



KTH Electrical Engineering

Bayesian methods for sparse and low-rank matrix problems

MARTIN SUNDIN

Doctoral Thesis in Electrical Engineering
Stockholm, Sweden 2016

TRITA-EE 2016:087
ISSN 1653-5146
ISBN 978-91-7729-044-5

KTH, School of Electrical Engineering
Department of Signal Processing
SE-100 44 Stockholm
SWEDEN

Akademisk avhandling som med tillstånd av Kungl Tekniska högskolan framlägges till offentlig granskning för avläggande av teknologie doktorsexamen i elektro- och systemteknik onsdagen den 14 september 2016 klockan 10.15 i hörsal F3, Lindstedtsvägen 26, Stockholm.

© 2016 Martin Sundin, unless otherwise noted.

Tryck: Universitetsservice US AB

Abstract

Many scientific and engineering problems require us to process measurements and data in order to extract information. Since we base decisions on information, it is important to design accurate and efficient processing algorithms. This is often done by modeling the signal of interest and the noise in the problem. One type of modeling is *Compressed Sensing*, where the signal has a sparse or low-rank representation. In this thesis we study different approaches to designing algorithms for sparse and low-rank problems.

Greedy methods are fast methods for sparse problems which iteratively detects and estimates the non-zero components. By modeling the detection problem as an array processing problem and a Bayesian filtering problem, we improve the detection accuracy. Bayesian methods approximate the sparsity by probability distributions which are iteratively modified. We show one approach to making the Bayesian method the Relevance Vector Machine robust against sparse noise.

Bayesian methods for low-rank matrix estimation typically use probability distributions which only depends on the singular values or a factorization approach. Here we introduce a new method, the Relevance Singular Vector Machine, which uses precision matrices with prior distributions to promote low-rank. The method is also applied to the robust Principal Component Analysis (PCA) problem, where a low-rank matrix is contaminated by sparse noise.

In many estimation problems, there exists theoretical lower bounds on how well an algorithm can perform. When the performance of an algorithm matches a lower bound, we know that the algorithm has optimal performance and that the lower bound is tight. When no algorithm matches a lower bound, there exists room for better algorithms and/or tighter bounds. In this thesis we derive lower bounds for three different Bayesian low-rank matrix models.

In some problems, only the amplitudes of the measurements are recorded. Despite being non-linear, some problems can be transformed to linear problems. Earlier works have shown how sparsity can be utilized in the problem, here we show how the low-rank can be used.

In some situations, the number of measurements and/or the number of parameters is very large. Such *Big Data* problems require us to design new algorithms. We show how the Basis Pursuit algorithm can be modified for problems with a very large number of parameters.

Sammanfattning

Många vetenskapliga och ingenjörproblem kräver att vi behandlar mätningar och data för att finna information. Eftersom vi grundar beslut på information är det viktigt att designa noggranna och effektiva behandlingsalgoritmer. Detta görs ofta genom att modellera signalen vi söker och bruset i problemet. En typ av modellering är *Compressed Sensing* där signalen har en gles eller lågrangs-representation. I denna avhandling studerar vi olika sätt att designa algoritmer för glesa och lågrangsproblem.

Giriga metoder är snabba metoder för glesa problem som iterativt detekterar och skattar de nollskilda komponenterna. Genom att modellera detektionsproblemet som ett gruppantennproblem och ett Bayesianskt filtreringsproblem förbättrar vi prestandan hos algoritmerna. Bayesianska metoder approximerar glesheten med sannolikhetsfördelningar som iterativt modifieras. Vi visar ett sätt att göra den Bayesianska metoden Relevance Vector Machine robust mot glest brus.

Bayesianska metoder för skattning av lågrangsmatriser använder typiskt sannolikhetsfördelningar som endast beror på matrisens singularvärden eller en faktoriseringmetod. Vi introducerar en ny metod, Relevance Singular Vector Machine, som använder precisionsmatriser med a-priori fördelningar för att införa låg rang. Metoden används också för robust Principal Komponent Analys (PCA), där en lågrangsmatris har störts av glest brus.

I många skattningsproblem existerar det teoretiska undre gränser för hur väl en algoritm kan prestera. När en algoritm möter en undre gräns vet vi att algoritmen är optimal och att den undre gränsen är den bästa möjliga. När ingen algoritm möter en undre gräns vet vi att det existerar utrymme för bättre algoritmer och/eller bättre undre gränser. I denna avhandling härleder vi undre gränser för tre olika Bayesianska lågrangsmodeller.

I vissa problem registreras endast amplituderna hos mätningarna. Några problem kan transformeras till linjära problem, trots att de är olinjära. Tidigare arbeten har visat hur gleshet kan användas i problemet, här visar vi hur låg rang kan användas.

I vissa situationer är antalet mätningar och/eller antalet parametrar mycket stort. Sådana *Big Data*-problem kräver att vi designar nya algoritmer. Vi visar hur algoritmen Basis Pursuit kan modifieras när antalet parametrar är mycket stort.

Acknowledgments

Even though this thesis bears the name of one, it would not have been possible without the contribution of many. Many members of the signal processing lab at KTH, past and present, have contributed to the research of this thesis.

First and foremost I wish to thank my supervisor Professor Magnus Jansson for accepting me as a PhD student and introducing me to the fascinating field of Compressed Sensing. Magnus has always been available for technical discussions, new projects and have always provided good feedback and suggestions. Magnus eye for quality has been very helpful, greatly improving the quality of the work from the first draft to the final product. Many thanks to Magnus for proofreading the thesis during his vacation and providing valuable feedback.

I am also greatly thankful to my co-supervisor Dr. Saikat Chatterjee. Without Saikat, this thesis would have been a lot sparser. As a newer ending source of new ideas and suggestions, Saikat has helped me to see new directions and ask new questions about all things.

I want to express my gratitude to my highly skilled team of co-authors: Dr. Dave Zachariah for many discussions on a wide range of topics ranging from economy to cosmology, Dennis Sundman for his extensive computer skills and optimism, Adria Casamitjana for his insights on Spanish and Catalan culture, Dr. Kezhi Li for our low-rank matrix and review discussions and Cristian Rojas for many interesting thoughts on research and academia.

The members of the signal processing has made my stay fun, interesting and challenging. Thanks to the Professors Magnus Jansson, Peter Händel, Mats Bengtsson and Joakim Jaldén for keeping a well organized and friendly department, teaching interesting courses and good seminars. I am grateful to my fellow PhD students, past and present, for many interesting discussions, lunches and conferences. Great thanks to all of you. Fortunately, you are so many that I, unfortunately, cannot give each one of you the proper credit you deserve. I am also thankful of Tove Schwartz for help with the administrative details and Niclas and Pontus for the IT support.

I want to thank Professor Yoram Bresler of the University of Illinois, Urbana-Champaign for serving as opponent in my PhD dissertation. I am also thankful of Professor Maria Sandsten of Lund University, Professor Mats Viberg of Chalmers University of Technology and Professor Subhrakanti Dey of Uppsala University for serving in the thesis grading committee. Great thanks also to Dr. Johan Karlsson of KTH for our restaurant visits and for serving as replacement in the grading committee.

Finally, I am grateful for the support, inspiration and motivation I have received from my dear family, my dear parents and my dear brothers in all stages of my PhD studies, life and work.

Martin Sundin
Stockholm, August 2016

Contents

Contents	viii
1 Introduction	1
1.1 Thesis scope and contributions	2
1.2 Copyright notice	6
2 Background	7
2.1 Bayesian modeling	7
2.2 Sparsity	9
2.3 Robust methods	13
2.4 Low rank matrices	14
2.5 Robust principal component analysis	17
2.6 Bayesian Cramér-Rao bounds	18
2.7 Phase retrieval	19
3 Improving greedy pursuit methods	21
3.1 Beamforming for sparse recovery	25
3.2 Beamforming in the presence of noise	30
3.3 Bayesian filtering for greedy pursuits	32
3.4 Conditional prior based OMP	37
3.5 Computational complexity	39
3.6 Numerical evaluation	40
3.7 Conclusion	47
4 Outlier robust relevance vector machine	49
4.1 RVM for combined sparse and dense noise (SD-RVM)	51
4.2 SD-RVM for block sparse signals	55
4.3 Simulation experiments	56
4.4 Derivation of update equations	66
4.5 Conclusion	72
5 Relevance singular vector machine for low-rank matrix reconstruction	73

5.1	Low-rank matrix estimation	74
5.2	The one-sided precision based model	76
5.3	Two-sided precision based model	79
5.4	Practical algorithms	82
5.5	Simulation experiments	85
5.6	Conclusion	100
6	Bayesian learning for robust PCA	101
6.1	Robust principal component analysis	101
6.2	Robust RSVM	102
6.3	Numerical experiments	105
6.4	Conclusion	108
7	Bayesian Cramér-Rao bounds for low-rank matrix estimation	109
7.1	Introduction	109
7.2	Priors for low-rank matrices	112
7.3	Bayesian Cramér-Rao bounds for low-rank matrix reconstruction	115
7.4	Numerical experiments	125
7.5	Conclusion	145
8	Low-rank phase retrieval	147
8.1	Introduction	147
8.2	The low-rank phase retrieval problem	149
8.3	Low-rank PhaseLift	150
8.4	Extensions of low-rank phase retrieval	153
8.5	Experiments with random measurement matrices	157
8.6	Conclusion	159
8.7	Derivations and proofs	160
9	Fast solution of the ℓ_1-norm	173
9.1	Introduction	173
9.2	The geometry of basis pursuit	177
9.3	Greedy ℓ_1 -minimization	178
9.4	Numerical comparison	180
9.5	Derivations and proofs	185
9.6	Conclusion	188
10	Conclusion	189
	Bibliography	191

Introduction

Measurements and experience are important parts of all scientific activities. Even more important is the information and conclusions we extract from them. This importance is illustrated in several historical examples. For example, in 1901, a discouraged Wilbur Wright stated that “man would not fly in a thousand years” after their second design of a glider plane crashed in several trials [WKW02]. Even though the theory of flight was well understood, it was not known if the force would ever be enough to lift an airplane and cargo. One parameter in the lift equation which describes the mechanics of flight is the Smeaton coefficient, which relates speed and lifting force. In 1901, it was widely believed that the value of the Smeaton coefficient was 0.0054. With this value of the coefficient, the brother’s glider should be able to carry one man. After several crashes, the Wright brothers began to question the value of the Smeaton coefficient. They therefore began to build their own wind tunnel to estimate the coefficient. After several measurements, the Wright brothers determined the Smeaton coefficient to be closer to 0.0033. With this new value, they were able to construct a glider with better wings in 1902 and perform the first ever powered controlled flight in 1903. Without a better estimate of the Smeaton coefficient, the Wright brothers would probably never have made their flight and the development of aviation would have been delayed.

In earlier times, a main difficulty was to perform the actual experiments and measurements in order to obtain data. As “it is a capital mistake to theorize before one has data” [Doy94], data collection was often the main obstacle for many engineering and scientific problems. Today, experiments are becoming easier to perform with more and more ubiquitous (and cheaper) sensors. With easier data collection, the main problems today are instead the communication, storage and processing of data. To efficiently process data and measurements, it is necessary to construct numerical procedures, algorithms, which combine the data to give us the estimates and information we seek. The theory of such algorithms is commonly called estimation theory in signal processing, regression in machine learning and quantitative finance and inference in statistics. In this thesis, we consider the problem of designing algorithms for a certain classes of estimation problems.

Let x_1, x_2, \dots, x_n denote n parameters in our problem. The parameters can be organized in an n -dimensional vector $\mathbf{x} \in \mathbb{R}^n$ and the measurement process can be written as

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n},$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a known matrix representing the linear sensing operation, $\mathbf{n} \in \mathbb{R}^m$ is additive noise and $\mathbf{y} \in \mathbb{R}^m$ is the observed measurements. In many scenarios, the parameter vector \mathbf{x} has some special structure. For example, often only a few parameters are able to explain the majority of the data. This leads to a *sparse representation* where many elements of \mathbf{x} are zero. Can we exploit this knowledge to construct better estimation methods? It turns out that the answer is yes. The theory of exploiting sparsity and other structures is often called *Compressed Sensing* [CW08]. In this thesis we will consider different methods for estimation of sparse vectors and low-rank matrices. Further details is given in Chapter 2.

1.1 Thesis scope and contributions

This thesis investigates different estimation methods for sparse and low-rank problems. The estimations methods are typically grouped into three classes: greedy search methods, Bayesian methods and convex optimization based methods. We touch on each class in this thesis. The contribution can roughly be divided into three parts, each related to one class of algorithms. We first presents two methods for improving greedy pursuit methods in Chapter 3. Next we consider Bayesian estimation methods for sparse and low-rank problems in Chapters 4-7. Lastly we consider convex methods for a non-linear estimation problem in Chapter 8 and fast minimization of the ℓ_1 -norm in Chapter 9. We summarize the structure of the thesis in Table 1.1.

	Structure		Estimation method		
	Sparse	Low-Rank	Greedy	Bayesian	Convex
Chapter 2	X		X	X	
Chapter 4	X			X	
Chapter 5		X		X	
Chapter 6	X	X		X	
Chapter 7		X		X	
Chapter 8		X			X
Chapter 9	X		X		X

Table 1.1: Overview of the structures and methods used in the respective chapters.

Some of the results presented in the thesis have already been published in journals and conferences, and some are under review. Parts of the thesis are adopted from the corresponding research papers nearly verbatim.

Chapter 2

In Chapter 2 we give the background of the work. We present some modern engineering problems and show how they are related to the work of the thesis. We have tried to do so by including a minimal amount of mathematics and concentrate on the main concepts and ideas. If you merely wish to understand the context of the work and how it relates to other problems, this is the chapter for you.

Chapter 3

Greedy search algorithms are fast and computationally efficient algorithms for finding sparse representations. It is desirable to improve the performance of the algorithms without increasing the complexity (too much). In Chapter 3 we first consider how the algorithms detect non-zero coefficients and improve the performance using beamformer techniques from array signal processing. Next, we examine how Bayesian filtering methods can be used to improve detection and estimation by modeling the parameters as random variables. The chapter is based on

- [SSJ13] M. Sundin, D. Sundman and M. Jansson, Beamformers for sparse recovery. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5920-5924, Vancouver, Canada, May 2013.
- [SJC13] M. Sundin, M. Jansson and S. Chatterjee, Conditional prior based LMMSE estimation of sparse signals. In *Proceedings of the 21st European Signal Processing Conference (EUSIPCO)*, pages 1-5, Marrakech, Morocco, September 2013.

Chapter 4

When measurements are contaminated by outliers (in addition to the usual dense noise), estimation becomes more difficult. Many sparse estimation methods have been adapted to handle outliers by treating the outliers as part of a sparse parameter vector to be estimated. However, this procedure increases the complexity of the algorithms. It is therefore desirable to estimate the problem parameters without explicitly estimating the outliers. In Chapter 4 we show how the Bayesian estimation method the Relevance Vector Machine can be adapted for measurements with outliers without explicitly estimating the outliers. The chapter is based on

- [SCJ14] M. Sundin, S. Chatterjee and M. Jansson, Combined modeling of sparse and dense noise improves Bayesian RVM. In *Proceedings of the 22nd European Signal Processing Conference (EUSIPCO)*, pages 1841-1845, Lisbon, Portugal, September 2014.

- [SCJ15b] M. Sundin, M. Jansson and S. Chatterjee, Combined modeling of sparse and dense noise for improvement of Relevance Vector Machine. *Submitted journal paper*.

We note that [SCJ15b] is an extended journal version of [SCJ14].

Chapter 5

The low-rank reconstruction problem is closely related to the sparse recovery problem. However, a hierarchical Bayesian method, like the Relevance Vector Machine, has not been developed for the low-rank reconstruction problem. In Chapter 5 we develop such a method by introducing left and right *precision matrices*. We show how the prior distribution of the precision matrices is related to the prior distribution of the matrix to be estimated and compare the performance with existing algorithms through numerical simulations. The chapter is based on

- [SCJR14] M. Sundin, S. Chatterjee, M. Jansson, C.R. Rojas, Relevance Singular Vector Machine for low-rank matrix sensing. In *2014 International Conference on Signal Processing and Communications (SPCOM)*, Bangalore, India, July 2014.
- [SRJC] M. Sundin, S. Chatterjee, M. Jansson, C.R. Rojas, Relevance Singular Vector Machine for low-rank matrix reconstruction. *Accepted for publication in IEEE Transactions of Signal Processing*.

We note that [SRJC] is an extended journal version of [SCJR14].

Chapter 6

Principal Component Analysis (PCA) is an important method for finding underlying patterns in data and measurements. However, PCA is not robust to outlier noise in the data. To design estimation methods for robust PCA problem requires combining methods for sparse and low-rank problems. In Chapter 6 we combine the robust estimation technique from Chapter 4 and the low-rank estimation method from Chapter 5 to construct a Bayesian learning algorithm for robust PCA. The chapter is based on

- [SCJ15a] M. Sundin, S. Chatterjee and M. Jansson, Bayesian learning for robust principal component analysis. In *Proceedings of the 23rd European Signal Processing Conference (EUSIPCO)*, pages 2361 - 2365, Nice, France, September 2015.

Chapter 7

A fundamental tool in evaluating the performance of estimation algorithms is the theoretical lower bounds. The Cramér-Rao Bound (CRB) is a theoretical lower bound

for unbiased estimators of deterministic parameters. When the parameters to be estimated are random, the CRB does not hold in general since the prior distribution brings more information about the parameters. The Bayesian CRB (also called the van Trees inequality) is a lower bound for random parameters. In Chapter 7 we consider the Bayesian CRB for different low-rank matrix reconstruction problems. We show that the extension of the CRB to the Bayesian setting with random parameters is not unambiguous and that several different bounds can be derived. The chapter is based on

- [SCJa] M. Sundin, M. Jansson and S. Chatterjee, Bayesian Cramér-Rao bounds for factorized model based low rank matrix reconstruction. *Accepted to the European Signal Processing Conference (EUSIPCO) 2016.*
- [SCJb] M. Sundin, M. Jansson and S. Chatterjee, Bayesian Cramér-Rao bounds for low-rank matrix reconstruction. *In preparation.*

We note that [SCJb] is an extended journal version of [SCJa].

Chapter 8

In many scenarios, such as X-ray crystallography, the problem contains non-linear measurements. However, some problems can be transferred into non-linear problems in another variable. One such problem is the phase retrieval problem. The phase retrieval problem can be *lifted* to a positive semidefinite problem, by relaxing the problem one then obtains the convex optimization problem PhaseLift, which can readily be solved. The PhaseLift algorithm has been developed to sparse phase retrieval problems, but lacks a low-rank matrix analogue. In Chapter 8 we show how the low-rank phase retrieval problem can be lifted to a semidefinite program using the theory of Kronecker product approximations. The chapter is based on

- [SCJc] M. Sundin, M. Jansson and S. Chatterjee, Convex recovery for low-rank phase retrieval. *In preparation.*

Chapter 9

The power of convex relaxation techniques for sparse problems is that efficient *off-the-shelf* algorithms exist that can solve almost any convex optimization problem. It is thus not difficult to implement sparse estimation methods based on convex optimization. While the general methods are efficient, they can sometimes be beaten in performance by specialized methods that solve the problem in another way. In Chapter 9 we present such a method for the basis-pursuit problem. By considering the geometry of the basis pursuit method we are able to derive conditions for optimality of the solution. We then develop an algorithm based on these conditions. The method has the advantage of exploiting the sparsity of the solution and does therefore not need to keep all variables in memory. This makes the method suited

for problems with large number of variables. We illustrate this point by using the algorithm to find a wavelet decomposition of an image over all wavelet families in Matlab. The chapter is based on

- [SCJ15c] M. Sundin, M. Jansson and S. Chatterjee, Greedy minimization of the L1-norm with high empirical success. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3816-3820, Brisbane, Australia, April 2015.

Chapter 10

In the last chapter, we summarize the work presented in the thesis.

Contributions Outside the Scope of the Thesis

Besides the listed contributions, the author of this thesis has also contributed to other related works listed below.

- [ZSJC12] D. Zachariah, M. Sundin, M. Jansson and S. Chatterjee, Alternating least-squares for low-rank matrix reconstruction. In *IEEE Signal Processing Letters*, vol. 19, no. 4, pages 231-234, April 2012.
- [LSR⁺16] K. Li, M. Sundin, C.R. Rojas, S. Chatterjee and M. Jansson, Alternating strategies with internal ADMM for low-rank matrix reconstruction. In *Signal Processing*, vol. 121, pages 153-159, April 2016.
- [CSGC15] A. Casamitjana, M. Sundin, P. Ghosh, S. Chatterjee, Bayesian learning for time-varying linear prediction of speech. In *Proceedings of the 23rd European Signal Processing Conference (EUSIPCO)*, pages 325 - 329, Nice, France, September 2015.
- [SVJC] M. Sundin, A. Venkitaraman, M. Jansson, S. Chatterjee, A convex constraint for graph connectedness. *Conference paper. In preparation.*

1.2 Copyright notice

Parts of the material presented in this thesis are partly verbatim based on the thesis author's joint works which are previously published or submitted to conferences and journals held by or sponsored by the Institute of Electrical and Electronics Engineer (IEEE). IEEE holds the copyright of the published papers and will hold the copyright of the submitted papers if they are accepted. Materials (e.g., figure, graph, table, or textual material) are reused in this thesis with permission.

Background

In signal processing, one important task is to extract a signal from noisy measurements. This is usually done by 1) modeling the noise, 2) modeling the signal and 3) using the models to construct an algorithm (an estimator) to extract the information from the data. How accurate we manage to extract the information naturally depends on how accurate and flexible our models are. A very precise model can be very efficient if it is correct and very poor if it is incorrect, while a very flexible model can account for many different models but may not be as accurate. In this thesis we will mainly discuss a combination of two different models, Bayesian models and parsimonious models. Bayesian modeling is a way to model prior knowledge and uncertainty using probability theory. Using Bayesian methods, it is possible both to extract information and quantify the uncertainty of the information. Parsimonious models are models where the *information content* is a fraction of the signal content, i.e. the signal is strongly redundant and can be compressed a lot without losing any information. Many natural signals are parsimonious, something which makes them easier to extract. Here we especially consider sparse and low-rank models. We will show how Bayesian methods can be used to handle different parsimonious models.

2.1 Bayesian modeling

Probability theory allows us to calculate the probability of an event and it also allows us to calculate the probability of a second event given that a first event has occurred. This is commonly known as conditional probability. Bayes rule allows us to reverse conditional probabilities, allowing us to compute the probability that the first event happened given that we observed the second event. Let $P(A)$ be the probability of an event A , $P(B)$ be the probability of an event B and $P(B|A)$ be the probability of event B given that A has occurred. Bayes rule allows us to compute the reverse probability $P(A|B)$, i.e. the probability that A did occur given

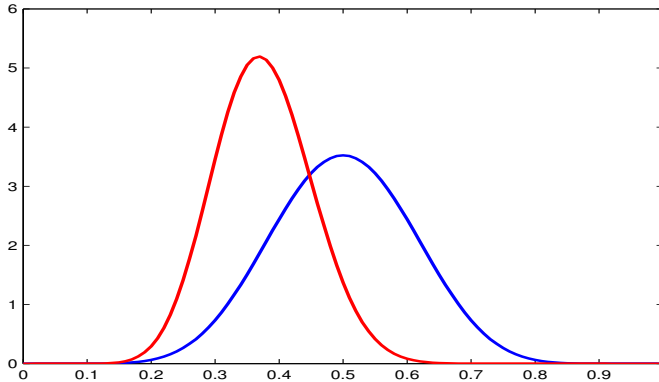


Figure 2.1: The prior (in blue) and the posterior (in red) probability of getting heads when flipping a coin. The prior suggests that the coin most probably is balanced while the posterior (given after observing $N_h = 5$ heads and $N_t = 15$ tails) suggests that the coin is biased.

that we observed the event B , as

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}.$$

Bayes rule is useful since it allows us to find the most probable cause of an event.

Consider the problem of deciding whether a coin is balanced or not. Assume that the probability of heads is q , then the probability of tails is $1 - q$. We have a high degree of confidence that the coin is balanced ($p = \frac{1}{2}$). We model this prior knowledge by assigning a prior distribution $p(q)$ to the probability q . After we observe N_h heads and N_t tails in $N_h + N_t$ independent trials, Bayes rule gives us that the posterior distribution of q is

$$p(q|N_h, N_t) = \frac{p(\text{heads}|q)^{N_h} p(\text{tails}|q)^{N_t} p(q)}{\int_0^1 p(\text{heads}|q)^{N_h} p(\text{tails}|q)^{N_t} p(q) dq} = \frac{q^{N_h} (1 - q)^{N_t} p(q)}{\int_0^1 q^{N_h} (1 - q)^{N_t} p(q) dq}.$$

In the coin-flipping example, it is common to use a Beta-distribution as a prior distribution. An example of a prior and posterior distribution is shown in Figure 2.1.

Bayesian methods are useful even when the prior knowledge is weak. In that case, the prior is often chosen to be as *non-informative* as possible. This can be done by selecting the prior to be *approximately flat* or by using the *Jeffries prior* which is invariant under change of variables. Prior distribution for which the posterior is of the same type as the prior distribution are called *conjugate priors*. Conjugate priors are useful since statistical inference can be performed more easily than for non-conjugate priors.

