \|\boldsymbol{y}-\boldsymbol{K}\boldsymbol{a}-c\boldsymbol{1}-\boldsymbol{u}\|_2^2+\lambda\boldsymbol{a}^T\boldsymbol{K}^T\boldsymbol{a}\leq\varepsilon. \end{aligned} \tag{6.1}$$

In the alternative formulation, the regularization is performed via the use of the ℓ_2-norm of the unknown parameters, which is a standard regularization technique, [86]. Thus, the problem is now formulated as

$$\begin{aligned} \min_{\boldsymbol{u},\boldsymbol{a}\in\mathbb{R}^N,c\in\mathbb{R}} \quad & \|\boldsymbol{u}\|_0 \\ \text{subject to} \quad & \|\boldsymbol{y}-\boldsymbol{K}\boldsymbol{a}-c\boldsymbol{1}-\boldsymbol{u}\|_2^2+\lambda\left\|\boldsymbol{a}\right\|_2^2+\lambda c^2\leq\varepsilon, \end{aligned} \tag{6.2}$$

Obviously, the difference lies solely on the regularization term.

Since both tasks (6.1) and (6.2) are combinatorial, due to the minimization of the ℓ_0-norm, a straightforward computation of a solution is impossible. To bypass this obstacle, we will derive a modified version of the OMP algorithm, that attempts to solve (6.1) and (6.2), see [38].

First, one should notice that in both cases, the quadratic inequality constraints could also be written in a more compact form as follows:

$$\mathrm{J}(\boldsymbol{z})=\|\boldsymbol{y}-\boldsymbol{A}\boldsymbol{z}\|_2^2+\lambda\boldsymbol{z}^T\boldsymbol{B}\boldsymbol{z}\leq\varepsilon, \tag{6.3}$$

where

$$\boldsymbol{A}=\begin{bmatrix}\boldsymbol{K} & \boldsymbol{1} & \boldsymbol{I}_N\end{bmatrix},\ \boldsymbol{z}=\begin{pmatrix}\boldsymbol{\alpha}\\ c\\ \boldsymbol{u}\end{pmatrix}, \tag{6.4}$$

and for the choice of matrix $\boldsymbol{B}$ either one of the following matrices could be used:

$$\boldsymbol{B}=\begin{bmatrix}\boldsymbol{K}^T & \boldsymbol{0} & \boldsymbol{O}_N\\ \boldsymbol{0}^T & 0 & \boldsymbol{0}^T\\ \boldsymbol{O}_N & \boldsymbol{0} & \boldsymbol{O}_N\end{bmatrix} \text{ or } \begin{bmatrix}\boldsymbol{I}_N & \boldsymbol{0} & \boldsymbol{O}_N\\ \boldsymbol{0}^T & 1 & \boldsymbol{0}^T\\ \boldsymbol{O}_N & \boldsymbol{0} & \boldsymbol{O}_N\end{bmatrix}, \tag{6.5}$$

depending on whether the model (6.1) or (6.2) is adopted, respectively. The process adopted is similar to the one for the linear case. The $2N+1$ column vectors of the matrix $\boldsymbol{A}$ are divided into two complementary subsets: the active set, $\widetilde{\mathcal{S}}_k$, which contains the indices of the active columns of the matrix at the k-th step, and the inactive set, $\widetilde{\mathcal{S}}_k^c$, which contains the remaining ones. Thus, $\boldsymbol{A}_{\widetilde{\mathcal{S}}_k}$ denotes the column vectors of matrix $\boldsymbol{A}$ restricted over the subset $\widetilde{\mathcal{S}}_k$. For $\boldsymbol{B}$ defined in (6.5), at each k-th step, $\boldsymbol{B}_{\widetilde{\mathcal{S}}_k,\widetilde{\mathcal{S}}_k}$ comprises the first $N+1+k$ rows and columns of the matrix. Moreover, the set, $\mathcal{S}_k$, defined in (4.2) is adopted for denoting the support for the sparse vector estimate and refers strictly to columns of the identity matrix; however, here, the cardinality of the initial set, $\widetilde{\mathcal{S}}_0$, is $N+1$.

Initially, only the first $N+1$ columns of matrix $\boldsymbol{A}$ are activated. Thus, $k=0$ leads to the initialization of the active set $\widetilde{\mathcal{S}}_0=\{1,2,\ldots,N+1\}$ with the corresponding initial matrices:

$$\boldsymbol{A}_{\widetilde{\mathcal{S}}_0}=[\boldsymbol{K}\ \boldsymbol{1}],$$

and

$$\boldsymbol{B}_{\widetilde{\mathcal{S}}_0,\widetilde{\mathcal{S}}_0}=\begin{bmatrix}\boldsymbol{K}^T & \boldsymbol{0}\\ \boldsymbol{0}^T & 0\end{bmatrix} \text{ or } \boldsymbol{I}_{N+1},$$

depending on the model selection, i.e., (6.1) or (6.2), respectively. Hence, the solution to the initial LS problem is given by

$$\hat{\boldsymbol{z}}_{(0)} \ :=\ \operatorname*{argmin}_{\boldsymbol{z}}\mathrm{J}_{(0)}(\boldsymbol{z})=\left(\boldsymbol{A}_{\widetilde{\mathcal{S}}_0}^T\boldsymbol{A}_{\widetilde{\mathcal{S}}_0}+\lambda\boldsymbol{B}_{\widetilde{\mathcal{S}}_0,\widetilde{\mathcal{S}}_0}\right)^{-1}\boldsymbol{A}_{\widetilde{\mathcal{S}}_0}^T\boldsymbol{y}.$$

Algorithm 9 Kernel Greedy Algorithm for Robust Denoising: KGARD

```
procedure KGARD($\boldsymbol{K},\ \boldsymbol{y},\ \lambda,\ \epsilon$)
    $k \leftarrow 0$
    $\widetilde{\mathcal{S}}_0 \leftarrow \{1, 2, ..., N+1\},\ \mathcal{S}_0^c \leftarrow \{1, ..., N\}$, $\boldsymbol{A} = [\boldsymbol{K}\ \boldsymbol{1}\ \boldsymbol{I}_N]$, $\boldsymbol{B}$ in (6.5)
    $\hat{\boldsymbol{z}}_{(0)} \leftarrow \left(\boldsymbol{A}_{\widetilde{\mathcal{S}}_0}^T \boldsymbol{A}_{\widetilde{\mathcal{S}}_0} + \lambda \boldsymbol{B}_{\widetilde{\mathcal{S}}_0, \widetilde{\mathcal{S}}_0}\right)^{-1} \boldsymbol{A}_{\widetilde{\mathcal{S}}_0}^T \boldsymbol{y}$        ▷ Initial reg. LS solution step.
    $\boldsymbol{r}_{(0)} \leftarrow \boldsymbol{y} - \boldsymbol{A}_{\widetilde{\mathcal{S}}_0} \hat{\boldsymbol{z}}_{(0)}$
    while $\|\boldsymbol{r}_{(k)}\|_2 > \epsilon$ do
        $k \leftarrow k+1$
        $j_k \leftarrow \text{argmax}_{j \in \mathcal{S}_{k-1}^c} |r_{(k-1),j}|$, $i_k = j_k + |\widetilde{\mathcal{S}}_0|$        ▷ Selection step.
        $\widetilde{\mathcal{S}}_k \leftarrow \widetilde{\mathcal{S}}_{k-1} \cup \{i_k\},\ \mathcal{S}_k^c \leftarrow \mathcal{S}_{k-1}^c \setminus \{j_k\}$
        $\hat{\boldsymbol{z}}_{(k)} \leftarrow \left(\boldsymbol{A}_{\widetilde{\mathcal{S}}_k}^T \boldsymbol{A}_{\widetilde{\mathcal{S}}_k} + \lambda \boldsymbol{B}_{\widetilde{\mathcal{S}}_k, \widetilde{\mathcal{S}}_k}\right)^{-1} \boldsymbol{A}_{\widetilde{\mathcal{S}}_k}^T \boldsymbol{y}$        ▷ Reg. LS solution step.
        $\boldsymbol{r}_{(k)} \leftarrow \boldsymbol{y} - \boldsymbol{A}_{\widetilde{\mathcal{S}}_k} \hat{\boldsymbol{z}}_{(k)}$
    Output: $\hat{\boldsymbol{z}}_{(k)} = \left(\hat{\boldsymbol{\alpha}}_{(k)}^T, \hat{c}_{(k)}, \hat{\boldsymbol{u}}_{(k)}^T\right)^T$ after $k$ iterations.
```

Next, the method computes the initial residual, $\boldsymbol{r}_{(0)} = \boldsymbol{y} - \boldsymbol{A}_{\widetilde{\mathcal{S}}_0} \hat{\boldsymbol{z}}_{(0)}$, and identifies an outlier[1], as the most prominent value of the residual vector. The selected index, say $j_1 \in \mathcal{S}_0^c$, corresponds to another index $i_1 = j_1 + N + 1$, which is then added into the set of active columns, i.e., $\widetilde{\mathcal{S}}_1 = \widetilde{\mathcal{S}}_0 \cup \{i_1\}$. Thus, the matrix $\boldsymbol{A}_{\widetilde{\mathcal{S}}_0}$ is augmented by the column vector, $\boldsymbol{e}_{j_1}$, drawn from matrix $\boldsymbol{I}_N$ to form matrix $\boldsymbol{A}_{\widetilde{\mathcal{S}}_1}$. Accordingly, the matrix $\boldsymbol{B}_{\widetilde{\mathcal{S}}_0, \widetilde{\mathcal{S}}_0}$ is augmented by a zero row and a zero column, forming $\boldsymbol{B}_{\widetilde{\mathcal{S}}_1, \widetilde{\mathcal{S}}_1}$. The new LS task is solved again (over the matrices $\boldsymbol{A}_{\widetilde{\mathcal{S}}_1}$, $\boldsymbol{B}_{\widetilde{\mathcal{S}}_1, \widetilde{\mathcal{S}}_1}$) and the new residual, $\boldsymbol{r}_{(1)}$, is computed. The process, summarized in Algorithm 9, is repeated until the residual drops below a predefined threshold.

Simply stated, the algorithm alternates between a regularized LS task and a column selection step that enlarges the solution subspace iteratively, in order to minimize the residual error. Although it shares resemblance to its predecessor for the linear regression task, i.e., GARD, it is different in many ways. The basic differences are:

- A regularized LS problem is solved instead of a LS task; that is,

$$\min_{\boldsymbol{z}} \mathrm{J}_{(k)}(\boldsymbol{z}) = \min_{\boldsymbol{z}} \left\{ \|\boldsymbol{y} - \boldsymbol{A}_{\widetilde{\mathcal{S}}_k} \boldsymbol{z}\|_2^2 + \lambda \boldsymbol{z}^T \boldsymbol{B}_{\widetilde{\mathcal{S}}_k, \widetilde{\mathcal{S}}_k} \boldsymbol{z} \right\}, \tag{6.6}$$

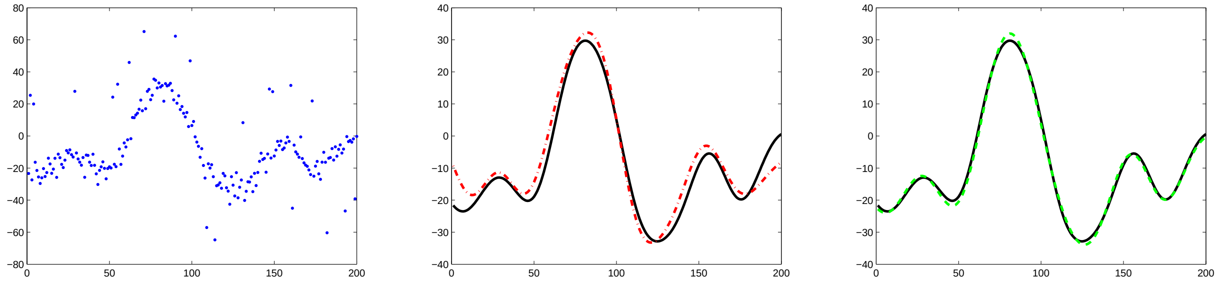
  where k is the current step index.

- The initialization for the set of active columns of matrix $\boldsymbol{A}$ is different. Besides, we should take into account that the number of the data is less than the number of unknowns.

These differences, which are imposed by the different structure of the problem which is formulated in an RKHS, lead to a completely distinct performance analysis for the method with respect to its predecessor for the linear case, i.e., GARD.

The gain of the robust estimation with KGARD over the standard KRR task is demonstrated in Figure 6.1. We consider our input data as 400 equidistant points over the interval $[0, 1)$ and generate the uncorrupted data via a nonlinear function $\underline{f} \in \mathcal{H}$ as a (sparse) linear

[1] If outliers are not present the algorithm terminates and no outlier estimate exists in the solution $\hat{\boldsymbol{z}}_0$.



(a) (b) (c)

Figure 6.1: The significance of robust estimation: (a) Data corrupted by both inlier and 10% of outlier noise. (b) The black and the red dashed lines correspond to the uncorrupted data and the non-robust estimation performed, respectively. The MSE over the training set is 10.79. (c) The black and the green dashed lines correspond to the uncorrupted data and the robust estimation performed with KGARD, respectively. The MSE over the training set is 1.21.

combination of Gaussian kernels with $\sigma = 0.1$ centered at a small number (i.e., 8 to 35) of those points (randomly selected). Next, the data is separated into two sets, the training and the validation (testing) subsets. The training subset consists of the 200 odd indexed points of the entire set (first, third, e.t.c.) and the validation subset includes the remaining ones (even indices). The noisy data emerge from (5.32), where the inlier noise $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, 4^2\boldsymbol{I}_N)$ and the vector of outliers $\underline{\boldsymbol{u}}$ has non-zero values equal to 40 or -40 uniformly distributed over N coordinates at a fraction of 10%. Finally, the MSE is measured over 1000 Monte-Carlo runs (independent experiments) for both the training and the validation set (more details on the experiment can be found in Section 6.4.2).

In Figure 6.1(a), we have plotted the noisy data (blue dots) of the training set (for a specific simulation). The red dashed line in Figure 6.1(b) corresponds to the estimation performed by the simple KRR task. Figure 6.1(c) corresponds to the robust estimation, performed via the KGARD. Comparing the two figures the advantages of the KGARD method are clear. Although both versions (6.1) and (6.2) are suitable for dealing with the sparse minimization task, in practice, the selection of the task in (6.2) turns out to be a better choice. In order to justify our claim, we have performed a comparative evaluation for the precedent experiment. The MSE attained via the $\mathcal{H}$-norm regularization is $MSE = 1.35$, over both the training and validation set. However, when performing the estimation with the ℓ_2-norm regularization, the respective value is *reduced* to $MSE = 1.21$, a performance improved by 10.4%. Hence, in the following, $\boldsymbol{B}$ will be adopted to denote the matrix used with the regularization performed with the ℓ_2-norm (defined in (6.5)). Finally, it should be noted that all λ and ϵ parameters have been chosen to correspond to the values resulting to the best performance, after extensive experimentation.

Remark 6.1. *In order to simplify the notation, in the following we adopt $\boldsymbol{A}_{(k)}$ and $\boldsymbol{B}_{(k)}$ to refer to the matrices $\boldsymbol{A}_{\widetilde{\mathcal{S}}_k}$ and $\boldsymbol{B}_{\widetilde{\mathcal{S}}_k,\widetilde{\mathcal{S}}_k}$, respectively, at the k-th step.*

Remark 6.2. *Once a column has been selected at the k-th step, it cannot be selected again in any subsequent step, since the corresponding residual coordinate is zero. In other words, the algorithm always selects a column from the last part of $\boldsymbol{A}$, i.e., matrix $\boldsymbol{I}_N$, that is not included in $\mathcal{S}_k$.*

Remark 6.3. *For the following the implementation based on* (6.2) *is adopted, due to the improved performance. Thus, $\boldsymbol{B}$ denotes the respective matrix defined in* (6.5).

6.2.1 Efficient Implementations

As we have already discussed for GARD, faster implementations are also possible for KGARD. Initially, the inversion of matrix $\boldsymbol{A}_{(0)}^T\boldsymbol{A}_{(0)} + \lambda\boldsymbol{B}_{(0)}$ plus the multiplication of $\boldsymbol{A}_{(0)}^T\boldsymbol{y}$ requires $(N+1)^3$ flops. At each of the steps, the required complexity is $\mathrm{O}\left((N+k+1)^3\right)$, while the total cost for the method is

$$\mathrm{O}\left(3(k+1)N^3 + (3/2)N^2k^2 + (5/3)Nk^3 + k^4/4\right).$$

Such a cost is acceptable, since $k << N$ is assumed[2]. However, the complexity of the method could be further reduced, since a large part of the inverted matrix, at each of the subsequent steps, remains unchanged. To this end, several methods could be employed [76].

The first technique, which has been applied to the proposed scheme, is the *Matrix Inversion Lemma* (*MIL*) (see Appendix B). The initial computational cost requirement is cubic, due to the inversion of the matrix

$$\boldsymbol{M}_{(0)} := \boldsymbol{A}_{(0)}^T\boldsymbol{A}_{(0)} + \lambda\boldsymbol{B}_{(0)} = \begin{bmatrix} \boldsymbol{K}^T\boldsymbol{K} + \lambda\boldsymbol{I}_N & \boldsymbol{K}^T\boldsymbol{1} \\ \boldsymbol{1}^T\boldsymbol{K} & N+\lambda \end{bmatrix}. \tag{6.7}$$

At each step, the column vector $\boldsymbol{e}_{j_k}$ is selected from matrix $\boldsymbol{I}_N$ and matrices $\boldsymbol{A}_{(k-1)}$, $\boldsymbol{B}_{(k-1)}$ are augmented, forming $\boldsymbol{A}_{(k)}$ and $\boldsymbol{B}_{(k)}$. Next, follows the inversion of the new matrix $\boldsymbol{M}_{(k)}$. However, with the application of the *MIL*, the inversion at each step is avoided, due to the computation of

$$\boldsymbol{M}_{(k)} := (\boldsymbol{A}_{(k)}^T\boldsymbol{A}_{(k)} + \lambda\boldsymbol{B}_{(k)}) = \begin{bmatrix} \boldsymbol{M}_{(k-1)} & \boldsymbol{A}_{(k-1)}^T\boldsymbol{e}_{j_k} \\ \boldsymbol{e}_{j_k}^T\boldsymbol{A}_{(k-1)} & 1 \end{bmatrix}. \tag{6.8}$$

and its inverse, recursively. Thus, the required cost for this update drops to $\mathrm{O}\left((N+1+k)^2\right)$ per iteration. However, since the inversion of matrix $\boldsymbol{M}_{(0)}$ could not be avoided, the total cost after k steps becomes now:

$$\mathrm{O}\left(2N^3 + 2kN^2 + k^3/3 + (3/2)k^2N\right).$$

An alternative technique, which also offers an efficient implementation is the *Cholesky decomposition* for matrix $\boldsymbol{M}_{(k)}$. This is summarized in the following steps:

- *Replace* the initial regularized Least Squares solution **step 4** of Algorithm 9, with:
  Factorization step: $\boldsymbol{M}_{(0)} = \boldsymbol{L}_{(0)}\boldsymbol{L}_{(0)}^T$

  Solve $\boldsymbol{L}_{(0)}\boldsymbol{L}_{(0)}^T\hat{\boldsymbol{z}}_{(0)} = \boldsymbol{A}_{(0)}^T\boldsymbol{y}$ using:
  - forward substitution $\boldsymbol{L}_{(0)}\boldsymbol{q} = \boldsymbol{A}_{(0)}^T\boldsymbol{y}$
  - backward substitution $\boldsymbol{L}_{(0)}^T\hat{\boldsymbol{z}}_{(0)} = \boldsymbol{q}$

[2] The k step reflects on the detection of an outlier, which in most applications is relatively low, since the outlier vector is assumed to be sparse.

Complexity: $\mathrm{O}\left((N+1)^3/3 + (N+1)^2\right)$

- Replace the regularized Least Squares solution **step 10** of Algorithm 9, with:
  Compute $\boldsymbol{d}$ such that: $\boldsymbol{L}_{(k-1)}\boldsymbol{d} = \boldsymbol{A}^T_{(k-1)}\boldsymbol{e}_{j_k}$
  Compute: $b = \sqrt{1 - ||\boldsymbol{d}||_2^2}$
  Matrix Update: $\boldsymbol{L}_{(k)} = \begin{bmatrix} \boldsymbol{L}_{(k-1)} & \boldsymbol{0} \\ \boldsymbol{d}^T & b \end{bmatrix}$
  Solve $\boldsymbol{L}_{(k)}\boldsymbol{L}^T_{(k)}\hat{\boldsymbol{z}}_{(k)} = \boldsymbol{A}^T_{(k)}\boldsymbol{y}$ using:
    - forward substitution $\boldsymbol{L}_{(k)}\boldsymbol{p} = \boldsymbol{A}^T_{(k)}\boldsymbol{y}$
    - backward substitution $\boldsymbol{L}^T_{(k)}\hat{\boldsymbol{z}}_{(k)} = \boldsymbol{p}$

  Complexity: $\mathrm{O}\left((5/2)N^2 + 4kN + (3/2)k^2\right)$ per iteration.