Sometimes we want to choose a prior distribution in which the parameters of the prior distribution are themselves random variables. Such priors are called *hierarchi-*

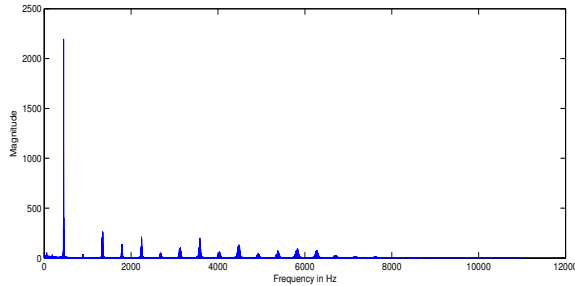


Figure 2.2: The energy in each frequency of a violin tone.

cal priors since the model consists of several *layers* of random variables. Hierarchical priors are often very flexible and can thus model several different distributions. To do inference in hierarchical priors often requires approximate inference methods.

2.2 Sparsity

Many tones of musical instruments have their energy concentrated to just a few frequencies, for example, a violin tone has its energy concentrated to one main frequency with residual energy in several smaller overtones, see Figure 2.2. The violin tone thus has an approximately sparse representation in the frequency domain. Other instruments also have sparse representations, for example the beats of a drum can be sparsely represented in the time domain. Most natural signals have sparse representations in some domain.

The fact that sparse signals contain less information than an arbitrary signal can be exploited to reconstruct the signals also when the number of measurements is not sufficient for standard reconstruction techniques. One example of this is from Magnetic Resonance Imaging (MRI). In MRI, a sample (e.g. a part of the body) is exposed to a magnetic field which varies in space. The magnetic field causes the magnetic moments of the hydrogen atoms to align with the magnetic field. The atoms are then exposed to a magnetic pulse which excites the magnetic moments. When the atoms relax back into equilibrium they emit radiation with frequency proportional to the strength of the magnetic field. This produces a signal whose amplitude is proportional to the density of hydrogen atoms. By computing the Fourier transform of the signal we can find the average concentration of hydrogen atoms in the area which has the same magnetic field strength. The measurement process thus gives a sub-sampled Fourier transform of the image. Each measurement takes about a half second, making MRI imaging a time-consuming process (imaging the brain takes about 20–45 minutes). The many measurements needed for MRI makes it hard to apply to sensitive parts of the body. It is therefore desirable to reduce the number of measurements needed.

An often used toy-model for MRI is the Shepp-Logan phantom [SL74] shown in



Figure 2.3: (a) The Shepp-Logan phantom. (b) Total variation of the Shepp-Logan phantom.

Figure 2.3 (a). The phantom image is used since it shares many of the properties of standard MRI images. The figure is not sparse, but has large areas where the intensity/color is constant. This means that the total variation (local difference of pixel values), shown in Figure 2.3 (b), is sparse. Real MRI images have similar sparsity since the body consists of regions with different tissue (e.g. muscle, organ and bones). A MRI image can thus be reconstructed by finding the image with the sparsest total variation which give rise to the observed measurements.

The image can be represented by a matrix $\mathbf{X} \in \mathbb{R}^{p \times q}$ in which the components $X_{i,j}$ represent the gray-scale intensity at pixel (i, j) . The total variation of the image is a matrix $\mathbf{V}(\mathbf{X})$ with elements

$$V_{ij}(\mathbf{X}) = \sqrt{(X_{i,j} - X_{i,j+1})^2 + (X_{i,j} - X_{i+1,j})^2}.$$

The problem of finding the image with the sparsest total variation can thus be expressed as the optimization problem

$$\begin{aligned} \hat{\mathbf{X}} &= \arg \min_{\mathbf{X}} \|\mathbf{V}(\mathbf{X})\|_0 \\ &\text{subject to } Y_{i,j} = \text{DFT}_2(\mathbf{X})_{i,j}, \text{ for all } (i, j) \in \Omega \end{aligned} \quad (2.1)$$

where $\hat{\mathbf{X}}$ is the reconstructed image, $\text{DFT}_2(\mathbf{X})_{i,j}$ denotes the (i, j) component of the image Fourier transform, Ω is the set of observed Fourier components and

$$\|\mathbf{V}\|_0 = |\{(i, j) : V_{i,j} \neq 0\}|$$

is the ℓ_0 -norm of \mathbf{V} (the number of non-zero components). The ℓ_0 -norm is not a norm in the proper mathematical sense but serves as a useful notation. A problem with the optimization problem (2.1) is that the ℓ_0 -norm is a hard to minimize, i.e. we often need to try every possible combination to find the minimum. This requires too much time, so the combinatorial solution is not useful in practice. An alternative is to instead minimize a function which approximates the ℓ_0 -norm, common choices

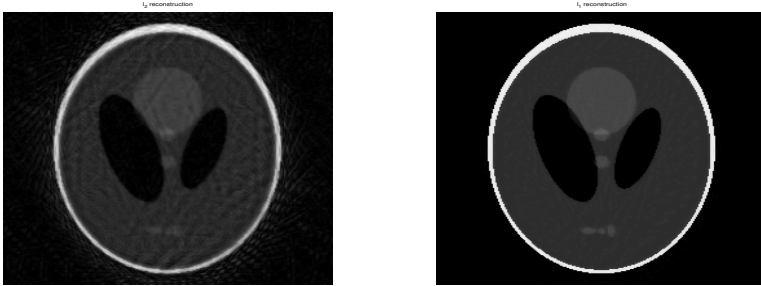


Figure 2.4: Reconstruction of the Shepp-Logan phantom from MRI measurements using the ℓ_2 -norm and ℓ_1 -norm.

are the ℓ_1 -norm and ℓ_2 -norm

$$\|\mathbf{V}\|_1 = \sum_{i,j} |V_{i,j}|, \quad \|\mathbf{V}\|_2 = \sqrt{\sum_{i,j} |V_{i,j}|^2}.$$

The advantage of using the ℓ_1 and ℓ_2 -norm is that the problem becomes convex and can thus be solved using standard methods from convex optimization. In Figure 2.4 we show the reconstructed image using the ℓ_2 and ℓ_1 -norm. We find that while both images resemble the original, the image reconstructed using the ℓ_2 -norm has several artifacts and the image reconstructed using the ℓ_1 -norm is very close to the original. The difference is because the total variation is sparse and the ℓ_1 -norm better promotes sparsity than the ℓ_2 -norm.

The general sparse reconstruction problem can be described as follows. Let $\mathbf{x} \in \mathbb{R}^n$ be a sparse parameter vector and assume that we linearly measure \mathbf{x} as

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}, \quad (2.2)$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a known measurement matrix and \mathbf{n} is additive noise. It is typically assumed that the noise is i.i.d. Gaussian. The problem is to recover \mathbf{x} from \mathbf{y} . When \mathbf{x} is not sparse, we in general need more measurements than parameters, i.e. $m \geq n$, to reconstruct \mathbf{x} while sparse \mathbf{x} can be reconstructed also when $m < n$.

To find the sparsest solution, we need to minimize the ℓ_0 -norm of \mathbf{x} , i.e. the number of non-zero components. The solution of trying every combination, an exhaustive search, is often infeasible since it takes too long time. For this reason, several other methods have been developed for finding sparse solutions. The methods are often classified as convex optimization based, greedy search based methods and Bayesian methods.

Convex optimization based methods formulates the estimation problem as an optimization problem. This is done by making the residual $\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2$ small while simultaneously minimizing a function $g(\mathbf{x})$ which is “small” when \mathbf{x} is sparse. A common choice is to use the ℓ_1 -norm where $g(\mathbf{x}) = \|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|$, the estimator

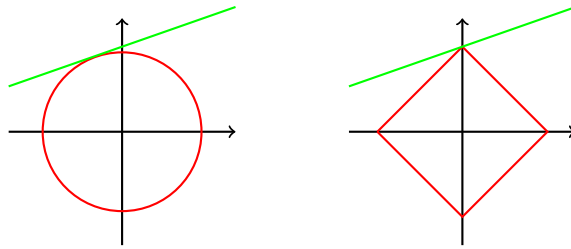


Figure 2.5: The intuition to why the ℓ_1 -norm gives a sparse solution. The green line shows the values of \mathbf{x} such that $\mathbf{y} = \mathbf{A}\mathbf{x}$ and the red lines shows the minimal ℓ_2 and ℓ_1 -ball that intersect the green line. The point of intersection is the resulting estimate. We see that the ℓ_1 -norm is more likely to recover a sparse solution.

is then often called Basis Pursuit Denoising (BPDN) [CRT06]. The problem can be formulated as

$$\min \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda g(\mathbf{x}) \quad (\text{Tikhonov regularization})$$

$$\begin{aligned} &\min g(\mathbf{x}), \\ &\text{subject to } \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2 \leq \epsilon \end{aligned} \quad (\text{Morozov regularization})$$

$$\begin{aligned} &\min \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2, \\ &\text{subject to } g(\mathbf{x}) \leq \delta \end{aligned} \quad (\text{Ivanov regularization})$$

where $\lambda, \epsilon, \delta > 0$ are positive constants. For the ℓ_1 -norm, the Tikhonov regularization is often referred to as the LASSO estimator [Tib96] (for the Least Angle Shrinkage and Selection Operator). The reason for using the ℓ_1 -norm, rather than the ℓ_2 -norm is that the ℓ_1 -norm is more likely to recover a sparse solution, as shown in Figure 2.5. This argument can be made mathematically precise [CRT06].

When the sparsity, $\|\mathbf{x}\|_0 = K$, is known, a good alternative to optimization based methods is greedy search methods. A greedy search is a method which adds elements sequentially by making a greedy decision in each iteration. Greedy methods are usually much faster than optimization based methods because of their lower complexity. To increase the accuracy often means resorting to more computationally complex methods that are more time demanding. It is desirable to find methods to improve the accuracy of greedy search methods without increasing the computational complexity. In Chapter 3, we will show a method to improve the accuracy of greedy search methods, *without* increasing their computational complexity. We will also relate the approach to Bayesian filtering methods.

When neither the sparsity nor noise power is known, Bayesian methods are often preferable. Bayesian methods model sparsity by assigning prior distributions to the

parameters and the noise and then iteratively updating the distributions to obtain a good estimate. Bayesian methods are often able to *learn* both the sparsity and noise power from data alone.

2.3 Robust methods

Sparse reconstruction is closely related to robust estimation methods. In the measurement model (2.2), it was assumed that the noise components have the same variance, i.e. the noise is the same in all measurements. However, in many scenarios, some noise components are very large (outliers). This can severely perturb the final estimate. To perform estimation from measurements with outliers requires robust estimation methods.

Outliers often occur when some datapoints are not well described by the model. Consider, for example, the problem of predicting house prices. It is plausible to assume that the house price increase with the number of rooms. In Figure 2.6 we show the house prices versus the average number of rooms from the Boston housing dataset [AN07]. The presence of outliers show that other factors also influence the house price. Using normal regression methods (least squares) we obtain the red line in the figure while removing many outliers gives the green line. Since the lines differ we find that the outliers affect our prediction and that a better prediction can be made when taking the outliers into account. By removing the outliers, the trend only models the majority of house prices and not the outliers.

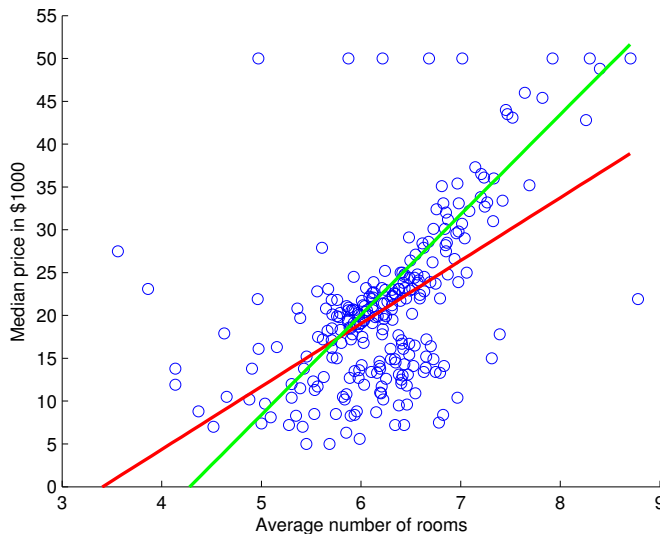


Figure 2.6: Predicting the trend of house prices from the Boston housing dataset. Not taking outliers into account gives the red line of regression, while taking outliers into account gives the green line of regression. The green line better shows the trend in house prices.

The model for measurements with outliers can be written as

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e} + \mathbf{n}, \quad (2.3)$$

where $\mathbf{e} \in \mathbb{R}^m$ is a sparse vector containing the outliers, $\mathbf{n} \in \mathbb{R}^m$ is (dense) measurement noise, $\mathbf{y} \in \mathbb{R}^m$ is the observed measurements, $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a known measurement matrix and $\mathbf{x} \in \mathbb{R}^n$ is the parameter vector of interest. Since the number of outliers is small, it is natural to use sparsity seeking methods. A standard approach is to concatenate \mathbf{x} and \mathbf{e} into a single vector and estimate the full vector using sparsity seeking methods. In chapter 4 we will introduce a Bayesian method for estimating \mathbf{x} *without* estimating \mathbf{e} .

2.4 Low rank matrices

Another parsimonious model is low-rank matrices. The rank of a matrix is the number of linearly independent column (or row) vectors of the matrix. A low-rank matrix is thus a matrix where the columns can be represented as linear combinations of a low number of (unknown) basis vectors. This can be interpreted as that there are a low number of factors which explain/describe the data in the matrix. Low rank matrices are used in many applications such as system identification [Faz02, ZSJC12], localization [CP10] and recommender systems [KBV09].

Recommender systems analyze the preferences of customers and try to recommend products the customers might like. Such systems are used by the online retailer Amazon, the movie streaming service Netflix and many others. The recommendation problem is to predict the ratings users are likely to give unseen products, i.e. find ratings to products the user has not viewed. Finding a high missing rating means that the product is likely to be bought by the customer. By recommending the product to the customer it is therefore possible to make a sale and earn money. A small example is shown in Table 2.1. We see that user 1 probably prefers product 2 since user 1 is *similar* to user 2, it is therefore good to recommend product 3 to user 1. Recommendation systems formalize the notion similarity so that recommendations can be made automatically by a computer.

The main assumption of many recommendation systems is that the user-product matrix of ratings is low-rank. This is because a person often prefers a product based on some features such as e.g. genre, actors or director in the case of movies. The ratings are thus modeled as

$$[\text{user } i\text{'s rating of product } j] = \sum_{k=1}^r [i\text{'s rating of feature } k] \cdot [\% \text{ of } j \text{ in feature } k].$$

The main advantage of this model, compared to e.g. content based recommendations, is that the features do not need to be known a-priori. The system can thus learn $[i\text{'s rating of feature } k]$ and $[\% \text{ of } j \text{ in feature } k]$ for each user, product and feature in order to find the missing ratings. Using user-product ratings to infer

	Product 1	Product 2	Product 3	Product 4
User 1	1	?	?	5
User 2	2	5	?	5
User 3	1	?	3	?
User 4	1	4	?	?
User 5	?	5	2	?

Table 2.1: A user-product rating matrix with ratings (1 – 5) and unknown ratings (question marks). The goal of recommender systems is to find high missing rating so that the product can be recommended to the user.

unseen ratings is commonly called *collaborative filtering*. The key assumption in collaborative filtering is to assume that the number of features is small.

The general low-rank matrix reconstruction (LRMR) problem is to recover a low-rank matrix $\mathbf{X} \in \mathbb{R}^{p \times q}$ with elements X_{ij} (where $1 \leq i \leq p$ and $1 \leq j \leq q$) from measurements

$$y_k = \sum_{i=1}^p \sum_{j=1}^q A_{kij} X_{ij} + n_k \quad (2.4)$$

where $1 \leq k \leq m$, the coefficients A_{kij} are known and n_k is additive noise. In collaborative filtering, each matrix \mathbf{A}_k chooses a single element from \mathbf{X} , this is often called *matrix completion*. The LRMR problem (2.4) is a more general problem and contains matrix completion as an important special case. Similarly as for sparsity, the rank is hard to minimize directly. It is therefore common to instead minimize a penalty function which approximate the rank function. Rank can be related to sparsity through the singular value decomposition (SVD). The SVD of a matrix $\mathbf{X} \in \mathbb{R}^{p \times q}$ is a factorization

$$\mathbf{X} = \mathbf{U} \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \vdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \sigma_k \end{bmatrix} \mathbf{V}^\top,$$

where $k = \min(p, q)$, the matrices \mathbf{U} and \mathbf{V} are unitary and the parameters $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k \geq 0$ are called the *singular values*. The rank of a matrix equals the number of non-zero singular values, so promoting sparsity in the singular values is equivalent to promoting low-rank. The LRMR problem can thus be written as e.g.

$$\min g(\mathbf{X}), \quad \text{subject to } \|\mathbf{y} - \mathbf{A}\text{vec}(\mathbf{X})\|_2^2 \leq \epsilon$$

	Product 1	Product 2	Product 3	Product 4
User 1	1	5.0	1.9	5
User 2	2	5	2.3	5
User 3	1	2.7	3	2.7
User 4	1	4	1.6	4.0
User 5	1.3	5	2	5.0

Table 2.2: Completion of the user-product rating matrix in 2.1 using the Nuclear norm.

To promote low-rank, different penalty functions $g(\mathbf{X})$ can be used such as

$$g(\mathbf{X}) = \sum_{i=1}^{\min(p,q)} \sigma_i(\mathbf{X}) = \text{tr}((\mathbf{X}\mathbf{X}^\top)^{1/2}), \quad (\text{Nuclear norm})$$

$$g(\mathbf{X}) = \sum_{i=1}^{\min(p,q)} (\sigma_i(\mathbf{X})^2 + \epsilon)^{s/2} = \text{tr}((\mathbf{X}\mathbf{X}^\top + \epsilon\mathbf{I}_p)^{s/2}), \quad (\text{Schatten } s\text{-norm})$$

$$g(\mathbf{X}) = \sum_{i=1}^{\min(p,q)} \log(\sigma_i(\mathbf{X})^2 + \epsilon) = \log \det(\mathbf{X}\mathbf{X}^\top + \epsilon\mathbf{I}_p). \quad (\text{log-determinant})$$

The “ball” of the penalty functions (the set of matrices such that $g(\mathbf{X}) = \text{constant}$) are illustrated in Figure 2.7 for $\mathbf{X} = \text{diag}(x, y)$. We find that the Schatten s -norm reduces to the Nuclear norm when $s = 1$ and $\epsilon = 0$. One advantage of the Nuclear norm is that it is convex and thus has a unique minima. It can be shown that the Nuclear norm recovers the true matrix \mathbf{X} with high probability from a sufficient number of random measurements [CP10]. For example, by performing Nuclear norm minimization for the product recommendation problem in Table 2.1, we obtain the product predictions in Table 2.2. We find that, as expected, product 2 should be recommended to user 1.

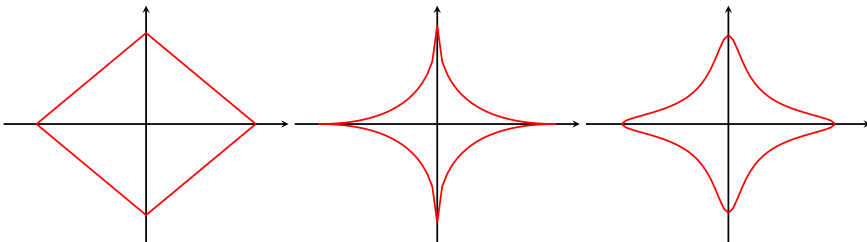


Figure 2.7: The “balls” of the Nuclear norm, the Schatten-0.5 norm and the log-determinant penalty for $\mathbf{X} = \text{diag}(x, y)$.

One disadvantage of the Nuclear norm is that the noise power needs to be known a-priori. When the noise power is unknown, Bayesian methods are preferable since they can learn the noise power from the data. In Chapter 5 we will present one Bayesian method for LRMR where the distributions of the hyper-parameters can be related to certain penalty functions.

2.5 Robust principal component analysis

An important special case of low-rank matrix reconstruction is robust Principal Component Analysis (PCA). In regular PCA, we measure all elements of a low-rank matrix as

$$\mathbf{Y} = \mathbf{X} + \mathbf{N},$$

where $\mathbf{Y} \in \mathbb{R}^{p \times q}$ is the observed measurements, \mathbf{N} is additive measurement noise and $\mathbf{X} \in \mathbb{R}^{p \times q}$ is the low-rank matrix of interest. PCA can be interpreted as extracting the *most informative features* of a dataset. Finding the mean of the dataset means finding a special rank-1 approximation while a rank- r approximation is related to finding the r *most informative deviations* from mean (principal components). Consider, for example, the hand-written 5's from the MNIST dataset [LCB98] in Figure 2.8. By stacking the vectorized images we obtain a matrix with the singular values shown in Table 2.3. We find that the 25 first (out of 784) singular values contain 85% of the total squared Frobenius norm of the matrix, this can be interpreted as that 85% the information is contained in the 25 first singular vectors. Calculating the 5 first principal components we find the images shown in Figure 2.9. We see that the first image is similar to the mean of the dataset while the other images shows the most common deviations from the mean.

However, PCA is not a robust technique, meaning that outlier noise can severely perturb the singular vectors, thus distorting the result. To design robust methods for PCA thus requires combining low-rank and sparse methods. The measurement

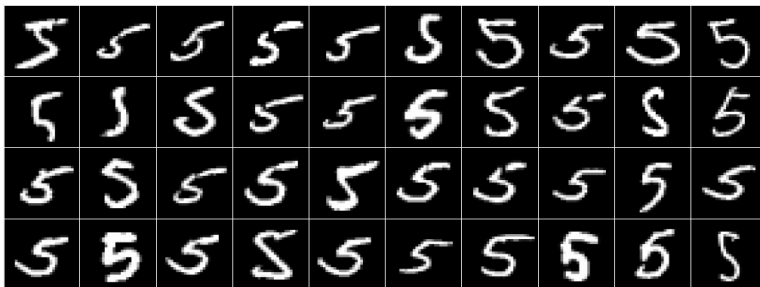


Figure 2.8: Handwritten 5's from the MNIST dataset.

k	1	5	10	25
fraction	48%	67%	75%	85%

Table 2.3: The fraction of the squared Frobenius norm contained in the first k singular values. The full dataset has 784 singular values.

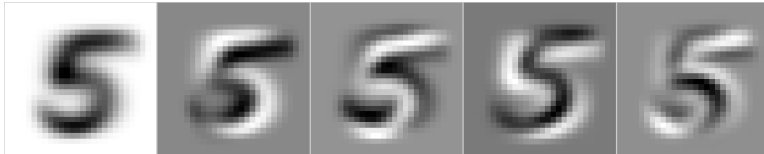


Figure 2.9: First 5 singular vectors of the 5's from the MNIST dataset.

model can be expressed as

$$\mathbf{Y} = \mathbf{X} + \mathbf{S} + \mathbf{N} \in \mathbb{R}^{p \times q},$$

where \mathbf{Y} is the observed measurements, \mathbf{X} is a low-rank matrix, \mathbf{S} is a sparse matrix containing the outliers and \mathbf{N} is dense additive noise. The problem has similarities with the matrix completion problem since the values of some components are very noisy. However, unlike in the matrix completion problem, the positions of the noisy elements are unknown in the robust PCA model. In Chapter 6 we discuss robust PCA in more detail and present a Bayesian estimation method.

2.6 Bayesian Cramér-Rao bounds

In this thesis, we seek to develop estimation methods which makes the error as small as possible. When developing new estimators, is it possible to indefinitely decrease the error by developing better and better estimation techniques? It turns out that the answer is negative. There exists theoretical limits to how well a parameter can be estimated. The Mean Squared Error (MSE) of an estimator is the expected square error of the estimate. It is a sum of its squared bias and variance as

$$\text{MSE} = \text{bias}^2 + \text{variance}.$$

The bias is the difference between the average (numerical) answer of the estimator and the true answer and the variance is the average squared deviation from the average answer. A high bias occurs when the estimation method has a large systematic error. A high variance occurs when the method is sensitive to measurement noise. An estimator with zero bias is called *unbiased*.

Assume that we want to estimate a parameter x from a measurement y . Because of the noise, y is a random variable with distribution $p(y|x)$, and the estimate $\hat{x} = \hat{x}(y)$ is also a random variable (since it depends on y). How well can we estimate

x ? If the estimator $\hat{x}(y)$ is unbiased, the Cramér-Rao bound (CRB) [Kay98] gives the following lower bound on the MSE,

$$\text{MSE} = \text{variance} \geq \text{CRB} = \frac{1}{J_y}, \quad (2.5)$$

where J_y is the Fisher information

$$J_y = \mathcal{E} \left[\left(\frac{\partial \log p(y|x)}{\partial x} \right)^2 \right].$$

When the parameters are random, the CRB is no longer a valid bound since the prior distribution $p(x)$ give additional information about x . A bound for random parameters is given by the Bayesian Cramér-Rao bound (BCRB) which takes the prior distribution into account. The BCRB is given by

$$\text{MSE} = \text{variance} \geq \text{BCRB} = \frac{1}{J_y + J_x}, \quad (2.6)$$

where J_x is given by

$$J_x = \mathcal{E} \left[\left(\frac{\partial \log p(x)}{\partial x} \right)^2 \right],$$

where the expected value is taken with respect to the distribution $p(y, x) = p(y|x)p(x)$. The BCRB is also known as the van-Trees inequality and the Borovkov-Sakhanenko inequality. The CRB (2.5) and BCRB (2.6) have multivariate counterparts for the estimation of several variables. In Chapter 7, we compute Bayesian Cramér-Rao bounds for different models of random low-rank matrices.

2.7 Phase retrieval

In many scenarios the measurements are non-linear. Non-linear problems are often harder to solve and require different estimation techniques. One example of non-linear measurements is X-ray crystallography where a molecule or crystal is illuminated by X-rays and a diffraction pattern is measured as shown in Figure 2.10. In the process, the amplitude of the measurement is recorded, but the phase information is lost. Since finding the true parameters is related to finding the phase of the measurements, this estimation problem is often called *phase retrieval*.

Phase retrieval is traditionally solved by iteratively estimating the parameters and the phases. The disadvantage of the traditional methods is that they require many measurements to perform well. A more modern method is to *lift* the non-linear problem to a linear problem with rank constraints. As before, we can approximate the constraints by penalty functions to give a problem we can solve numerically. This solution method is called PhaseLift. PhaseLift can be adapted to sparse parameters

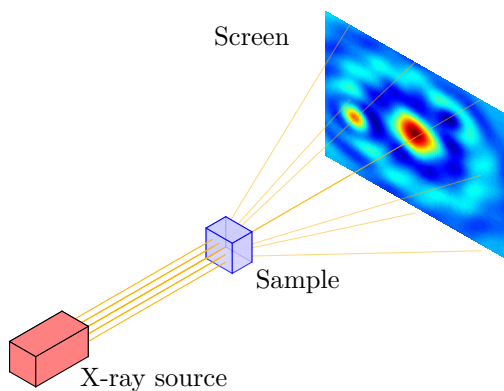


Figure 2.10: In X-ray diffraction, a sample is exposed to X-ray radiation and a diffraction pattern is measured. The diffraction pattern gives information about the atomic structure of the sample.

without problems since the lifting procedure preserves the sparsity. However, it does not work when the parameters comprise a low-rank matrix. In Chapter 8, we will show how PhaseLift can be adapted to low-rank matrices.

Improving greedy pursuit methods

Greedy pursuits are fast and effective methods for finding sparse representations. Even though the methods are sometimes less accurate than convex optimization based methods, their simplicity and speed often make them preferable for many problems. Greedy method finds a solution by iteratively make a choice that gives the largest gain in the present iteration. Because of their lower accuracy, it is desirable to improve the performance of greedy pursuit methods while not decreasing the speed (too much).

In this chapter we present two methods for improving the performance of greedy search methods. Both methods modify how the algorithm detects non-zero entries. The first method is deterministic and models the problem as an array processing problem while the second method uses a Bayesian approach and models the problem as a stochastic filtering problem. The methods are shown to be equivalent in a certain limit.

The goal of sparse reconstruction algorithms is to recover a sparse vector $\mathbf{x} \in \mathbb{R}^n$ from measurements

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}, \quad (3.1)$$

where $\mathbf{A} = [\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_n] \in \mathbb{R}^{m \times n}$ is the sensing matrix, $\mathbf{n} \in \mathbb{R}^m$ is additive noise and $\mathbf{x} \in \mathbb{R}^n$ is a sparse vector. We here assume that the sparsity

$$\|\mathbf{x}\|_0 = |\{i|x_i \neq 0\}| = K.$$

is known a-priori. This assumption is necessary for many greedy search algorithms in order to know when to stop the algorithm.

3.0.1 Exhaustive search

When the sparsity is known, the sparse reconstruction problem can be written as the optimization problem

$$\begin{aligned} \min_{\mathbf{x}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2 \\ \text{s.t. } \|\mathbf{x}\|_0 \leq K \end{aligned} \quad (3.2)$$

When the support set I is known, the solution is given by the least squares estimate

$$\begin{aligned}\hat{\mathbf{x}}_I &= \mathbf{A}_I^+ \mathbf{y}, \\ \hat{\mathbf{x}}_{I^c} &= \mathbf{0},\end{aligned}$$

where $I^c = [n] \setminus I$ is the complement set of I . The problem (3.2) can be solved by trying all possible support sets I such that $|I| = K$ and chose the solution which gives the smallest residual. This strategy is commonly called the *exhaustive search* and requires solving $\binom{n}{K}$ systems of equations. The exhaustive search is typically too slow to be useful in practice. Faster methods, such as greedy search methods, are therefore often used to solve the sparse approximation problem in reasonable time.

3.0.2 Orthogonal Matching Pursuit

One effective greedy pursuit method is Orthogonal Matching Pursuit (OMP) [TG07]. The method works by iteratively detecting non-zero components and estimating their value using least squares. In the first iteration, the algorithm estimates the support set to be the empty set, $I = \emptyset$, and the residual to be the measurements, $\mathbf{r} = \mathbf{y}$. OMP then adds the index which best describes the residual to the support set as

$$\begin{aligned}\hat{i} &= \arg \min_i \min_{x_i} \|\mathbf{r} - \mathbf{a}_i x_i\|_2 = \arg \max_i |\mathbf{a}_i^\top \mathbf{r}|, \\ I \cup \{\hat{i}\} &\rightarrow I,\end{aligned}\tag{3.3}$$

The method (3.3) for detecting non-zero components uses a *matched filter*. The non-zero components are estimated using least squares estimation as

$$\hat{\mathbf{x}}_I = \arg \min_{\mathbf{x}_I} \|\mathbf{y} - \mathbf{A}_I \mathbf{x}_I\|_2^2.$$

The steps are repeated until the support set contains K elements, $|I| = K$. The algorithm can be written as in Algorithm 1.

Data: $\mathbf{y}, \mathbf{A}, K$.

Initialization: $\mathbf{r} = \mathbf{y}, \hat{I} = \emptyset, \hat{\mathbf{x}}_I = \mathbf{0}$

while $|\hat{I}| < K$ **do**

$\hat{i} = \arg \max_i \mathbf{a}_i^\top \mathbf{r} $
$I \cup \{\hat{i}\} \rightarrow I$
$\hat{\mathbf{x}}_I = \mathbf{A}_I^+ \mathbf{y}$
$\mathbf{r} = \mathbf{y} - \mathbf{A}_I \hat{\mathbf{x}}_I$

end

Result: Estimated support set, \hat{I} , and components, $\hat{\mathbf{x}}_I$.

Algorithm 1: The OMP algorithm.

OMP is an extension of the Matching Pursuit algorithm [MZ93] which, unlike OMP, does not use least square estimation. Because in each iteration, the residual, \mathbf{r} , becomes orthogonal to the prediction, $\mathbf{A}_I \hat{\mathbf{x}}_I$, OMP is an orthogonal version of matching pursuit (hence the name). Several improvements to the OMP algorithm has been proposed, see e.g. [WS12, SM15, CSS11, SCS12, CSVS12, CSVS11, BD08, YdH15, RG09, SGIH13, SACH14, WS11, GAH98, RNL02].

How well OMP is able to recover a sparse vector depends on the sensing matrix \mathbf{A} . If the column vectors are close together, it is harder to find which vector is active. How close the column vectors are can be measured by the *mutual coherence*.

Definition 3.1. The mutual coherence, $\mu(\mathbf{A})$, of a sensing matrix \mathbf{A} (with column vectors of unit ℓ_2 -norm) is the maximum absolute inner product of two column vectors of \mathbf{A} , i.e.

$$\mu(\mathbf{A}) = \max_{i \neq j} |\mathbf{a}_i^\top \mathbf{a}_j|.$$

We see that when the mutual coherence is small, the column vectors of \mathbf{A} are nearly orthogonal. It then becomes easier to decompose \mathbf{y} as a linear combinations of atoms in \mathbf{A} . When the mutual coherence is large, some column vectors are close together and it becomes harder to distinguish which vector is active. This can be formulated through the following theorem.

Theorem 3.0.1. Let $\mu(\mathbf{A})$ be the mutual coherence of the sensing matrix \mathbf{A} . If

$$K < \frac{1}{2} \left(1 + \frac{1}{\mu(\mathbf{A})} \right), \quad (3.4)$$

then OMP recovers all K -sparse vectors exactly from measurements $\mathbf{y} = \mathbf{A}\mathbf{x}$.

Proof. We can assume that the non-zero components of \mathbf{x} are x_1, x_2, \dots, x_K and that the first component has the largest absolute value. We write the measurements, \mathbf{y} , as

$$\mathbf{y} = \mathbf{A}\mathbf{x} = \sum_{k=1}^K \mathbf{a}_k x_k.$$

The OMP algorithm recovers a component in the support set, I , if

$$\max_{1 \leq i \leq K} \left| \mathbf{a}_i^\top \left(\sum_{k=1}^K \mathbf{a}_k x_k \right) \right| > \max_{K+1 \leq j \leq n} \left| \mathbf{a}_j^\top \left(\sum_{k=1}^K \mathbf{a}_k x_k \right) \right|, \quad (3.5)$$

Using the triangle inequality, we can bound the left-hand side of (3.5) from below as

$$\begin{aligned} & \max_{1 \leq i \leq K} \left| \mathbf{a}_i^\top \left(\sum_{k=1}^K \mathbf{a}_k x_k \right) \right| \\ & \geq \max_{1 \leq i \leq K} |\mathbf{a}_i^\top \mathbf{a}_1| \cdot |x_1| - \sum_{k=2}^K |x_k| \cdot |\mathbf{a}_i^\top \mathbf{a}_k| \\ & \geq |x_1| - |x_1| \mu(\mathbf{A}) K. \end{aligned} \quad (3.6)$$

Similarly, we can bound the right-hand side of (3.5) from above as

$$\begin{aligned} & \max_{K+1 \leq j \leq n} \left| \mathbf{a}_j^\top \left(\sum_{k=1}^K \mathbf{a}_k x_k \right) \right| \\ & \leq \max_{K+1 \leq j \leq n} \sum_{k=1}^K |x_k| \cdot |\mathbf{a}_j^\top \mathbf{a}_k| \\ & \leq |x_1| \mu(\mathbf{A}) K \end{aligned} \quad (3.7)$$

We find that if (3.7) is smaller than (3.6), then (3.5) holds, i.e. when

$$|x_1| \mu(\mathbf{A}) K < |x_1| - |x_1| \mu(\mathbf{A}) (K - 1).$$

Rearranging the terms gives us that

$$K < \frac{1}{2} \left(1 + \frac{1}{\mu(\mathbf{A})} \right). \quad (3.8)$$

So, when (3.8) holds, OMP recovers a component in the support set. Since (3.8) does not depend on \mathbf{y} or \mathbf{x} , OMP also recovers the other components in subsequent iterations and thus the full vector. This proves the theorem. \square

Borrowing terminology from array signal processing, we can interpret the theorem as saying that we can resolve more sources when the *sidelobes* $|\mathbf{a}_i^\top \mathbf{a}_j|$ are small. We also see that the theorem gives the worst case scenario where all sidelobes are large. Often a few sidelobes are large and the remaining small. This is not captured by the mutual coherence which overestimates the sidelobes. Another way to measure sidelobes is the *cumulative coherence*, or *Babel function*, of \mathbf{A} defined as [Ela, T⁺04]

$$\mu_1(p) = \max_{J, |J| \leq p} \max_{l \notin J} \sum_{k \in J} |\mathbf{a}_l^\top \mathbf{a}_k|.$$

The cumulative coherence is the maximum sum of coherences rather than the maximum coherence. Clearly $\mu_1(1) = \mu(\mathbf{A})$. The cumulative coherence can be calculated by computing $|\mathbf{A}^\top \mathbf{A}|$, summing the largest $p + 1$ components in each column, finding the maximum and subtracting 1. The cumulative coherence provides tighter bounds on the performance of OMP as follows.

Theorem 3.0.2. *If \mathbf{A} is a matrix with cumulative coherence $\mu_1(s)$ and*

$$\mu_1(K) + \mu_1(K - 1) < 1, \quad (3.9)$$

then OMP recovers all K -sparse vectors from measurements $\mathbf{y} = \mathbf{A}\mathbf{x}$.

Proof. As in the proof of theorem 3.0.1, OMP recovers an atom in the support set if (3.5) holds. Using the cumulative coherence, we can bound the left-hand side of (3.5) from below as

$$\begin{aligned} \max_{1 \leq j \leq K} \left| \mathbf{a}_j^\top \left(\sum_{k=1}^K \mathbf{a}_k x_k \right) \right| & \geq \max_{1 \leq i \leq K} |x_1| |\mathbf{a}_i^\top \mathbf{a}_1| - |x_1| \sum_{k=2}^K |\mathbf{a}_i^\top \mathbf{a}_k| \\ & \geq |x_1| - |x_1| \mu_1(K - 1). \end{aligned}$$

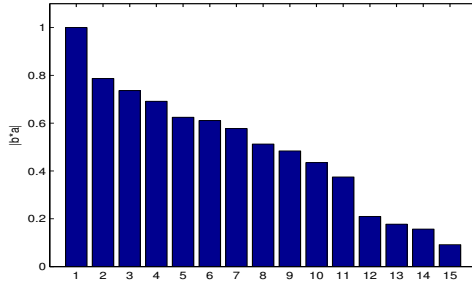


Figure 3.1: Sidelobes (sorted by magnitude) of a random 5×15 dictionary.

Similarly, we can bound the right hand side of (3.5) as

$$\max_{K+1 \leq j \leq n} \left| \mathbf{a}_j^\top \left(\sum_{k=1}^K \mathbf{a}_k x_k \right) \right| \leq \max_{K+1 \leq j \leq n} \sum_{k=1}^K |x_k| \cdot |\mathbf{a}_j^\top \mathbf{a}_k| \leq |x_1| \mu_1(K).$$

This gives us that if

$$1 - \mu_1(K-1) > \mu_1(K),$$

then OMP recovers the first atom of the support set. Since $\mu_1(p+1) \geq \mu_1(p)$, it follows that (3.9) is a sufficient condition for OMP to recover the subsequent atoms and thus the complete support set. This proves the theorem. \square

Both the mutual and cumulative coherence will be useful tools for improving the performance of OMP, as explained in the next section.

3.1 Beamforming for sparse recovery

The conditions (3.4) and (3.9) shows that the main obstacle for recovering non-zero components is because of the interference between different columns. The limitation is because of the matched filter. The matched filter is optimal for detecting a known signal in noise when no interfering atoms exists. But is the matched filter still optimal when several signals are present? We here show that the matched filter is *not* optimal by constructing detectors with better detection performance. In this section we construct detectors which minimize the sidelobes. We refer to the detectors as a beamformers because of its similarity with array processing techniques. We here concentrate on the OMP algorithm, although the approach can also be used to improve the performance of other greedy algorithms.

The beamformer detects non-zero components as

$$\hat{i} = \arg \max_i |\mathbf{b}_i^\top \mathbf{r}|, \quad (3.10)$$

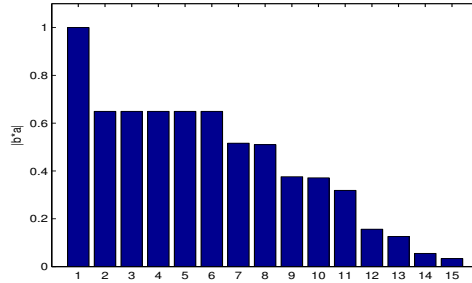


Figure 3.2: Sidelobes (sorted by magnitude) of the maximum-sidelobe beamformer.

where \mathbf{b}_i is a vector such that

$$\mathbf{b}_i^\top \mathbf{a}_i = 1.$$

We use $\mathbf{B} = [\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n]$ to denote the matrix of beamformers for all components. We refer to OMP with beamformer (i.e. replacing (3.3) by (3.10)) as OMPb, beamformer-aided OMP. The problem is now to design the beamformer so that it improves the estimation performance of OMP.

One method for designing a beamformer is to minimize the maximum sidelobe, i.e. the mutual-coherence of each atom. This leads to the optimization problem

$$\begin{aligned} \mathbf{b}_i &= \arg \min_{\mathbf{b}} \max_{j \neq i} |\mathbf{b}^\top \mathbf{a}_j| \\ &\text{subject to } \mathbf{b}^\top \mathbf{a}_i = 1. \end{aligned}$$

The beamformer minimizes the maximum sidelobe and is therefore referred to as the *maximum-sidelobe beamformer*. The optimization problem is convex and can be solved using e.g. linear programming. Minimizing the sidelobes shown in Figure 3.1 gives the sidelobes shown in Figure 3.2. We find that while many sidelobes have decreased, some have also increased. The maximum-sidelobe beamformer improves the recovery guarantee 3.0.1, but does not always improve performance. To find the optimal beamformer, we must analyze which measure of coherence to minimize. We will see that different performance measures naturally lead to different coherence measures. We begin by considering the worst case scenario.

3.1.1 Worst case beamformer

In the proof of Theorem 3.0.2 we notice that the inequalities in the proof are tighter when the non-zero components of \mathbf{x} have the same amplitude. The worst case beamformer for \mathbf{a}_i can thus be found as

$$\begin{aligned} \mathbf{b}_i &= \arg \min_{\mathbf{b}} \max_{\mathbf{x}} |\mathbf{b}^\top \mathbf{A} \mathbf{x}| \\ &\text{subject to } \|\mathbf{x}\|_\infty \leq 1, \|\mathbf{x}\|_0 \leq K, x_i = 0, \mathbf{b}^\top \mathbf{a}_i = 1. \end{aligned} \tag{3.11}$$

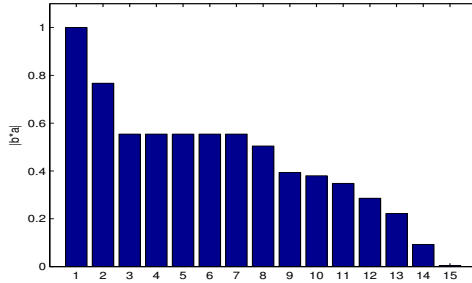


Figure 3.3: Sidelobes (sorted by magnitude) of the worst-case beamformer for $K = 3$.

The objective can be maximized with respect to \mathbf{x} , resulting in the expression

$$\mathbf{b}_i = \arg \min_{\mathbf{b}} \max_j \sum_{j \in J} |\mathbf{b}^\top \mathbf{a}_j|, \quad (3.12)$$

$$\text{subject to } J \subset [n] \setminus \{i\}, |J| \leq K, \mathbf{b}^\top \mathbf{a}_i = 1.$$

The worst case beamformer when \mathbf{x} is K -sparse is thus the one which minimizes the ℓ_1 -norm of the K largest sidelobes. The objective in (3.12) is convex and can thus be solved using e.g. the cvx toolbox [GBY08].

In Figure 3.3 we show the sidelobes of the worst case beamformer when the sidelobes of the matched filter is given by Figure 3.1. We note that although the second sidelobe is larger than the second sidelobe for the maximum-coherence beamformer, the subsequent sidelobes are smaller. Although this beamformer improves the worst-case performance, the improvement of the average performance is small compared to the matched filter. To improve the average-case performance, we need to construct a beamformer using probabilistic arguments.

3.1.2 Average case beamformer

The average performance of an algorithm can be found by randomly generating measurement data and averaging the error over the realizations. In such simulations, many different distributions can be chosen for the non-zero components of \mathbf{x} [Stu11]. One common scenario is to let the non-zero components be i.i.d. Gaussian distributed. In this case, the measurement $\mathbf{y} = \mathbf{A}\mathbf{x}$ is also Gaussian. Finding the optimal beamformer for this average case thus means finding a beamformer which has the largest probability of recovering non-zero components. Lemma 3.1 is useful for constructing an average-case beamformer.

Lemma 3.1. *Let $\mathbf{c}, \mathbf{d}, \mathbf{z} \in \mathbb{R}^m$. If $\mathbf{z} \in \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}_n)$ and \mathbf{c}, \mathbf{d} are fixed, then*

$$\Pr(|\mathbf{c}^\top \mathbf{z}| > |\mathbf{d}^\top \mathbf{z}|) = \frac{1}{\pi} \arccos \left(\frac{\|\mathbf{d}\|_2^2 - \|\mathbf{c}\|_2^2}{\|\mathbf{c} - \mathbf{d}\|_2 \cdot \|\mathbf{c} + \mathbf{d}\|_2} \right).$$

Proof. By symmetry we have that

$$\begin{aligned} P &= \Pr(|\mathbf{c}^\top \mathbf{z}| > |\mathbf{d}^\top \mathbf{z}|) \\ &= 2\Pr(\mathbf{c}^\top \mathbf{z} \geq 0, (\mathbf{c} - \mathbf{d})^\top \mathbf{z} \geq 0, (\mathbf{c} + \mathbf{d})^\top \mathbf{z} \geq 0) \end{aligned}$$

The last probability is given by the obtuse angle between the hyperplanes $(\mathbf{c} - \mathbf{d})^\top \mathbf{x} = 0$ and $(\mathbf{c} + \mathbf{d})^\top \mathbf{x} = 0$ divided by 2π . The angle between the hyperplanes is π minus the angle between the normal vectors. Using that $\pi - \arccos(t) = \arccos(-t)$ we find that the probability becomes

$$\begin{aligned} P &= \frac{2}{2\pi} \arccos\left(-\frac{(\mathbf{c} - \mathbf{d})^\top (\mathbf{c} + \mathbf{d})}{\|\mathbf{c} - \mathbf{d}\|_2 \cdot \|\mathbf{c} + \mathbf{d}\|_2}\right) \\ &= \frac{1}{\pi} \arccos\left(\frac{\|\mathbf{d}\|_2^2 - \|\mathbf{c}\|_2^2}{\|\mathbf{c} - \mathbf{d}\|_2 \cdot \|\mathbf{c} + \mathbf{d}\|_2}\right). \end{aligned}$$

This completes the proof. □

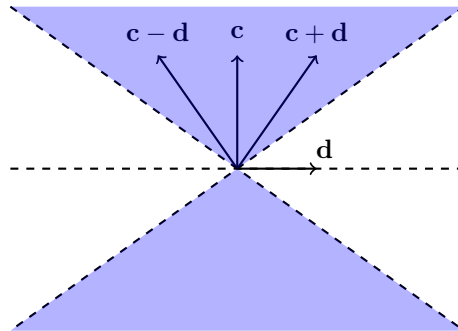


Figure 3.4: Illustration of the region in the proof of Lemma 3.1. The shaded area contains all vectors \mathbf{z} such that $|\mathbf{c}^\top \mathbf{z}| > |\mathbf{d}^\top \mathbf{z}|$.

From lemma 3.1 we find that the probability increases when the length of \mathbf{c} increases and the length of \mathbf{d} decreases, this can also be seen in Figure 3.5.

The lemma easily extends to non-white random Gaussian vectors for which $\text{Cov}(\mathbf{z}) = \mathbf{C}$ by setting $\mathbf{c}_i = \mathbf{C}^{-1/2} \tilde{\mathbf{c}}_i$. The inner products are then replaced by $\tilde{\mathbf{c}}_i^\top \tilde{\mathbf{c}}_j = \mathbf{c}_i^\top \mathbf{C} \mathbf{c}_j$.