Employing the Cholseky decomposition plus the update step leads to a reduction of the total computational cost to

$$\mathrm{O}\left((N+1)^3/3 + k^3/2 + N^2(5k/2+1)\right),$$

which is the implementation adopted throughout this work for KGARD (recall that $k << N$). Finally, another technique that could also be applied is the QR factorization. However, this leads to a slightly more demanding implementation compared to the Cholesky decomposition.

6.2.2 Further Improvements on KGARD's Performance

In order to simplify the theoretical analysis and reduce the number of the corresponding equations, the proposed algorithm employs the same regularization parameter for all kernel coefficients. However, one may employ a more general scheme as follows:

$$\begin{array}{cc} \min\limits_{\boldsymbol{a},\boldsymbol{u}\in\mathbb{R}^N, c\in\mathbb{R}} & \|\boldsymbol{u}\|_0 \\ \text{subject to} & \|\boldsymbol{y} - \boldsymbol{K}\boldsymbol{a} - c\boldsymbol{1} - \boldsymbol{u}\|_2^2 + \|\boldsymbol{\Psi}\boldsymbol{a}\|_2^2 + \lambda c^2 \leq \varepsilon, \end{array} \tag{6.9}$$

where $\boldsymbol{\Psi}$ is a more general regularization matrix (Tikhonov matrix), [86]. For example, as the accuracy of the kernel based methods usually drops near the border of the input domain, it is reasonable to increase the regularization effect at these points. This can be easily achieved by employing a diagonal matrix with positive elements on the diagonal and increasing the regularization factors that correspond to the points near the border. This is demonstrated in the experimental Section 6.4.

6.3 Theoretical Analysis

Our main focus in this section is to study the theoretical properties of the proposed algorithmic scheme. In particular: a) we prove that the inversion matrix is non-singular and thus the solution to the Least Squares task at each step is unique and b) we provide the necessary condition which guarantees that the proposed method identifies first the correct locations of all the outliers, for the case where only outliers exist in the noise. The reason that, the analysis is carried out for the case where inlier noise is not present, is due to the fact that the analysis gets highly involved. The absence of the inlier noise makes the analysis easier and it highlights some theoretical aspects on why the method works. It must be emphasized that, such a theoretical analysis is carried out for the first time and it is absent in the previously published works. However, as it is demonstrated in the experiments, the method succeeds in identifying outlier locations in many more cases even when the theoretical result does not hold. This leads to the conclusion that the provided condition can be loosen up significantly in the future. Note that this is in line with most bounds and conditions that have been derived in the context of sparse models. In practice, even in the more extreme cases, where inlier and outlier noise coexist, the method manages to identify the majority of the outliers.

6.3.1 Non-singularity of the Inversion Matrix

In the current section, we are interested in the properties of the proposed scheme, i.e., KGARD. It will be shown that, the matrix inverted at each step is non-singular thus the solution at each step is unique. To this end, it is first required to establish expressions equivalent to the task in (6.6) for the regularized Least Squares task that is solved at each iteration's k-th step.

First, it can be readily seen that the minimization of J_k in (6.6) for all $k = 0, 1, 2, \dots$ is equivalent to the following set of *normal equations*[3]:

$$\boldsymbol{M}_{(k)}\boldsymbol{z} = \boldsymbol{A}_{(k)}^T\boldsymbol{y}, \tag{6.10}$$

for $\lambda > 0$. Next, follows a lemma that guarantees the inversion of the matrix in (6.10) and hence the existence of a unique solution for the task in (6.6).

Lemma 6.1. *The matrix $\boldsymbol{M}_{(k)}$ in* (6.8) *is (strictly) positive definite for every $\lambda > 0$, hence invertible.*

Proof. Consider a non-zero vector $\boldsymbol{z} \in \mathbb{R}^{2N+1}$, so that $\boldsymbol{z} = \left(\boldsymbol{\alpha}^T, \beta, \boldsymbol{\gamma}^T\right)^T$ is decomposed, such that $\boldsymbol{\alpha} \in \mathbb{R}^N$, $\beta \in \mathbb{R}$ and $\boldsymbol{\gamma} \in \mathbb{R}^k$. Then, it is easy to show that

$$\boldsymbol{z}^T\boldsymbol{M}_{(k)}\boldsymbol{z} = \left\|\boldsymbol{K\alpha} + \beta\boldsymbol{1} + \boldsymbol{I}_{\mathcal{S}_k}\boldsymbol{\gamma}\right\|_2^2 + \lambda\left\|\boldsymbol{\alpha}\right\|_2^2 + \lambda\beta^2 > 0,$$

which implies that $\boldsymbol{M}_{(k)}$ is a (strictly) positive definite matrix. □

Moreover, the task in (6.6) is equivalent to:

$$\min_{\boldsymbol{z}} \mathrm{J}_{(k)}(\boldsymbol{z}) = \min_{\boldsymbol{z}} \left\| \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{0} \end{pmatrix} - \boldsymbol{D}_{(k)}\boldsymbol{z} \right\|_2^2, \tag{6.11}$$

where $\boldsymbol{D}_{(k)} = \begin{bmatrix} \boldsymbol{A}_{(k)} \\ \sqrt{\lambda}\boldsymbol{B}_{(k)} \end{bmatrix}$. It is well established that, problem (6.11) has a unique solution if and only if the null spaces of $\boldsymbol{A}_{(k)}$ and $\boldsymbol{B}_{(k)}$ intersect only trivially, i.e., $\mathcal{N}(\boldsymbol{A}_{(k)}) \cap \mathcal{N}(\boldsymbol{B}_{(k)}) = \{0\}$ [89, 90, 92]. Since $\boldsymbol{M}_{(k)}$ is (strictly) positive definite, the columns of $\boldsymbol{D}_{(k)}$ are linearly independent and the minimizer $\hat{\boldsymbol{z}}_{(k)} \in \mathbb{R}^{N+1+k}$ of (6.11) (or equivalently (6.6)) is unique, [91].

[3] Notice that matrix $\boldsymbol{B}$ (hence every $\boldsymbol{B}_{(k)}$) is a projection matrix.

6.3.2 Robust Enhancement via the Regularization Term

This section is devoted to the presentation of an important property that enhances robustness for the proposed method, i.e., KGARD. Recall the analysis provided in Chapter 2, where we discussed about the problems associated with the identification of the outliers via the residual. In the following analysis, it will be shown that the regularization term reduces the values of the hat matrix in the diagonal. In other words, the effect of leverage points is significantly reduced. Although the proof is carried out for the initial step, it could also be extended to every subsequent step.

At the initialization of KGARD, $\boldsymbol{A}_{(0)} = [\boldsymbol{K}\ \boldsymbol{1}]$ and $\boldsymbol{B}_{(0)} = \boldsymbol{I}_{N+1}$. The following analysis (which is also adopted for the proof of the outlier identification) is based on the *Singular Value Decomposition* (SVD) (see Appendix C) for matrix $\boldsymbol{A}_{(0)} = [\boldsymbol{K}\ \boldsymbol{1}]$, i.e., $\boldsymbol{A}_{(0)} = \boldsymbol{QSV}^T$. The matrices $\boldsymbol{Q}, \boldsymbol{V}$ are orthogonal, while $\boldsymbol{S}$ is the matrix of dimension $N \times (N+1)$ of the form $\boldsymbol{S} = \begin{bmatrix}\boldsymbol{\Sigma} & \boldsymbol{0}\end{bmatrix}$. The matrix $\boldsymbol{A}_{(0)}^T\boldsymbol{A}_{(0)}$ is positive semi-definite, thus all of its eigenvalues are non-negative. Hence, $\boldsymbol{\Sigma}$ is the diagonal matrix with entries the singular values of matrix $\boldsymbol{A}_{(0)}$, i.e., $\sigma_i \geq 0,\ i = 1, ..., N$.

If no regularization was performed, according to the ordinary LS task, the hat matrix would equal $\boldsymbol{H} = \boldsymbol{QQ}^T$, see (2.8). However, since at every step a regularized LS task is solved (instead of an ordinary LS one)

$$\boldsymbol{A}_{(0)}^T\boldsymbol{A}_{(0)} + \lambda\boldsymbol{I}_{N+1} = \boldsymbol{V} \underbrace{\begin{bmatrix}\boldsymbol{\Sigma}^2 + \lambda\boldsymbol{I}_N & \boldsymbol{0} \\ \boldsymbol{0}^T & \lambda\end{bmatrix}}_{\boldsymbol{\Lambda}} \boldsymbol{V}^T = \boldsymbol{V\Lambda V}^T, \tag{6.12}$$

and the new hat matrix is expressed as

$$\widetilde{\boldsymbol{H}} = \boldsymbol{A}_{(0)}(\boldsymbol{A}_{(0)}^T\boldsymbol{A}_{(0)} + \lambda\boldsymbol{I}_{N+1})^{-1}\boldsymbol{A}_{(0)}^T = \boldsymbol{QGQ}^T, \tag{6.13}$$

where

$$\boldsymbol{G} = \boldsymbol{\Sigma}(\boldsymbol{\Sigma}^2 + \lambda\boldsymbol{I}_N)^{-1}\boldsymbol{\Sigma} \text{ is a diagonal matrix with entries } g_{ii} = \frac{\sigma_i^2}{\sigma_i^2 + \lambda},\ i = 1, 2, ..., n, \tag{6.14}$$

where $\lambda > 0$ is the regularization parameter and σ_i the i-th singular value of the matrix $\boldsymbol{A}_{(0)}$. Thus, this leads to a residual expression similar to (2.12), simply by replacing matrix $\boldsymbol{H}$, by $\widetilde{\boldsymbol{H}}$. Hence, from (6.13), it is a matter of simple manipulations to establish for the diagonal elements of the new hat matrix that they satisfy

$$\tilde{h}_{ii} = \frac{\sigma_i^2}{\sigma_i^2 + \lambda} h_{ii}. \tag{6.15}$$

In simple words, the regularization performed down-weights the diagonal elements of the hat matrix. Equation (6.15) is of great importance, since it guarantees that $\tilde{h}_{ii} < h_{ii}$ for any $\lambda > 0$. Furthermore, it is readily seen that as $\lambda \to 0$ the detection of outlier via the residual is forbidden (recall that here $\overline{h} \simeq 1$), while as $\lambda \to \infty$ then $\tilde{h}_{ii} \to 0$ and thus occurrences of leverage points tend to disappear. In simple words, the regularization performed on the specific task guards the method against occurrences of leverage points. Of course, this fact alone does not guarantee that one could safely detect an outlier via the residual. This is due to the following two reasons:

a) the outliers values could be too small (engaging with the inlier noise) or b) the fraction of outliers contaminating the data could be enormously large; in such cases the summation term in (2.12) could easily be the dominant one. Based on the previous discussion, we adopt the assumptions that the outliers are relatively few (the vector $\underline{\boldsymbol{u}}$ is sparse) and also that the outlier values are (relatively) large. From a practical point of view, the latter assumption is natural, since we want to detect values that greatly deviate from healthy measurements. The first assumption is, also, in line with the use of the greedy approach. It is well established by now that greedy techniques work well for relatively small sparsity levels. These assumptions are also verified by the obtained experimental results. Finally, it should be stressed here that a condition similar to (2.9) does no longer hold (nor does the corresponding discussion), since for the new hat matrix, $\widetilde{\boldsymbol{H}}$, the idempotent property is no longer satisfied.

6.3.3 Identification of the Outliers for the Noiseless Case

The following theorem establishes a bound on the largest singular value of matrix $\boldsymbol{A}_{(0)}$, which guarantees that the method first identifies the correct locations of all the outliers, for the case where only outliers exist in the noise. However, since the ϵ parameter controls the number of iterations for which the method identifies an outlier, it is not guaranteed that it will stop once all the outliers are identified, unless the correct value is somehow given. Thus, it is possible that a few other locations that correspond to healthy measurements are classified as outliers.

Theorem 6.1. *Let $\boldsymbol{K}$ be a full rank, square, real valued matrix. Suppose, that*

$$\boldsymbol{y} = [\boldsymbol{K}\ \boldsymbol{1}]\underbrace{\begin{pmatrix}\underline{\boldsymbol{\alpha}}\\ \underline{c}\end{pmatrix}}_{\underline{\boldsymbol{\theta}}} + \underline{\boldsymbol{u}},$$

where $\underline{\boldsymbol{u}}$ is a sparse (outlier) vector. KGARD is guaranteed to identify first the correct locations of all the outliers[4]*, if the maximum singular value of matrix $\boldsymbol{A}_{(0)} := [\boldsymbol{K}\ \boldsymbol{1}]$, satisfies:*

$$\sigma_{max}(\boldsymbol{A}_{(0)}) < \gamma\sqrt{\lambda}, \tag{6.16}$$

where

$$\gamma = \sqrt{\frac{\min|\underline{u}| - \sqrt{2\lambda}||\underline{\boldsymbol{\theta}}||_2}{2||\underline{\boldsymbol{u}}||_2 - \min|\underline{u}| + \sqrt{2\lambda}||\underline{\boldsymbol{\theta}}||_2}}, \tag{6.17}$$

$\min|\underline{u}|$ *is the smallest absolute value of the sparse vector over the non-zero coordinates and $\lambda > 0$ is a sufficiently large*[5] *regularization parameter for KGARD.*

Proof. The proof is based on the SVD decomposition (see Appendix C) for matrix $\boldsymbol{A}_{(0)}$. For simplification, the notation σ_M will be used to denote its maximum singular value.

The proposed method attempts to solve at each step the regularized Least Squares (LS) task in (6.6), which is equivalent to (6.11). Thus, the regularized LS solution at each k-th step

[4] However, the theorem does not guarantee that only the locations of the outliers will be identified. If the value of ϵ is too small, then KGARD will select locations that do not correspond to true outlier indices.

[5] Since the regularization parameter is defined by the user, such a value can be achieved.

is expressed as:

$$\hat{\boldsymbol{z}}_{(k)} = (\boldsymbol{D}_{(k)}^T \boldsymbol{D}_{(k)})^{-1} \boldsymbol{D}_{(k)}^T \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{0} \end{pmatrix} = (\boldsymbol{A}_{(k)}^T \boldsymbol{A}_{(k)} + \lambda \boldsymbol{B}_{(k)})^{-1} \boldsymbol{A}_{(k)}^T \boldsymbol{y}, \tag{6.18}$$

and the respective residual of the lower dimensional space as

$$\boldsymbol{r}_{(k)} = \boldsymbol{y} - \boldsymbol{A}_{(k)} \hat{\boldsymbol{z}}_{(k)} = \boldsymbol{y} - \boldsymbol{A}_{(k)} (\boldsymbol{A}_{(k)}^T \boldsymbol{A}_{(k)} + \lambda \boldsymbol{B}_{(k)})^{-1} \boldsymbol{A}_{(k)}^T \boldsymbol{y}. \tag{6.19}$$

Step $k = 0$:
Initially, $\widetilde{\mathcal{S}}_0 = \{1, \dots, N+1\}$ (no index has been selected for the outlier estimate), thus $\boldsymbol{A}_{(0)} = [\boldsymbol{K} \ \boldsymbol{1}]$ and $\boldsymbol{B}_{(0)} = \boldsymbol{I}_{N+1}$. Hence, the expression for the initial regularized LS solution $\hat{\boldsymbol{z}}_0$ is obtained from equation (6.18) for $k = 0$. Combining (6.19) for $k = 0$ with (6.12) leads to

$$\boldsymbol{r}_{(0)} = \boldsymbol{y} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T \boldsymbol{y}, \tag{6.20}$$

Furthermore, since $\boldsymbol{y} = \boldsymbol{A}_{(0)} \underline{\boldsymbol{\theta}} + \underline{\boldsymbol{u}}$, substituting in (6.20) leads to

$$\boldsymbol{r}_{(0)} = \underline{\boldsymbol{u}} + \boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T \underline{\boldsymbol{\theta}} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T \underline{\boldsymbol{u}}, \tag{6.21}$$

where $\boldsymbol{F} = \boldsymbol{S} - \boldsymbol{G}\boldsymbol{S} = [\underbrace{\boldsymbol{\Sigma} - \boldsymbol{G}\boldsymbol{\Sigma}}_{\breve{\boldsymbol{F}}} \ \boldsymbol{0}]$ and $\boldsymbol{G}$ is defined in (6.14). Matrix $\breve{\boldsymbol{F}}$ is also diagonal with values

$$\breve{f}_{ii} = \frac{\lambda \sigma_i}{\sigma_i^2 + \lambda}, \ i = 1, 2, ..., N.$$

At this point, it is required to explore some of the unique properties of matrices $\boldsymbol{G}$ and $\boldsymbol{F}$. Recall that the (matrix) 2-norm of a diagonal matrix is equal to the maximum absolute value of the diagonal entries. Hence, it is clear that

$$||\boldsymbol{G}||_2 = \sigma_M^2 / (\sigma_M^2 + \lambda) \text{ and } ||\boldsymbol{F}||_2 = ||\breve{\boldsymbol{F}}||_2 \leq \sqrt{\lambda}/2, \tag{6.22}$$

since $\mathrm{g}(\sigma) = \frac{\sigma^2}{\sigma^2 + \lambda}$ is a strictly increasing function of $\sigma \geq 0$ and $\breve{\mathrm{f}}(\sigma) = \frac{\lambda \sigma}{\sigma^2 + \lambda}$ receives a unique maximum, which determines the upper bound for the matrix 2-norm.

Finally, it should be noted that if no outliers exist in the noise, the algorithm terminates due to the fact that the norm of the initial residual is less than (or equal to) ϵ. However, this scenario is rather insignificant, since no robust modeling is required. Thus, if our goal is for the method to be able to handle various types of noise that includes outliers (e.g. Gaussian noise plus impulses), we assume that $\|\boldsymbol{r}_{(0)}\|_2 > \epsilon$. In such a case, KGARD will select an outlier index from the set $\mathcal{S}_0^c = \{1, 2, ..., N\}$.

At the first selection step, as well as at all subsequent steps, we should impose a condition so that the method identifies and selects an index that belongs to the support of the sparse outlier vector. To this end, let $\mathcal{S}$ denote the non-empty support set for the sparse outlier vector $\underline{\boldsymbol{u}}$. In order for KGARD to select an atom $\boldsymbol{e}_i$, from columns of the matrix $\boldsymbol{I}_N$ that belongs to $\mathcal{S}$, we should impose

$$|r_{(0),i}| > |r_{(0),j}|, \text{ for all } i \in \mathcal{S} \text{ and } j \in \mathcal{S}^c. \tag{6.23}$$

The key is to establish appropriate bounds, which guarantee the selection of a correct index that belongs to $\mathcal{S}$. Therefore, we first need to develop bounds on the following inner products.

Using (6.22), the Cauchy-Schwarz inequality and the fact that $\boldsymbol{Q}, \boldsymbol{V}$ are orthonormal, it is easy to verify that

$$\begin{aligned} |\langle \boldsymbol{e}_l, \boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\rangle| &= \left|(\boldsymbol{Q}^T\boldsymbol{e}_l)^T\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\right| \leq \left\|\boldsymbol{Q}^T\boldsymbol{e}_l\right\|_2 \left\|\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\right\|_2 \leq \\ &\leq \|\boldsymbol{F}\|_2 \left\|\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\right\|_2 \leq \frac{\sqrt{\lambda}}{2}\|\underline{\boldsymbol{\theta}}\|_2 \end{aligned} \tag{6.24}$$

as well as

$$\begin{aligned} |\langle \boldsymbol{e}_l, \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}\rangle| &= \left|(\boldsymbol{Q}^T\boldsymbol{e}_l)^T\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}\right| \leq \left\|\boldsymbol{Q}^T\boldsymbol{e}_l\right\|_2 \left\|\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}\right\|_2 \leq \\ &\leq \|\boldsymbol{G}\|_2 \left\|\boldsymbol{Q}^T\underline{\boldsymbol{u}}\right\|_2 = \frac{\sigma_M^2}{\sigma_M^2+\lambda}\|\underline{\boldsymbol{u}}\|_2, \end{aligned} \tag{6.25}$$

for all $l = 1, 2, ..., N$. Thus, we have that

$$\begin{aligned} |r_{(0),i}| &= |\langle \boldsymbol{r}_{(0)}, \boldsymbol{e}_i\rangle| = |\langle \underline{\boldsymbol{u}} + \boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}, \boldsymbol{e}_i\rangle| \geq \\ &\geq |\underline{u}_i| - |\langle \boldsymbol{e}_i, \boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\rangle| - |\langle \boldsymbol{e}_i, \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}\rangle| \geq \\ &\geq \min|\underline{u}| - \frac{\sqrt{\lambda}}{2}\|\underline{\boldsymbol{\theta}}\|_2 - \frac{\sigma_M^2}{\sigma_M^2+\lambda}\|\underline{\boldsymbol{u}}\|_2 > \\ &> \min|\underline{u}| - \frac{\sqrt{2\lambda}}{2}\|\underline{\boldsymbol{\theta}}\|_2 - \frac{\sigma_M^2}{\sigma_M^2+\lambda}\|\underline{\boldsymbol{u}}\|_2, \end{aligned} \tag{6.26}$$

for any $i \in \mathcal{S}$ and

$$\begin{aligned} |r_{(0),j}| &= |\langle \boldsymbol{r}_{(0)}, \boldsymbol{e}_j\rangle| = |\langle \boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}, \boldsymbol{e}_j\rangle| \leq \\ &\leq |\langle \boldsymbol{e}_j, \boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\rangle| + |\langle \boldsymbol{e}_j, \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}\rangle| \leq \\ &\leq \frac{\sqrt{\lambda}}{2}\|\underline{\boldsymbol{\theta}}\|_2 + \frac{\sigma_M^2}{\sigma_M^2+\lambda}\|\underline{\boldsymbol{u}}\|_2 < \\ &< \frac{\sqrt{2\lambda}}{2}\|\underline{\boldsymbol{\theta}}\|_2 + \frac{\sigma_M^2}{\sigma_M^2+\lambda}\|\underline{\boldsymbol{u}}\|_2, \end{aligned} \tag{6.27}$$

for all $j \in \mathcal{S}^c$, where equation (6.21) and inequalities (6.24) and (6.25) have also been used. Hence, imposing (6.23) leads to (6.16). It should be noted that, a reason that could lead to the violation of (6.16) is for the term $\min|\underline{u}| - \sqrt{2\lambda}\|\underline{\boldsymbol{\theta}}\|_2$ to be non-positive. Thus, since the regularization parameter is fine tuned by the user we should select a value for λ, such that $\lambda < \left(\min|\underline{u}|/\|\underline{\boldsymbol{\theta}}\|_2\right)^2/2$. If the condition is guaranteed, then at the first selection step a column indexed as $j_1 \in \mathcal{S}$ is selected. The set of active columns that participate in the LS solution of the current step is then $\mathcal{S}_1 = \{j_1\} \subseteq \mathcal{S}$ and thus $\boldsymbol{A}_{(1)} = \begin{bmatrix}\boldsymbol{A}_{(0)} & \boldsymbol{e}_{j_1}\end{bmatrix}$ and $\boldsymbol{B}_{(1)} = \begin{bmatrix}\boldsymbol{I}_{N+1} & \boldsymbol{0}\\ \boldsymbol{0}^T & 0\end{bmatrix}$. After the selection of the first column the inversion of the following matrix is performed:

$$\boldsymbol{D}_{(1)}^T\boldsymbol{D}_{(1)} = \boldsymbol{A}_{(1)}^T\boldsymbol{A}_{(1)} + \lambda\boldsymbol{B}_{(1)} = \begin{bmatrix}\boldsymbol{A}_{(0)}^T\boldsymbol{A}_{(0)} + \lambda\boldsymbol{I}_{N+1} & \boldsymbol{A}_{(0)}^T\boldsymbol{e}_{j_1}\\ \boldsymbol{e}_{j_1}^T\boldsymbol{A}_{(0)} & 1\end{bmatrix}.$$

Applying the *Matrix Inversion Lemma* combined with (6.12) leads to

$$(\boldsymbol{D}_{(1)}^T\boldsymbol{D}_{(1)})^{-1} = \begin{bmatrix}\boldsymbol{V}\boldsymbol{\Lambda}^{-1}\boldsymbol{V}^T + \frac{1}{\beta}\boldsymbol{V}\boldsymbol{\Gamma}^T\boldsymbol{Q}^T\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{\Gamma}\boldsymbol{V}^T & -\frac{1}{\beta}\boldsymbol{V}\boldsymbol{\Gamma}^T\boldsymbol{Q}^T\boldsymbol{e}_{j_1}\\ -\frac{1}{\beta}\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{\Gamma}\boldsymbol{V}^T & \frac{1}{\beta}\end{bmatrix},$$

where $\boldsymbol{\Gamma} = [\boldsymbol{\Sigma}(\boldsymbol{\Sigma}^2 + \lambda \boldsymbol{I}_N)^{-1} \; \boldsymbol{0}]$ and $\beta = 1 - \boldsymbol{e}_{j_1}^T \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T \boldsymbol{e}_{j_1} = 1 - \left\|\boldsymbol{G}^{1/2}\boldsymbol{Q}^T \boldsymbol{e}_{j_1}\right\|_2^2$. The regularized LS solution is obtained from (6.18) for $k = 1$ and after substitution into (6.19) leads to the new residual vector:

$$\boldsymbol{r}_{(1)} = \boldsymbol{y} - \boldsymbol{A}_{(1)}\hat{\boldsymbol{z}}_{(1)} = \boldsymbol{P}_{(1)}\underline{\boldsymbol{u}} + \boldsymbol{P}_{(1)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}} - \boldsymbol{P}_{(1)}\boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}, \tag{6.28}$$

where $\boldsymbol{P}_{(1)} = \boldsymbol{I}_N + \frac{1}{\beta}\boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T - \frac{1}{\beta}\boldsymbol{e}_{j_1}\boldsymbol{e}_{j_1}^T$.

The process of the augmentation of the active set (by the selection of an atom/column), continues, until the norm of the residual vector drops below the user-defined threshold. Thus, in order for KGARD to select an index from the set $\mathcal{S}$ and as long as the ϵ parameter is tuned sufficiently small we should impose

$$|r_{(1),i}| > |r_{(1),j}|, \text{ for all } i \in \mathcal{S} \setminus \mathcal{S}_1 \text{ and } j \in \mathcal{S}^c.$$

In order to simplify (6.28) we need to decompose the sparse vector $\underline{\boldsymbol{u}}$ into two parts. Based on (4.27) and with the use of simple linear algebra, we obtain $\boldsymbol{P}_{(1)}(\underline{\boldsymbol{u}} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}) = \tilde{\boldsymbol{u}}_{(1)} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\tilde{\boldsymbol{u}}_{(1)}$, where

$$\tilde{\boldsymbol{u}}_{(1)} = \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}}) + \frac{1}{\beta}\left(\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}})\right) \cdot \boldsymbol{e}_{j_1}. \tag{6.29}$$

Hence, the final form of the residual at step $k = 1$ is:

$$\boldsymbol{r}_{(1)} = \tilde{\boldsymbol{u}}_{(1)} + \boldsymbol{P}_{(1)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\tilde{\boldsymbol{u}}_{(1)}. \tag{6.30}$$

Notice here that, $\mathrm{supp}(\underline{\boldsymbol{u}}) = \mathrm{supp}(\tilde{\boldsymbol{u}}_{(1)}) = \mathcal{S}$ and that the first and third terms of the residual in (6.30) are independent of matrix $\boldsymbol{P}_{(1)}$. Next, we focus on taking bounds on every term of the right-hand part in (6.30). Recall that the matrix $\boldsymbol{P}_{(1)}$ was not present in (6.21). The fact that it could not be excluded from the second term of the residual in (6.30) adds some extra difficulty to the task. However, we could overcome such an obstacle simply by noticing that there is no need to establish bounds on the matrix $\boldsymbol{P}_{(1)}$ after using the sub-multiplicative norm property on the product $\boldsymbol{P}_{(1)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T$. Instead, we are only interested in establishing a bound on the norm of the vector $\boldsymbol{P}_{(1)}^T\boldsymbol{e}_l$ for every $l \neq j_1$. Therefore, the l-th row of the matrix $\boldsymbol{P}_{(1)}$, i.e.,

$$\boldsymbol{P}_{(1)}^T\boldsymbol{e}_l = \boldsymbol{e}_l + \omega \cdot \boldsymbol{e}_{j_1},$$

is a 2-sparse vector with $\omega = \frac{1}{\beta}\left(\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\boldsymbol{e}_l\right)$. Moreover, it is readily seen that,

$$|\omega| \leq \frac{1}{\beta}\left\|\boldsymbol{G}\right\|_2 \leq \frac{\sigma_M^2}{\lambda} < 1, \tag{6.31}$$

since $1/\beta \leq (\sigma_M^2 + \lambda)/\lambda$ and $\sigma_M < \sqrt{\lambda}$ as observed from (6.16) and (6.17) ($\gamma < 1$). Thus, we have that

$$\left\|\boldsymbol{P}_{(1)}^T\boldsymbol{e}_l\right\|_2 = \sqrt{1 + |\omega|^2} < \sqrt{2}. \tag{6.32}$$

Exploiting the latter bound we have that

$$\begin{aligned} |\langle \boldsymbol{e}_l, \boldsymbol{P}_{(1)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\rangle| &= \left|(\boldsymbol{Q}^T\boldsymbol{P}_{(1)}^T\boldsymbol{e}_l)^T\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\right| \leq \left\|\boldsymbol{P}_{(1)}^T\boldsymbol{e}_l\right\|_2 \left\|\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\right\|_2 < \\ &< \sqrt{2}\left\|\boldsymbol{F}\right\|_2 \left\|\underline{\boldsymbol{\theta}}\right\|_2 \leq \frac{\sqrt{2\lambda}}{2}\left\|\underline{\boldsymbol{\theta}}\right\|_2. \end{aligned} \tag{6.33}$$

Moreover,

$$\left|\frac{1}{\beta}\boldsymbol{e}_{j_1}^T\boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}})\right| \le \frac{\sigma_M^2}{\lambda}\|\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}})\|_2 < \frac{\min|\underline{u}|}{\|\underline{\boldsymbol{u}}\|_2}\|\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_1}(\underline{\boldsymbol{u}})\|_2 < \min|\underline{u}| \le \left|\underline{u}_{j_1}\right|,$$

which according to (6.29) leads to

$$\left\|\tilde{\boldsymbol{u}}_{(1)}\right\|_2 < \|\underline{\boldsymbol{u}}\|_2 . \tag{6.34}$$

Similarly, we have that

$$\begin{aligned}
|r_{(1),i}| &= |\langle \boldsymbol{r}_{(1)}, \boldsymbol{e}_i\rangle| = |\langle \tilde{\boldsymbol{u}}_{(1)} + \boldsymbol{P}_{(1)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\tilde{\boldsymbol{u}}_{(1)}, \boldsymbol{e}_i\rangle| \ge \\
&\ge |\underline{u}_i| - |\langle \boldsymbol{e}_i, \boldsymbol{P}_{(1)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\rangle| - |\langle \boldsymbol{e}_i, \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\tilde{\boldsymbol{u}}_{(1)}\rangle| > \\
&> \min|\underline{u}| - \frac{\sqrt{2\lambda}}{2}\|\underline{\boldsymbol{\theta}}\|_2 - \frac{\sigma_M^2}{\sigma_M^2+\lambda}\left\|\tilde{\boldsymbol{u}}_{(1)}\right\|_2 > \\
&> \min|\underline{u}| - \frac{\sqrt{2\lambda}}{2}\|\underline{\boldsymbol{\theta}}\|_2 - \frac{\sigma_M^2}{\sigma_M^2+\lambda}\|\underline{\boldsymbol{u}}\|_2,
\end{aligned} \tag{6.35}$$

for any $i \in \mathcal{S}\setminus\mathcal{S}_1$ and

$$\begin{aligned}
|r_{(1),j}| &= |\langle \boldsymbol{r}_{(1)}, \boldsymbol{e}_j\rangle| = |\langle \boldsymbol{P}_{(1)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\tilde{\boldsymbol{u}}_{(1)}, \boldsymbol{e}_j\rangle| \le \\
&\le |\langle \boldsymbol{e}_j, \boldsymbol{P}_{(1)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\rangle| + |\langle \boldsymbol{e}_j, \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\tilde{\boldsymbol{u}}_{(1)}\rangle| < \\
&< \frac{\sqrt{2\lambda}}{2}\|\underline{\boldsymbol{\theta}}\|_2 + \frac{\sigma_M^2}{\sigma_M^2+\lambda}\left\|\tilde{\boldsymbol{u}}_{(1)}\right\|_2 < \\
&< \frac{\sqrt{2\lambda}}{2}\|\underline{\boldsymbol{\theta}}\|_2 + \frac{\sigma_M^2}{\sigma_M^2+\lambda}\|\underline{\boldsymbol{u}}\|_2,
\end{aligned} \tag{6.36}$$

for all $j \in \mathcal{S}^c$, where (6.25), (6.30), (6.33) and (6.34) are used. Thus, once more, by imposing that the lower bound in (6.35) has to be greater than the upper bound in (6.36) we are led to (6.16). Hence, it is guaranteed that at the current step the column indexed $j_2 \in \mathcal{S}$ is selected and thus $\mathcal{S}_2 = \{j_1, j_2\} \subseteq \mathcal{S}$. Now that we have demonstrated how the method works for the first simple step, we present the general selection step analysis for KGARD.

General k-th step:
At the k-th step, $\mathcal{S}_k = \{j_1, j_2, ..., j_k\} \subseteq \mathcal{S}$ and thus

$$\boldsymbol{A}_{(k)} = \begin{bmatrix}\boldsymbol{A}_{(0)} & \boldsymbol{I}_{\mathcal{S}_k}\end{bmatrix} \text{ and } \boldsymbol{B}_{(k)} = \begin{bmatrix}\boldsymbol{I}_{N+1} & \boldsymbol{O}_{(N+1)\times k} \\ \boldsymbol{O}_{(N+1)\times k}^T & \boldsymbol{O}_k\end{bmatrix}.$$

The LS step requires the inversion of the matrix

$$\boldsymbol{D}_{(k)}^T\boldsymbol{D}_{(k)} = \boldsymbol{A}_{(k)}^T\boldsymbol{A}_{(k)} + \lambda\boldsymbol{B}_{(k)} = \begin{bmatrix}\boldsymbol{A}_{(0)}^T\boldsymbol{A}_{(0)} + \lambda\boldsymbol{I}_{N+1} & \boldsymbol{A}_{(0)}^T\boldsymbol{I}_{\mathcal{S}_k} \\ \boldsymbol{I}_{\mathcal{S}_k}^T\boldsymbol{A}_{(0)} & \boldsymbol{I}_k\end{bmatrix}.$$

Applying the *Matrix Inversion Lemma* combined with (6.12) leads to

$$(\boldsymbol{D}_{(k)}^T\boldsymbol{D}_{(k)})^{-1} = \begin{bmatrix}\boldsymbol{V}\boldsymbol{\Lambda}^{-1}\boldsymbol{V}^T + \boldsymbol{V}\boldsymbol{\Gamma}^T\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}_k}\widetilde{\boldsymbol{W}}_{(\mathcal{S}_k)}^{-1}\boldsymbol{I}_{\mathcal{S}_k}^T\boldsymbol{Q}\boldsymbol{\Gamma}\boldsymbol{V}^T & -\boldsymbol{V}\boldsymbol{\Gamma}^T\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}_k}\widetilde{\boldsymbol{W}}_{(\mathcal{S}_k)}^{-1} \\ -\widetilde{\boldsymbol{W}}_{(\mathcal{S}_k)}^{-1}\boldsymbol{I}_{\mathcal{S}_k}^T\boldsymbol{Q}\boldsymbol{\Gamma}\boldsymbol{V}^T & \widetilde{\boldsymbol{W}}_{(\mathcal{S}_k)}^{-1}\end{bmatrix},$$

where $\widetilde{\boldsymbol{W}}_{(\mathcal{S}_k)} = \boldsymbol{I}_k - \boldsymbol{I}_{\mathcal{S}_k}^T \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T \boldsymbol{I}_{\mathcal{S}_k}$. In turn, substitution into (6.19) leads to:

$$\boldsymbol{r}_{(k)} = \boldsymbol{P}_{(k)}\underline{\boldsymbol{u}} + \boldsymbol{P}_{(k)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}} - \boldsymbol{P}_{(k)}\boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}, \tag{6.37}$$

where $\boldsymbol{P}_{(k)} = \boldsymbol{I}_N + \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\boldsymbol{I}_{\mathcal{S}_k}\widetilde{\boldsymbol{W}}_{(\mathcal{S}_k)}^{-1}\boldsymbol{I}_{\mathcal{S}_k}^T - \boldsymbol{I}_{\mathcal{S}_k}\widetilde{\boldsymbol{W}}_{(\mathcal{S}_k)}^{-1}\boldsymbol{I}_{\mathcal{S}_k}^T$. If we would like for the method to select an index from the set $\mathcal{S}$, we should impose

$$|r_{(k),i}| > |r_{(k),j}|, \text{ for all } i \in \mathcal{S} \setminus \mathcal{S}_k \text{ and } j \in \mathcal{S}^c.$$

Now $\boldsymbol{P}_{(k)}(\underline{\boldsymbol{u}} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\underline{\boldsymbol{u}}) = \tilde{\boldsymbol{u}}_{(k)} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\tilde{\boldsymbol{u}}_{(k)}$, where

$$\tilde{\boldsymbol{u}}_{(k)} = \boldsymbol{I}_{\mathcal{S}_k}\widetilde{\boldsymbol{W}}_{(\mathcal{S}_k)}^{-1}\boldsymbol{I}_{\mathcal{S}_k}^T\boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_k}(\underline{\boldsymbol{u}}) + \mathrm{F}_{\mathcal{S}\setminus\mathcal{S}_k}(\underline{\boldsymbol{u}}). \tag{6.38}$$

Hence, the final form for the residual is:

$$\boldsymbol{r}_{(k)} = \tilde{\boldsymbol{u}}_{(k)} + \boldsymbol{P}_{(k)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}} - \boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\tilde{\boldsymbol{u}}_{(k)}. \tag{6.39}$$

Following a similar path for $l \notin \mathcal{S}_k$ we conclude that

$$\boldsymbol{P}_{(k)}^T\boldsymbol{e}_l = \boldsymbol{e}_l + \boldsymbol{I}_{\mathcal{S}_k}\widetilde{\boldsymbol{W}}_{(\mathcal{S}_k)}^{-1}\boldsymbol{I}_{\mathcal{S}_k}^T\boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\boldsymbol{e}_l,$$

is a $(k+1)$-sparse vector. Furthermore, it is readily seen that

$$\left\|\widetilde{\boldsymbol{W}}_{(\mathcal{S}_k)}^{-1}\boldsymbol{I}_{\mathcal{S}_k}^T\boldsymbol{Q}\boldsymbol{G}\boldsymbol{Q}^T\boldsymbol{e}_l\right\|_2 \leq \frac{\sigma_M^2}{\lambda} < 1, \tag{6.40}$$

which leads to $\left\|\boldsymbol{P}_{(k)}^T\boldsymbol{e}_l\right\|_2 < \sqrt{2}$. Moreover,

$$|\langle\boldsymbol{e}_l, \boldsymbol{P}_{(k)}\boldsymbol{Q}\boldsymbol{F}\boldsymbol{V}^T\underline{\boldsymbol{\theta}}\rangle| < \frac{\sqrt{2\lambda}}{2}\left\|\underline{\boldsymbol{\theta}}\right\|_2 \text{ and } \|\tilde{\boldsymbol{u}}_{(k)}\|_2 < \|\underline{\boldsymbol{u}}\|_2. \tag{6.41}$$

The bounds for the residual are now expressed as

$$|r_{(k),i}| = |\langle\boldsymbol{r}_{(k)}, \boldsymbol{e}_i\rangle| > \min|\underline{u}| - \frac{\sqrt{2\lambda}}{2}\left\|\underline{\boldsymbol{\theta}}\right\|_2 - \frac{\sigma_M^2}{\sigma_M^2 + \lambda}\left\|\underline{\boldsymbol{u}}\right\|_2, \tag{6.42}$$

for any $i \in \mathcal{S} \setminus \mathcal{S}_k$, and

$$|r_{(k),j}| = |\langle\boldsymbol{r}_{(k)}, \boldsymbol{e}_j\rangle| < \frac{\sqrt{2\lambda}}{2}\left\|\underline{\boldsymbol{\theta}}\right\|_2 + \frac{\sigma_M^2}{\sigma_M^2 + \lambda}\left\|\underline{\boldsymbol{u}}\right\|_2, \tag{6.43}$$

for all $j \in \mathcal{S}^c$, where (6.39) and (6.41) are used. Finally, by imposing the lower bound of (6.42) to be greater than the upper bound of (6.43) leads to the inequality (6.16). At the k-th step, it is proved that unless the residual length is below the predefined threshold the algorithm will select another correct atom from the identity matrix and the procedure is repeated until $\mathcal{S}_k = \mathcal{S}$. At this point, if the ϵ parameter is carefully tuned the residual of KGARD drops below the user-defined threshold and the procedure terminates. If not, then extra indices that correspond to healthy observations are classified as outliers and $\mathcal{S} \subset \mathcal{S}_k$. □

Careful tuning of the ϵ parameter seems to play an important role regarding the performance of KGARD. This is the user-defined parameter that controls the number of iterations for the method (thus the convergence speed) and also the sparsity for the outlier estimate vector. Assuming that the ϵ value is set to a relatively small value, the algorithm will first select the correct locations of the outliers and then continue until all columns of $\boldsymbol{I}_N$ are selected (in such case $k = N$ is the maximum number of iterations for KGARD). Consequently, we can easily see that the norm of the residual vector will eventually drop below $\epsilon > 0$ (and if all columns are selected $\boldsymbol{r}_{(k)} = \mathbf{0}$). Simply stated, the procedure will continue and model other samples (not originating from a noisy source) as outliers filling up the outlier estimate vector $\hat{\boldsymbol{u}}$, which will no longer be sparse. On the contrary, if ϵ is set to relatively large values, the algorithm will stop within a few only iterations, which leads to the identification of only a few of the true outliers in the dataset. Hence, sensible tuning of ϵ should be applied. Finally, it should be stated that the algorithm is not very sensitive to the choice of ϵ, i.e., small changes in its value do not affect the sparsity level of the outlier estimate.