Let $I \subset [n]$ be the support set of a sparse vector \mathbf{x} . We use Lemma 3.1 to construct an average-case beamformer by setting

$$\mathbf{c}_i = \mathbf{b}_i^\top \mathbf{A}_I, \quad \mathbf{z} = \mathbf{x}_I.$$

From Lemma 3.1 we get that the probability to choose $i \in I$ over $j \notin I$ is large when $\|\mathbf{c}_i\|_2$ is large and $\|\mathbf{c}_j\|_2$ is small. Since the support set is unknown, we minimize

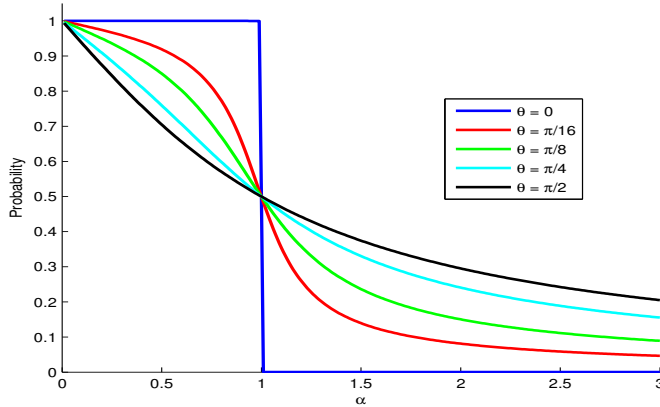


Figure 3.5: $\Pr(|\mathbf{c}^\top \mathbf{z}| > |\mathbf{d}^\top \mathbf{z}|)$ when $\|\mathbf{d}\|_2 = \alpha\|\mathbf{c}\|_2$ and $\mathbf{c}^\top \mathbf{d} = \alpha\|\mathbf{c}\|_2^2 \cos(\theta)$.

the maximum length of \mathbf{c}_i over all support sets not containing i while keeping $\mathbf{b}_i^\top \mathbf{a}_i$ fixed, i.e. we choose the beamformer as

$$\mathbf{b}_i = \arg \min_{\mathbf{b}} \left(\max_{|J| \leq K, i \notin J} \sum_{j \in J} (\mathbf{b}^\top \mathbf{a}_j)^2 \right) \quad \text{s.t.} \quad \mathbf{b}^\top \mathbf{a}_i = 1. \quad (3.13)$$

The optimization problem is convex and can be solved using `cvx` [GBY08] or methods from e.g. [OT03]. For $K = n - 1$, the beamformer can be found analytically as

$$\mathbf{b}_i = \frac{(\mathbf{A}\mathbf{A}^\top)^{-1} \mathbf{a}_i}{\mathbf{a}_i^\top (\mathbf{A}\mathbf{A}^\top)^{-1} \mathbf{a}_i}, \quad (3.14)$$

i.e. $\mathbf{B} = (\mathbf{A}^+)^{\top} \mathbf{D}$ where \mathbf{D} is a diagonal matrix with entries $D_{ii} = 1/(\mathbf{A}^+ \mathbf{A})_{ii}$ and \mathbf{A}^+ is the Moore-Penrose pseudoinverse of \mathbf{A} . We see that (3.14) can be interpreted as a Capon method [SM05] for recovery of sparse random vectors. Another approach which also produces the beamformer (3.14) is to minimize the expected length of \mathbf{c}_i rather than the maximum length.

Another motivation for using the pseudoinverse as a beamformer is to choose \mathbf{B} so that $\mathbf{B}^\top \mathbf{A} \mathbf{x}$ is as close as possible to \mathbf{x} in the mean square sense, i.e. we choose \mathbf{B} to minimize

$$\begin{aligned} \mathcal{E}[\|\mathbf{x} - \mathbf{B}^\top \mathbf{A} \mathbf{x}\|_2^2] &= \text{tr}((\mathbf{I} - \mathbf{B}^\top \mathbf{A}) \mathcal{E}[\mathbf{x} \mathbf{x}^\top] (\mathbf{I} - \mathbf{B}^\top \mathbf{A})^\top) \\ &= \frac{K \sigma_x^2}{n} \|\mathbf{I} - \mathbf{B}^\top \mathbf{A}\|_F^2 \end{aligned}$$

where \mathbb{E} denotes the expectation value, we assumed that all support sets are chosen with equal probability and that the components of \mathbf{x}_I are random variables with

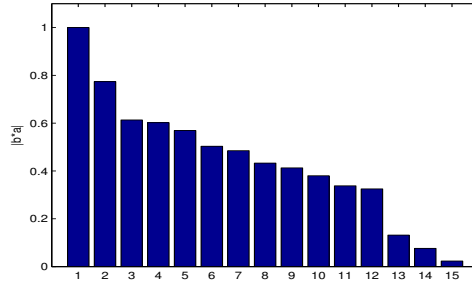


Figure 3.6: Sidelobes (sorted by magnitude) of the average-case beamformer (3.14).

$\mathbb{E}[x_i x_j | \{i, j\} \subset I] = \sigma_x^2 \delta_{ij}$. This gives us the minimizer

$$\mathbf{B} = (\mathbf{A}^+)^{\top}$$

Note that we did not make any assumptions on the distribution of \mathbf{x}_I , so this argument holds also for non-Gaussian random signals, e.g. binary (± 1) signals. This beamformer is different from (3.14) since in general $\mathbf{b}_i^{\top} \mathbf{a}_i \neq 1$.

3.2 Beamforming in the presence of noise

The optimal beamformers need to be adjusted when the measured signal is contaminated by noise. This is because measurement noise introduces an additional source of error which needs to be mitigated. In theory, exact recovery is not possible under additive noise, however, one is able to recover the support set with some probability.

3.2.1 Worst case beamformer for noisy measurements

When the measurements are noisy, extra care is needed when constructing the beamformer. The presence of random noise means that we design the beamformer to maximize the probability of recovering a component $i \in I$. The following theorem gives us a way to constructing a worst-case beamformer for noisy measurements.

Theorem 3.2.1. *Assume that the additive noise is zero-mean Gaussian distributed, $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$, that $\mathbf{b}_i^{\top} \mathbf{a}_i = 1$ for all $i = 1, 2, \dots, n$ and let*

$$c = 1 - \mu_1(\mathbf{A}, \mathbf{B}, K) - \mu_1(\mathbf{A}, \mathbf{B}, K - 1) > 0$$

where $\mu_1(\mathbf{A}, \mathbf{B}, K)$ is the cross cumulative coherence [SV08]

$$\mu_1(\mathbf{A}, \mathbf{B}, K) = \max_{i, |J| \leq K, i \notin J} \sum_{j \in J} |\mathbf{b}_i^{\top} \mathbf{a}_j|.$$

Then the probability P that OMPb recovers the component x_i of \mathbf{x} with maximum modulus in the first iteration obeys

$$P \geq 1 - 2Q\left(\frac{c|x_i|}{\sqrt{\mathbf{b}_i^\top \mathbf{C} \mathbf{b}_i}}\right) \quad (3.15)$$

where $Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-t^2/2} dt$ is the tail probability of the normal distribution.

Proof. A sufficient condition for OMPb to recover $i \in I$ is

$$\frac{2}{|x_i|} |\mathbf{b}_i^\top \mathbf{n}| < 1 - \sum_{j \in I \setminus \{i\}} |\mathbf{b}_i^\top \mathbf{a}_j| - \max_{l \notin I} \sum_{j \in I} |\mathbf{b}_l^\top \mathbf{a}_j| \quad (3.16)$$

Using that

$$\begin{aligned} \sum_{j \in I \setminus \{i\}} |\mathbf{b}_i^\top \mathbf{a}_j| &\leq \mu_1(\mathbf{A}, \mathbf{B}, K - 1) \\ \max_{l \notin I} \sum_{j \in I} |\mathbf{b}_l^\top \mathbf{a}_j| &\leq \mu_1(\mathbf{A}, \mathbf{B}, K) \end{aligned}$$

we find that (3.16) holds provided that $|\mathbf{b}_i^\top \mathbf{n}| < c|x_i|/2$. Using this we find that

$$P \geq \Pr\left(|\mathbf{b}_i^\top \mathbf{n}| < \frac{c|x_i|}{2}\right) \quad (3.17)$$

When the noise is $\mathcal{N}(\mathbf{0}, \mathbf{C})$ distributed, then $z_i = \mathbf{b}_i^\top \mathbf{n}$ is $\mathcal{N}(0, \mathbf{b}_i^\top \mathbf{C} \mathbf{b}_i)$ distributed. Using that $P(|z_i| < \epsilon) = 1 - 2Q(2\epsilon/\sigma_i)$ we arrive at the result. \square

Note that (3.17) also holds for non-Gaussian noise distributions, but for such cases it is harder to obtain an expression similar to (3.15). Theorem 3.2.1 gives that the probability of recovering the largest component increases with increasing Signal-to-Noise Ratio (SNR), as can be expected. One way to maximize P is to maximize the argument of the Q -function. The argument is, however, a non-convex function of \mathbf{B} and is therefore difficult to maximize. A more accessible approach is to find the beamformer as

$$\begin{aligned} \mathbf{b}_i &= \arg \min_{\mathbf{b}} \left(\max_{|J|=K, i \notin J} \sum_{j \in J} |\mathbf{b}^\top \mathbf{a}_j| + \lambda \mathbf{b}^\top \mathbf{C} \mathbf{b} \right) \\ &\text{s.t. } \mathbf{b}^\top \mathbf{a}_i = 1 \end{aligned} \quad (3.18)$$

where $\lambda \geq 0$ is a design parameter.

3.2.2 Average case beamformer for noisy measurements

To find the average case beamformer for the noisy setting, we can still utilize Lemma 3.1 by redefining the vectors involved. For measurements (3.1) with $\text{supp}(\mathbf{x}) = I$, $\mathbf{x}_I \sim \mathcal{N}(\mathbf{0}, \sigma_x^2 \mathbf{I})$ and $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$ we set

$$\mathbf{b}_i^\top \mathbf{y} = \mathbf{b}_i^\top (\mathbf{A}_I \mathbf{x}_I + \mathbf{n}) = (\mathbf{c}_i^\top, \mathbf{b}_i^\top) \begin{pmatrix} \mathbf{x}_I \\ \mathbf{n} \end{pmatrix},$$

where $\mathbf{c}_i = \mathbf{A}_I^\top \mathbf{b}_i$. The probability to choose the index i over the index j then becomes

$$P(|\mathbf{b}_i^\top (\mathbf{A} \mathbf{x} + \mathbf{n})| > |\mathbf{b}_j^\top (\mathbf{A} \mathbf{x} + \mathbf{n})|) = \frac{1}{\pi} \arccos \left(\frac{\sigma_x^2 (\|\mathbf{c}_j\|_2^2 - \|\mathbf{c}_i\|_2^2) + (\mathbf{b}_j^\top \mathbf{C} \mathbf{b}_j - \mathbf{b}_i^\top \mathbf{C} \mathbf{b}_i)}{\sqrt{(\sigma_x^2 \|\mathbf{c}_j\|_2^2 + \sigma_x^2 \|\mathbf{c}_i\|_2^2 + \mathbf{b}_i^\top \mathbf{C} \mathbf{b}_i + \mathbf{b}_j^\top \mathbf{C} \mathbf{b}_j)^2 - 4(\sigma_x^2 \mathbf{c}_i^\top \mathbf{c}_j + \mathbf{b}_i^\top \mathbf{C} \mathbf{b}_j)^2}} \right)$$

To maximize the probability of recovery, we need to minimize the length of \mathbf{b}_j while maximizing the length of \mathbf{c}_i relative to the length of \mathbf{c}_j . One approach is, as before, to penalize the length of \mathbf{b}_i by setting

$$\mathbf{b}_i = \arg \min_{\mathbf{b}} \left(\max_{|J| \leq K, i \notin J} \sum_{j \in J} |\mathbf{b}^\top \mathbf{a}_j|^2 + \lambda \mathbf{b}^\top \mathbf{C} \mathbf{b} \right) \quad (3.19)$$

s.t. $\mathbf{b}^\top \mathbf{a}_i = 1$,

where λ is a design parameter. Again, setting $K = n - 1$, we obtain the beamformer

$$\mathbf{b}_i = \frac{(\mathbf{A} \mathbf{A}^\top + \lambda \mathbf{C})^{-1} \mathbf{a}_i}{\mathbf{a}_i^\top (\mathbf{A} \mathbf{A}^\top + \lambda \mathbf{C})^{-1} \mathbf{a}_i}. \quad (3.20)$$

When considering the expected cumulative-cross-coherence rather than the maximum cross-coherence for K sparse vectors, one obtains a similar beamformer with $\lambda_{\text{average}} = n\lambda/K$ in (3.20). We see that both (3.18) and (3.20) converge to \mathbf{a}_i in the limit $\lambda \rightarrow \infty$. Next we investigate how the detection problem can be modeled as a Bayesian filtering problem.

3.3 Bayesian filtering for greedy pursuits

A basic problem in signal processing is to extract a signal from noisy observations. Since the signal is unknown, but has known properties such as first and second order statistics, the signal and noise are often modeled as random processes. By using the statistics of the signal it is possible to construct a filter which minimizes the error in a probabilistic sense, e.g. the mean square error. Two common filters

are the Wiener and Kalman filters [Kay98, KSH00]. The standard linear filtering theory is not directly applicable to the sparse signals. Rather, the sparse signal reconstruction problem is both a detection (finding which components are non-zero) and estimation (finding the values of the non-zero components) problem. To construct filters for sparse signals, we first need to model the random signals.

To model the support set of a sparse signal we assign a prior probability to the possible support sets

$$p(I) = \{\text{probability that } \text{supp}(\mathbf{x}) = I\}.$$

Further we choose a distribution for the components. The distribution of the components of a sparse random vector are conditioned on whether the index of the component is in the support set or not as follows

$$\begin{aligned} p(x_i|i \in I) &= p(x_i), \\ p(x_i|i \notin I) &= \delta(x_i). \end{aligned}$$

Assuming that all support sets contains K elements and are equally probable we get that

$$p(I) = \binom{n}{K}^{-1}.$$

We find that the probability of an index i belonging to the support set is

$$p(i \in I) = \binom{n}{K}^{-1} \binom{n-1}{K-1} = \frac{K}{n}.$$

From now on we assume that the non-zero components are Gaussian distributed as $p(x_i) = \mathcal{N}(x_i|0, \sigma_x^2)$ and that the noise is Gaussian distributed as $p(\mathbf{n}) = \mathcal{N}(\mathbf{n}|\mathbf{0}, \sigma_n^2 \mathbf{I}_m)$. When the support set is fixed, the measurements \mathbf{y} is Gaussian distributed with

$$p(\mathbf{y}|I) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{C}_I) = \frac{1}{(2\pi)^{m/2} |\mathbf{C}_I|^{1/2}} e^{-\frac{1}{2} \mathbf{y}^\top \mathbf{C}_I^{-1} \mathbf{y}}$$

where the covariance of \mathbf{y} is given by

$$\mathbf{C}_I = \sigma_x^2 \mathbf{A}_I \mathbf{A}_I^\top + \sigma_n^2 \mathbf{I}_n.$$

For a known support set, the Minimum Mean Square Error (MMSE) estimator of \mathbf{x} is given by

$$\begin{aligned} \hat{\mathbf{x}}_{MMSE}(I, \mathbf{y})_I &= \mathcal{E}[\mathbf{x}_I | \mathbf{y}, I] = \sigma_x^2 \mathbf{A}_I^\top \mathbf{C}_I^{-1} \mathbf{y}, \\ \hat{\mathbf{x}}_{MMSE}(I, \mathbf{y})_{I^c} &= \mathcal{E}[\mathbf{x}_{I^c} | \mathbf{y}, I] = \mathbf{0}. \end{aligned}$$

When the support set is random, the MMSE estimator becomes

$$\hat{\mathbf{x}}_{MMSE} = \mathcal{E}[\mathbf{x}|\mathbf{y}] = \sum_{\mathcal{C} \subset [n]} p(I|\mathbf{y}) \hat{\mathbf{x}}_{MMSE}(I, \mathbf{y}), \quad (3.21)$$

where the a-posteriori probability $p(I|\mathbf{y})$ of a support set I is given by Bayes rule

$$p(I|\mathbf{y}) = \frac{p(\mathbf{y}|I)p(I)}{p(\mathbf{y})} = \frac{e^{-\frac{1}{2}\mathbf{y}^\top \mathbf{C}_I^{-1} \mathbf{y}}}{Z|\mathbf{C}_I|^{1/2}},$$

where Z is a normalization constant.

The MMSE estimator is optimal with respect to the mean square error (MSE). A disadvantage of the MMSE estimator is that it requires computing $\binom{n}{K}$ matrix inverses. The computational complexity is thus of the order of the exhaustive search, making the estimator intractable for many problems. Even for *small* problems, the estimator is computationally demanding. For example, when $n = 100$ and $K = 5$ the estimator requires about 75 million 5×5 matrix inverses. This means that it takes a standard laptop computer about 11 days to compute the estimate.

The intractability of the MMSE estimator for sparse Bayesian reconstruction has given rise to several approximate estimators. One such estimator is the approximate MMSE estimator by Selen and Larsson [LS07] which approximates the sum by a partial sum over more *significant* support sets and finds these subsets using a greedy search method. However, the search strategy employed by the approximate MMSE estimator requires subsets of all cardinalities to have non-zero probability. This makes the estimator unable to handle problems where the cardinality is known. Another method which can handle support sets of fixed cardinality is the randOMP algorithm by Elad and Yavneh [EY09]. The randOMP algorithm computes several estimates using an OMP algorithm which selects the atoms at random by a probabilistic rule. The final estimate is the average of the random estimates. Both the approximate MMSE estimator and randOMP uses the standard matched filter to detect non-zero components.

The Wiener filter exploits first and second order statistics, i.e. expectation values and correlations, to construct a linear estimator which minimizes the MMSE. Given measurements \mathbf{y} , the Wiener filter is the linear estimator

$$\hat{x} = \mathbf{b}^\top \mathbf{y} + c,$$

where \mathbf{b} and c are chosen to minimize the Mean Square Error (MSE)

$$\text{MSE} = \mathcal{E}[(\hat{x} - x)^2].$$

Minimizing the MSE gives the estimator

$$\hat{x} = \mathcal{E}[x] + \mathbf{C}(x, \mathbf{y})\mathbf{C}(\mathbf{y})^{-1}(\mathbf{y} - \mathcal{E}[\mathbf{y}]),$$

where

$$\begin{aligned}\mathbf{C}(\mathbf{y}) &= \text{Cov}(\mathbf{y}, \mathbf{y}) = \mathcal{E}[(\mathbf{y} - \mathcal{E}[\mathbf{y}])(\mathbf{y} - \mathcal{E}[\mathbf{y}])^\top], \\ \mathbf{C}(x, \mathbf{y}) &= \text{Cov}(x, \mathbf{y}) = \mathcal{E}[(x - \mathcal{E}[x])(\mathbf{y} - \mathcal{E}[\mathbf{y}])^\top],\end{aligned}$$

are a-priori covariance and cross-correlation matrices.

3.3.1 Detecting active components

Directly applying the Wiener filter to the sparse reconstruction problem leads to the MMSE estimator (3.21). To avoid the high complexity of the MMSE estimator we construct a Wiener filter conditioned on the support set. We do this by developing a Bayesian detector and estimator by conditioning the prior distribution on hypotheses about the support, i.e. the prior distribution is $p(\mathbf{x}|H_i)$ where H_i is a hypothesis. We then chose the hypothesis which best describes the data. One extreme is the exhaustive search which corresponds to testing the hypotheses $H_i = \{I_i = I\}$ for $i = 1, 2, 3, \dots, \binom{n}{K}$. Another extreme is the least restrictive hypothesis $H_i = \{i \in I\}$ for $i = 1, 2, \dots, n$.

We first consider the noise-free case. Under the hypothesis $i \notin I$, the LMMSE estimator is the trivial estimator

$$\hat{x}_i | (i \notin I) = 0.$$

For the hypothesis $H_i = \{i \in I\}$, the LMMSE estimator is $\hat{x}_i | (i \in I) = \mathbf{b}^\top \mathbf{y}$, where \mathbf{b} minimizes the conditional MSE

$$\begin{aligned}\mathcal{E}[(x_i - \mathbf{b}^\top \mathbf{y})^2 | i \in I] &= \mathcal{E}_I \left[(1 - \mathbf{b}^\top \mathbf{a}_i)^2 \sigma_x^2 + \sum_{j \in I \setminus \{i\}} (\mathbf{b}^\top \mathbf{a}_j)^2 \sigma_x^2 \middle| i \in I \right] \\ &= (1 - \mathbf{b}^\top \mathbf{a}_i)^2 \sigma_x^2 + \sigma_x^2 \sum_{j \neq i} (\mathbf{b}^\top \mathbf{a}_j)^2 P(j \in I | i \in I).\end{aligned}\tag{3.22}$$

When all support sets are equally probable, we get that for $j \neq i$

$$P(j \in I | i \in I) = \frac{\binom{n-2}{K-2}}{\binom{n-1}{K-1}} = \frac{K-1}{n-1} = \rho_1.$$

Using this, we find that

$$\mathbf{b} = ((1 - \rho_1) \mathbf{a}_i \mathbf{a}_i^\top + \rho_1 \mathbf{A} \mathbf{A}^\top)^{-1} \mathbf{a}_i.$$

The conditional LMMSE estimator thus becomes

$$\begin{aligned}\hat{x}_i | (i \in I) &= \mathbf{a}_i^\top ((1 - \rho_1) \mathbf{a}_i \mathbf{a}_i^\top + \rho_1 \mathbf{A} \mathbf{A}^\top)^{-1} \mathbf{y} \\ &= \frac{\mathbf{a}_i^\top (\mathbf{A} \mathbf{A}^\top)^{-1} \mathbf{y}}{\rho_1 + (1 - \rho_1) \mathbf{a}_i^\top (\mathbf{A} \mathbf{A}^\top)^{-1} \mathbf{a}_i},\end{aligned}$$

where we simplified the expression using the Sherman-Morrison formula [HJ12].

In the limit $\rho_1 \rightarrow 1$, the estimator becomes the unconditional estimator $\hat{x}_i = \mathbf{e}_i^\top \mathbf{A}^+ \mathbf{y}$, where \mathbf{A}^+ denotes the Moore-Penrose pseudoinverse of \mathbf{A} . This corresponds to the limit $K \rightarrow n$. On the other hand, in the limit $\rho_1 \rightarrow 0$, the estimator becomes the average case beamformer (3.14). This corresponds to the limit $K \rightarrow 1$. We thus find that the unconditional estimator corresponds to the conditional LMMSE estimator with $K = n$, while the average-case beamformer corresponds to $K = 1$. For $K = 1$, the minimum of (3.22) is non-unique. The solution which minimizes the ℓ_2 norm of \mathbf{b} is the matched filter $\mathbf{b} = \mathbf{a}_i$.

3.3.2 Estimating active components

Many greedy search algorithms, such as OMP, estimate the support set by estimating the components in a partial support set, forming a prediction and then subtracting the prediction from the measurements to infer new atoms. The Bayesian modeling with the conditional prior gives us a way of constructing a Bayesian estimator. Assuming that $|I_s| = s \leq K$ and $I_s \subset I$, the conditional LMMSE estimator of \mathbf{x}_{I_s} is given by $\hat{\mathbf{x}}_{I_s} | (I_s \subset I) = \mathbf{B}_s^\top \mathbf{y}$, where $\mathbf{B}_s \in \mathbb{R}^{m \times s}$ minimizes the MSE

$$\begin{aligned} \text{MSE} | (I_s \subset I) &= \mathcal{E} [\|\mathbf{x}_{I_s} - \mathbf{B}_s^\top \mathbf{y}\|_2^2 | I_s \subset I] \\ &= \sigma_x^2 \|\mathbf{I} - \mathbf{B}_s^\top \mathbf{A}_{I_s}\|_F^2 + \sigma_x^2 \sum_{j \notin I_s} \|\mathbf{B}_s^\top \mathbf{a}_j\|_2^2 P(j \in I | I_s \subset I). \end{aligned}$$

We find that

$$P(j \in I | I_s \subset I) = \frac{\binom{n-s-1}{K-s-1}}{\binom{n-s}{K-s}} = \frac{K-s}{n-s} = \rho_s,$$

for $j \notin I_s$. This gives us that the LMMSE estimator becomes

$$\hat{\mathbf{x}}_{I_s} | (I_s \subset I) = \mathbf{A}_{I_s}^\top \left((1 - \rho_s) \mathbf{A}_{I_s} \mathbf{A}_{I_s}^\top + \rho_s \mathbf{A} \mathbf{A}^\top \right)^{-1} \mathbf{y}.$$

We have that $\rho_K = 0$, so

$$\hat{\mathbf{x}}_{I_K} | (I_K = I) = \mathbf{A}_{I_K}^+ \mathbf{y}.$$

The conditional Bayesian estimator thus reduces to the usual least square estimator when the partial support set has size K .

3.3.3 Noisy and correlated signals

When the measurements are corrupted by noise and/or the signal is correlated, the expressions for the estimator becomes somewhat more involved. For correlated signals, the estimators depend on conditional cross-covariance matrices of two vectors

\mathbf{u} and \mathbf{v} given by

$$\begin{aligned} \mathbf{C}(\mathbf{u}, \mathbf{v}|I_s) &= \mathcal{E}[\mathbf{u}\mathbf{v}^\top | I_s \subset I] \\ &= \sum_{J, |J|=K} P(J = I | I_s \subset I) \mathcal{E}[\mathbf{u}\mathbf{v}^\top | \text{supp}(\mathbf{x}) = J]. \end{aligned}$$

We also use $\mathbf{C}(\mathbf{u}|I_s) = \mathbf{C}(\mathbf{u}, \mathbf{u}|I_s)$ to denote the conditional cross-covariance of \mathbf{u} with itself. Assuming that the noise is correlated with covariance matrix $\mathbf{C}(\mathbf{w})$, the MSE of $\hat{\mathbf{x}}_{I_s} | (I_s \subset I) = \mathbf{B}^\top \mathbf{y}$ becomes

$$\begin{aligned} \text{MSE} | (I_s \subset I) &= \mathcal{E} [\|\mathbf{x}_{I_s} - \mathbf{B}^\top \mathbf{y}\|_2^2 | I_s \subset I] \\ &= \text{tr}(\mathbf{C}(\mathbf{x}_{I_s} | I_s)) + \text{tr}(\mathbf{B}^\top \mathbf{C}(\mathbf{y}|I_s) \mathbf{B}) - 2\text{tr}(\mathbf{B}^\top \mathbf{C}(\mathbf{y}, \mathbf{x}_{I_s} | I_s)), \end{aligned}$$

where $\mathbf{C}(\mathbf{y}|I_s) = \mathbf{A}\mathbf{C}(\mathbf{x}|I_s)\mathbf{A}^\top + \mathbf{C}(\mathbf{w})$ is the conditional covariance of \mathbf{y} and $\mathbf{C}(\mathbf{y}, \mathbf{x}_{I_s} | I_s \subset I) = \mathbf{A}\mathbf{C}(\mathbf{x}, \mathbf{x}_{I_s} | I_s)$ is the cross-covariance of \mathbf{y} and \mathbf{x}_{I_s} . Using this, we find that the LMMSE estimator becomes

$$\hat{\mathbf{x}}_{I_s} | (I_s \subset I) = \mathbf{C}(\mathbf{x}_{I_s}, \mathbf{y}|I_s) \mathbf{C}(\mathbf{y}|I_s)^{-1} \mathbf{y}.$$

With this notation we notice the similarity to the classical Wiener filter. We also note that when the non-zero signal components are i.i.d. and the noise is white, the LMMSE estimator reduces to the expected form

$$\hat{\mathbf{x}}_{I_s} | (I_s \subset I) = \mathbf{A}_{I_s}^\top ((1 - \rho_s) \mathbf{A}_{I_s} \mathbf{A}_{I_s}^\top + \rho_s \mathbf{A} \mathbf{A}^\top + \gamma \mathbf{I}_m)^{-1} \mathbf{y},$$

where $\gamma = \sigma_n^2 / \sigma_x^2 = \text{SNR}^{-1}$ is the inverse Signal-to-Noise-Ratio.