In principal, the analysis for the case where inbound noise is present can be carried out in a way similar to the the one provided for the noiseless case. However, this turns out to be excessively algebraically complex.

6.4 Experiments

For the entire section of experiments, the Gaussian kernel is employed and all results are averaged over 1000 Monte-Carlo runs (independent simulations). At each experiment, the parameters are optimized (via cross-validation) and the respective parameter values are given (for each method) so that results are reproducible. The specific (MATLAB) code can be found in `http://bouboulis.mysch.gr/kernels.html`.

6.4.1 Testing the Recovery of the Sparse Outlier vector's Support

In the current section, our main concern is to test on the validity of the condition (6.16), in practice. To this end, we have performed the following test for the case where only outliers exist in the noise.

We consider $N = 100$ equidistant points over the interval $[0, 1]$ and generate the output data via $\underline{\mathrm{f}}(x_i) = \sum_{j=1}^{N} \underline{\alpha}_j \kappa(x_i, x_j)$, where κ is the Gaussian kernel with $\sigma = 0.1$ and the vector of coefficients $\underline{\boldsymbol{\alpha}} = [\underline{\alpha}_1, \dots, \underline{\alpha}_N]$ is a sparse vector with the number of non-zero coordinates ranging between 2 and 23 and their values drawn from $\mathcal{N}(0, 0.5^2)$. Since no inlier noise exists, our corrupted data is generated via (5.24) for $\eta_i = 0$ and outlier values $\pm\underline{u}$. Moreover, since the condition (6.16) is valid for fixed values of the parameters involved, we have measured the capability of KGARD to recover the support of the sparse outlier vector, i.e., $\mathcal{S} = \mathrm{supp}(\underline{\boldsymbol{u}})$, while varying the values of the outliers. In Figure 6.2, KGARD's capability to identify the exact sparse outlier vector support is demonstrated, for a fraction of outliers at 10%. On the vertical axis we have measured the percentage of correct and wrong indices recovered, while varying the value $\underline{u}$ of the outliers. In parallel, the bar chart demonstrates the validity of the introduced condition (6.16). It is clear that, if the condition holds, KGARD identifies the correct support of the sparse outlier vector successfully. However, even if the condition is rarely satisfied, e.g.,

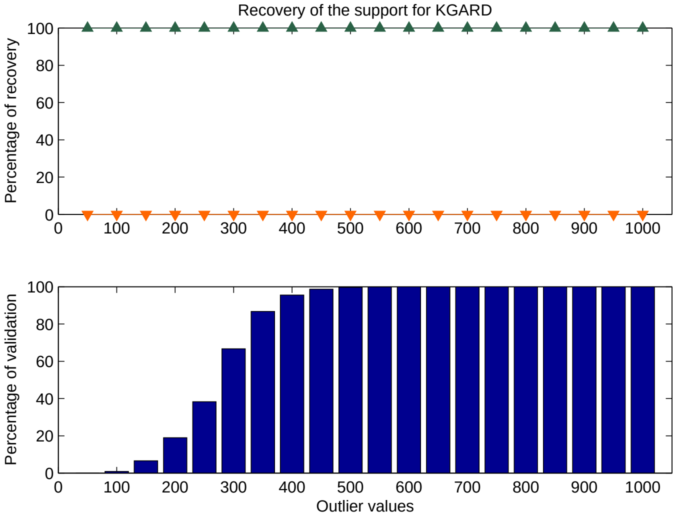


Figure 6.2: Percentage of the correct (green pointing up) and wrong (orange pointing down) indices that KGARD has selected, while varying the values $\pm \underline{u}$ of the outliers at the fixed fraction of 10%. Although the condition (6.16) is valid only for values greater than ±600 (and with high probability valid for values 400-599), the support of the sparse outlier vector has been correctly identified for much smaller values of outlier noise, too.

for $\underline{u} = 100$, the method still manages to identify the correct support. This fact leads to the conclusion that the condition imposed by (6.16) is rather strict.

Finally, in Table 6.1, the previous experiment has been performed for various fractions of outliers. In the second and third column, we have listed the percentage of correct and wrong indices selected by KGARD, for all values of outliers ranging from 50 to 1000. Moreover, in the final column, the minimum value of outliers, which renders the condition valid, is shown. For example, in the second row and for fraction of outliers at 10% the condition is valid only for values greater than 600 (last column in Table 6.1). However, the method manages to correctly identify the support (one-to-one index for columns two and three), not only for values $\underline{u}$ greater than 600, but for all outlier values, ranging from the minimum value of 50 to the maximum value of 1000. It should also be noted that, experiments have been performed with the use of various nonlinear functions and results were similar to the ones presented here.

Table 6.1: Percentage of correct and wrong indices that KGARD has selected, for outlier values $\underline{u}$ ranging from 50 to 1000. The correct support (second column) corresponds to true outliers (indices in $\mathcal{S}$), while the wrong support (third column) corresponds to locations which are wrongly classified as outliers (thus do not belong to $\mathcal{S}$). In the final column the minimum value $\underline{u}$ of outliers for which the support recovery condition is valid, is listed.

Outlier fraction	Correct support	Wrong support	Outlier value u
5 %	100 %	0 %	450
10 %	100 %	0 %	600
15 %	100 %	0 %	650
20 %	100 %	0 %	700
25 %	100 %	0 %	750
30 %	100 %	0 %	950

Table 6.2: Computed MSE for $\underline{f}(x) = 20sinc(2\pi x)$ over the training and validation set. Additionally, the percentage of correct and wrong indices that each method has classified as outliers and the Mean Implementation Time (MIT), for various levels of inlier and outlier noise, are evaluated.

Algorithm	MSE_{tr}	MSE_{val}	Cor. ind.	Wr. ind.	MIT (sec)	Inlier - Outlier
RB-RVM	0.0850	0.0851	-	-	0.298	20 dB - 5%
RAM ($\lambda = 0.07, \mu = 2.5$)	0.0344	0.0345	100 %	0.2 %	0.005	20 dB - 5%
KGARD ($\lambda = 0.2, \varepsilon = 10$)	**0.0285**	**0.0285**	100 %	0 %	0.004	20 dB - 5%
RB-RVM	0.0911	0.0912	-	-	0.298	20 dB - 10%
RAM ($\lambda = 0.07, \mu = 2.5$)	0.0371	0.0372	100 %	0.1 %	0.007	20 dB - 10%
KGARD ($\lambda = 0.2, \varepsilon = 10$)	**0.0305**	**0.0305**	100 %	0 %	0.008	20 dB - 10%
RB-RVM	0.0992	0.0994	-	-	0.299	20 dB - 15%
RAM ($\lambda = 0.07, \mu = 2$)	0.0393	0.0393	100 %	0.6 %	0.008	20 dB - 15%
KGARD ($\lambda = 0.3, \varepsilon = 10$)	**0.0330**	**0.0330**	100 %	0 %	0.012	20 dB - 15%
RB-RVM	0.1189	0.1184	-	-	0.305	20 dB - 20%
RAM ($\lambda = 0.07, \mu = 2$)	**0.0421**	**0.0422**	100 %	0.4 %	0.010	20 dB - 20%
KGARD ($\lambda = 1, \varepsilon = 10$)	0.0626	0.0626	100 %	0 %	0.017	20 dB - 20%
RB-RVM	0.3630	0.3631	-	-	0.327	15 dB - 5%
RAM ($\lambda = 0.15, \mu = 5$)	0.1035	0.1036	100%	0.7 %	0.005	15 dB - 5%
KGARD ($\lambda = 0.3, \varepsilon = 15$)	**0.0862**	**0.0862**	100 %	0.1 %	0.005	15 dB - 5%
RB-RVM	0.3828	0.3830	-	-	0.319	15 dB - 10%
RAM ($\lambda = 0.15, \mu = 5$)	0.1117	0.1118	100%	0.4 %	0.006	15 dB - 10%
KGARD ($\lambda = 0.3, \varepsilon = 15$)	**0.0925**	**0.0925**	100 %	0 %	0.008	15 dB - 10%
RB-RVM	0.4165	0.4166	-	-	0.317	15 dB - 15%
RAM ($\lambda = 0.15, \mu = 5$)	0.1186	0.1186	100%	0.3 %	0.007	15 dB - 15%
KGARD ($\lambda = 0.3, \varepsilon = 15$)	**0.1001**	**0.1003**	100 %	0 %	0.012	15 dB - 15%
RB-RVM	0.4793	0.4798	-	-	0.312	15 dB - 20%
RAM ($\lambda = 0.15, \mu = 4$)	**0.1281**	**0.1282**	100%	1.4 %	0.008	15 dB - 20%
KGARD ($\lambda = 0.7, \varepsilon = 15$)	0.1340	0.1349	100 %	0 %	0.016	15 dB - 20%

6.4.2 Evaluation of the Method: Mean-Square-Error (MSE)

In the current section, the previously established methods that deal with the robust nonlinear estimation with kernels, i.e., the Bayesian approach, RB-RVM, and the weighted ℓ_1-norm approximation method, RAM, are compared against KGARD in terms of the mean-square-error (MSE). Additionally, the evaluation includes the listing of the percentage of indices that each method has identified as outliers, for all methods, except for the Bayesian approach. Moreover, the *Mean Implementation Time* (MIT) is measured for each experiment. Finally, as already discussed in Section 6.2.2, we have increased the regularization value λ of KGARD near the borders for the first two experiments, as a means to increase the performance. In particular, at the first and last five points (borders), the regularizer is automatically multiplied by the factor of 5, with respect to the predefined value λ, which is set on the interior points. It should also be noted that, the observation data are generated via equation (5.24) for the rest of the experiments in the current section. The experiments and their results are analyzed next.

For the first experiment, we have selected the *sinc* function, which is well known in the machine learning community for its properties. We consider 398 equidistant points over the

Table 6.3: Performance evaluation for each method, where $\underline{f} \in \mathcal{H}$ is considered as a linear combination of a few kernels with the input data lying on the 1-dimensional space. The inlier noise is considered random Gaussian with $\sigma = 4$ and for various fractions of outliers we have computed: the training and validation MSE, the percentage of correct and wrong indices that each method has classified as outliers and the Mean Implementation Time (MIT).

Algorithm	MSE_{tr}	MSE_{val}	Cor. ind.	Wr. ind.	MIT (sec)	Outliers
RB-RVM	3.3405	3.3436	-	-	0.309	5%
RAM ($\lambda = 0.15, \mu = 33$)	1.2459	1.2473	100%	0 %	0.005	5%
KGARD ($\lambda = 0.3, \varepsilon = 57$)	**1.1567**	**1.1580**	99.8 %	1.2 %	0.004	5%
RB-RVM	3.6111	3.6176	-	-	0.308	10%
RAM ($\lambda = 0.15, \mu = 31$)	1.3085	1.3100	100%	0.1 %	0.005	10%
KGARD ($\lambda = 0.3, \varepsilon = 55$)	**1.2110**	**1.2120**	99.9 %	0.9 %	0.008	10%
RB-RVM	3.7902	3.7950	-	-	0.308	15%
RAM ($\lambda = 0.15, \mu = 28$)	1.3945	1.3972	100%	0.2 %	0.006	15%
KGARD ($\lambda = 0.3, \varepsilon = 53$)	**1.2922**	**1.2942**	100 %	0.8 %	0.012	15%
RB-RVM	4.0685	4.0705	-	-	0.307	20%
RAM ($\lambda = 0.15, \mu = 24$)	**1.5110**	**1.5109**	100%	0.8 %	0.007	20%
KGARD ($\lambda = 0.3, \varepsilon = 52$)	1.5173	1.5262	99.9 %	0.4 %	0.016	20%

interval $[-0.99, 1)$ for the input values and generate the uncorrupted output values via $\underline{f}(x_i) = 20sinc(2\pi x_i)$. Next, the set of points is split into two subsets, the training and the validation subset. The training subset, with points denoted by (y_i, x_i), consists of the $N = 199$ odd indexed points (first, third, e.t.c.), while the validation subset comprises the remaining points, which are denoted as (y_i', x_i'). The original data of the training set, is then contaminated by noise, as (5.24) suggests. The inlier part is considered to be random Gaussian noise of appropriate variance (measured in dB), while the outlier part consists of various fractions of outliers with constant values ±15, distributed uniformly over the support set. Finally, the kernel parameter σ has been set equal to $\sigma = 0.15$.

Table 6.2 depicts each method's performance, where the best results are marked in **bold**. In terms of the computed MSE, it is clear that KGARD attains the lower MSE for both the training and the validation error for all fractions of outliers, except for the fraction of 20%. This fact is also in line with what is known concerning the performance of the sparse greedy methods in practice; that is, their performance boosts as the sparsity level of the approximation is low. On the other hand, the RAM solver seems more suitable for larger fractions of outliers. Moreover, the computational cost is comparable for both methods (RAM and KGARD) and for small fractions of outliers. Regarding the identification of the sparse outlier vector's support, although both methods correctly identify the correct indices belonging to the true support, i.e., $\mathcal{S}$, RAM incorrectly classifies more indices compared to KGARD.

For the second experiment, the performance for each method is evaluated for the following set-up. The input data consists of 400 equidistant points over the interval $[0, 1)$. The uncorrupted observations are generated via $\underline{f}(x_i) = \sum_{j=1}^{400} \underline{\alpha}_j \kappa(x_j, x_i)$ by the Gaussian kernel with parameter $\sigma = 0.1$ and a sparse coefficient vector, $\boldsymbol{\underline{\alpha}}$, with non-zeros ranging between 4% and 18% and their values randomly drawn from the Gaussian distribution $\mathcal{N}(0, 20^2)$. In the sequel, the set of points is split into two subsets, the training and the validation subset. Similar to the

Table 6.4: Performance evaluation for each method, where $\underline{f} \in \mathcal{H}$ is considered as a linear combination of a few kernels with the input data lying on the 2-dimensional space. The inlier noise is considered white Gaussian with $\sigma = 3$ and for various fractions of outliers we have computed: the training and validation MSE,the percentage of correct and wrong indices that each method has classified as outliers and the Mean Implementation Time (MIT).

Algorithm	MSE_{tr}	MSE_{val}	Cor. ind.	Wr. ind.	MIT (sec)	Outliers
RB-RVM	3.9825	3.6918	-	-	0.416	5%
RAM ($\lambda = 0.2, \mu = 22$)	2.0534	1.8592	100%	0.1 %	0.010	5%
KGARD ($\lambda = 0.15, \varepsilon = 46$)	**1.7381**	**1.5644**	100 %	0.3 %	0.009	5%
RB-RVM	4.2382	3.8977	-	-	0.419	10%
RAM ($\lambda = 0.2, \mu = 18$)	2.2281	1.9926	100%	0.9 %	0.013	10%
KGARD ($\lambda = 0.15, \varepsilon = 44$)	**1.8854**	**1.6750**	100 %	0.5 %	0.016	10%
RB-RVM	4.5749	4.2181	-	-	0.418	15%
RAM ($\lambda = 0.2, \mu = 17$)	2.5944	2.2846	100%	1.6 %	0.016	15%
KGARD ($\lambda = 0.2, \varepsilon = 42$)	**2.1968**	**1.9375**	99.9 %	0.9 %	0.024	15%
RB-RVM	5.7051	5.0540	-	-	0.418	20%
RAM ($\lambda = 0.2, \mu = 16$)	3.0593	2.6703	99.9%	2.3 %	0.020	20%
KGARD ($\lambda = 0.4, \varepsilon = 42$)	**3.0293**	**2.6113**	99.9 %	1 %	0.033	20%

first experiment, the training subset consists of the $N = 200$ odd indexed points (first, third, e.t.c.), while the remaining (even indices) correspond to the validation/test subset. The uncorrupted observations of the training set is contaminated by white Gaussian noise with variance equal to 16. Finally, various fractions of outliers have been used (distributed uniformly over the training points) with values 40 or -40.

In Table 6.3, the performance for each method is depicted. Once again, KGARD attains the lowest MSE for all fractions of outliers up to 15%. It is readily seen that this holds, despite the fact that the support of the sparse outlier vector is not fully recovered (due to the existence of heavy inlier noise). Also notice that, for the case where 20% of outlier values are present, the MSE for RAM is lower than KGARD's, for both the training and the validation set. This comes at no surprise, since it is in line with what is the more general experience, in practice, concerning the comparative performance of ℓ_1-norm based optimization and the greedy algorithms.

For the final pilot experiment, KGARD's performance is tested for the case where the input data lies on a 2-dimensional subspace. To this end, we consider 31 points in $[0, 1]$ and split these points to form the training set, which comprises 16 odd indices and the rest 15 forming the validation set. Next, the 31^2 points are distributed over a squared lattice in plane $[0, 1] \times [0, 1]$, where each uncorrupted measurement is generated by $\underline{f}(\boldsymbol{x}_i) = \sum_{j=1}^{31^2} \underset{\sim}{\alpha}_j \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$, $(\sigma = 0.2)$ and a sparse coefficient vector $\underset{\sim}{\boldsymbol{\alpha}} = [\underset{\sim}{\alpha}_1, \ldots, \underset{\sim}{\alpha}_{31^2}]$ with non-zero values ranging between 4% and 17.5% and their values randomly drawn from $\mathcal{N}(0, 25.6^2)$. Thus, the training subset consists of $N = 16^2$ points, while the remaining 15^2 correspond to the validation/test subset. Next, the original observations of the training set are corrupted by inlier noise originating from $\mathcal{N}(0, 3^2)$ and outlier values ±40. The results are presented in Table 6.4 for various fractions of outliers, with the best values of the MSE marked in **bold**. It is evident that for the 2-dimensional nonlinear denoising task, KGARD's performance outperforms its competitors (in terms of MSE), for all fractions of the outliers.

Finally, it should also be noted that, although the RB-RVM method does not perform at the highest level, it utilizes the advantage that no tuning of parameters is required; however, this comes at substantially increased computational cost. On the contrary, the pair of tuning parameters for RAM, renders the method extremely difficult to be fully optimized (in terms of MSE), in practice. However, taking into account the physical interpretation of ϵ and λ associated with KGARD in the noise denoising task, we have developed a method for automatic user-free choice, as it is demonstrated in the next chapter.

Chapter 7

Applications to Image Denoising

7.1 Introduction

In this chapter, in order to test and verify the performance of the proposed algorithmic scheme, in practice, we use the KGARD framework to address one of the most popular problems in the field of image processing: the task of removing noise from a digital image.

First, we demonstrate how the proposed KGARD algorithmic scheme can be used to treat the image denoising task in cases where the noise model includes outliers. This is accomplished by diving the noisy image into smaller regions of interest (ROIs) and applying the robust scheme separately. However, since it requires the tuning of the algorithm's two parameters at each ROI, automatic selection of both λ and ε is established.

Next, two different denoising methods that deal with outlier noise are presented. The first one is directly based on KGARD algorithmic scheme. The second method splits the denoising procedure into two parts: the identification and removal of the impulses, which is first carried out via the KGARD, and finally the removal of the remaining component from the intermediate output via a cutting edge wavelet-based denoising method.

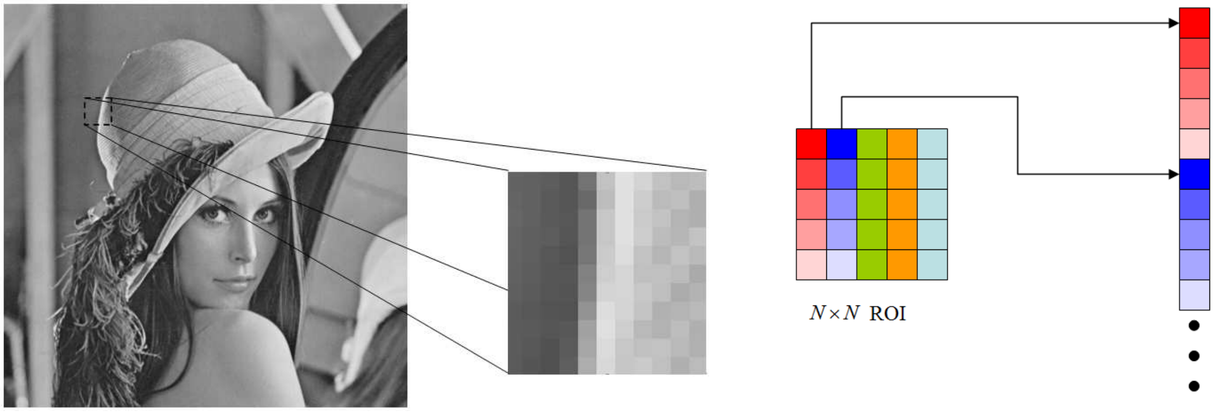
7.2 Modeling the Image and the Noise

The source of noise in a digital image can either be errors of the imaging system itself (e.g., hardware or software errors, transmission errors, quantization errors), errors that occur due to limitations of the imaging system (e.g., small size of the sensor) or errors that are generated by the environment (e.g., low light, heat, e.t.c.).

Typically, the noisy image is modeled as follows:

$$\mathrm{g}(x, x') = \underline{\mathrm{g}}(x, x') + \boldsymbol{\nu}(x, x'),$$

where $x, x' \in [0, 1]$ correspond to the normalized pixel coordinates, $\underline{\mathrm{g}}$ is the original noise free image and $\boldsymbol{\nu}$ the additive noise. Given the noisy image, g, the objective of any image denoising method is to obtain an estimate $\hat{\mathrm{g}}$ of the original image $\underline{\mathrm{g}}$. In most cases, we assume that



(a) (b)

Figure 7.1: (a) A square $N \times N$ region of intest (ROI). (b) Rearranging the pixels of a ROI.

the image noise is Gaussian additive, independent at each pixel, and independent of the signal intensity and/or that it contains spikes or impulses (i.e., salt and pepper noise). However, there are cases where the noise model follows other, than Gaussian, probability density functions (e.g., the Poisson distribution or the uniform distribution).

7.2.1 Dividing the Image into ROIs

In the proposed denoising method, we adopt the well known and popular strategy of dividing the noisy image into smaller $N \times N$ square regions of interest (ROIs), as illustrated in Figure 7.1. Then, we rearrange the pixels so that it forms a row vector. Instead of applying the denoising process to the entire image, we process each ROI individually in a sequential order. This is done for two reasons: (a) first, the time needed to solve the involved optimization tasks increases polynomially with N^2 and (b) working in each ROI separately enables us to change the parameters of the model in an adaptive manner in order to account for the different level of details in each ROI. Note that, the rearrangement shown in Figure 7.1 implies that the pixel (i, j) (i.e., i-th row, j-th column) is placed at the n-th position of the respective vector, where $n = (i - 1) \cdot N + j$.

7.2.2 Robust Modeling

In kernel ridge regression denoising methods, one assumes that each ROI represents the points on the surface of a continuous function, $\underline{\mathrm{g}}$, of two variables defined on $[0, 1] \times [0, 1]$. The pixel values of the clean and the noisy digitized ROIs are represented as $\underline{\zeta}_{ij} = \underline{\mathrm{g}}(x_i, x'_j)$ and ζ_{ij} respectively (both taking values in the interval $[0, 255]$), where $x_i = (i - 1)/(N - 1)$, $x'_j = (j - 1)/(N - 1)$, for $i, j = 1, 2, ..., N$. Moreover, as the original image $\underline{\mathrm{g}}$ is a relatively smooth function (with the exception on the edges) we assume that it lies in an RKHS induced by the Gaussian kernel, i.e., $\underline{\mathrm{g}} \in \mathcal{H}$, for some $\sigma > 0$. Specifically, in order to be consistent with the Representer Theorem, we will assume that $\underline{\mathrm{g}}$ takes the form of a finite linear representation of kernel functions, i.e.,

$$\underline{\mathrm{g}} = \sum_{i,j=1}^{N} \underline{\alpha}_{ij} \kappa(\cdot, (x_i, x'_j)). \tag{7.1}$$

After pixel rearrangement, equation (7.1) can be cast as:

$$\underline{\mathrm{g}} = \sum_{n=1}^{N^2} \underline{\alpha}_n \kappa(\cdot, \boldsymbol{x}_n),$$

where $n = (i-1) \cdot N + j$ and $\boldsymbol{x}_n = (x_i, x'_j)$. Hence, the intensity of the n-th pixel is given by

$$\underline{\zeta}_n = \underline{\mathrm{g}}(\boldsymbol{x}_n) = \sum_{m=1}^{N^2} \underline{\alpha}_m \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m). \tag{7.2}$$

The model considered in this paper assumes that the intensity of the pixels of the noisy ROI can be decomposed as follows:

$$\zeta_{ij} = \underline{\zeta}_{ij} + \underline{u}_{ij} + \eta_{ij},$$

for $i, j = 1, 2, ..., N$, where η_{ij} denotes the inlier noise component and $\underline{u}_{ij}$ a possible outlier at that pixel. In vector notation (after rearrangement) we can write:

$$\boldsymbol{\zeta} = \underline{\boldsymbol{\zeta}} + \underline{\boldsymbol{u}} + \boldsymbol{\eta}, \tag{7.3}$$

where $\boldsymbol{\zeta}, \underline{\boldsymbol{\zeta}}, \underline{\boldsymbol{u}}, \boldsymbol{\eta}, \in \mathbb{R}^{N^2}$ and $\underline{\boldsymbol{u}}$ is a sparse vector. Moreover, as the elements of $\underline{\boldsymbol{\zeta}}$ take the form (7.2), we can write $\underline{\boldsymbol{\zeta}} = \boldsymbol{K} \cdot \underline{\boldsymbol{\alpha}}$, where $\kappa_{nm} = \kappa(\boldsymbol{x}_n, \boldsymbol{x}_m)$. In this context, we can model the denoising task as the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{a}, \boldsymbol{u} \in \mathbb{R}^{N^2}, c \in \mathbb{R}} \quad & \|\boldsymbol{u}\|_0 \\ \text{subject to} \quad & \|\boldsymbol{\zeta} - \boldsymbol{K}\boldsymbol{a} - c\boldsymbol{1} - \boldsymbol{u}\|_2^2 + \lambda\|\boldsymbol{a}\|_2^2 + \lambda c^2 \leq \varepsilon, \end{aligned} \tag{7.4}$$

for some predefined $\lambda, \varepsilon > 0$. In a nutshell, problem (7.4) solves for the sparsest outlier's vector $\boldsymbol{u}$ and the respective $\boldsymbol{a}$ (i.e., the coefficients of the kernel expansion) that keep the error low; at the same time the regularization parameter λ controls the smoothness of the solution. The larger the λ is, the smoother the solution, i.e., $\hat{\underline{\boldsymbol{\zeta}}} = \boldsymbol{K}\hat{\boldsymbol{\alpha}}$, tends to become.

7.2.3 Implementation

The main mechanism of both algorithms, that are presented in this section, is simple. The image is divided into $N \times N$ ROIs and the KGARD algorithm is applied sequentially in each individual ROI. However, as the reconstruction accuracy of KRR methods drops near the borders of the respective domain, we have chosen to discard their values. This means that although KGARD is applied to the $N \times N$ ROI only the $L \times L$ values are used in the final reconstruction (those that are around the center of the ROI). In the sequel, we will name the $L \times L$ centered region as the "reduced ROI" or rROI for sort. Alternatively, one may consider that the image is actually

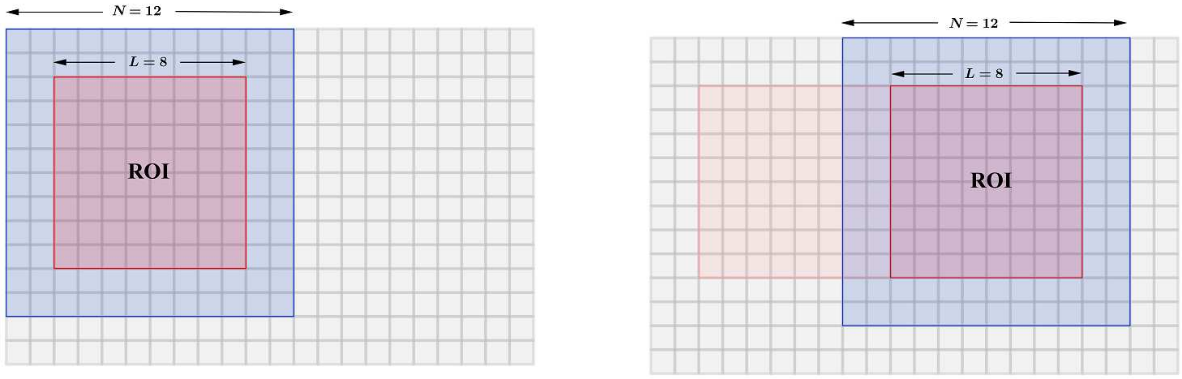
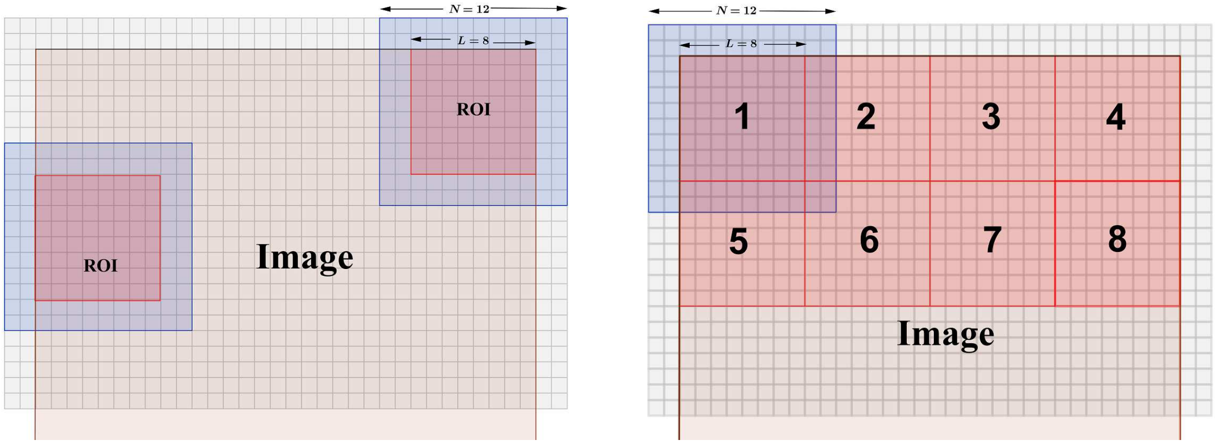


Figure 7.2: Two consecutive $N \times N$ ROIs. Observe that the two ROIs overlap.



(a)

(b)

Figure 7.3: (a) The algorithm has reached the right end of the image, hence it moves L pixels below. (b) In this example, $L = 8$, $N = 12$. The image has been padded using 2 pixels in all dimensions. The Figure shows the 8 first rROIs.

divided into $L \times L$ non-overlapping regions (the rROIs) and those regions are extended to the size $N \times N$. This means that the ROIs contain overlapping parts. We will also assume that the dimensions of the image are multipliers of L (if they are not we can add dummy pixels to the end) and select N so that $N - L$ is an even number.

After the reconstruction of a specific rROI, the algorithm moves to the next one, i.e., it moves L pixels to the right (see Figure 7.2), or, if the algorithm has reached the right end of the image it moves at the beginning of the line, which is placed L pixels below (see Figure 7.3(a)). Observe that, for this procedure to be valid the image has to be padded by adding $(N - L)/2$ pixels along all dimensions. In this paper, we chose to pad the image by repeating border elements[1]. For example, if we select $L = 8$ and $N = 12$, to apply this procedure on an image with dimensions[2] 32×32, we will end up with a total of 16 overlapping ROIs, 4 per line

[1] This can be done with the 'replicate' option of MatLab's function padarray.

[2] Observe that L divides 32.

Algorithm 10 Selection of the regularization parameter λ

```
1: Assume a user defined value $\lambda_0$.
2: Compute the magnitude of the gradient $\boldsymbol{T}$ given in (7.5), at each pixel.
3: Compute the mean gradient of each ROI, i.e., the mean value of the gradient's magnitude of all pixels that belong to the ROI.
4: Compute the mean value, $m$, and the standard deviation, $s$, of the aforementioned mean gradients.
5: ROIs with mean gradient larger than $m + s$ are assumed to be areas with fine details and the algorithm sets $\lambda = \lambda_0$.
6: All ROIs with mean gradient lower than $m - s/10$ are assumed to be smooth areas and the algorithm sets $\lambda = 15\lambda_0$.
7: For all other ROIs the algorithm sets $\lambda = 5\lambda_0$.
```

(see Figure 7.3(b)).

Another important aspect of the denoising algorithm is the automated selection of the parameters λ and ϵ that are involved with KGARD. This an important feature, as these parameters largely control both the quality of the estimation and the correct identification of the outliers. Thus, careful tuning at each specific ROI is required. Naturally, it would have been intractable to require a user pre-defined pair of values (i.e., λ, ϵ) for each specific ROI. Hence, we devised simple methods to adjust these values in each ROI depending on its features.

Automatic selection of the regularization parameter λ

This parameter controls the smoothing operation of the denoising process. The user enters a specific λ value, e.g. λ_0, so that it controls the strength of the smoothening. Then the algorithm adjusts this value at each ROI separately, so that λ is small at ROIs that contain a lot of "edges" and large at ROIs that contain smoother areas. The specific values given in Algorithm 10 are the result of extensive experimentation.

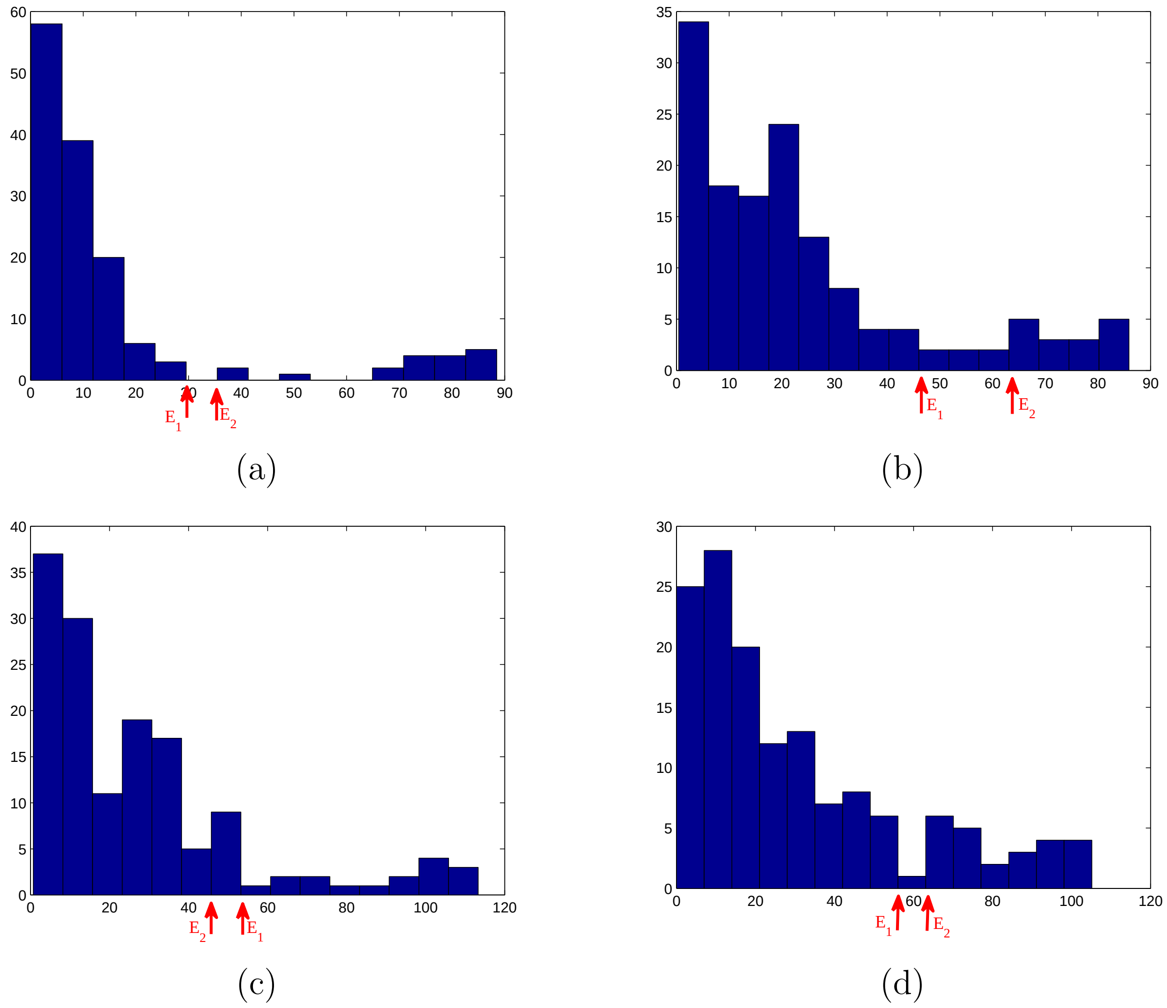
In order to extract information from images, we have used an *image gradient*; this is a directional change in the intensity or color of an image. In our case, the magnitude of the gradient is computed via the Sobel (Sobel-Feldman) operator, which is a discrete differentiation operator that computes an approximation of the gradient of the image intensity function, [93]. The operator uses two 3 × 3 kernels which are convolved with the original image to calculate approximations of the derivatives - one for the horizontal and one for the vertical changes, i.e., $\boldsymbol{T}_x$ and $\boldsymbol{T}_y$, respectively. The gradient magnitude $\boldsymbol{T}$ is then computed with values

$$\tau_{ij} = \sqrt{\tau_{x_{ij}} + \tau_{y_{ij}}}. \tag{7.5}$$

Whether a ROI has edges or not, is determined by the mean magnitude of the gradient at each pixel. The rationale is described in Algorithm 10.

Automatic computation of the termination parameter ϵ

The stopping criterion for KGARD, that has been adopted for the image denoising task, is slightly different than the one employed in Algorithm 9. In this case, instead of requiring the



(a)

(b)

(c)

(d)

Figure 7.4: Histograms of the residual vectors used in the automatic computation of ϵ.

norm of the residual vector to drop below ϵ, i.e., $\|\boldsymbol{r}_{(k)}\|_2 \leq \epsilon$, we require the maximum absolute valued coordinate of $\boldsymbol{r}_{(k)}$ to drop below ϵ $\left(\left\|\boldsymbol{r}_{(k)}\right\|_{\infty} \leq \epsilon\right)$. The estimation of ϵ for each particular ROI is carried out as follows. Initially, a user defined parameter E_0 is selected. At each step, a histogram chart of the values $|r_{(k),i}|$ is generated, using $\lfloor\frac{N^2}{10}\rfloor + 1$ equally spaced bins along the x-axis, between the minimum and maximum values of the coordinates. Let $\boldsymbol{h}$ denote the heights of the bars of the histogram and h_{min} be the minimum height of the histogram bars. Next, two real numbers, i.e., E_1, E_2, are assigned. In particular, the number E_1 represents the left endpoint of the first occurrence of a minimum-height bar (i.e., the first bar with height equal to h_{min}, moving from left to right). The number E_2 represents the left endpoint of the first bar, ℓ, with height h_ℓ (moving from left to right) that satisfies both $h_\ell - h_{\ell-1} \geq 1$ and $h_{\ell-1} \leq h_{min} + 5$, $\ell \geq 2$. This roughly corresponds to the first increasing bar, which in parallel is next to a bar with height close to the minimum height. Figure 7.4 demonstrates some typical examples regarding the computation of these numbers. Both E_1 and E_2 are reasonable choices for the value of ϵ (meaning that the bars to the right of these values may be assumed to represent outliers). Finally, the algorithm determines whether the histogram can be clearly divided into two parts, where one represents the errors due to outliers by using a simple rule:

Algorithm 11 KGARD for image denoising

```
Input: the original noisy image $\boldsymbol{I}$ and the parameters $\lambda_0$, $\sigma$, $E_0$, $N$, $L$.
Build the kernel matrix $\boldsymbol{K}$.
if the dimensions of the original image are not multiplies of $L$ then
    Add initial padding
Form $\hat{\boldsymbol{I}}$ and $\hat{\boldsymbol{O}}$ so that they have the same dimensions as $\boldsymbol{I}$.
Add padding with size $N - L$ around the image.
Divide the image into $N \times N$ ROIs and compute the regularization parameters of each ROI according to Algorithm 10.
for each ROI $\boldsymbol{R}$ do
    Rearrange the pixels of $\boldsymbol{R}$ to form the vector $\boldsymbol{\zeta}$.
    Run the modified KGARD algorithm on the set $\boldsymbol{\zeta}$ with parameter $\lambda$ (obtained from Algorithm 10) and stoping criterion as given in (7.6).
    Let $\hat{\boldsymbol{a}}$, $\hat{\boldsymbol{u}}$ be the estimated solution according to KGARD algorithm.
    Compute the denoised vector $\hat{\boldsymbol{\zeta}} = \boldsymbol{K}\hat{\boldsymbol{\alpha}}$.
    Rearrange the elements of $\hat{\boldsymbol{\zeta}}$ to form the denoised ROI $\hat{\boldsymbol{R}}$.
    Extract the centered $L \times L$ rROI from $\hat{\boldsymbol{R}}$.
    Use the values of the rROI to set the values of the corresponding pixels in $\hat{\boldsymbol{I}}$.
    Rearrange the elements of $\hat{\boldsymbol{u}}$ to form the outliers' ROI.
    Extract the centered $L \times L$ values of the outliers' ROI.
    Use these values to set the values of the corresponding outliers in $\hat{\boldsymbol{O}}$.
    Move to the next ROI.
Remove the initial padding on $\hat{\boldsymbol{I}}$ and $\hat{\boldsymbol{O}}$ (if needed).
Output: the denoised image $\hat{\boldsymbol{I}}$ and the outliers' image $\hat{\boldsymbol{O}}$.
```

if $\frac{\sqrt{\mathrm{var}(\boldsymbol{h}_{(k)})}}{\mathrm{mean}(\boldsymbol{h}_{(k)})} > 0.9$ then we assume that this is trivial (e.g., Figure 7.4(a)-(c)), otherwise it is harder to distinguish these areas (e.g., figure 7.4(d)). Note that, we use the notation $\boldsymbol{h}_{(k)}$ to refer to the heights of the histogram bar at the k step of the algorithm. The final computation of ϵ (at step k) is carried out as follows:

$$\epsilon_{(k)} = \begin{cases} \min\{E_0, E_1, E_2\}, & \text{if } \frac{\sqrt{\mathrm{var}(\boldsymbol{h}_{(k)})}}{\mathrm{mean}(\boldsymbol{h}_{(k)})} > 0.9 \\ \min\{E_0, E_1\}, & \text{otherwise.} \end{cases} \tag{7.6}$$

It should be noted that, the user defined parameter E_0 has little importance in the evaluation of ϵ. One may set it constantly to a value near 40 (as we did in all provided simulations). However, in cases where the image is corrupted by outliers only, a smaller value may be advisable although it does not have a great impact on the reconstruction quality.