3.4 Conditional prior based OMP

So far, we have discussed Bayesian detection of active components and estimation of their values. The detector and estimator are quite general and can be used in any detection based algorithm for sparse reconstruction, e.g. greedy search algorithms. Several greedy search algorithms have been developed such as Matching Pursuit (MP) [MZ93], OMP [TG07], Subspace Pursuit (SP) [DM09] and CoSamp [NT09]. For concreteness, we here focus on adopting the conditional Bayesian detection and estimation methods to the OMP algorithm to formulate the Conditional Prior based OMP (CpOMP) algorithm. The methodology can also be adopted to other pursuit algorithms in a similar way.

To adapt the conditional prior methods to OMP, we must consider the decision rule of when to include new atoms in the support set. The detection approach is to include atoms as

$$\hat{i} = \arg \max_j |\hat{x}_j|,$$

although it is also possible to include atoms according to the rule

$$\hat{i} = \arg \min_j \|\mathbf{r} - \mathbf{a}_j \hat{x}_j\|_2.$$

For the standard OMP, these selection rules are equivalent while for CpOMP they are not.

When noise is present, the single element estimators (detectors) are given by

$$\hat{x}_i = \frac{\mathbf{a}_i^\top (\mathbf{A}\mathbf{A}^\top + \frac{\gamma}{1-\rho_1} \mathbf{I}_m)^{-1} \mathbf{y}}{\rho_1 + (1-\rho_1) \mathbf{a}_i^\top (\mathbf{A}\mathbf{A}^\top + \frac{\gamma}{1-\rho_1} \mathbf{I}_m)^{-1} \mathbf{a}_i}$$

The estimator has the disadvantage that it needs to compute a new matrix inverse in each iteration. To lower the complexity, we use the approximation

$$\frac{\gamma}{1-\rho_1} \approx \gamma.$$

The conditional prior estimator then becomes

$$\hat{x}_i = \frac{\mathbf{a}_j^\top \mathbf{D} \mathbf{r}}{\rho_s + (1-\rho_s) \mathbf{a}_j^\top \mathbf{D} \mathbf{a}_j},$$

where

$$\mathbf{D} = (\mathbf{A}\mathbf{A}^\top + \gamma \mathbf{I}_m)^{-1}.$$

We refer to OMP with the conditional prior Bayesian detector and estimator as Conditional prior based OMP (CpOMP). It is possible to further improve the estimation performance by combining the conditional prior with projection based strategies [CSVS12].

3.4.1 Improved search using Projection based OMP

The Projection based OMP (POMP) search strategy [CSVS12] works by including more than one element in the support set and then removing elements which are considered to be irrelevant. Like OMP, the standard POMP algorithm [CSVS12] detects non-zero components using the matched filter

$$\hat{J} = \hat{I} \cup \{L \text{ largest components of } |\mathbf{A}^\top \mathbf{r}|\}.$$

The difference is that POMP includes the L largest components of $|\mathbf{A}^\top \mathbf{r}|$, stored in a set \hat{J} , in a preliminary support set estimate $\hat{I} \cup \hat{J}$, where \hat{I} is the estimated support set from previous iterations. POMP then estimates the coefficients of $\hat{\mathbf{x}}_{\hat{I} \cup \hat{J}}$ using least squares. Finally, POMP includes only the the component of largest amplitude in the support set estimate as

$$\hat{I} \cup \{\arg \max_{i \in \hat{J}} |\hat{x}_i|\} \rightarrow \hat{I}.$$

Data: $\mathbf{y}, \mathbf{A}, K$.

Initialization: $\mathbf{r} = \mathbf{y}$, $I_0 = \emptyset$, $\hat{\mathbf{x}}_I = \mathbf{0}$, $s = 0$

while $s < K$ **do**

$s + 1 \rightarrow s$
$c_i = \frac{ \mathbf{a}_j^\top \mathbf{D} \mathbf{r} }{\rho_s + (1 - \rho_s) \mathbf{a}_j^\top \mathbf{D} \mathbf{a}_j}$
$J = I_{s-1} \cup \{\text{indices of } L \text{ largest } c_i \text{ not in } I_{s-1}\}$
$\hat{\mathbf{x}}_J = \hat{\mathbf{x}}_J (J \subset I) = \mathbf{A}_J^\top \left((1 - \rho_{ J }) \mathbf{A}_J \mathbf{A}_J^\top + \rho_{ J } \mathbf{A} \mathbf{A}^\top \right)^{-1} \mathbf{y}$
$I_s = I_{s-1} \cup \{\text{index of largest element of } \hat{\mathbf{x}}_J \text{ in } J \setminus I_{s-1}\}$
$\hat{\mathbf{x}}_{I_s} (I_s \subset I) = \mathbf{A}_{I_s}^\top \left((1 - \rho_s) \mathbf{A}_{I_s} \mathbf{A}_{I_s}^\top + \rho_s \mathbf{A} \mathbf{A}^\top \right)^{-1} \mathbf{y}$
$\mathbf{r} = \mathbf{y} - \mathbf{A}_{I_s} \hat{\mathbf{x}}_{I_s}$

end

Result: Estimated support set, $\hat{I} = I_K$, and components, $\hat{\mathbf{x}}_I$.

Algorithm 2: The CpPOMP algorithm.

The POMP algorithm has a higher complexity than the standard OMP (corresponding to POMP with $L = 1$) and improves the estimation performance [CSVS12].

We adopt the Conditional Prior to POMP by replacing the least squares estimator by the conditional prior estimator and the matched filter by the conditional prior detection filter. Since the conditional prior can only handle support sets of size $|\hat{I}| \leq K$, we set $\rho_s = \max(0, K - s)/(n - s)$ (we here assume that $K + L < n + 1$). The CpPOMP algorithm is thus given by Algorithm 2.

3.5 Computational complexity

Different algorithms have different trade-offs between accuracy and complexity. For this reason it is interesting to investigate the computational complexity of CpOMP. We compare here compare the complexity of (the naive implementation of) OMP against (the equally naive implementation of) CpOMP.

Since OMP runs K iterations, it requires the computation of $\mathcal{O}(nK)$ correlations. This requires $\mathcal{O}(nmK)$ multiplications. OMP also solves solving K least squares problems, requiring $\mathcal{O}(mK^3)$ operations in total. So the complexity of OMP is $\mathcal{O}(mK^3 + mnK)$.

CpOMP requires computing the matrix $\mathbf{A}^\top \mathbf{D}$ requiring $\mathcal{O}(n^2m)$ operations, correlations requiring $\mathcal{O}(nmK)$ operations and computing K estimates $\hat{\mathbf{x}}_{\hat{I}}$ requiring $\mathcal{O}(Km^3)$ operations in total. The complexity of CpOMP is thus $\mathcal{O}(mn^2 + m^3K)$. The complexity of CpOMP is thus much higher than the complexity of OMP when $K \ll m$.

We note that efficient implementations of OMP (and other pursuit algorithms) exists, see e.g. [RZE08] and references therein.

3.6 Numerical evaluation

3.6.1 Simulation setup

To compare the methods we randomly generated the problem (3.1) for the chosen parameter values and estimate the sparse vector \mathbf{x} using different estimation algorithms. We then calculated the mean square error by averaging over the different problem realizations. The problem parameters are the number of measurements (n), the number of measurements (m), the number of non-zero elements in \mathbf{x} (K) and the Signal to Noise Ratio (SNR). The problem parameters are specified for each problem. In each simulation we vary one parameter and keep the other parameters fixed.

We generate the measurement matrix \mathbf{A} by drawing its elements from an i.i.d. zero-mean Gaussian distribution and normalizing the column vectors to unit length. This is equivalent to sampling the column vectors uniformly on the sphere. The support set (positions of the non-zero components) of \mathbf{x} are selected uniformly at random from all subsets of $[n] = \{1, 2, \dots, n\}$ of with K elements. The non-zero elements are then drawn from $\mathcal{N}(0, 1)$. The noise is drawn from $\mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I}_m)$ where the noise variance σ_n^2 is chosen such that the SNR

$$\text{SNR} = \frac{\mathcal{E}[\|\mathbf{A}\mathbf{x}\|_2^2]}{\mathcal{E}[\|\mathbf{n}\|_2^2]} = \frac{\frac{K}{n} \text{tr}(\mathbf{A}^\top \mathbf{A})}{m\sigma_n^2} = \frac{K}{m\sigma_n^2},$$

takes the desired value. The random variables are thus \mathbf{A} , \mathbf{x} and \mathbf{n} . For each realization we generated estimates $\hat{\mathbf{x}}$ of \mathbf{x} and \hat{I} of the support set $I = \text{supp}(\mathbf{x})$. We then numerically evaluated the error by averaging over all realizations. We measured the Normalized Mean Square Error (NMSE)

$$\text{NMSE} = \frac{\mathcal{E}[\|\hat{\mathbf{x}} - \mathbf{x}\|_2^2]}{\mathcal{E}[\|\mathbf{x}\|_2^2]},$$

which describes how far the estimate is from the true value, the Average Support Cardinality Error (ASCE)

$$\text{ASCE} = 1 - \frac{\mathcal{E}[\|\hat{I} \cap I\|]}{K},$$

which is a measure of how many elements of the estimated support set that are incorrect. We also measured the average cputime required by algorithms to compute the estimate.

In each simulation we generated 100 realizations of \mathbf{A} , and for each \mathbf{A} we generated 100 realizations of \mathbf{x} and \mathbf{n} .

3.6.2 Comparison of beamformer methods

In the first experiment we compared different beamformer methods to measure their effectiveness. In the simulation we varied K while setting $n = 100$, $m = 25$

and $\text{SNR} = 20$ dB. We compared the standard OMP with the maximum-sidelobe beamformer (OMPb max), the worst case beamformer (OMPb worst case), the average case pseudoinverse beamformer (OMPb pseudoinverse) and the Equiangular Tight Frame from [SV08] (OMPb ETF).

When measuring the NMSE, we found the results shown in Figure 3.7. We get that OMPb worst case actually performed worse than OMP by more than 2 dB for $K \geq 4$ but better than OMP by 1.5 dB for $K = 2$. This is because OMPb worst case is designed to improve the worst case performance and not the average case measured by the NMSE. The ETF beamformer has a performance similar to the standard OMP for $K \geq 5$ and performed better by more than 1 dB for $2 \leq K \leq 4$. Somewhat surprisingly, both the maximum-sidelobe beamformer and the pseudoinverse beamformer showed similar behavior and gave 1.5 dB lower NMSE than OMP for $3 \leq K \leq 7$. However, the pseudoinverse beamformer is to prefer since it is easier to motivate from theory and also easier to compute in practice.

When measuring the ASCE, we found the results shown in Figure 3.8. The ASCE-performance of the methods are similar to the NMSE performance with the exception that the ASCE of OMPb ETF now is worse than that of OMP. OMPb max and OMPb pseudoinverse recovers about 2% more of the support set than OMP for $K \geq 6$ while OMPb ETF recovers about 2% less than the standard OMP. OMPb worst case gives the worst performance and recovers about 7% less indices of the support set than OMP.

The experiments show that OMPb pseudoinverse and OMPb max gives the best performance while OMPb worst case gives a poorer performance than the standard OMP. Next we examine the performance of different Bayesian filtering methods.

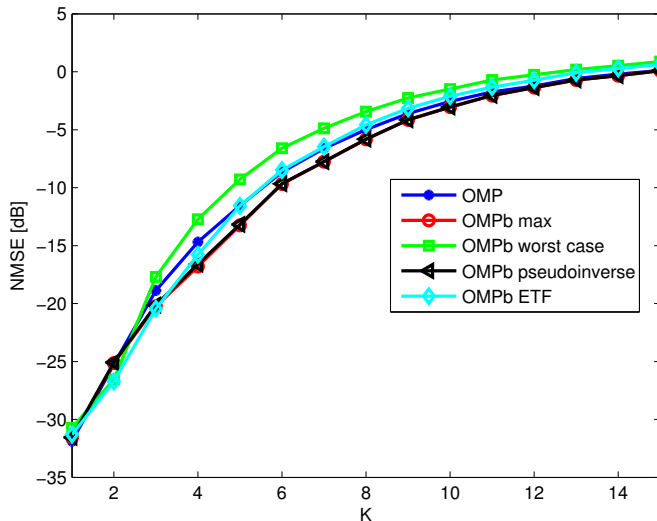


Figure 3.7: NMSE [dB] of beamforming methods for varying K .

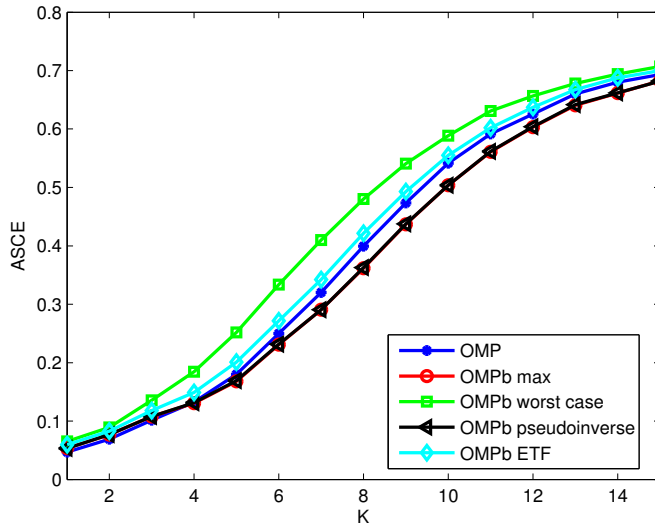


Figure 3.8: ASCE of beamforming methods for varying K .

3.6.3 Comparison of Bayesian filtering methods

In the second experiment we measured the performance of different Bayesian filtering methods for greedy pursuits when varying the number of non-zero elements, K . We compared the standard Orthogonal Matching Pursuit (OMP), the Projection based OMP (POMP), the Conditional prior OMP (CpOMP) and the Projection based CpOMP (CpPOMP). The algorithms POMP and CpPOMP both use $L = K$ in the simulation. In the simulation we set $n = 100$, $m = 25$ and $\text{SNR} = 20$ dB.

When measuring the NMSE we found the results shown in Figure 3.9. We see that CpPOMP and CpOMP gave the smallest error while POMP gave a smaller error than the standard OMP. The NMSE of CpOMP and CpPOMP was about 2 dB lower than the NMSE of OMP for $K \geq 2$ while the NMSE of POMP was 1.5 dB lower than the NMSE of OMP for $K \geq 3$.

When measuring the ASCE we found the results shown in Figure 3.10. We see that CpPOMP and CpOMP gave the smallest error while POMP gave a smaller error than the standard OMP. The ASCE of CpOMP and CpPOMP was about 4% lower than the ASCE of OMP for $K \geq 7$ while the ASCE of POMP was about 3% lower than the ASCE of OMP for $K \geq 7$.

From the experiments we get that CpOMP and CpPOMP give better performance than the standard OMP and POMP. In the next section we examine how the methods compare with OMPb and other methods.

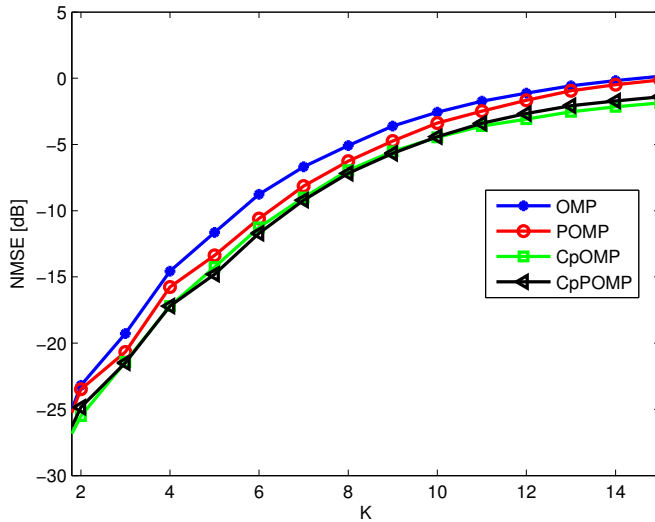


Figure 3.9: NMSE [dB] of different Bayesian filtering methods for varying K .

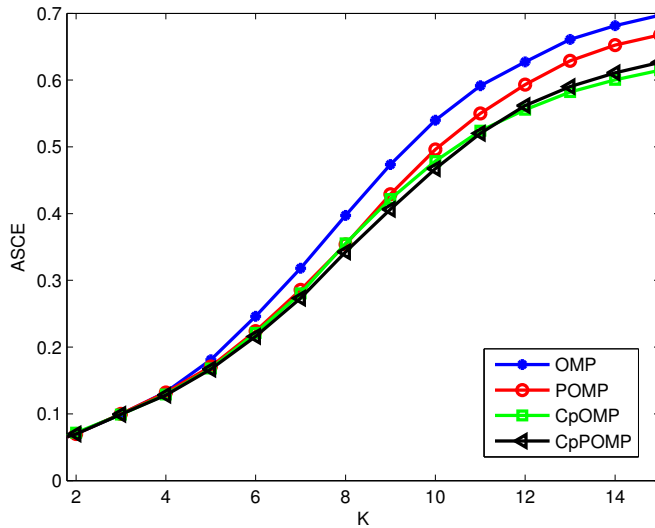


Figure 3.10: ASCE of different Bayesian filtering methods for varying K .

3.6.4 Comparison of Greedy pursuit methods

Here we compare the methods proposed in this chapter against estimation methods from the literature. We compare OMPb, CpOMP, the standard OMP, POMP, RandOMP and the convex Basis Pursuit (BP). Basis pursuit estimates the vector

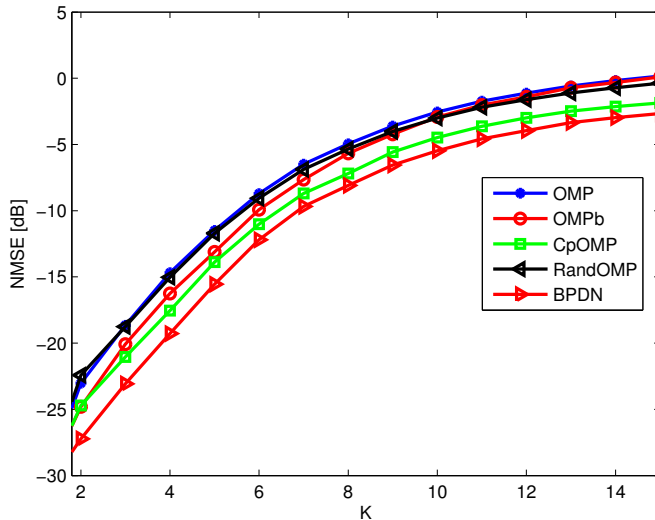


Figure 3.11: NMSE [dB] vs. K for different sparse estimation methods.

\mathbf{x} as

$$\begin{aligned} \hat{\mathbf{x}} &= \arg \min \|\mathbf{x}\|_1 \\ \text{s.t. } &\|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2 \leq \delta \end{aligned}$$

In the simulation we used $\delta = \sigma_n \sqrt{m + 8\sqrt{m}}$ as proposed in [CRT06]. Both RandOMP and BP can give estimates with more or less than K non-zero elements, we therefore chose the support set of the estimates to be the K components of $\hat{\mathbf{x}}$ with largest absolute value.

In the first experiment we varied the number of non-zero components, K for $n = 100$, $m = 25$, $\text{SNR} = 20$ dB and obtained the results shown in Figure 3.11. We find that all methods performed better than OMP with BPDN performing the best. BPDN gave about 4 dB lower NMSE than OMP for $2 \leq K \leq 5$ and 2.8 dB lower NMSE for $K \geq 9$. CpOMP gave more than 2.2 dB improvement over OMP for $3 \leq K \leq 8$ and 1.9 dB improvement for $K \geq 9$. RandOMP gave an improvement of around 0.3 dB for $4 \leq K \leq 8$ and more than 0.45 dB improvement for $K \geq 10$. The performance in terms of ASCE is shown in Figure 3.12. We find that BPDN, CpOMP and RandOMP have very similar ASCE performance while OMPb has better ASCE performance than OMP but worse than the other methods. Finally we measured the average cputime of the algorithms with varying K . The results are shown in Figure 3.13. We find that BPDN is the slowest method while OMP is the fastest method. On average BPDN requires 10^6 times the cputime of OMP to compute the estimate while Randomp requires $10^{2.4} \approx 25$ and CpOMP requires $10^{1.9} \approx 79$ times the cputime of OMP. The second fastest method was OMPb which required $10^{0.4} \approx 2.5$ times the cputime of OMP.

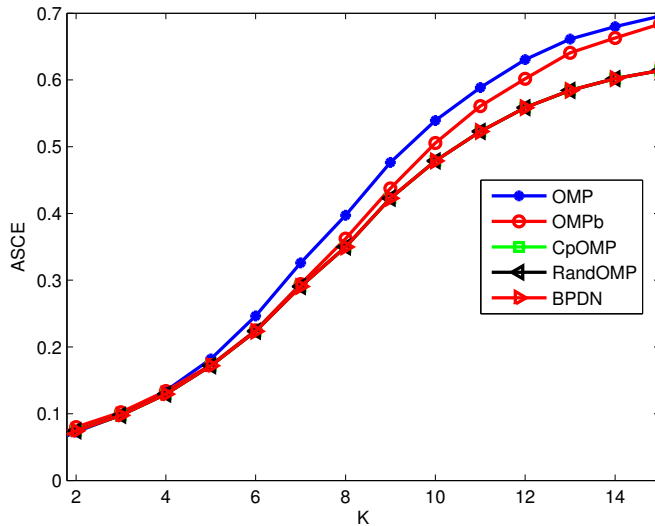


Figure 3.12: ASCE vs. K for different sparse estimation methods.

In the second experiment we set $n = 100$, $K = 5$, SNR = 20 dB and varied the number of measurements m . The results are shown in Figure 3.14. We find that BPDN gave the best performance for $m \leq 30$, CpOMP for $35 \leq m \leq 45$ and OMP and RandOMP for $m \geq 50$. For $10 \leq m \leq 30$, BPDN gave a 4 to 9.5 dB gain in NMSE over OMP and CpOMP a 2.5 to 6.5 dB gain. OMPb gave a 1 to 4.7 dB gain in NMSE over OMP for $20 \leq m \leq 40$. The ASCE and cputime of the algorithms was similar when varying m .

In a third experiment we measured the performance of the algorithms with

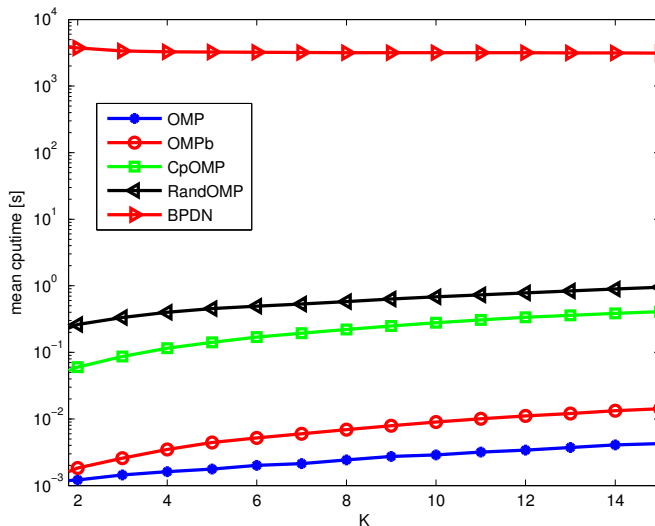


Figure 3.13: Average cputime vs. K for different sparse estimation methods.

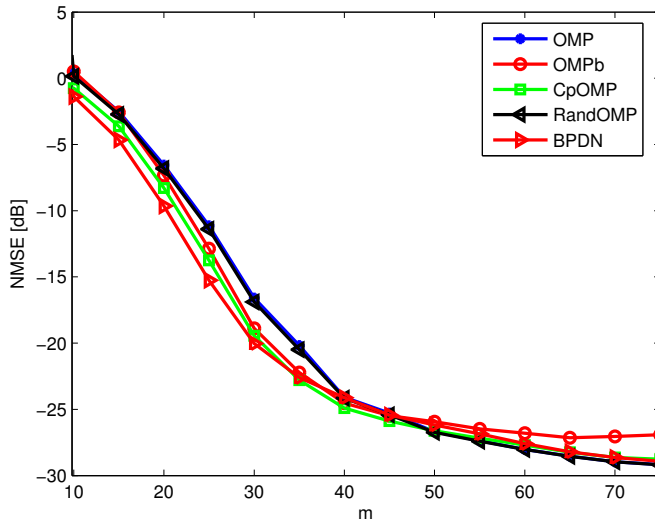


Figure 3.14: NMSE [dB] vs. m for different sparse estimation methods.

varying SNR. In the experiment we set $n = 100$, $m = 25$ and $K = 5$. The results are shown in Figure 3.15. We find that BPDN has the lowest NMSE of all methods with a gain over OMP by 1.3 to 2.4 dB for $\text{SNR} \leq 15$ dB and more than 8 dB for $\text{SNR} \geq 30$ dB. For $\text{SNR} \leq 10$ dB, RandOMP had the best performance of the greedy methods with a gain of 1.3 to 2.3 dB over OMP while for $\text{SNR} \geq 15$, CpOMP had the best performance with a gain of 1.7 to 3.4 dB. For all SNR, OMPb had an NMSE which was 0.6 to 2.1 dB worse than the NMSE of CpOMP.

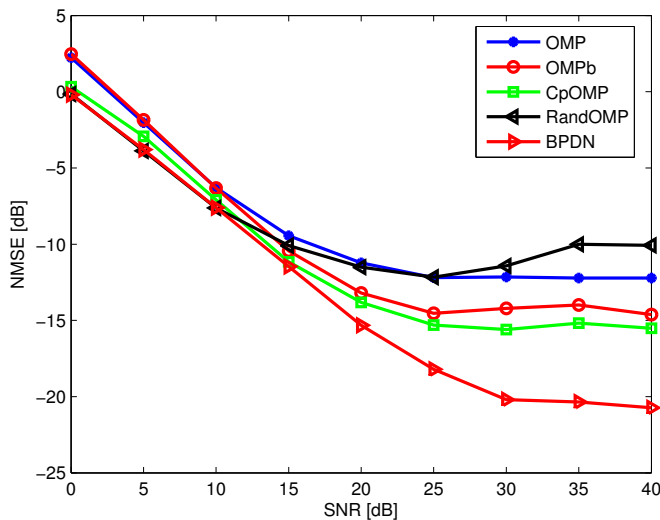


Figure 3.15: NMSE [dB] vs. SNR for different sparse estimation methods.

3.7 Conclusion

In this chapter we examined how greedy search algorithms such as OMP can be improved by improving the step in which new atoms are detected. We first considered the deterministic construction of beamformers and thereafter Bayesian filtering methods which can be described as conditional Wiener filters for sparse estimation. The Bayesian filtering method improves both the detection and estimation of components in the support set. The Bayesian filtering method has the advantage of capturing the random nature of the non-zero elements. For simplicity we here only considered the scenario of uniformly distributed support sets, but the conditional prior method can be extended to non-uniform scenarios. The experiments show that while the improved greedy search methods OMPb and CpOMP are not as effective as BPDN, they are much faster and outperform the standard OMP. The gain in performance comes at the price of higher complexity. CpOMP has a higher complexity than OMP, but lower than RandOMP and OMPb has a complexity which is closer to OMP. This shows that the OMP algorithm can be improved by improving the detection and estimation steps of the algorithm.

Outlier robust relevance vector machine

Measurements are typically assumed to be perturbed by noise. The noise accounts for measurement errors and model imperfections. Often, some measurements are corrupted by extra strong noise. This sparse noise accounts for saturation of sensors, missing values, quantizing errors or impulse bursts as well as datapoints which deviate from the model. In images, for example, the pixels can become saturated by *salt and pepper* noise which turns the pixels black or white. Since the sparse noise can strongly perturb the final estimate (as for the Boston housing dataset in Figure 2.6), it is important to make estimation methods robust against sparse noise. Because of its sparsity, the noise can be removed using sparse estimation techniques.

Most sparse estimation methods can be made robust by treating the sparse noise as additional parameters to be estimated. This approach is often successful, but leads to higher complexity since the parameter vector becomes larger. The *Relevance Vector Machine* (RVM) [Tip01] is a Bayesian method for sparse estimation. Bayesian methods are often preferable since they can estimate both the parameter vector and noise power from the measurements alone. We here show how the RVM can be made robust to sparse noise *without* explicitly estimating the components of the outliers.

4.0.1 System model

We consider the linear system model

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{e} + \mathbf{n}, \quad (4.1)$$

where $\mathbf{y} \in \mathbb{R}^m$ is the observed measurements, $\mathbf{x} \in \mathbb{R}^n$ is a sparse vector (for example weights in regression or sparse signal to estimate in compressed sensing) we wish to estimate, $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a known system matrix (for example, regressors or sampling system) representing the measurement process. Further, $\mathbf{e} \in \mathbb{R}^m$ and $\mathbf{n} \in \mathbb{R}^m$ are sparse and dense noise respectively. We assume that $\|\mathbf{x}\|_0 \ll n$ and $\|\mathbf{e}\|_0 \ll m$ are small and unknown, where $\|\cdot\|_0$ denotes the number of non-zero components

of a vector. The random vectors \mathbf{x} , \mathbf{e} and \mathbf{n} are independent. The model (4.1) is used in e.g. face recognition [WYG⁺09], image denoising [MVC10] and compressed sensing [JXC08].

4.0.2 Prior work

Almost all prior works [MVC10, LDB⁺09, JR10, VKC13] translate (4.1) into the equivalent setup

$$\mathbf{y} = \begin{bmatrix} \mathbf{A} & \mathbf{I}_m \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{e} \end{bmatrix} + \mathbf{n}, \quad (4.2)$$

where \mathbf{I}_m is the $m \times m$ identity matrix, $\begin{bmatrix} \mathbf{A} & \mathbf{I}_m \end{bmatrix}$ acts as the effective system matrix and $\begin{bmatrix} \mathbf{x}^\top & \mathbf{e}^\top \end{bmatrix}^\top$ acts as the effective parameter vector. The RB-RVM of [MVC10] uses the standard RVM approach for (4.2) directly. Hence RB-RVM learns model parameters for all three signals $\mathbf{x} = [x_1, x_2, \dots, x_n]^\top$, $\mathbf{e} = [e_1, e_2, \dots, e_m]^\top$ and \mathbf{n} , and thus estimates both \mathbf{x} and \mathbf{e} jointly.

RVM has high similarity with Sparse Bayesian Learning (SBL) [ZR11, ZR13, WR04, WPR04]. Sparse Bayesian learning has been used for structured sparse signals, for example block sparse signals [ZR11], where the problem of unknown signal block structure was treated using overlapping blocks. The model extension of RB-RVM shown in (4.2) for handling block sparse noise with unknown block structure is straight-forward to derive. However, in our formulation, as we are not estimating the noise explicitly, the use of block sparse noise with unknown block structure is non-trivial.

Further, convex optimization based methods have been used for sparse estimation problems [LDB⁺09, CDS01, CSVS12, ZCJ12]. For example, justice pursuit (JP) [LDB⁺09] uses the optimization technique of the standard basis pursuit denoising method [CDS01], as follows

$$\hat{\mathbf{x}}, \hat{\mathbf{e}} = \arg \min_{\mathbf{x}, \mathbf{e}} \|\mathbf{x}\|_1 + \|\mathbf{e}\|_1 \quad (4.3)$$

$$\text{s.t. } \|\mathbf{y} - \mathbf{A}\mathbf{x} - \mathbf{e}\|_2 \leq \epsilon, \quad (4.4)$$

where $\epsilon > 0$ is a model parameter. For unknown noise power, it is impossible to know ϵ a-priori. We mention that a fully Bayesian setup like the RVM does not require parameters set by a-priori.

4.0.3 Our contribution

Our main contribution is developing a robust RVM using a combined noise model. The method uses fewer parameters than the robust RVM of [MVC10], since the sparse noise is not estimated explicitly. This means that the method has lower computational complexity and also better performance in some instances. As an

extension we also consider the scenario where the signal \mathbf{x} and noise \mathbf{e} are block sparse. By using techniques from [ZR11] we generalize the methods to signals where the position of the blocks are unknown. The main technical contribution is to derive update equations that are used iteratively for estimation of parameters in the new RVM. We refer to the new RVM as the RVM for combined sparse and dense noise (SD-RVM). By an approximate analysis, the SD-RVM algorithm is shown to be equivalent to the minimization of a sparsity inducing cost function. Finally, the performance of SD-RVM is evaluated numerically using examples from compressed sensing, block sparse signal recovery, house price prediction and image denoising. Throughout the paper, we take an approach of comparing SD-RVM vis-a-vis the existing Robust Bayesian RVM (RB-RVM) of [MVC10].

4.1 RVM for combined sparse and dense noise (SD-RVM)

4.1.1 SD-RVM Method

For (4.1), we propose to use a combined model for the noise terms, as follows

$$\mathbf{e} + \mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{B}^{-1}), \quad (4.5)$$

where $\mathbf{B} = \text{diag}(\boldsymbol{\beta})$, $\boldsymbol{\beta} = [\beta_1, \beta_2, \dots, \beta_m]$ and $\beta_j > 0$ for $j \in [m] = \{1, 2, \dots, m\}$. The two noise terms are treated as a single combined noise where each noise component has its own precision. The rationale is that we do not need to separate the two noises. Although our model promotes sparsity in the noise we empirically find that it is able to model both sparse and non-sparse noise.

We model the components of the parameter vector \mathbf{x} as

$$x_i \sim \mathcal{N}(0, \gamma_i^{-1}), \quad (4.6)$$

where $\gamma_i > 0$ and $i \in [n] = \{1, 2, \dots, n\}$. From (4.5) and (4.6) we find that the maximum a posteriori (MAP) estimate of \mathbf{x} is

$$\begin{aligned} \hat{\mathbf{x}} &= \boldsymbol{\Sigma} \mathbf{A}^\top \mathbf{B} \mathbf{y}, \\ \boldsymbol{\Sigma} &= (\boldsymbol{\Gamma} + \mathbf{A}^\top \mathbf{B} \mathbf{A})^{-1}, \end{aligned}$$

where $\boldsymbol{\Gamma} = \text{diag}(\boldsymbol{\gamma})$ and $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \dots, \gamma_n]$.

To promote sparsity in $\hat{\mathbf{x}}$, the RVM assigns prior probabilities to the precisions [Tip01]. A common choice is to assign Gamma distributions

$$p(\gamma_i) = \text{Gamma}(\gamma_i | a, b) = \frac{b^a \gamma_i^{a-1} e^{-b\gamma_i}}{\Gamma(a)}, \quad (4.7)$$

$$p(\beta_j) = \text{Gamma}(\beta_j | c, d) = \frac{d^c \gamma_i^{c-1} e^{-d\beta_j}}{\Gamma(c)}, \quad (4.8)$$

where $a, b, c, d > 0$ are parameters. We will consider parameters for which the distributions are “nearly flat”, i.e. a, c are slightly larger than one and b, d are

slight larger than zero. Through expectation-maximization (EM) we find that the precisions are updated as

$$\gamma_i^{new} = \frac{1 + 2(a - 1)}{\hat{x}_i^2 + \Sigma_{ii} + 2b}, \quad (4.9)$$

$$\beta_j^{new} = \frac{1 + 2(c - 1)}{[\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}]_j^2 + [\mathbf{A}\Sigma\mathbf{A}^\top]_{jj} + 2d}, \quad (4.10)$$

where we use $[\cdot]_{ij}$ to denote the i, j element of a matrix. The derivations of (4.9) and (4.10) are given in Section 4.4.1. Next we show that the model is equivalent to MAP estimation with a sparsity promoting penalty function.

4.1.2 Relation to sparsity promoting penalty function

A common approach to sparse estimation is to estimate \mathbf{x} as

$$\begin{aligned} \hat{\mathbf{x}} = \arg \min_{\mathbf{x}} \frac{\beta}{2} h(\mathbf{n}) + g(\mathbf{x}), \\ \text{subject to } \mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n} \end{aligned} \quad (4.11)$$

where $g(\cdot)$ is a penalty function which promotes sparsity in \mathbf{x} , e.g. the ℓ_1 -norm, and typically $h(\mathbf{n}) = \|\mathbf{n}\|_2^2$. Here we show that SD-RVM corresponds to (4.11) for a certain sparsity promoting penalty function. Since several approximations are made in the derivation of the iterative update equations. It is interesting to see how the approximations affect the sparsity promoting penalty function.

To motivate that the standard RVM is sparsity promoting, one can note that the marginal distribution of x_i is a student-t distribution. For a fixed β (and $\mathbf{e} = \mathbf{0}$), the standard RVM is therefore an iterative method for minimizing [Tip01]

$$\frac{\beta}{2} h(\mathbf{n}) + \left(\frac{1}{2} + a\right) \sum_{i=1}^n \log(x_i^2 + 2b),$$

subject to the constraint $\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}$. The log-sum penalty function can be used as a sparsity promoting cost function, making it plausible that the RVM promotes sparsity.

For the SD-RVM, the precisions are updated by maximizing the log-likelihood

$$\begin{aligned} \mathcal{L} = \log p(\mathbf{y}, \boldsymbol{\gamma}, \boldsymbol{\beta}) = \text{constant} \\ - \frac{1}{2} \log \det(\mathbf{B}^{-1} + \mathbf{A}\boldsymbol{\Gamma}^{-1}\mathbf{A}^\top) \\ - \frac{1}{2} \mathbf{y}^\top (\mathbf{B}^{-1} + \mathbf{A}\boldsymbol{\Gamma}^{-1}\mathbf{A}^\top)^{-1} \mathbf{y} \\ + \sum_{i=1}^n (a \log \gamma_i - b \gamma_i) \\ + \sum_{j=1}^m (c \log \beta_j - d \beta_j). \end{aligned} \quad (4.12)$$

By the matrix determinant lemma [Har97],

$$\det(\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top) = \det(\mathbf{\Sigma}^{-1})\det(\mathbf{\Gamma}^{-1})\det(\mathbf{B}^{-1}).$$

We can approximate (4.12) using that

$$\begin{aligned} \log \det(\mathbf{\Sigma}^{-1}) &\approx \log \det((\mathbf{\Sigma}^{old})^{-1}) \\ &+ \sum_{i=1}^n \Sigma_{ii}^{old} (\gamma_i - \gamma_i^{old}) \\ &+ \sum_{j=1}^m [\mathbf{A}\mathbf{\Sigma}^{old}\mathbf{A}^\top]_{jj} (\beta_j - \beta_j^{old}), \end{aligned} \quad (4.13)$$

where the approximation is to first order in γ and β . We also introduce variables \mathbf{x} and $\tilde{\mathbf{e}}$ through the relation [RKH13]

$$\begin{aligned} \mathbf{y}^\top (\mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top + \mathbf{B}^{-1})^{-1}\mathbf{y} &= \min_{\mathbf{x}, \tilde{\mathbf{e}}} \sum_{i=1}^n \gamma_i x_i^2 + \sum_{j=1}^m \beta_j \tilde{e}_j^2, \\ &\text{such that } \mathbf{A}\mathbf{x} + \tilde{\mathbf{e}} = \mathbf{y} \end{aligned} \quad (4.14)$$

where now $\tilde{\mathbf{e}} = \mathbf{e} + \mathbf{n}$ as in (4.5). The minimization problem then becomes

$$\begin{aligned} \min_{\mathbf{x}, \tilde{\mathbf{e}}, \gamma, \beta} &\sum_{i=1}^n [(x_i^2 + \Sigma_{ii}^{old} + 2b)\gamma_i + (2a - 1)\log(\gamma_i)] + \\ &\sum_{j=1}^m [(e_j^2 + [\mathbf{A}\mathbf{\Sigma}^{old}\mathbf{A}^\top]_{jj} + 2d)\beta_j + (2c - 1)\log(\beta_j)]. \end{aligned} \quad (4.15)$$

such that $\mathbf{A}\mathbf{x} + \tilde{\mathbf{e}} = \mathbf{y}$

By minimizing (4.15) with respect to γ_i and β_j , the problem reduces to

$$\begin{aligned} \min_{\mathbf{x}, \tilde{\mathbf{e}}} &(2a - 1) \sum_{i=1}^n \log(x_i^2 + \Sigma_{ii}^{old} + 2b) \\ &+ (2c - 1) \sum_{j=1}^m \log(\tilde{e}_j^2 + [\mathbf{A}\mathbf{\Sigma}^{old}\mathbf{A}^\top]_{jj} + 2d), \end{aligned} \quad (4.16)$$

such that $\mathbf{A}\mathbf{x} + \tilde{\mathbf{e}} = \mathbf{y}$

where we have ignored additive constants. Because of the approximations, the constants Σ_{ii}^{old} and $[\mathbf{A}\mathbf{\Sigma}^{old}\mathbf{A}^\top]_{jj}$ make the penalty function penalize different components of \mathbf{x} and $\tilde{\mathbf{e}}$ differently. In a similar fashion it can be shown that the standard RVM and RB-RVM are also equivalent to similar penalty functions. A two-dimensional illustration of the penalty function is shown in Figure 4.1.

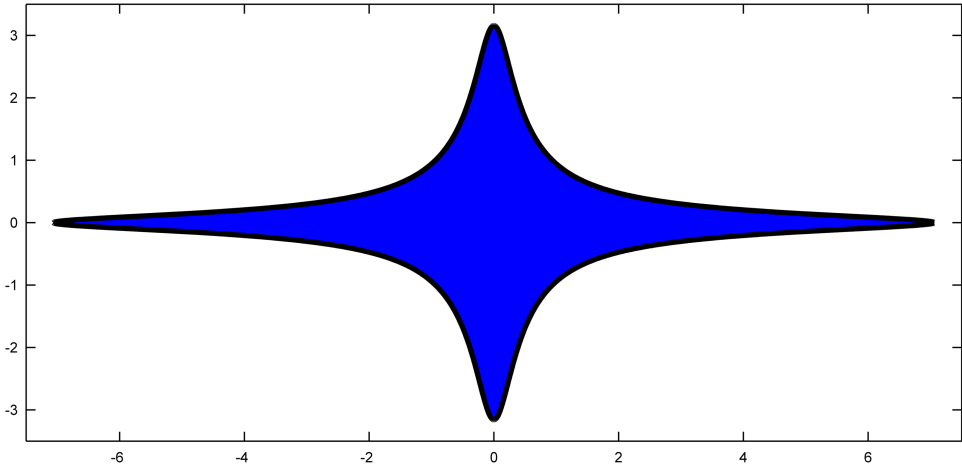


Figure 4.1: The non-symmetric *log-ball* consisting of points $(x_1, x_2) \in \mathbb{R}^2$ such that $\log(x_1^2 + 0.02) + \log(x_2^2 + 0.1) \leq 0$. SD-RVM is equivalent to finding the smallest non-symmetric log-ball that intersects the linear subspace $\mathbf{Ax} + \tilde{\mathbf{e}} = \mathbf{y}$.

4.1.3 Computational complexity

In this section we quantify the computational complexity of the SD-RVM. The complexity is given in number of multiplications per iteration, since multiplications are typically the most demanding numerical operation [TBI97] and the number of iterations depends on the stopping criterion used. We give the complexity for the naive implementation of the algorithm. Each iteration of SD-RVM requires $\mathcal{O}(n^3)$ multiplications to compute the matrix Σ using Gauss-Jordan elimination [TBI97]. Updating the precisions requires $\mathcal{O}(nm)$ flops since the residual $\mathbf{y} - \mathbf{Ax}$ needs to be computed. Hence the computational complexity of SD-RVM is

$$\mathcal{O}(nm + n^3) = \mathcal{O}(n \cdot \max(m, n^2)).$$

It is interesting to compare the complexity of SD-RVM to that of RB-RVM. Again with the assumption of a naive implementation, each iteration of RB-RVM requires the inversion of a $(n+m) \times (n+m)$ matrix to compute Σ_{RB} . Updating the precisions requires $\mathcal{O}(nm)$ flops and hence the computational complexity of RB-RVM is

$$\mathcal{O}(\max(nm, (n+m)^3)) = \mathcal{O}((n+m)^3).$$

We thus find that the numerical complexity of SD-RVM is smaller than that of RB-RVM. In Section 4.3.1 we provide numerical evaluations to quantify algorithm run time requirements that confirm that SD-RVM is typically faster than RB-RVM.

4.2 SD-RVM for block sparse signals

In many applications, the signal of interest is block sparse [BCDH10], i.e. the signal can be partitioned into blocks of which many blocks only contain zeros. In some situations the positions and size of the blocks is known while in other situations they are unknown [ZR11].

4.2.1 Known block structure

To describe a block sparse signal $\mathbf{x} \in \mathbb{R}^n$ with known block structure we partition $[n] = \{1, 2, \dots, n\}$ into blocks

$$[n] = \{1, 2, \dots, n\} = I_1 \cup I_2 \cup \dots \cup I_p,$$

where $|I_i| = n_i$ and $I_i \cap I_j = \emptyset$ for $i \neq j$. The signal is block sparse when only a few blocks of the signal are non-zero. The component-wise SD-RVM generalizes to this scenario by requiring that the precisions are equal in each block, i.e. we choose the prior distribution for the components of block I_i to be

$$\mathbf{x}_{I_i} \sim \mathcal{N}(\mathbf{0}, \gamma_i^{-1} \mathbf{I}_{n_i}).$$

where $\mathbf{x}_{I_i} \in \mathbb{R}^{n_i}$ denotes the vector consisting of the components of \mathbf{x} with indices in I_i .

Similarly we can partition the components of the sparse noise $\tilde{\mathbf{e}} \in \mathbb{R}^m$ into blocks

$$[m] = \{1, 2, \dots, m\} = J_1 \cup J_2 \cup \dots \cup J_q,$$

where $|J_j| = m_j$, $J_j \cap J_i = \emptyset$ for $i \neq j$ and the block J_j of \mathbf{e} is given the prior distribution

$$\mathbf{e}_{J_j} \sim \mathcal{N}(\mathbf{0}, \beta_j^{-1} \mathbf{I}_{m_j}).$$

As before, the precisions are given gamma distributions (4.7) as priors. Using this model, we derive the update equations of precisions as below

$$\gamma_i^{new} = \frac{n_i + 2(a-1)}{\|\hat{\mathbf{x}}_{I_i}\|_2^2 + \text{tr}(\boldsymbol{\Sigma}_{I_i}) + 2b}, \quad (4.17)$$

$$\beta_j^{new} = \frac{m_j + 2(c-1)}{\|(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}})_{J_j}\|_2^2 + \text{tr}([\mathbf{A}\boldsymbol{\Sigma}\mathbf{A}^\top]_{J_j}) + 2d}, \quad (4.18)$$

where $\boldsymbol{\Sigma}_{I_i}$ denotes the $n_i \times n_i$ submatrix of $\boldsymbol{\Sigma}$ formed by elements appropriately indexed by I_i . By setting $I_i = \{i\}$ and $J_j = \{j\}$ we obtain the update equations for component-wise sparse signal and noise. We see that (4.18) reduces to the update equations of the standard RVM when $I_i = \{i\}$ and $J_j = J = [m]$. The derivation of the update equations (4.17) and (4.18) is found in Section 4.4.3.