Direct KGARD implementation

The first denoising method, which we call "Kernel GARD Denoising" (or KGARD for short), is described in Algorithm 11. The algorithm requires five user defined parameters: (a) the regularization parameter, λ_0, (b) the Gaussian kernel width, σ, (c) the E_0 threshold for the automatic computation of the termination parameter, (d) the size of the ROI, N and (e) the

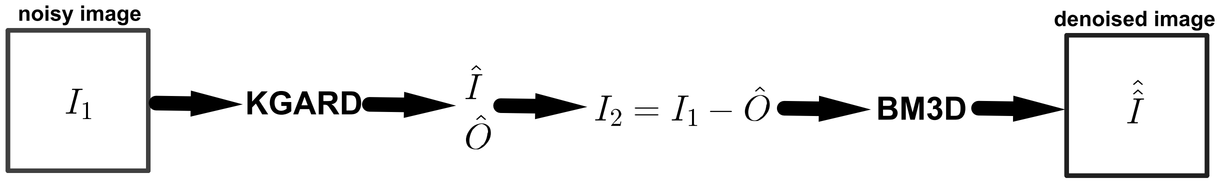


Figure 7.5: The KGARD-BM3D denoising method. First the KGARD applied to the noisy image extracts the outliers and then the BM3D removes the remaining of the noise.

size of the rROIs, that are used in the reconstruction, i.e., L. Nevertheless, these parameters are somehow interrelated. We will discuss these issues in the next sections.

KGARD combined with BM3D (KGARD-BM3D)

The second denoising method is actually a two-step procedure which combines the outliers detection properties of KGARD with the denoising capabilities of a standard off-the-shelf denoising method. The KGARD algorithm is applied onto the noisy image, but this time the produced denoised image $\hat{\boldsymbol{I}}$ is discarded and only the positions and values of the reconstructed outliers are taken into consideration. These are subtracted from the original noisy image and a cutting edge wavelet-based method with the name BM3D (Block Matching and 3-D filtering) is applied to the result, [43]. In this setting, (which is the one we propose) the KGARD is actually used to detect the outliers and remove their contribution (estimate), while the BM3D methods takes over afterwards to clean the remaining of the noise. Figure 7.5 illustrates this procedure. This method requires the same parameters as KGARD, plus the parameter s, which is involved in the BM3D algorithm. The BM3D filter is built upon the assumption that the image is corrupted by Gaussian noise. Hence, the parameter s is the variance of that Gaussian noise and is either known a-priori or it is given as a user-defined estimate. Moreover, it has been demonstrated that BM3D can also efficiently remove other types of noise, if s is adjusted properly, [46].

7.2.4 Parameter Selection

This section is devoted to prove guidelines for the selection of the user defined parameters for the proposed denoising algorithms. Typical values of N range between 8 and 16. In most cases, values near 8 or even lower increase the time required to complete the denoising process with no significant improvements. However, if the image contains a lot of "fine details" this may be advisable. In such case, smaller values for the width of the Gaussian kernel, σ, may also enhance the results since the regression task is more robust to abrupt changes. However, we should note that σ is inversely associated with the size of the ROI, N, thus if one increases N, one should decrease σ proportionally, i.e., keeping the product $N \cdot \sigma$ constant. For example, if $N = 12$ and $\sigma = 0.3$, then the kernel width covers 3.6 pixels. It is straightforward to see that, if N decreases to say 8, then the kernel width that will provide a coverage of 3.6 pixels is $\sigma = 0.45$. We have observed that the values $N = 12$ and $\sigma = 0.3$ (which result to a product equal to $N \cdot \sigma = 3.6$) are adequate to remove moderate noise from a typical image. In cases where the image has many details and edges, N and σ should be adjusted to provide a lower product (e.g.,

$N = 12$ and $\sigma = 0.15$, so that $N \cdot \sigma = 1.8$). For images corrupted by high noise, this product should become larger. Finally, the λ value controls the regularization on the final result. Large values imply a strong smoothing operation, while smaller ones (close to zero) reduce the effect of regularization leading to a better fit, but also may lead to overfitting.

For the experiments presented in this paper, we fixed the size of the ROIs using $N = 12$ and $L = 8$. These are reasonable choices that provide fast[3] results with high reconstruction accuracy. Hence, only the values of σ and λ_0 need to be adjusted according to the density of the details in the image and the amount of noise. We have found that the values of σ that provide adequate results range between 0.1 and 0.4. Similarly, typical values of λ_0 range from 0.1 to 1. Finally, the constant E_0 was set equal to 40 for all cases.

The s parameter of the BM3D method is adjusted according to the amount of noise that is presented at each image. It ranges between very small values (e.g, $s = 10$), when only a small amount of noise is present, to significantly larger values (e.g., $s = 25$ or $s = 50$) if the image is highly corrupted.

7.3 Experiments on Images Corrupted by Synthetic Noise

In this section, we present an extensive set of experiments on grayscale images that have been corrupted by mixed noise, which comprises a Gaussian component and a set of impulses (± 100). The intensity of the Gaussian noise ranges between 15 dB and 25 dB and the percentage of impulses between 5% and 20%. The tests are performed on three very popular images in greyscale: the *Lena*, the *boat* and the *Barbara* images that are included in Waterloo's image repository. The images are 512×512 pixels in size. Each test has been performed 50 times and the respective mean PSNRs are reported, where

$$\text{PSNR} = 10 \log_{10} \left(\frac{Max_I^2}{MSE} \right),$$

where Max_I is the maximum possible pixel value (here 255) of an image $P_1 \times P_2$ in size (here $P_1 = P_2 = 512$) and

$$MSE = \frac{1}{P_1 P_2} \sum_{i=1}^{P_1} \sum_{j=1}^{P_2} [I(i,j) - \hat{I}(i,j)]^2,$$

with I and $\hat{I}$ corresponding to the "clean" image and its noisy approximation, respectively. The parameters have been tuned so that to provide the best result for each method (in terms of MSE).

In Table 7.1, the two proposed methods are applied to the *Lena* image and they are compared with BM3D (the state-of-the-art wavelet-based method) and an image denoising method based on RB-RVM. For the latter, we chose a simple implementation, similar to the one we propose in our methods: the image is divided into ROIs and the RB-RVM algorithm is applied to each ROI sequentially. The parameters were selected to provide the best possible

[3] A typical denoising task using either KGARD or KGARD-BM3D implemented in MATLAB takes less than a minute on a moderate computer.

results in terms of PSNR. The size of the ROIs for the Lena image has been set to $N = 12$ and $L = 8$.

In Tables 7.2 and 7.3, the results of the BM3D and the KGARD-BM3D methods applied to the *boat* and *Barbara* images, respectively, are displayed. The size of the ROIs has been set to $N = 12$ and $L = 8$ for the *boat* image, whereas $N = 12$ and $L = 4$ for the *Barbara* image since it has more finer details (e.g., the stripes of the pants). Moreover, one can observe that for this image we have used a lower value for σ and λ as indicated in Section 7.2.4.

Figures 7.6, 7.7 and 7.8 show the obtained denoised images on a specific experiment (20 dB Gaussian noise and 10% outliers). The experiments show that the proposed method (KGARD-BM3D) enhances significantly the denoising capabilities of BM3D, especially for low and moderate intensities of the Gaussian noise. If the Gaussian component becomes prominent (e.g., at 15 dB) then the two methods provide similar results.

Finally, it should be noted that, we chose not to include RAM or any ℓ_1-based denoising method, as this would require efficient techniques to adaptively control its parameters, i.e., λ, μ at each ROI (similar to the case of KGARD). Such an action remains an open issue. Having to play with both parameters, makes the tuning computationally demanding. This is because the number of iterations for the method to converge to a reasonable solution increases substantially, once the parameters are moved away from their optimal (in terms of MSE) values. For example, if the parameters are not optimally tuned, the denoising process may take more than an hour to complete in MATLAB on a moderate computer.

Table 7.1: Denoising performed on the *Lena* image corrupted by various types and intensities of noise using the proposed methods, the robust RVM (RB-RVM) approach and the state-of-the-art wavelet method BM3D.

Method	Parameters	Gaussian Noise	Impulses (±100)	PSNR
BM3D	$s = 30$	25 dB	5%	32.2 dB
RB-RVM	$\sigma = 0.3$	25 dB	5%	31.78 dB
KGARD	$\sigma = 0.3$, $\lambda = 1$	25 dB	5%	33.91 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 10$	25 dB	5%	**36.12 dB**
BM3D	$s = 30$	25 dB	10%	30.84 dB
RB-RVM	$\sigma = 0.3$	25 dB	10%	31.25 dB
KGARD	$\sigma = 0.3$, $\lambda = 1$	25 dB	10%	33.49 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 10$	25 dB	10%	**35.67 dB**
BM3D	$s = 45$	25 dB	20%	29.28 dB
RB-RVM	$\sigma = 0.4$	25 dB	20%	30.3 dB
KGARD	$\sigma = 0.4$, $\lambda = 1$	25 dB	20%	32.04 dB
KGARD-BM3D	$\sigma = 0.4$, $\lambda = 1$, $s = 15$	25 dB	20%	**33.69 dB**
BM3D	$s = 30$	20 dB	5%	31.83 dB
RB-RVM	$\sigma = 0.4$	20 dB	5%	29.3 dB
KGARD	$\sigma = 0.3$, $\lambda = 1$	20 dB	5%	32.35 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 15$	20 dB	5%	**34.24 dB**
BM3D	$s = 35$	20 dB	10%	30.66 dB
RB-RVM	$\sigma = 0.4$	20 dB	10%	29.09 dB
KGARD	$\sigma = 0.3$, $\lambda = 1$	20 dB	10%	31.94 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 15$	20 dB	10%	**33.81 dB**
BM3D	$s = 50$	20 dB	20%	29.86 dB
RB-RVM	$\sigma = 0.4$	20 dB	20%	28.29 dB
KGARD	$\sigma = 0.4$, $\lambda = 1$	20 dB	20%	30.72 dB
KGARD-BM3D	$\sigma = 0.4$, $\lambda = 1$, $s = 15$	20 dB	20%	**32.06 dB**
BM3D	$s = 35$	15 dB	5%	30.87 dB
RB-RVM	$\sigma = 0.6$	15 dB	5%	26.74 dB
KGARD	$\sigma = 0.3$, $\lambda = 1.5$	15 dB	5%	29.12 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 25$	15 dB	5%	**31.18 dB**
BM3D	$s = 40$	15 dB	10%	29.94 dB
RB-RVM	$\sigma = 0.4$	15 dB	10%	25.85 dB
KGARD	$\sigma = 0.3$, $\lambda = 2$	15 dB	10%	28.47 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 25$	15 dB	10%	**30.77 dB**
BM3D	$s = 40$	15 dB	20%	28.78 dB
RB-RVM	$\sigma = 0.4$	15 dB	20%	25 dB
KGARD	$\sigma = 0.4$, $\lambda = 3$	15 dB	20%	27.87 dB
KGARD-BM3D	$\sigma = 0.4$, $\lambda = 1$, $s = 35$	15 dB	20%	**29.66 dB**

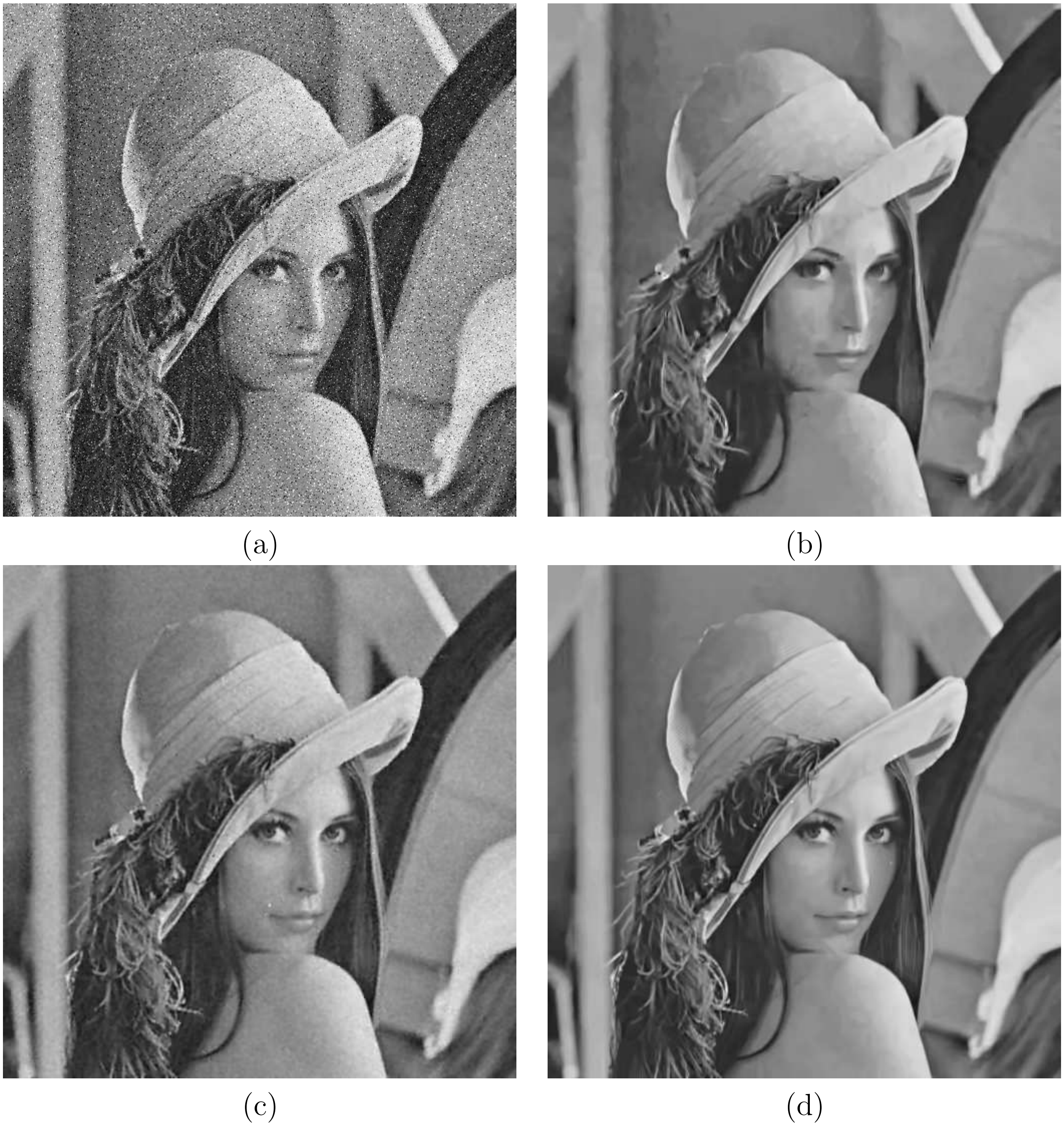


Figure 7.6: (a) The *Lena* image corrupted by 20 dB of Gaussian noise and 10% outliers. (b) Denoising with BM3D (30.66 dB). (c) Denoising with KGARD (31.94 dB). (d) Denoising with joint KGARD-BM3D (33.81 dB).

Table 7.2: Denoising performed on the *boat* image corrupted by various types and intensities of noise using the state-of-the-art wavelet method BM3D with and without outlier detection.

Method	Parameters	Gaussian Noise	Impulses (±100)	PSNR
BM3D	$s = 25$	25 dB	5%	30.57 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 10$	25 dB	5%	**34.61 dB**
BM3D	$s = 30$	25 dB	10%	29.41 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 10$	25 dB	10%	**33.86 dB**
BM3D	$s = 45$	25 dB	20%	27.64 dB
KGARD-BM3D	$\sigma = 0.4$, $\lambda = 1$, $s = 15$	25 dB	20%	**31.62 dB**
BM3D	$s = 30$	20 dB	5%	30.16 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 10$	20 dB	5%	**32.19 dB**
BM3D	$s = 35$	20 dB	10%	28.97 dB
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 15$	20 dB	10%	**31.52 dB**
BM3D	$s = 50$	20 dB	20%	27.49 dB
KGARD-BM3D	$\sigma = 0.4$, $\lambda = 1$, $s = 15$	20 dB	20%	**29.7 dB**
BM3D	$s = 35$	15 dB	5%	**29.1 dB**
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 25$	15 dB	5%	28.54 dB
BM3D	$s = 40$	15 dB	10%	**28.13 dB**
KGARD-BM3D	$\sigma = 0.3$, $\lambda = 1$, $s = 25$	15 dB	10%	28.11 dB
BM3D	$s = 50$	15 dB	20%	**27.07 dB**
KGARD-BM3D	$\sigma = 0.4$, $\lambda = 1$, $s = 40$	15 dB	20%	26.99 dB

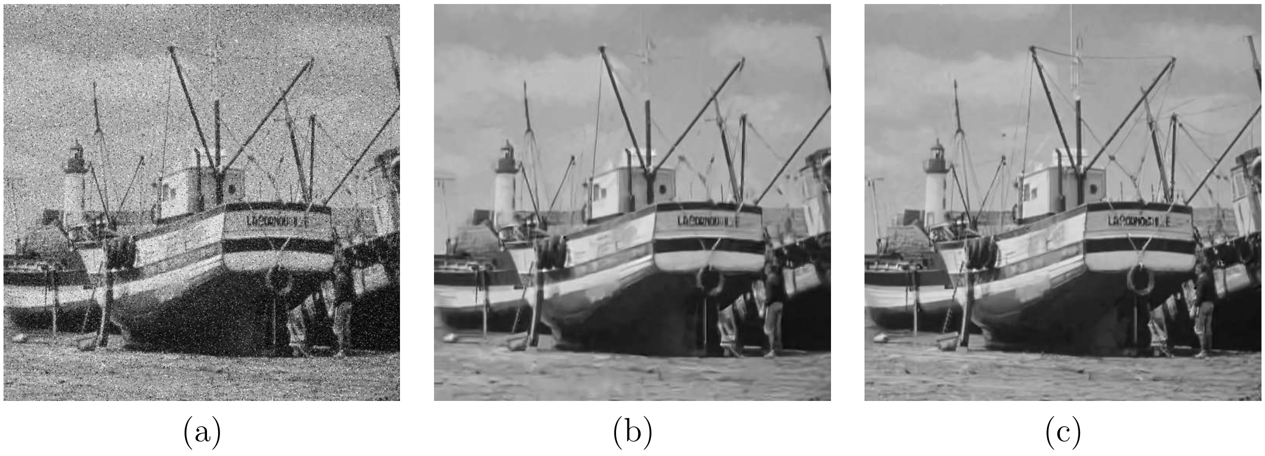
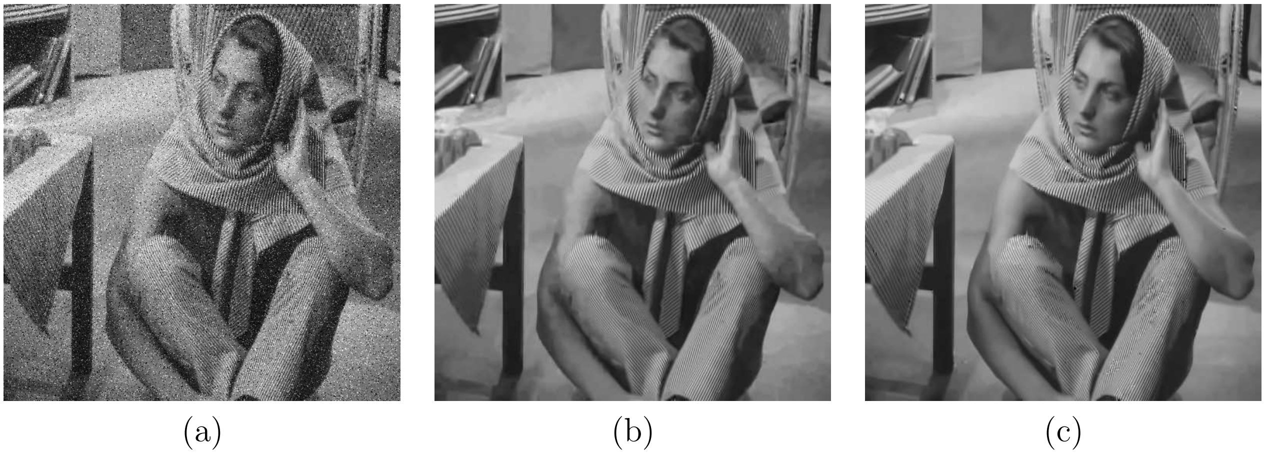


Figure 7.7: (a) The *boat* image corrupted by 20 dB of Gaussian noise and 10% outliers. (b) Denoising with BM3D (28.97 dB). (c) Denoising with joint KGARD-BM3D (31.52 dB).

Table 7.3: Denoising performed on the *Barbara* image corrupted by various types and intensities of noise using the state-of-the-art wavelet method BM3D with and without outlier detection.

Method	Parameters	Gaussian Noise	Impulses (±100)	PSNR
BM3D	$s = 25$	25 dB	5%	31.06 dB
KGARD-BM3D	$\sigma = 0.15$, $\lambda = 0.1$, $s = 15$	25 dB	5%	**33.45 dB**
BM3D	$s = 30$	25 dB	10%	29.4 dB
KGARD-BM3D	$\sigma = 0.15$, $\lambda = 0.1$, $s = 20$	25 dB	10%	**31.25 dB**
BM3D	$s = 45$	25 dB	20%	27.78 dB
KGARD-BM3D	$\sigma = 0.15$, $\lambda = 0.2$, $s = 30$	25 dB	20%	**28.03 dB**
BM3D	$s = 25$	20 dB	5%	30.69 dB
KGARD-BM3D	$\sigma = 0.15$, $\lambda = 0.1$, $s = 15$	20 dB	5%	**32.24 dB**
BM3D	$s = 35$	20 dB	10%	29.2 dB
KGARD-BM3D	$\sigma = 0.15$, $\lambda = 0.1$, $s = 20$	20 dB	10%	**30.43 dB**
BM3D	$s = 50$	20 dB	20%	**27.68 dB**
KGARD-BM3D	$\sigma = 0.15$, $\lambda = 0.15$, $s = 30$	20 dB	20%	27.48 dB
BM3D	$s = 30$	15 dB	5%	29.71 dB
KGARD-BM3D	$\sigma = 0.15$, $\lambda = 0.1$, $s = 25$	15 dB	5%	**29.97 dB**
BM3D	$s = 40$	15 dB	10%	28.41 dB
KGARD-BM3D	$\sigma = 0.15$, $\lambda = 0.1$, $s = 30$	15 dB	10%	**28.73 dB**
BM3D	$s = 50$	15 dB	20%	**27.27 dB**
KGARD-BM3D	$\sigma = 0.15$, $\lambda = 0.1$, $s = 45$	15 dB	20%	26.39 dB



(a) (b) (c)

Figure 7.8: (a) The *Barbara* image corrupted by 20 dB of Gaussian noise and 10% outliers. (b) Denoising with BM3D (29.2 dB). (c) Denoising with joint KGARD-BM3D (30.43 dB).

Chapter 8

Summary of the Thesis and Conclusions

In the preceding chapters, we have addressed the task of robust linear and nonlinear regression via sparse modeling methods, within the context of machine learning. The proposed methods are built on the popular Orthogonal Matching Pursuit algorithm (OMP) by imposing sparsity constraints on the outliers. The results of this novel approach are summarized in this final chapter and conclusions are also provided.

8.1 The Linear Regression Task

The novel iterative scheme, i.e., Greedy Algorithm for Robust Denoising (GARD), was developed as a robust tool for the task of linear regression and it is based on the popular OMP algorithm, which has been introduced for sparse modeling optimization tasks. GARD alternates between an OMP selection step, which identifies the outliers, and a Least Squares estimation, that attempts to fit the data. Thus, it establishes a bridge between the Outlier Diagnostics and the Robust Regression approaches.

The simplicity of the proposed algorithm contributed to the establishment of sound theoretical results. First, it was proved that the algorithm always converges to a solution in a finite number of iteration steps. However, this, by itself, is not a strong evidence for the method's performance. To this end, the focus was turned on establishing results concerning the identification of the outliers; two scenarios were treated separately. In the first one, we assumed that no inlier noise exists (only outliers exist in the noise) and we established that under a sufficient bound/condition on the RIP constant that: a) the outliers are identified and b) GARD's solution is exact (zero error). Although, at first, it seems that this is of limited interest in practice, as a matter of fact, it paves the way to establish the more complex condition for the noisy case. In the second scenario, where both outlier and bounded inlier noise are present, a condition is accordingly established for the recovery of the sparse outlier vector's support. It should be noted that such a condition is derived for the first time in the robust regression framework. Moreover, if this condition is satisfied, it also follows that the method is stable, i.e., that the estimation error is bounded by a term depending on the RIP constant. In fact, this is an improvement upon previously established results; the contribution is that the latter bound is independent of the methods' capability to estimate the outliers (in terms of both support and values), thus it is constant. On the contrary, in [40], the authors take into account the mismatch between the estimated and the true outlier vector, since the recovery of the outlier vector's support is not guaranteed. Hence, if the estimator does not succeed to correctly identify the outliers, the bound could reach arbitrary large values. Our results prove that GARD:

- Correctly identifies the outliers according to Theorem 4.4.
- At the end of the process, the associated error defined in (4.44) is well bounded. Thus, the solution is stable.

The experiments performed with the GARD algorithm indicate the overall advantages of the method. In particular, GARD: a) has an overall enhanced tolerance to the outliers, compared to its competitors, b) it performs an improved estimation (attains the lowest MSE) and c) it has low computational requirements. Finally, the experiments verify that it outperforms the ℓ_1-norm minimization schemes, for low sparsity levels; however, when the fraction of outliers that contaminates the data increases beyond a level (depending on the dimensionality of the unknown vector and the number of data), ℓ_1-norm minimization techniques demonstrate greater tolerance, something which is inline with the experimental evidence concerning the greedy optimization methods that are developed for sparse signal recovery.

Moreover, the study of GARD's properties, provided some answers towards several issues regarding the ordinary LS estimator. In Chapter 2, we have pointed out the direction taken by Statisticians, i.e., to analyze the residual according to the diagonal elements of the hat matrix, which led to the definition of leverage points. Although the existence of a single leverage point was known to put a threat to the LS estimator or an attempt to detect outliers via its residual, many questions remained unanswered. For example, when do leverage points occur and if so is it certain that they lead to erroneous data (recall that "good" leverage points may not affect the estimation at all)? Moreover, what are the conditions that should be satisfied so that occurrences of leverage points are limited or even prohibited? Also, is it possible for an outlier to be identified via the residual in cases where medium leverage points exist? Finally, since leverage points arise in the input data, is it valid to expect from a model that is designed for the removal of noise in the outputs (low breakdown point estimator) to detect them?

Our gained knowledge indicates that the occurrence of leverage points is limited or even prohibited if the number of data is sufficiently larger than the number of unknowns. The claim is justified in the analysis of Section 2.2 in Chapter 2. Of course, this is only for the case where no outlier noise exists in the input data. Thus, the LS residual is granted as a reliable source for detecting a single outlier, which is also theoretically justified by the result of Theorem 4.4. The obtained bound with respect to the RIP constant is stronger than the analysis based on the residual. This is due to the following reasons:

1. According to Remark 4.13, it takes into account the separation between the outliers and the inlier noise.
2. It guarantees the identification of an outlier regardless of the diagonal values of the hat matrix, since the values (in the absolute sense) of the outliers are also taken into consideration.

8.2 The Nonlinear Regression Task

The study of the respective kernel-based nonlinear regression task was introduced in Chapter 6. Since it could not be viewed as a generalization of the linear regression task, the modeling was modified. Besides, this is also the reason that for the analysis a different path had to be followed and eventually led to the bound of the maximum singular value in Theorem 6.1. Recall that we assumed that the unknown nonlinear function belongs to an RKHS, thus we have dealt with a nonparametric task, in contrast to the linear one, which is a parametric one. Since in such models overfitting issues occur, the incorporation of a regularization term ensures that the estimated nonlinear function is relatively smooth. The resulting algorithm, KGARD, alternates between an OMP selection step and a Kernel Ridge Regression (KRR) step at each iteration.

The study of this greedy-based selection scheme led to some interesting results:

- The solution to the regularized Least Squares task at each step is unique.
- For the case where only outliers exist in the noise, the bound on the maximum singular value of the matrix $\boldsymbol{A}_{(0)} = [\boldsymbol{K}\ \boldsymbol{1}]$ guarantees that the method identifies the outliers, first.

However, since a regularization term is also involved, KGARD's termination parameter ϵ is not directly related to the noise level. Recall that for the linear task, the noiseless case implies that $\epsilon = 0$; however, for the nonlinear one, since each projection is oblique, one should set for $\epsilon > 0$. Hence, unless perfect tuning of the parameter is performed, it is not guaranteed that no other extra indices, i.e., those corresponding to healthy observations, are classified as outliers. Unfortunately, an additional condition could not be derived for the noisy case, since it was technically demanding. On the other hand, it should be noted that such a result (on the identification for the noiseless case) has not been ever established in the respective literature. Although the authors in [69] deal with a convex task for the AM solver and an approximation to the ℓ_0-norm minimization one for the RAM solver (which performs better), there are no theoretical justifications that these method succeed in identifying the outliers. Also in [70], where the authors follow a Bayesian approach model, no such results are derived.

We have already discussed the influence of leverage points in any LS-based estimation method; indeed, KGARD also seems to be highly affected by such abnormalities. Although the conditions that guarantee the outlier identification are derived for the linear case (both noiseless and noisy) with GARD, for the KGARD no such condition is derived although it has been seriously attempted for the case both inlier and outlier noise are present. This seems discouraging at first, since it puts the proposed method at risk. However, from the analysis provided in Section 6.3.2, it follows that the occurrences of leverage points are limited if not prohibited. Notice in (6.15), by performing a regularization and for any parameter $\lambda > 0$, that the elements of the hat matrix are always downweighted. Hence, one could always find such a value, so that the average of h_{ii}'s, i.e., $\bar{h}$, is relatively small. Albeit we would have preferred the establishment of better guarantees, unfortunately this was not possible and it remains an open problem.

On the experimental section, various simulations were performed designating the overall advantages of KGARD against its competitors. First, in Section 6.3.3, the validity of the established bound in (6.16) was verified. Next, various tests with synthetic data were performed, while we have measured the MSE, the Mean Implementation Time (MIT) and the number of correct and wrong indices that each method has classified as outliers. There it was shown that, KGARD attains a low MSE with the exception of cases where heavy noise exists (both inlier and outlier); there, RAM does slightly better. It is also evident that KGARD's computational requirements are very low, especially for cases where the outliers are only few. Finally, the robust greedy-based framework was applied to the task of image denoising in Chapter 7, for the removal of salt and pepper noise. In such cases, classical methods, such as wavelets display limited performance. To this end, two novel methods were derived; the first one is exclusively based on KGARD, while the second one first employs KGARD to identify the outliers and remove their contribution and then a wavelet-based method in order to remove the remaining noise components. The experiments, which were performed with salt and pepper noise plus inlier noise, indicate the gains of the two proposed algorithms. The results on the combined KGARD-BM3D method exhibit gains in terms of PSNR, which are significantly increased for lower fractions of outliers; on the other hand, when the fraction of the outliers increases, we may still benefit from the cooperation of the two methods, unless the level of the inlier noise is also high. In such an extreme case, the KGARD algorithm does not contribute to the improvement of the denoising process.

8.3 Future Directions

Greatly influenced by the simplicity and the performance of the proposed robust scheme, at this final section we present several possible directions for future work.

8.3.1 Generalization of the Method with Noise in the Input Data

Most of the methods that are developed for the robust regression task assume that outlier noise most commonly occurs in the outputs, i.e., the y_i's. However, if leverage points exist, which is often related to noise on the inputs (not always), these methods become unreliable. Thus, inspired by the performance and simplicity of the proposed schemes, an extension of GARD (or KGARD) for this demanding task would be an interesting direction. Since the residual of the LS estimator could not be used any further due to the existence of leverage points, one could resort to other more robust signle-outlier detection-based residuals, such as the (internally or externally) studentized residuals in [20] or the Cook's distance in [94]. The estimation part could either remain unchanged (LS-based) or performed according to another estimator, such as the Total Least Squares (TLS) one. The fact that the implementation of such a scheme is simple and thus the computational requirements low, would also be a great advantage, in contrast to most of the existing methods, which, in general, operate with high computational efforts.

8.3.2 GARD in a Distributed Network Environment

Another interesting research direction is the development of a model, based on the proposed greedy scheme, that makes cooperative usage of the estimated parameters in order to improve the estimation in a distributed environment. The method should simultaneously be able to detect the outliers that may arise within the sensing network and also enhance the performance of the estimator, compared to the model with a single sensor. However, several modeling scenarios could be used, depending on the application. Moreover, other issues, such as termination criteria, which are application dependent, should be taken into account, accordingly.

Abbreviations

ADMM	Alternating Direction Method of Multipliers
AM	Alternating direction Method
AWGN	Additive White Gaussian Noise
BLUE	Best Linear Unbiased Estimator
BM3D	Block Matching and 3-D filtering
BP	Basis Pursuit
EM	Expectation Maximization
GARD	Greedy Algorithm for Robust Denoising
GM-est	Generalized Maximum Likelihood estimator
IRLS	Iteratively Reweighted Least Squares
KGARD	Kernel Greedy Algorithm for Robust Denoising
KRR	Kernel Ridge Regression
LASSO	Least Absolute Shrinkage and Selection Operator
LAV	Least Absolute Values
LMedS	Least Median of Squares Regression
LS	Least Squares
LTS	Least Trimmed Squares
M-est	Maximum likelihood estimator
MIL	Matrix Inversion Lemma
MIT	Mean Implementation Time
ML	Maximum Likelihood
MSE	Mean-square-error
MVUE	Minimum Variance Unbiased Estimator
OMP	Orthogonal Matching Pursuit
PSNR	Peak signal-to-noise ratio
RAM	Refined Alternating directions Method of Multipliers
RANSAC	RANndom SAmple Consensus
RBF	Radial Basis Function
RB-RVM	Robust Relevance Vector Machine
RIP	Restricted Isometry Property
RKHS	Reproducing Kernel Hilbert Space
RKRR	Robust Kernel Ridge Regression
ROI	Region Of Interest
ROMP	Robust Orthogonal Matching Pursuit
RR	Robust Regression
rROI	Reduced Region Of Interest
RVM	Relevance Vector Machine
SBL	Sparse Bayesian Learning
SOCP	Second Order Cone Programming
SVR	Support Vector Regression
WAM	Weighted Alternating directions Method of Multipliers
WLS	Weighted Least Squares

Appendix

A. The Inversion of Matrix $\boldsymbol{I}_N - \boldsymbol{A}$

The geometric series with ratio r

$$\sum_{k=0}^{\infty} r^k = 1 + r + r^2 + \cdots = \frac{1}{1-r}, \text{ if } |r| < 1$$

and diverges if $|r| \geq 1$.

Lemma. *For a matrix $\boldsymbol{A} \in \mathbb{R}^{N \times N}$ if $\|\boldsymbol{A}\|_2 < 1$, then the matrix $\boldsymbol{I}_N - \boldsymbol{A}$ is non-singular and*

$$\left\|(\boldsymbol{I}_N - \boldsymbol{A})^{-1}\right\|_2 \leq \frac{1}{1 - \|\boldsymbol{A}\|_2}.$$

Proof. The first part of the proof relies on contradiction. Suppose that matrix $\boldsymbol{I}_N - \boldsymbol{A}$ is singular; that is, there exists $\boldsymbol{x} \neq \boldsymbol{0}$ such that $(\boldsymbol{I}_N - \boldsymbol{A})\boldsymbol{x} = \boldsymbol{0}$ which leads to $\|\boldsymbol{A}\boldsymbol{x}\|_2 = \|\boldsymbol{x}\|_2$. However, $\|\boldsymbol{x}\|_2 = \|\boldsymbol{A}\boldsymbol{x}\|_2 \leq \|\boldsymbol{A}\|_2 \|\boldsymbol{x}\|_2 < \|\boldsymbol{x}\|_2$, which is a contradiction. Hence, the $N \times N$ matrix $\boldsymbol{I}_N - \boldsymbol{A}$ is non-singular.

Furthermore, consider the obvious identity

$$\left(\sum_{k=0}^{N} \boldsymbol{A}^k\right)(\boldsymbol{I}_N - \boldsymbol{A}) = \boldsymbol{I}_N - \boldsymbol{A}^{N+1}.$$

However, $\|\boldsymbol{A}^k\|_2 \leq \|\boldsymbol{A}\|_2^k$, and since $\|\boldsymbol{A}\|_2 < 1$, we have that $\boldsymbol{A}^k \to \boldsymbol{O}_{N \times N}$ as $k \to \infty$. As a result,

$$\lim_{N \to \infty} \left(\sum_{k=0}^{N} \boldsymbol{A}^k\right)(\boldsymbol{I}_N - \boldsymbol{A}) = \boldsymbol{I}_N,$$

so that the Neumann series $(\boldsymbol{I}_N - \boldsymbol{A})^{-1} = \sum_{k=0}^{\infty} \boldsymbol{A}^k$ converges. Thus, it is obtained that

$$\left\|(\boldsymbol{I}_N - \boldsymbol{A})^{-1}\right\|_2 = \left\|\sum_{k=0}^{\infty} \boldsymbol{A}^k\right\|_2 \leq \sum_{k=0}^{\infty} \left\|\boldsymbol{A}^k\right\|_2 \leq \sum_{k=0}^{\infty} \|\boldsymbol{A}\|_2^k = \frac{1}{1 - \|\boldsymbol{A}\|_2},$$

by employing the geometric series and the proof is complete. □

B. Matrix Inversion Lemma

Let $\boldsymbol{A}$ and $\boldsymbol{D}-\boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B}$ be non-singular matrices; then the Matrix Inversion Lemma (MIL) (also known as the Woodbury matrix identity) is the following formula:

$$\left(\boldsymbol{A}-\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C}\right)^{-1}=\boldsymbol{A}^{-1}+\boldsymbol{A}^{-1}\boldsymbol{B}\left(\boldsymbol{D}-\boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B}\right)^{-1}\boldsymbol{C}\boldsymbol{A}^{-1}.$$

A very useful application of the Matrix Inversion Formula is for the inversion of the matrix

$$\boldsymbol{M}=\begin{bmatrix}\boldsymbol{A} & \boldsymbol{B}\\ \boldsymbol{C} & \boldsymbol{D}\end{bmatrix},$$

which is given in block form. If the $N\times N$ and $K\times K$ matrices, $\boldsymbol{A}$ and $\boldsymbol{D}$, respectively, are invertible, then

$$\boldsymbol{M}^{-1}=\begin{bmatrix}(\boldsymbol{A}-\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1} & -\boldsymbol{A}^{-1}\boldsymbol{B}(\boldsymbol{D}-\boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\\ -\boldsymbol{D}^{-1}\boldsymbol{C}(\boldsymbol{A}-\boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{C})^{-1} & (\boldsymbol{D}-\boldsymbol{C}\boldsymbol{A}^{-1}\boldsymbol{B})^{-1}\end{bmatrix}.$$

As a special case, if $K=1$ and for the form

$$\boldsymbol{M}=\begin{bmatrix}\boldsymbol{A} & \boldsymbol{b}\\ \boldsymbol{b}^T & c\end{bmatrix},$$

the inverse of $\boldsymbol{M}$ is

$$\boldsymbol{M}^{-1}=\begin{bmatrix}\left(\boldsymbol{A}-\frac{\boldsymbol{b}\boldsymbol{b}^T}{c}\right)^{-1} & -\frac{1}{\beta}\boldsymbol{A}^{-1}\boldsymbol{b}\\ -\frac{1}{\beta}\boldsymbol{b}^T\boldsymbol{A}^{-1} & \frac{1}{\beta}\end{bmatrix}=\begin{bmatrix}\boldsymbol{A}^{-1}+\frac{1}{\beta}\boldsymbol{A}^{-1}\boldsymbol{b}\boldsymbol{b}^T\boldsymbol{A}^{-1} & -\frac{1}{\beta}\boldsymbol{A}^{-1}\boldsymbol{b}\\ -\frac{1}{\beta}\boldsymbol{b}^T\boldsymbol{A}^{-1} & \frac{1}{\beta}\end{bmatrix},$$

where $\beta=c-\boldsymbol{b}^T\boldsymbol{A}^{-1}\boldsymbol{b}$.