4.2.2 Unknown block structure

In some situations the signal can have an unknown block structure, i.e. the signal is block sparse, but the dimensions and positions of the blocks are unknown. This scenario can be handled by treating the signal as a superposition of block sparse signals [ZR11] (see illustration in Figure 4.2). This approach also describes the scenario (4.1) when \mathbf{e} is component wise sparse and \mathbf{n} is dense (e.g. Gaussian). The precision of each component is then a combination of the precisions of the blocks to which the component belongs. Let γ_i be the precision of the component x_i and $\tilde{\gamma}_k$ be the precision of block I_k . We model the signal as

$$\begin{aligned} x_i &\sim \mathcal{N}(0, \gamma_i^{-1}), \\ \gamma_i^{-1} &= \sum_{k, i \in I_k} \tilde{\gamma}_k^{-1}. \end{aligned} \quad (4.19)$$

We model the noise in a similar way with precisions β_j for component j and precisions $\tilde{\beta}_l$ for the block with support J_l . To promote sparsity, the precisions of the underlying blocks are given gamma distributions as priors. In each iteration we update the underlying precisions $\tilde{\gamma}_k$. The component-wise precisions are then updated using (4.19). With this model, the update equations for the precisions become

$$\tilde{\gamma}_k^{new} = \frac{\frac{1}{\tilde{\gamma}_k} \text{tr}(\mathbf{\Gamma}_k) + 2(a-1)}{\frac{1}{\tilde{\gamma}_k^2} \|\mathbf{\Gamma}_i \hat{\mathbf{x}}\|_2^2 + \frac{1}{\tilde{\gamma}_k} \text{tr}(\mathbf{\Gamma}_k \mathbf{\Sigma} \mathbf{\Gamma}_k) + 2b}, \quad (4.20)$$

$$\tilde{\beta}_l^{new} = \frac{\frac{1}{\tilde{\beta}_l} \text{tr}(\mathbf{B}_l) + 2(c-1)}{\frac{1}{\tilde{\beta}_l^2} \|\mathbf{B}_l(\mathbf{y} - \mathbf{A} \hat{\mathbf{x}})\|_2^2 + \frac{1}{\tilde{\beta}_l} \text{tr}(\mathbf{B}_l \mathbf{A}^\top \mathbf{\Sigma} \mathbf{A} \mathbf{B}_l) + 2d}, \quad (4.21)$$

where $\mathbf{\Gamma}_k$ is the diagonal matrix with $[\mathbf{\Gamma}_k]_{ii} = \gamma_i$ if $i \in I_k$ and $[\mathbf{\Gamma}_k]_{ii} = 0$ otherwise. We denote the corresponding matrix for β_l by \mathbf{B}_l . The component-wise precisions are updated using (4.19) and similar for β_j .

We see that when the underlying blocks are disjoint, then $\gamma_i = \tilde{\gamma}_k$ for all $i \in I_k$ and $\beta_j = \tilde{\beta}_l$ for all $j \in J_l$. The update equations then reduce to the update equations (4.18) for the block sparse model with known block structure.

4.3 Simulation experiments

In this section we evaluate the performance of SD-RVM compared to other methods using several scenarios – for simulated and real signals. For simulated signals, we considered the sparse and block sparse recovery problem in compressed sensing. For real signals, we considered prediction of house prices using the Boston housing dataset [AN07] and denoising of images contaminated by *salt and pepper* noise. We used the cvx toolbox [GBY08] to implement JP.

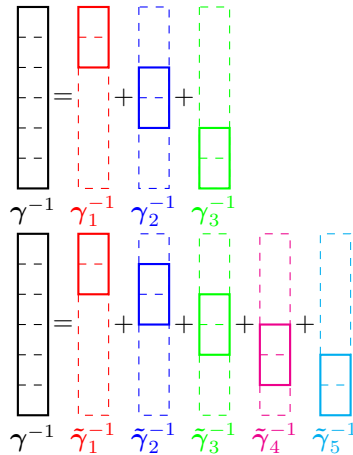


Figure 4.2: Illustration of non-overlapping and overlapping block parameterizations.

4.3.1 Compressed sensing

In compressed sensing we want to estimate a sparse vector $\mathbf{x} \in \mathbb{R}^n$ from m measurements (4.1), where $m \ll n$. We evaluated the average performance by randomly generating each test case. First we generated measurement matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ by drawing their components from a $\mathcal{N}(0, 1)$ distribution and rescaling the column vectors to unit norm. We selected the positions of the active components of \mathbf{x} and \mathbf{e} uniformly at random and draw their values from $\mathcal{N}(0, 1)$. The dense noise \mathbf{n} was generated by a $\mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I}_m)$ distribution. We compared SD-RVM to the standard RVM, RB-RVM and JP. For JP (4.3) we assumed σ_n is known and used $\epsilon = \sigma_n \sqrt{m + 2\sqrt{2m}}$ as proposed in [CRT06].

In the simulations we varied the measurement rate m/n (ratio of the number of measurements and the signal dimension) for measurements without outliers and with 5% outliers. We chose $n = 100$ and fixed the signal-to-dense-noise-ratio (SDNR)

$$\text{SDNR} = \frac{\mathcal{E}[\|\mathbf{Ax}\|_2^2]}{\mathcal{E}[\|\mathbf{n}\|_2^2]} = \frac{\|\mathbf{x}\|_0}{m\sigma_n^2},$$

to 20 dB. By generating 100 measurement matrices and 100 vectors \mathbf{x} and \mathbf{e} for each matrix we numerically evaluated the Normalized Mean Square Error (NMSE)

$$\text{NMSE} = \frac{\mathcal{E}[\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2]}{\mathcal{E}[\|\mathbf{x}\|_2^2]}.$$

Note that it is trivial to obtain an NMSE of 0 dB (i.e. $\text{NMSE} = 1$) by setting $\hat{\mathbf{x}} = \mathbf{0}$. An estimate with an NMSE higher than 0 dB is therefore not informative. The simulation results for measurements without outliers are shown in Figure 4.3 and for measurements with 10% outliers in Figure 4.4.

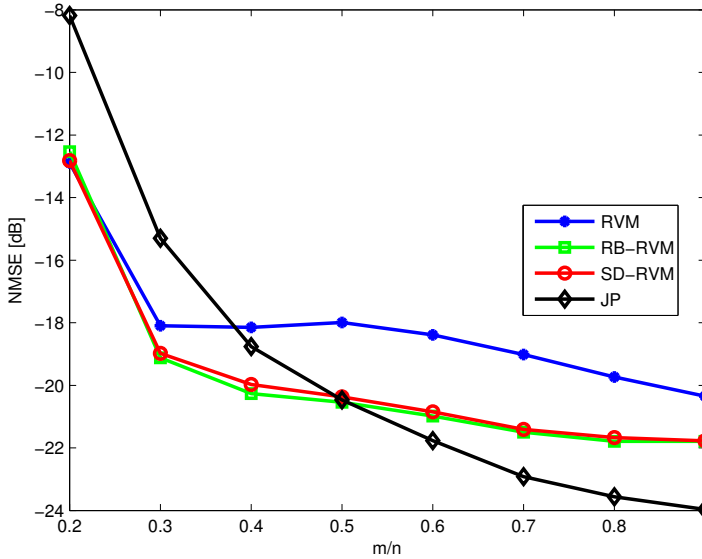


Figure 4.3: NMSE vs. m/n for outlier free measurements.

In the experiments we found that SD-RVM had a performance comparable to that of RB-RVM. SD-RVM performed slightly better than RB-RVM (about 0.1 dB) for $m/n \geq 0.3$. JP performed worse than RB-RVM and SD-RVM for $m/n \leq 0.4$, but better for $m/n \geq 0.5$. The standard RVM showed good performance for measurements without sparse noise, but performed worse than the other methods.

When 10% of the measurements were contaminated by sparse noise, the RVM algorithm failed to give a good estimate and gave an NMSE larger than zero. SD-RVM performed better than RB-RVM for $m/n \geq 0.4$ with a gain of 0.25 to 0.75 dB in NMSE. The experiment shows that the performance of SD-RVM and RB-RVM degrades with 6 dB for $m/n = 0.2$ and 3 dB for $m/n = 0.4$ when sparse noise is introduced. The performance of JP degrades with 1.5 to 4.6 dB when sparse noise is introduced.

We also measured the cputime required for each algorithm to estimate the time needed for each algorithm to give an estimate. We measured the cputime by randomly generating the compressed sensing problem with a $n = 100$ dimensional vector \mathbf{x} with $K = 5$ non-zero components, $m = 35$ measurements with $K_w = 4$ measurements contaminated by sparse noise. From 625 realization we found the histogram of cputimes shown in Figure 4.5. The maximum, minimum, mean and median cputimes are shown in Table 4.1. We found that the runtimes of the RVM algorithms (the standard RVM, RB-RVM and SD-RVM) were longer than the runtime of JP. SD-RVM was fastest of the RVM methods while RB-RVM was the slowest. The experiment shows that SD-RVM is faster than RB-RVM even though

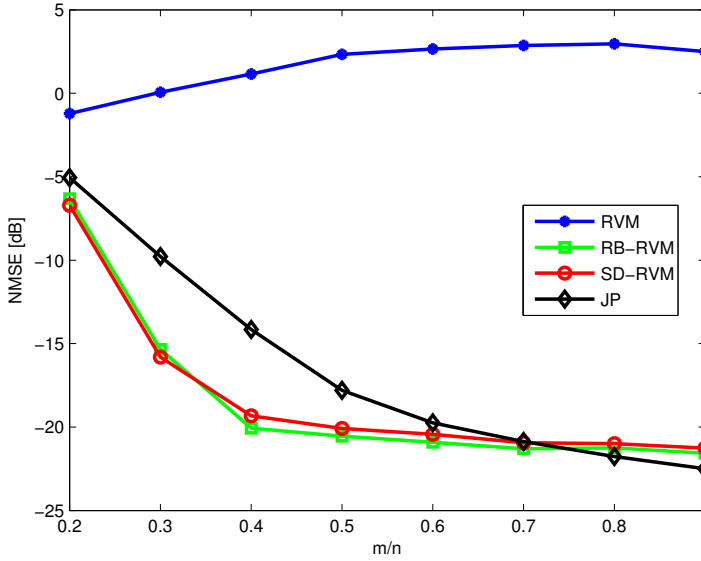


Figure 4.4: NMSE vs. m/n for 5% outliers contaminated measurements.

the algorithms have comparable NMSE performance.

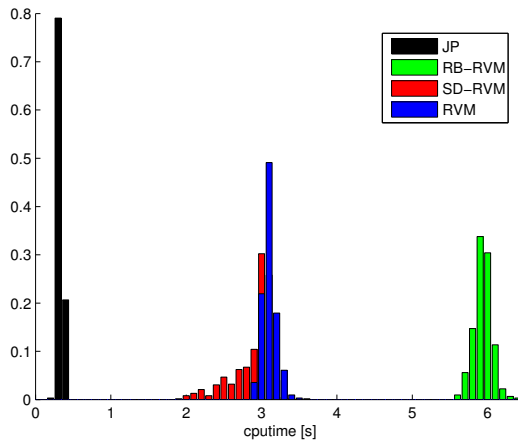


Figure 4.5: Histogram of cputimes for the different algorithms and the compressed sensing problem. Time is in seconds.

Table 4.1: Maximum, minimum, mean and median cputime of the different estimation algorithms for $n = 100$, $m = 35$, $K = 5$, $K_w = 4$ non-zero sparse noise components and 625 problem realizations. Time is in seconds.

Algorithm	Max	Min	Mean	Median
RVM	3.58	2.87	3.11	3.10
RB-RVM	6.40	5.63	5.94	5.94
SD-RVM	3.37	1.91	2.90	2.99
JP	0.44	0.25	0.32	0.32

4.3.2 Block sparse signals

Estimating block sparse signals is closely related to the component-wise sparse recovery problem [BCDH10]. For block sparse signals, the signal components are partitioned into blocks and only a few blocks contain non-zero components. A Bayesian method for block sparse signals is block Sparse Bayesian Learning (BSBL) in which the problem of unknown block structure can be solved by using several overlapping blocks (as in Section 4.2.2) [ZR11,ZR13].

Justice Pursuit can be modified to the block sparse case in a similar way as BSBL. For known block structure, JP becomes the standard block sparse promoting sum of ℓ_2 norms [SPH09]. Let n_{min} be the minimum block length in \mathbf{x} and m_{min}

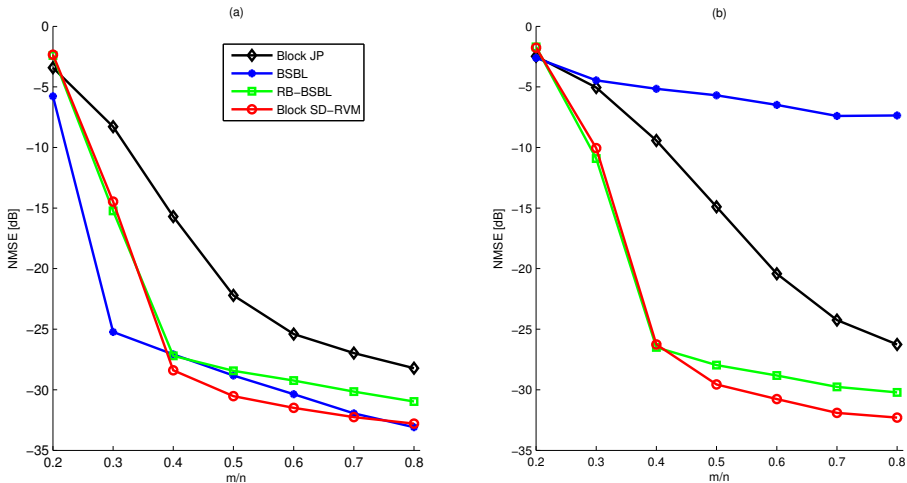


Figure 4.6: NMSE vs. m/n for signals with known block structure. In (a) no outliers are present while in (b) 5% outliers are present.

be the minimum block length in \mathbf{e} . By defining matrices

$$\begin{aligned}\mathbf{Z} &= [\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_{n-n_{\min}+1}] \in \mathbb{R}^{n_{\min} \times (n-n_{\min}+1)}, \\ \mathbf{W} &= [\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_{m-m_{\min}+1}] \in \mathbb{R}^{m_{\min} \times (m-m_{\min}+1)},\end{aligned}$$

the block-JP algorithm for unknown block structure can be written as

$$\begin{aligned}\min_{\mathbf{x}, \mathbf{e}} \quad & \sum_{l=1}^{n-n_{\min}+1} \|\mathbf{z}_l\|_2 + \sum_{a=1}^{m-m_{\min}+1} \|\mathbf{w}_a\|_2, \\ \text{subject to } x_i &= \sum_{l=1+(i-n_{\min})_+}^{\min(i, n-n_{\min}+1)} Z_{i+1-l, l}, \quad 1 \leq i \leq n, \\ e_j &= \sum_{a=1+(j-m_{\min})_+}^{\min(j, m-m_{\min}+1)} W_{j+1-a, a}, \quad 1 \leq j \leq m, \\ & \|\mathbf{y} - \mathbf{A}\mathbf{x} - \mathbf{e}\|_2 \leq \epsilon\end{aligned}\tag{4.22}$$

where $(x)_+ = \max(x, 0)$ and we set $\epsilon = \sigma_n \sqrt{m + \sqrt{8m}}$ as before.

To numerically evaluate the performance of the block sparse algorithms we varied the measurement rate m/n for measurements with 5% sparse noise. We set the signal dimension to $n = 100$ and fixed the SDNR to 20 dB. We divided the signal \mathbf{x} into 20 blocks of equal size of which 3 blocks were non-zero. The sparse noise consisted of blocks with 5 components in each block. In the sparse noise, 5% of the blocks were active. For known block structure, the blocks were chosen uniformly at random from a set of predefined and non-overlapping blocks while for unknown block structure, the first component of each block was chosen uniformly at random, making it possible for the blocks to overlap. The active components of the signal and the sparse noise were drawn from $\mathcal{N}(0, 1)$. By generating 50 measurement matrices \mathbf{A} and 50 signals \mathbf{x} and sparse noises \mathbf{e} for each matrix we numerically evaluated the NMSE.

For known block structure we found that block SD-RVM gave 1.1 to 2.2 improvement in NMSE versus RB-BSBL and 4.5 to 12.6 dB improvement versus block JP for $m/n \geq 0.4$ when no outliers are present. The non-robust BSBL method gave a lower NMSE than the other methods for $m/n < 0.4$ and a performance between SD-RVM and RB-RVM for $m/n \geq 0.4$. For measurements with outliers, the improvement of SD-RVM was 1.6 to 2.2 dB versus RB-RVM and 6 to 14.7 dB versus block JP for $m/n \geq 0.5$. The BSBL method performed worse when outliers were present with an NMSE of about -4.5 to -7.4 dB. The NMSE of the block sparse SD-RVM was lower than the NMSE of block JP by more than 4.7 dB for $m/n \geq 0.3$. The results are shown in Figure 4.6.

For unknown block structure we found that BSBL gave the best performance when no outliers were present. Block SD-RVM and block JP had similar performance with block SD-RVM performing better for $m/n < 0.4$ and block JP

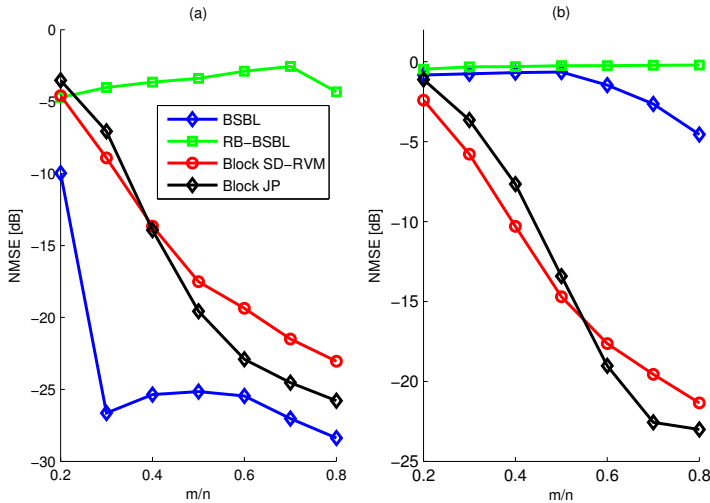


Figure 4.7: NMSE vs. m/n for signals with unknown block structure. In (a) no outliers are present and in (b) 5% outliers are present in measurements.

performing better for $m/n > 0.4$. Somewhat unexpected, RB-BSBL gave a poor performance with an NMSE of -2.5 to -4.7 dB. This shows that it is non-trivial to make BSBL robust against sparse noise. When outliers were present, the performance of BSBL and RB-BSBL worsened to give an NMSE of about 0 dB. SD-RVM and block JP had similar performance with block SD-RVM performing better for $m/n \leq 0.5$ while block JP performed better for $m/n \geq 0.6$. The results are shown in Figure 4.7.

4.3.3 House price prediction

A real world example of sparse regression with sparse noise is the prediction of house prices. The sparse regression vector signifies that many features are redundant and does not influence the final price significantly. The sparse noise represents the fact that some houses, e.g. very expensive houses and very inexpensive houses, does not follow the main trend but can be treated as outliers. The Boston housing dataset [AN07, HR78] consists of 506 house prices in suburbs of Boston together with the values of 13 features for each house (see Table 4.2). The problem is to predict the median house price for part of the dataset (test data) by using the complement dataset (training data).

To predict unseen house prices, we make the simplified assumption that the house prices are a linear function of the feature values. We then model the house prices as

$$p_i = \mathbf{w}_i^T \mathbf{x} + n_i + e_i,$$

Table 4.2: Features in the Boston housing dataset.

Number	Feature
1.	CRIM: per capita crime rate by town
2.	ZN: proportion of residential land zoned for lots over 25,000 sq.ft.
3.	INDUS: proportion of non-retail business acres per town
4.	CHAS: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise)
5.	NOX: nitric oxides concentration (parts per 10 million)
6.	RM: average number of rooms per dwelling
7.	AGE: proportion of owner-occupied units built prior to 1940
8.	DIS: weighted distances to five Boston employment centres
9.	RAD: index of accessibility to radial highways
10.	TAX: full-value property-tax rate per \$10,000
11.	PTRATIO: pupil-teacher ratio by town
12.	B: $1000(B_k - 0.63)^2$ where B_k is the proportion of blacks by town
13.	LSTAT: % lower status of the population
14.	MEDV: Median value of owner-occupied homes in \$1000's

where p_i is the price of house i , $\mathbf{w}_i \in \mathbb{R}^{13}$ contains the feature values of house i , $\mathbf{x} \in \mathbb{R}^{13}$ is the regression vector, n_i is (Gaussian) noise and e_i is sparse noise.

We arrange the features of the houses in the test set into a matrix \mathbf{W}_{test} . Given an estimate $\hat{\mathbf{x}}$ of the regression vector, we estimate the median house price as

$$\hat{m} = \text{median}(\mathbf{W}_{test}^\top \hat{\mathbf{x}}).$$

We used a fraction ρ of the dataset as training data and the rest as test set. In each problem realization we uniformly at random chose the samples for the training set and the test set. We then estimated the regression vector using the different methods and computed the estimated median value. We computed the mean absolute error ε , median absolute error ε_m and mean cputime T (in seconds) over 100 realizations.

We found that SD-RVM gave 4.5% to 12% lower mean absolute error than RB-RVM except for $\rho = 0.7$ and 25% to 44% lower error than RVM (see Table 4.3). SD-RVM was also 96% to 98% faster than RB-RVM.

For $\rho = 0.7$ we also examined the relevance of the different features by computing the mean absolute estimate $\mathcal{E}[|\hat{x}_i|]$. A high mean absolute value indicates that the feature is non-zero in many problem realizations while a low mean absolute value indicates that the feature is small or zero in most realizations. The results are shown in Figure 4.8. We find that RB-RVM and SD-RVM attribute similar importance to the different features. They estimate the number of rooms to be the most important feature followed by the pollution in the area and if the house is close to the Charles River. The RVM and least squares approaches reached similar conclusions.

Table 4.3: Prediction of median houseprice using the Boston Housing dataset. Mean absolute error ME, median absolute error MED and mean cputime T (in seconds), for different fractions, ρ , of the dataset used as training set.

ρ	RVM			RB-RVM			SD-RVM		
	ME	MED	T	ME	MED	T	ME	MED	T
0.3	0.75	0.71	0.17	0.44	0.40	8.10	0.42	0.35	0.32
0.4	0.69	0.67	0.23	0.49	0.43	15.17	0.46	0.41	0.43
0.5	0.76	0.73	0.23	0.49	0.47	22.63	0.43	0.39	0.63
0.6	0.67	0.63	0.22	0.51	0.46	30.80	0.48	0.44	0.73
0.7	0.78	0.80	0.26	0.51	0.44	57.37	0.58	0.56	1.26

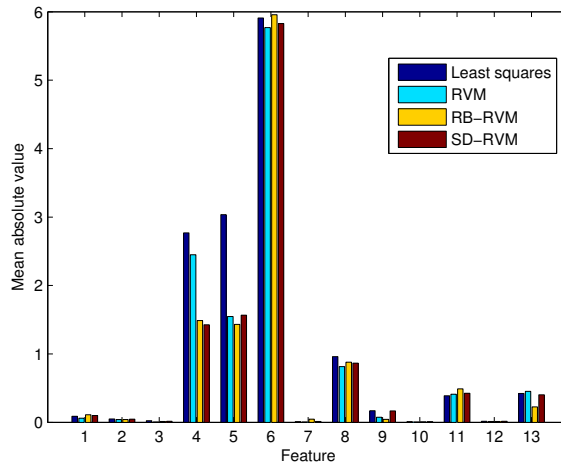


Figure 4.8: Mean absolute value, $\mathcal{E}[|\hat{x}_i|]$, of the regression parameters for $\rho = 0.7$ and the different estimation methods.

4.3.4 Image denoising

In images, pixels can sometimes become strongly disturbed by noise. This shows as completely black or completely white pixels in the image. Such noise is sometimes called *salt and pepper noise* and is because of e.g. analog-to-digital conversion, bit quantization errors or dead pixels. Fortunately, the number of corrupted pixels is often small, allowing sparse techniques to be used to denoise the image. A gray scale image (represented in double-precision) can be modeled as a matrix with elements in the interval from 0 to 1. Salt and Pepper [MVC10] noise makes some pixels black (0) or white (1). To denoise a image we need to reconstruct the corrupted pixels using the neighboring pixels. Many algorithms has been developed to denoise images with salt and pepper noise. One of the basic algorithms is the median filter.

The median filter estimates the value of each pixel by the median in a square patch. In our simulation we used a 3×3 patch since it gave the smallest error. Another method is to model the image in each patch using an image kernel [TFM06]. This amounts to modeling a pixel y_i as

$$y_i = \beta_0 + \beta_1^\top (\mathbf{x} - \mathbf{x}_i) + \beta_2^\top \text{vech}((\mathbf{x} - \mathbf{x}_i)(\mathbf{x} - \mathbf{x}_i)^\top) + n_i,$$

where n_i is noise, \mathbf{x} is the position of the central pixel in the patch, \mathbf{x}_i is the position of pixel i , β_0 , β_1 and β_2 are parameters to be estimated and vech is the *half-vectorization operator* for symmetric matrices [TFM06], e.g.

$$\text{vech} \left(\begin{pmatrix} a & b \\ b & c \end{pmatrix} \right) = \begin{pmatrix} a \\ b \\ c \end{pmatrix}.$$

Given the regression parameters, the value of the central pixel is estimated as $\hat{y} = \hat{\beta}_0$. Since pixels close to the central pixel are more important, the errors are weighted by a kernel, $K(\mathbf{x}, \mathbf{x}_i)$. The estimation problem thus becomes

$$\min_{\beta_0, \beta_1, \beta_2} \sum_{i=1}^P \left| y_i - \beta_0 - \beta_1^\top (\mathbf{x} - \mathbf{x}_i) - \beta_2^\top \text{vech}((\mathbf{x} - \mathbf{x}_i)(\mathbf{x} - \mathbf{x}_i)^\top) \right|^2 K(\mathbf{x}, \mathbf{x}_i),$$

where a standard kernel is [TFM06]

$$K(\mathbf{x}, \mathbf{x}_i) = \exp(-\|\mathbf{x} - \mathbf{x}_i\|_2^2 / r^2) (1 + \mathbf{x}^\top \mathbf{x}_i)^p,$$

the kernel is a composition of a Gaussian and polynomial kernel [MVC10, TFM06]. In the simulation we used a 5×5 patch, $r = 2.1$ and $p = 1$ as in [MVC10]. As the kernel model is a deterministic model, we need to reinterpret the model in order to use the Bayesian methods.