C. Matrix Decomposition

QR factorization

A $N\times M$ matrix $\boldsymbol{A}$ with $\text{rank}(\boldsymbol{A})=M,\ (M<N)$ may be decomposed as

$$\boldsymbol{A}=\boldsymbol{Q}\boldsymbol{R},$$

where $\boldsymbol{Q}$ is orthogonal and $\boldsymbol{R}$ is an upper triangular matrix with positive diagonal elements.

Cholesky decomposition

A symmetric, positive definite matrix $\boldsymbol{X}$ may be decomposed as

$$\boldsymbol{X}=\boldsymbol{L}\boldsymbol{L}^T,$$

where $\boldsymbol{L}$ is a lower triangular matrix with positive diagonal elements.

Singular Value Decomposition (SVD)

The Singular Value Decomposition of a $N \times M$, with $N < M$, full rank matrix $\boldsymbol{A}$ is given by

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T,$$

where $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal, i.e., $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}_N$ and $\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{V}\boldsymbol{V}^T = \boldsymbol{I}_M$. The matrix $\boldsymbol{S}$ is of dimension $N \times M$ of the form $\boldsymbol{S} = [\boldsymbol{\Sigma}\ \boldsymbol{0}]$, where $\boldsymbol{\Sigma}$ is the diagonal matrix with entries the singular values of matrix $\boldsymbol{A}$.

Bibliography

[1] S. Theodoridis, *Machine Learning: A Bayesian and Optimization Perspective.* Academic Press, 2015.

[2] A. M. Legendre, *Nouvelles méthodes pour la détermination des orbites des comètes.* No. 1, F. Didot, 1805.

[3] C. F. Gauss, *Theoria motus corporum coelestium in sectionibus conicis solem ambientium auctore Carolo Friderico Gauss.* Hamburgi sumtibus Frid. Perthes et IH Besser, 1809.

[4] C.-F. Gauss, *Theoria combinationis observationum erroribus minimis obnoxiae.-Gottingae, Henricus Dieterich 1823.* Henricus Dieterich, 1823.

[5] W. J. Dixon *et al.*, "Analysis of extreme values," *The Annals of Mathematical Statistics*, vol. 21, no. 4, pp. 488–506, 1950.

[6] F. E. Grubbs, "Procedures for detecting outlying observations in samples," *Technometrics*, vol. 11, no. 1, pp. 1–21, 1969.

[7] D. M. Hawkins, *Identification of outliers*, vol. 11. Springer, 1980.

[8] V. Barnett and T. Lewis, *Outliers in statistical data*, vol. 3. Wiley New York, 1994.

[9] D. S. Moore and G. P. McCabe, *Introduction to the Practice of Statistics.* WH Freeman/Times Books/Henry Holt & Co, 1989.

[10] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *ACM SIGMOD Record*, vol. 29, pp. 427–438, ACM, 2000.

[11] J. Tang, Z. Chen, A. W.-C. Fu, and D. W. Cheung, "Enhancing effectiveness of outlier detections for low density patterns," in *Advances in Knowledge Discovery and Data Mining*, pp. 535–548, Springer, 2002.

[12] Z. Chen, J. Tang, and A. W.-C. Fu, "Modeling and efficient mining of intentional knowledge of outliers," in *Database Engineering and Applications Symposium, 2003. Proceedings. Seventh International*, pp. 44–53, IEEE, 2003.

[13] J. Tang, Z. Chen, A. W. Fu, and D. W. Cheung, "Capabilities of outlier detection schemes in large datasets, framework and methodologies," *Knowledge and Information Systems*, vol. 11, no. 1, pp. 45–84, 2007.

[14] P. J. Huber, “The 1972 wald lecture robust statistics: A review,” *The Annals of Mathematical Statistics*, pp. 1041–1067, 1972.

[15] P. J. Huber *et al.*, “Robust estimation of a location parameter,” *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964.

[16] P. J. Rousseeuw and B. C. Van Zomeren, “Unmasking multivariate outliers and leverage points,” *Journal of the American Statistical Association*, vol. 85, no. 411, pp. 633–639, 1990.

[17] A. M. Leroy and P. J. Rousseeuw, “Robust regression and outlier detection,” *J. Wiley&Sons, New York*, 1987.

[18] P. J. Huber, *Wiley Series in Probability and Mathematics Statistics.* Wiley Online Library, 1981.

[19] R. A. Maronna, R. D. Martin, and V. J. Yohai, *Robust statistics.* J. Wiley, 2006.

[20] P. J. Rousseeuw and A. M. Leroy, *Robust regression and outlier detection*, vol. 589. John Wiley & Sons, 2005.

[21] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[22] S. S. Chen, D. L. Donoho, and M. A. Saunders, “Atomic decomposition by basis pursuit,” *SIAM journal on scientific computing*, vol. 20, no. 1, pp. 33–61, 1998.

[23] E. J. Candes, J. K. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements,” *Communications on pure and applied mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

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

[25] M. Lustig, D. Donoho, and J. M. Pauly, “Sparse mri: The application of compressed sensing for rapid mr imaging,” *Magnetic resonance in medicine*, vol. 58, no. 6, pp. 1182–1195, 2007.

[26] E. J. Candès, J. Romberg, and T. Tao, “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” *IEEE Transactions on Information Theory*, vol. 52, no. 2, pp. 489–509, 2006.

[27] J. A. Tropp and A. C. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[28] G. Davis, S. Mallat, and M. Avellaneda, “Adaptive greedy approximations,” *Constructive approximation*, vol. 13, no. 1, pp. 57–98, 1997.

[29] S. Mallat, *A wavelet tour of signal processing: the sparse way.* Academic press, 2008.

[30] D. Needell and R. Vershynin, "Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 310–316, 2010.

[31] D. Needell and J. A. Tropp, "Cosamp: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, 2009.

[32] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[33] R. J. Tibshirani *et al.*, "The lasso problem and uniqueness," *Electronic Journal of Statistics*, vol. 7, pp. 1456–1490, 2013.

[34] G. Papageorgiou, P. Bouboulis, S. Theodoridis, and K. Themelis, "Robust linear regression analysis-the greedy way," in *Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European*, pp. 16–20, IEEE, 2014.

[35] G. Papageorgiou, P. Bouboulis, and S. Theodoridis, "Robust linear regression analysis-a greedy approach," *IEEE Transactions on Signal Processing*, vol. 63, no. 15, pp. 3872–3887, 2014.

[36] G. Papageorgiou, P. Bouboulis, and S. Theodoridis, "Robust kernel-based regression using orthogonal matching pursuit," in *Machine Learning for Signal Processing (MLSP), 2013 IEEE International Workshop on*, pp. 1–6, IEEE, 2013.

[37] P. Bouboulis, G. Papageorgiou, and S. Theodoridis, "Robust image denoising in RKHS via orthogonal matching pursuit," in *Cognitive Information Processing (CIP), 2014 4th International Workshop on*, pp. 1–6, IEEE, 2014.

[38] G. Papageorgiou, P. Bouboulis, and S. Theodoridis, "Robust non-linear regression analysis: A greedy approach employing kernels and application to image denoising," January 2016.

[39] E. J. Candès, "The restricted isometry property and its implications for compressed sensing," *Comptes Rendus Mathematique*, vol. 346, no. 9, pp. 589–592, 2008.

[40] K. Mitra, A. Veeraraghavan, and R. Chellappa, "Analysis of sparse regularization based robust regression approaches," *IEEE Transactions on Signal Processing*, vol. 61, no. 5, pp. 1249–1257, 2013.

[41] J. Portilla, V. Strela, M. Wainwright, and E. P. Simoncelli, "Image denoising using scale mixtures of gaussians in the wavelet domain," *IEEE Transactions on Image Processing*, vol. 12, no. 11, pp. 1338–1351, 2003.

[42] L. Sendur and I. Selesnick, "Bivariate shrinkage functions for wavelet-based denoising exploiting interscale dependency," *IEEE Transactions on Signal Processing*, vol. 50, no. 11, pp. 2744–2756, 2002.

[43] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3d transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.

[44] K. Seongjai, "PDE-based image restoration : A hybrid model and color image denoising," *IEEE Transactions on Image Processing*, vol. 15, no. 5, pp. 1163–1170, 2006.

[45] H. Takeda, S. Farsiu, and P. Milanfar, "Kernel regression for image processing and reconstruction," *IEEE Tranactions on Image Processing*, vol. 16, no. 2, pp. 349–366, 2007.

[46] P. Bouboulis, K. Slavakis, and S. Theodoridis, "Adaptive kernel-based image denoising employing semi-parametric regularization," *IEEE Transactions on Image Processing*, vol. 19, no. 6, pp. 1465–1479, 2010.

[47] Y. She and A. B. Owen, "Outlier detection using nonconvex penalized regression," *Journal of the American Statistical Association*, vol. 106, no. 494, 2011.

[48] D. C. Hoaglin and R. E. Welsch, "The hat matrix in regression and anova," *The American Statistician*, vol. 32, no. 1, pp. 17–22, 1978.

[49] S. Van Huffel and J. Vandewalle, *The total least squares problem: computational aspects and analysis*, vol. 9. Siam, 1991.

[50] F. Y. Edgeworth, "On observations relating to several quantities," *Hermathena*, vol. 6, no. 13, pp. 279–285, 1887.

[51] R. W. Hill, *Robust Regression When There Are Outliers in the Carriers.* PhD thesis, Harvard University, Boston, 1977. Unpublished.

[52] C. L. Mallows, "On some topics in robustness." Unpublished memorandum, Bell Telephone Laboratories, Murray Hill, NJ, 1975.

[53] A. M. Bruckstein, D. L. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *SIAM review*, vol. 51, no. 1, pp. 34–81, 2009.

[54] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM journal on computing*, vol. 24, no. 2, pp. 227–234, 1995.

[55] Y. C. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Conference on Signals, Systems and Computers, Conference Record of The Twenty-Seventh Asilomar*, pp. 40–44, IEEE, 1993.

[56] J. A. Tropp, "Greed is good: Algorithmic results for sparse approximation," *IEEE Transactions on Information Theory*, vol. 50, no. 10, pp. 2231–2242, 2004.

[57] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[58] M. Vetterli and T. Kalker, "Matching pursuit for compression and application to motion compensated video coding," in *IEEE International Conference on Image Processing, 1994 (ICIP-94.)*, vol. 1, pp. 725–729, IEEE, 1994.

[59] A. M. Bruckstein, D. L. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," *SIAM review*, vol. 51, no. 1, pp. 34–81, 2009.

[60] L. Rebollo-Neira and D. Lowe, "Optimized orthogonal matching pursuit approach," *Signal Processing Letters, IEEE*, vol. 9, no. 4, pp. 137–140, 2002.

[61] D. L. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck, "Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 58, no. 2, pp. 1094–1121, 2012.

[62] J. Wang, S. Kwon, and B. Shim, "Generalized orthogonal matching pursuit," *IEEE Transactions on Signal Processing*, vol. 60, no. 12, pp. 6202–6216, 2012.

[63] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear algebra and its applications*, vol. 284, no. 1, pp. 193–228, 1998.

[64] S. P. Boyd and L. Vandenberghe, *Convex optimization.* Cambridge University press, 2004.

[65] D. W. Peaceman and H. H. Rachford, Jr, "The numerical solution of parabolic and elliptic differential equations," *Journal of the Society for Industrial & Applied Mathematics*, vol. 3, no. 1, pp. 28–41, 1955.

[66] J. Douglas, Jr, "On the numerical integration of ∂ˆ2u/∂xˆ2+∂ˆ2u/∂yˆ2=∂u/∂t by implicit methods," *Journal of the Society for Industrial & Applied Mathematics*, vol. 3, no. 1, pp. 42–65, 1955.

[67] S. Boyd, "Alternating direction method of multipliers," in *Talk at NIPS Workshop on Optimization and Machine Learning*, 2011.

[68] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[69] G. Mateos and G. B. Giannakis, "Robust nonparametric regression via sparsity control with application to load curve data cleansing," *IEEE Transactions on Signal Processing*, vol. 60, no. 4, pp. 1571–1584, 2012.

[70] K. Mitra, A. Veeraraghavan, and R. Chellappa, "Robust RVM regression using sparse outlier model," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1887–1894, IEEE, 2010.

[71] Y. Jin and B. D. Rao, "Algorithms for robust linear regression by exploiting the connection to sparse signal recovery," in *International Conference on Acoustics Speech and Signal Processing (ICASSP)*, pp. 3830–3833, IEEE, 2010.

[72] M. E. Tipping, "Sparse Bayesian learning and the relevance vector machine," *The journal of machine learning research*, vol. 1, pp. 211–244, 2001.

[73] D. P. Wipf and B. D. Rao, "Sparse bayesian learning for basis selection," *IEEE Transactions on Signal Processing*, vol. 52, no. 8, pp. 2153–2164, 2004.

[74] D. Wipf, J. Palmer, and B. Rao, "Perspectives on sparse bayesian learning," *Computer Engineering*, vol. 16, no. 1, p. 249, 2004.

[75] S. A. Razavi, E. Ollila, and V. Koivunen, "Robust greedy algorithms for compressed sensing," in *Signal Processing Conference (EUSIPCO), Proceedings of the 20th European*, pp. 969–973, IEEE, 2012.

[76] B. L. Sturm and M. G. Christensen, "Comparison of orthogonal matching pursuit implementations," in *Signal Processing Conference (EUSIPCO), Proceedings of the 20th European*, pp. 220–224, IEEE, 2012.

[77] D. L. Donoho, M. Elad, and V. N. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Transactions on Information Theory*, vol. 52, no. 1, pp. 6–18, 2006.

[78] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods.* Cambridge University Press, 2000.

[79] K. Slavakis, P. Bouboulis, and S. Theodoridis, *Signal Processing Theory and Machine Learning: Online Learning in Reproducing Kernel Hilbert Spaces*, ch. 17. Academic Press Library in Signal Processing, 2014.

[80] A. J. Smola and B. Schölkopf, *Learning with Kernels.* The MIT Press, 2002.

[81] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, 4th Edition.* Academic press, 2008.

[82] J. Mercer, "Functions of positive and negative type, and their connection with the theory of integral equations," *Philosophical transactions of the royal society of London. Series A, containing papers of a mathematical or physical character*, pp. 415–446, 1909.

[83] K. Slavakis, P. Bouboulis, and S. Theodoridis, "Online learning in reproducing kernel hilbert spaces," *Signal Processing Theory and Machine Learning*, pp. 883–987, 2013.

[84] W. Rudin, *Real and complex analysis.* Tata McGraw-Hill Education, 1987.

[85] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[86] V. Y. A. A. N. Tikhonov, "Solutions of ill-posed problems," *Mathematics of Computation*, vol. 32, no. 144, pp. 1320–1322, 1978.

[87] P. Bouboulis and S. Theodoridis, "Kernel methods for image denoising," in *Regularization, optimization, kernels, and support vector machines* (J. Suykens, M. Signoretto, and A. Argyriou, eds.), CRC Press, 2015.

[88] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted ℓ 1 minimization," *Journal of Fourier analysis and applications*, vol. 14, no. 5-6, pp. 877–905, 2008.

[89] W. Gander, *On the linear least squares problem with a quadratic constraint.* Computer Science Department, Stanford University, 1978.

[90] W. Gander, “Least squares with a quadratic constraint,” *Numerische Mathematik*, vol. 36, no. 3, pp. 291–307, 1980.

[91] Å. Björck, *Numerical methods for least squares problems.* No. 51, Society for Industrial and Applied Mathematics, 1996.

[92] M. Rojas and D. C. Sorensen, “A trust-region approach to the regularization of large-scale discrete forms of ill-posed problems,” *SIAM Journal on Scientific Computing*, vol. 23, no. 6, pp. 1842–1860, 2002.

[93] R. O. Duda, P. E. Hart, *et al.*, *Pattern classification and scene analysis*, vol. 3. Wiley New York, 1973.

[94] R. D. C. S. Weisberg, *Residuals and influence in regression.* Monographs on statistics and applied probability (Series), Chapman and Hall/CRC, 1st ed., 1982.

Index

ℓ_p-norms, 64
- ℓ_0(pseudo)-norm, 47, 64
- ℓ_0-norm minimization task, 78
- ℓ_1-norm, 47
- ℓ_1-regression, 55
- ℓ_2-norm, 64

Active columns, 74
Alternating Direction Method of Multipliers (ADMM), 69, 113

Basis Pursuit (BP), 67
Best linear unbiased estimator (BLUE), 53
Bias, 108
BM3D filter, 146
Breakdown point, 53, 58

Cardinality, 74
Cholesky, 75, 162
Classification, 43
Combinatorial optimization, 47

Decomposition
- noisy observations, 84, 87
- signal, 79
- unique, 80

Empirical risk, 109
Estimation
- nonparametric, 44
- parametric, 44

Expectation Maximization (EM), 115

Full rank matrix, 52

Generalized LASSO, 69
Greedy Algorithm for Robust Denoising (GARD), 48, 73, 153
- complexity, 75
- complexity evaluation, 92
- error evaluation, 95
- support recovery test, 93

Hat matrix, 52, 125
Hypercube, 89
Hyperparameters, 70, 97, 115

Image denoising, 49
Inactive set, 74
Iteratively Reweighted Least Squares (IRLS), 56

Kernel
- matrix, 101
- trick, 101

Kernel functions, 102
- Gaussian rbf, 103, 107
- Laplacian rbf, 103
- polynomial kernel, 103
- spline kernel, 103

Kernel Greedy Algorithm for Robust Denoising (KGARD), 48, 117
Kernel Ridge Regression (KRR), 48, 109
KGARD for image denoising, 145

Lévy alpha-stable, 97
Least Absolute Shrinkage and Selection Operator (LASSO), 67
Least Absolute Values (LAV), 55
Least Median of Squares Regression (LMedS), 47, 59
Least Squares (LS), 44, 52
- estimator, 48
- regression estimate, 52
- regularized, 119, 125
- residual, 52, 53, 126

Least Trimmed Squares (LTS), 47, 59

Leverage points, 54, 87, 126

M-est, 55
- Generalized M-est (GM-est), 59

Machine Learning, 43
Matrix Inversion Lemma (MIL), 75, 162
Maximum Likelihood, 51
Mean-square-error (MSE), 90
Median Absolute Deviation (MAD), 71
Minimum-norm solution, 64
Monte-Carlo runs, 93, 96, 97, 120, 132
Moore-Penrose pseudo-inverse, 87

NP-Hard, 47, 65

Orthogonal Matching Pursuit (OMP), 48, 66
Outlier, 45, 47
- subspace, 79

Peak signal-to-noise ratio (PSNR), 147
Positive definite, 101
Positive semi-definite, 125
Projection matrix, 82
Projection theorem, 102

QR, 75, 77, 162
Quadratic optimization, 68

RANdom SAmple Consensus (RANSAC), 47, 60
Refined Alternating directions Method of Multipliers (RAM), 112
Regions of interest (ROI), 140
- reduced (rROI), 141

Regression, 43, 51
- function, 43
- subspace, 79

Representer Theorem, 108
Reproducing Kernel Hilbert Space (RKHS), 44, 48, 99
Restricted Isometry Property (RIP), 48, 79, 86
- constant, 79

Robust
- diagnostics, 46
- linear regression, 67
- nonlinear regression, 48
- regression, 44, 45
- robust function, 55
- statistics, 46

Robust Kernel Ridge Regression (RKRR), 110
Robust Orthogonal Matching Pursuit (ROMP), 70, 76
Robust Relevance Vector Machine (RB-RVM), 114

Second Order Cone Programming (SOCP), 68
Singular Value Decomposition (SVD), 125, 127, 163
Singular values, 125
- smallest, 86

Skewness, 97
Smallest principal angle, 78, 79
Sobel operator, 143
Soft-thresholding, 69
Sparse
- denoising, 65
- modeling, 47, 65

Sparse Bayesian Learning (SBL), 69
Standard Euclidean basis, 74
Support Vector Regression (SVR), 111

Tikhonov
- matrix, 122
- regularization parameter, 123

Underdetermined system of linear equations, 52

Vapnik's ϵ-sensitive loss, 109

Wavelets, 49
Weighted Alternating directions solver (WAM), 114
Weighted Least Squares (WLS), 56, 71