The image denoising model can be interpreted as that we observe measurements

$$\tilde{y}_i = \sqrt{K(\mathbf{x}, \mathbf{x}_i)} y_i = \sqrt{K(\mathbf{x}, \mathbf{x}_i)} \left(\beta_0 + \beta_1^\top (\mathbf{x} - \mathbf{x}_i) + \beta_2^\top \text{vech}((\mathbf{x} - \mathbf{x}_i)(\mathbf{x} - \mathbf{x}_i)^\top) \right) + n_i + e_i, \quad (4.23)$$

where some elements have been subjected to sparse noise. To avoid over-fitting, it is beneficial to promote sparsity in $[\beta_0, \beta_1^\top, \beta_2^\top]^\top$ [BDE09, MES08, EA06]. We can thus treat the image denoising problem as finding a sparse solution to (4.23).

To evaluate the performance of the median filter, the RVM, the RB-RVM and the SD-RVM, we added ρ percent of salt and pepper noise in 7 different images (the

Table 4.4: Mean cputime (in seconds) for denoising images corrupted by salt and pepper noise averaged over 7 images.

Algorithm	Mean cputime
Median filter	1.27
RVM	4673
RB-RVM	1635
SD-RVM	672

standard images Boat, Baboon, Barbara, Elaine, House, Lena and Peppers used in image processing) and denoised them. To reduce the length of the simulation we cropped the images to a quarter in size. The corrupted pixels were set to either black or white with equal probability. For the RVM, RB-RVM and SD-RVM, the value of the central pixel was estimated by forming a 5×5 square patch around each pixel.

For the image denoising problem we compared the algorithms by computing the Peak Signal to Noise Ratio (PSNR)

$$\text{PSNR} = -10 \cdot \log_{10} \left(\frac{\mathcal{E}[\|\mathbf{X} - \hat{\mathbf{X}}\|_F^2]}{\mathcal{E}[\max_{i,j} |X_{ij}|^2](pq)} \right),$$

where the size of the image is $p \times q$ and the expectation is taken over the different images and over different noise realizations for each image. All images in the simulation were of size $p = q = 128$ with $\max_{i,j} |X_{ij}|^2 = 1$. Figure 4.9 shows one realization of the problem, where for the first image (the boat image) SD-RVM gives the highest PSNR, for the second image (the Elaine image) the RB-RVM gave the highest PSNR and for the third image (the Peppers image) the median filter gave the highest PSNR. In the simulations we varied ρ and used 3 noise realizations for each image. The mean PSNR of the algorithms is shown in Figure 4.10.

We found that the median filter, the RB-RVM and SD-RVM gave the same PSNR for $\rho \leq 0.1$ while SD-RVM outperformed RB-RVM and the median filter for $\rho \geq 0.2$. The PSNR of SD-RVM was 0.5 to 1.3 dB higher than that of the median filter for $\rho \geq 0.2$ and 0.8 to 1.3 dB higher than that of the RB-RVM for $\rho \geq 0.3$. The mean cputime of SD-RVM was 41% of the mean cputime of RB-RVM (see Table 4.4) while the median filter was by far the fastest method.

4.4 Derivation of update equations

4.4.1 Derivation of update equations for SD-RVM

To update the precisions we maximize the distribution $p(\mathbf{y}, \boldsymbol{\gamma}, \boldsymbol{\beta}) = p(\mathbf{y}|\boldsymbol{\gamma}, \boldsymbol{\beta})p(\boldsymbol{\gamma})p(\boldsymbol{\beta})$ (obtained by marginalizing over \mathbf{x}), with respect to γ_i and β_j . The precisions are

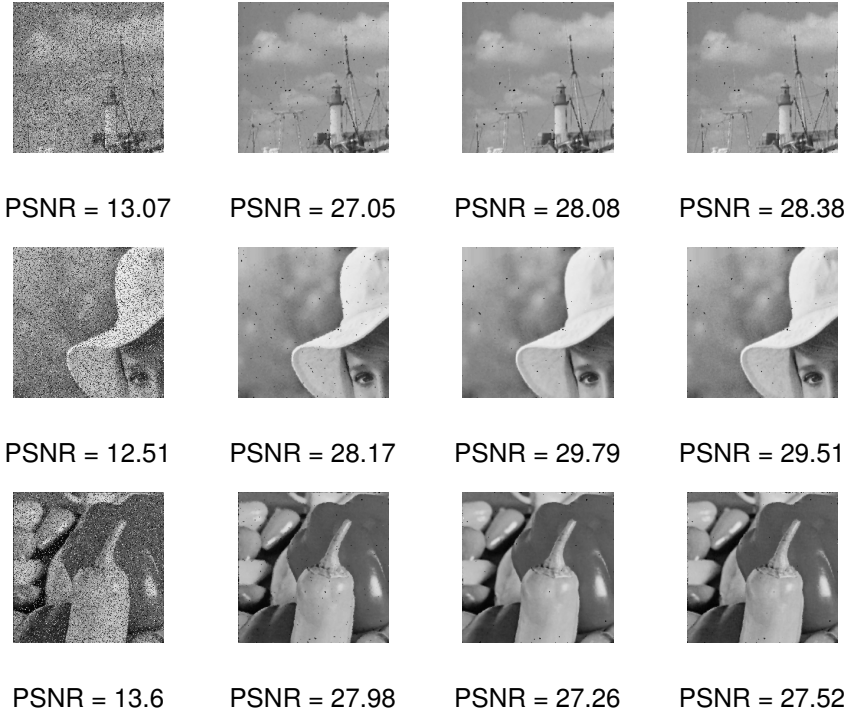


Figure 4.9: One realization of salt and pepper noise denoising for $\rho = 0.2$. Columns from left to right: Noisy image, median filter, RB-RVM and SD-RVM. Peak Signal to Noise Ratio (PSNR) has been rounded to two decimals.

Gamma distributed as in (4.7) and (6.5). The log-likelihood of the parameters is given by (4.32).

For fixed precisions γ and β , the Maximum A Posteriori (MAP) estimate of \mathbf{x} becomes

$$\begin{aligned}
 \hat{\mathbf{x}} &= \arg \max_{\mathbf{x}} \log p(\mathbf{y}, \mathbf{x} | \gamma, \beta) \\
 &= \arg \min_{\mathbf{x}} (\mathbf{y} - \mathbf{Ax})^\top \mathbf{B} (\mathbf{y} - \mathbf{Ax}) + \mathbf{x}^\top \mathbf{\Gamma} \mathbf{x} \\
 &= \mathbf{\Sigma} \mathbf{A}^\top \mathbf{B} \mathbf{y},
 \end{aligned}$$

where $\mathbf{\Sigma} = (\mathbf{\Gamma} + \mathbf{A}^\top \mathbf{B} \mathbf{A})^{-1}$. The form of the MAP estimate is the same for all models considered in this paper.

We maximize \mathcal{L} w.r.t. γ_i by setting the derivative to zero. We use that

$$\frac{\partial}{\partial \gamma_i} (\mathbf{y}^\top (\mathbf{B}^{-1} + \mathbf{A} \mathbf{\Gamma}^{-1} \mathbf{A}^\top)^{-1} \mathbf{y}) = \hat{x}_i^2, \quad (4.24)$$

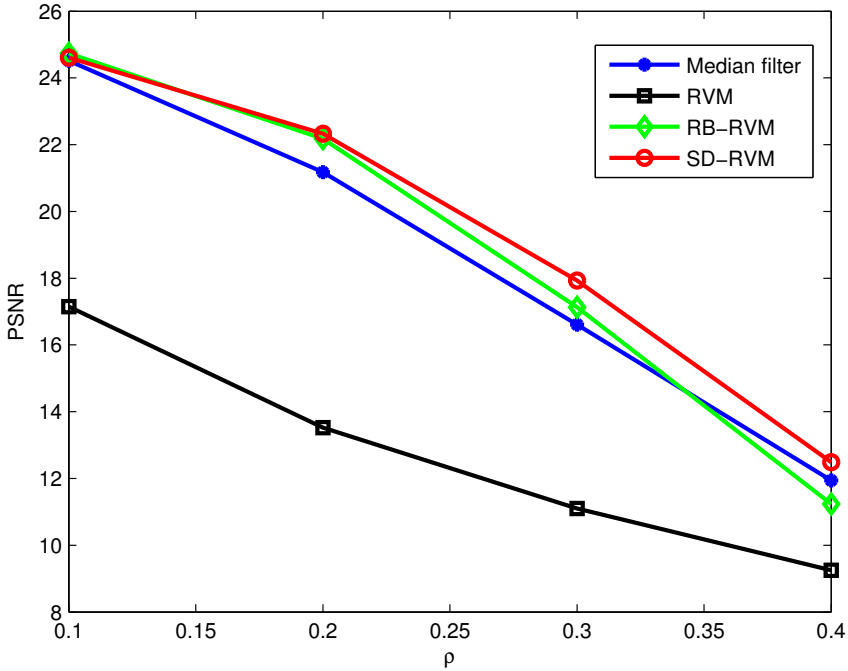


Figure 4.10: PSNR vs. percentage of salt and pepper noise (ρ) averaged over 7 images with 3 noise realizations for each image and each value of ρ .

(we show (4.24) in Section 4.4.2) and the determinant lemma to find that \mathcal{L} is maximized w.r.t. γ_i when

$$-\frac{1}{2}\Sigma_{ii} + \frac{1}{2\gamma_i} + \frac{a}{\gamma_i} - b - \frac{1}{2}\hat{x}_i^2 = 0. \quad (4.25)$$

Instead of solving for γ_i (which would require solving a non-linear equation since Σ and $\hat{\mathbf{x}}$ depend on γ_i) we approximate the equation as

$$1 + 2(a - 1) - (\hat{x}_i^2 + \Sigma_{ii} + 2b)\gamma_i^{new} = 0. \quad (4.26)$$

The approximation results in the same update equations as expectation-maximization and are different from the update equations of the original RVM [Tip01, Mac92]. The update equation then becomes

$$\gamma_i^{new} = \frac{1 + 2(a - 1)}{\hat{x}_i^2 + \Sigma_{ii} + 2b}.$$

Setting $a = 1$ and $b = 0$ we obtain (4.9).

For the noise precisions we use that

$$\frac{\partial}{\partial \beta_j} [\mathbf{y}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y}] = [\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}]_j^2. \quad (4.27)$$

We show the identity (4.27) in 4.4.2. We find that \mathcal{L} is maximized w.r.t. β_j when

$$-\frac{1}{2}\text{tr}(\mathbf{\Sigma}\mathbf{A}_{j,:}^\top\mathbf{A}_{j,:}) + \frac{1}{2\beta_j} - \frac{1}{2}[\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}]_j^2 + \frac{c}{\beta_j} - d = 0,$$

where $\mathbf{A}_{j,:}$ denotes the j 'th row vector of \mathbf{A} . Rewriting the equation as

$$1 + 2(c-1) - ([\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}]_j^2 + \mathbf{A}_{j,:}\mathbf{\Sigma}\mathbf{A}_{j,:}^\top + 2d)\beta_j^{new} = 0,$$

and using that $\mathbf{A}_{j,:}\mathbf{\Sigma}\mathbf{A}_{j,:}^\top = [\mathbf{A}\mathbf{\Sigma}\mathbf{A}^\top]_{jj}$, we find that

$$\beta_j^{new} = \frac{1 + 2(c-1)}{[\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}]_j^2 + [\mathbf{A}\mathbf{\Sigma}\mathbf{A}^\top]_{jj} + 2d}.$$

Setting $c = 1$ and $d = 0$ we obtain (4.10).

4.4.2 Derivation of (4.24) and (4.27)

Proof of (4.24). Since

$$\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top = \mathbf{B}^{-1} + \sum_{i=1}^n \gamma_i^{-1} \mathbf{a}_i \mathbf{a}_i^\top,$$

where \mathbf{a}_i is the i 'th column vector of \mathbf{A} we find that

$$\frac{\partial}{\partial \gamma_i} (\mathbf{y}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y}) = \gamma_i^{-2} (\mathbf{a}_i^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y})^2.$$

Using that

$$\begin{aligned} & \mathbf{\Gamma}^{-1}\mathbf{A}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y} \\ &= \mathbf{\Gamma}^{-1}\mathbf{A}^\top (\mathbf{B} - \mathbf{B}\mathbf{A}(\mathbf{\Gamma} + \mathbf{A}^\top\mathbf{B}\mathbf{A})^{-1}\mathbf{A}^\top\mathbf{B})\mathbf{y} \\ &= \mathbf{\Gamma}^{-1}\mathbf{A}^\top\mathbf{B}\mathbf{y} - \mathbf{\Gamma}^{-1} \underbrace{\mathbf{A}^\top\mathbf{B}\mathbf{A}}_{=\mathbf{\Sigma}^{-1}-\mathbf{\Gamma}} \underbrace{(\mathbf{\Gamma} + \mathbf{A}^\top\mathbf{B}\mathbf{A})^{-1}}_{=\mathbf{\Sigma}} \mathbf{A}^\top\mathbf{B}\mathbf{y} \\ &= \mathbf{\Sigma}\mathbf{A}^\top\mathbf{B}\mathbf{y} = \hat{\mathbf{x}}, \end{aligned} \quad (4.28)$$

we find that

$$\frac{\partial}{\partial \gamma_i} (\mathbf{y}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y}) = \gamma_i^{-2} (\gamma_i \hat{x}_i)^2 = \hat{x}_i^2.$$

□

Proof of (4.27). Since

$$\mathbf{y}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y} = \mathbf{y}^\top \mathbf{B}\mathbf{y} - \mathbf{y}\mathbf{B}\mathbf{A} \underbrace{(\mathbf{\Gamma} + \mathbf{A}^\top \mathbf{B}\mathbf{A})^{-1}}_{=\mathbf{\Sigma}} \mathbf{A}^\top \mathbf{B}\mathbf{y},$$

we get that

$$\begin{aligned} & \frac{\partial}{\partial \beta_j} (\mathbf{y}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y}) \\ &= y_j^2 - 2y_j \mathbf{A}_{j,:} \mathbf{\Sigma} \mathbf{A}^\top \mathbf{B}\mathbf{y} + \mathbf{y}^\top \mathbf{B}\mathbf{A} \mathbf{\Sigma} \mathbf{A}_{j,:}^\top \mathbf{A}_{j,:} \mathbf{\Sigma} \mathbf{A}^\top \mathbf{B}\mathbf{y} \\ &= y_j^2 - 2y_j \mathbf{A}_{j,:} \hat{\mathbf{x}} + (\mathbf{A}_{j,:} \hat{\mathbf{x}})^2 = [\mathbf{y} - \mathbf{A}\hat{\mathbf{x}}]_j^2. \end{aligned}$$

□

4.4.3 Update equations for known block structure

Let $\mathbf{\Gamma}$ and \mathbf{B} be diagonal matrices with

$$[\mathbf{\Gamma}]_{kk} = \gamma_i, \text{ if } k \in I_i, \quad [\mathbf{B}]_{ll} = \beta_j, \text{ if } l \in J_j,$$

and zero otherwise.

To update the precisions we maximize the marginal distribution

$$p(\mathbf{y}, \boldsymbol{\gamma}, \mathbf{B}) = p(\mathbf{y}|\boldsymbol{\gamma}, \mathbf{B})p(\boldsymbol{\gamma})p(\boldsymbol{\beta}),$$

with respect to $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$, where $p(\boldsymbol{\gamma})$ and $p(\boldsymbol{\beta})$ are as in (4.7) and (6.5). The log-likelihood of the parameters is

$$\begin{aligned} \mathcal{L} &= \text{const.} - \frac{1}{2} \log \det(\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top) - \frac{1}{2} \mathbf{y}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y} \\ &+ \sum_{i=1}^p ((a-1) \log \gamma_i - b\gamma_i) + \sum_{j=1}^q ((c-1) \log \beta_j - d\beta_j). \end{aligned}$$

We get that \mathcal{L} is maximized when

$$\frac{\partial \mathcal{L}}{\partial \gamma_i} = -\frac{1}{2} \text{tr}(\boldsymbol{\Sigma}_{I_i}) + \frac{n_i}{2\gamma_i} + \frac{a-1}{\gamma_i} - b - \frac{1}{2\gamma_i^2} \|\mathbf{A}_{I_i}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y}\|_2^2 = 0, \quad (4.29)$$

where $\boldsymbol{\Sigma}_{I_i} \in \mathbb{R}^{n_i \times n_i}$ is the submatrix of $\boldsymbol{\Sigma}$ consisting of the columns and rows in I_i . Further, using (4.28) we get that

$$\mathbf{A}_{I_i}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y} = \gamma_i \hat{\mathbf{x}}_{I_i}. \quad (4.30)$$

Thus, (4.29) is fulfilled when

$$-\frac{1}{2} \text{tr}(\boldsymbol{\Sigma}_{I_i}) + \frac{n_i}{2\gamma_i} + \frac{a-1}{\gamma_i} - b - \frac{1}{2} \|\hat{\mathbf{x}}_{I_i}\|_2^2 = 0.$$

As before, we rewrite the equation as

$$n_i + 2(a - 1) - (\|\hat{\mathbf{x}}_{I_i}\|_2^2 + \text{tr}(\mathbf{\Sigma}_{I_i}) + 2b)\gamma_i^{new} = 0. \quad (4.31)$$

Solving (4.31) for γ_i^{new} gives us the update equation (4.17).

To find the update equation for β_j we use that

$$\begin{aligned} & \frac{\partial}{\partial \beta_j} [\mathbf{y}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y}] \\ &= \|\mathbf{y}_{J_j}\|_2^2 - 2\mathbf{y}_{J_j}^\top \mathbf{A}_{J_j,:} \mathbf{\Sigma} \mathbf{A}^\top \mathbf{B} \mathbf{y} + \mathbf{y}^\top \mathbf{B} \mathbf{A} \mathbf{\Sigma} \mathbf{A}_{J_j,:}^\top \mathbf{A}_{J_j,:} \mathbf{\Sigma} \mathbf{A}^\top \mathbf{B} \mathbf{y} \\ &= \|(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}})_{J_j}\|_2^2, \end{aligned}$$

where $\mathbf{A}_{J_j,:}$ consists of the row vectors with row indexes in J_j . We find that

$$\frac{\partial \mathcal{L}}{\partial \beta_j} = -\frac{1}{2} \text{tr}(\mathbf{\Sigma} \mathbf{A}_{J_j,:}^\top \mathbf{A}_{J_j,:}) + \frac{m_j}{2\beta_j} - \frac{1}{2} \|(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}})_{J_j}\|_2^2 + \frac{c-1}{\beta_j} - d = 0.$$

Rewriting the equation as

$$1 + 2(c - 1) = (\|(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}})_{J_j}\|_2^2 + \text{tr}(\mathbf{A}_{J_j,:} \mathbf{\Sigma} \mathbf{A}_{J_j,:}^\top) + 2d)\beta_j^{new},$$

and using that $\text{tr}(\mathbf{A}_{J_j,:} \mathbf{\Sigma} \mathbf{A}_{J_j,:}^\top) = \text{tr}([\mathbf{A}\mathbf{\Sigma}\mathbf{A}^\top]_{J_j})$ gives us the update equation (4.18).

4.4.4 Update equations for unknown block structure

When the block structure is unknown, we use the overparametrized model in section 4.2.2. The log-likelihood of the parameters is

$$\begin{aligned} \mathcal{L} &= \log p(\mathbf{y}|\boldsymbol{\gamma}, \boldsymbol{\beta})p(\boldsymbol{\gamma})p(\boldsymbol{\beta}) \quad (4.32) \\ &= \text{const.} - \frac{1}{2} \log \det(\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top) - \frac{1}{2} \mathbf{y}^\top (\mathbf{B}^{-1} + \mathbf{A}\mathbf{\Gamma}^{-1}\mathbf{A}^\top)^{-1}\mathbf{y} \\ &\quad + \sum_{i=1}^p ((a-1) \log \tilde{\gamma}_i - b\tilde{\gamma}_i) + \sum_{j=1}^q ((c-1) \log \tilde{\beta}_j - d\tilde{\beta}_j). \end{aligned}$$

We search to maximize (4.32) with respect to the underlying variables $\tilde{\gamma}_k$ and $\tilde{\beta}_l$. Using that $\frac{\partial \gamma_i^{-1}}{\partial \tilde{\gamma}_k} = -\tilde{\gamma}_k^{-2}$, $\frac{\partial \gamma_i}{\partial \tilde{\gamma}_k} = \gamma_i^2 \tilde{\gamma}_k^{-2}$, when $i \in I_k$ and zero otherwise and (4.28) we find that \mathcal{L} is maximized when

$$\frac{\partial \mathcal{L}}{\partial \tilde{\gamma}_k} = -\frac{1}{2\tilde{\gamma}_k^2} \text{tr}(\mathbf{\Sigma}\mathbf{\Gamma}_k^2) + \frac{1}{2\tilde{\gamma}_k^2} \text{tr}(\mathbf{\Gamma}_k) - \frac{1}{2\tilde{\gamma}_k^2} \hat{\mathbf{x}}^\top \mathbf{\Gamma}_k^2 \hat{\mathbf{x}} + \frac{a-1}{\tilde{\gamma}_k} - b = 0, \quad (4.33)$$

By rewriting (4.33) as

$$\frac{1}{\tilde{\gamma}_k} \text{tr}(\mathbf{\Gamma}_k) + 2(a-1) = \left(\frac{1}{\tilde{\gamma}_k^2} \|\mathbf{\Gamma}_k \hat{\mathbf{x}}\|_2^2 + \frac{1}{\tilde{\gamma}_k^2} \text{tr}(\mathbf{\Gamma}_k \mathbf{\Sigma} \mathbf{\Gamma}_k) + 2b \right) \tilde{\gamma}_k^{new}. \quad (4.34)$$

Solving (4.34) for $\tilde{\gamma}_k^{new}$ gives us the update equation (4.20).

For the noise precisions, we similarly find that

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \tilde{\beta}_l} &= -\frac{1}{2\tilde{\beta}_l^2} \text{tr}(\mathbf{B}_l \mathbf{A}^\top \boldsymbol{\Sigma} \mathbf{A} \mathbf{B}_l) + \frac{1}{2\tilde{\beta}_l^2} \text{tr}(\mathbf{B}_l) \\ &\quad - \frac{1}{2\tilde{\beta}_l^2} \|\mathbf{B}_l(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}})\|_2^2 + \frac{c-1}{\tilde{\beta}_l} - d = 0. \end{aligned}$$

By rewriting the expression as

$$\frac{1}{\tilde{\beta}_l} \text{tr}(\mathbf{B}_l) + 2(c-1) = \left(\frac{1}{\tilde{\beta}_l^2} \|\mathbf{B}_l(\mathbf{y} - \mathbf{A}\hat{\mathbf{x}})\|_2^2 + \frac{1}{\tilde{\beta}_l^2} \text{tr}(\mathbf{B}_l \mathbf{A}^\top \boldsymbol{\Sigma} \mathbf{A} \mathbf{B}_l) + 2d \right) \tilde{\beta}_l^{new},$$

we find the update equation (4.21).

We see that the form of update equations depends on how the equations are rewritten. The form used here has the advantage of reducing to (4.18) when the underlying blocks are disjoint.

4.5 Conclusion

We proposed a modification of the Relevance Vector Machine (RVM) which is robust against sparse noise while not needing to estimate the sparse noise explicitly. We denote the proposed method by SD-RVM for Sparse and Dense noise combined RVM. The method is faster than the robust RVM (RB-RVM) of [MVC10] where the sparse noise is treated as an additional estimation variable. Simulations show that the method is faster than the RB-RVM for compressed sensing and has similar performance. We also show how the method can be modified to block-sparse signals and noise and simulate the method for housing price prediction and image denoising.

Relevance singular vector machine for low-rank matrix reconstruction

Low-rank matrices is a parsimonious model which, like the sparse model, can be used to improve estimation performance. However, unlike sparsity, which is a property of the individual components, low-rank is a property of the entire matrix. While sparsity gives a parsimonious representation of data in a known basis, low-rank matrices can be interpreted as representing data in an unknown basis. One prominent low-rank model is collaborative filtering where a recommender system attempts to find if you will like a product you have not yet seen. The basic assumption of low-rank models is that the problem has a high degree of inherent (linear) structure.

The recommendation problem can be modeled as follows: what you prefer depends on what persons you are similar to and what those persons prefer. Despite the simplistic nature of the model, it has been shown to be a good model for the recommendation problem [KBV09, CP10]. Other applications of low-rank modeling is clustering documents by topic and system identification [CP10, LSR⁺16].

Several approaches have been developed to solve the low-rank estimation problem. A prominent class of methods are convex optimization based techniques which generalizes the ℓ_1 -norm for sparse problems to low-rank problems. The low-rank penalty corresponding to the ℓ_1 -norm is the nuclear norm which is the sum of the singular values of a matrix [CP10, CR09, Faz02]. In the same way as the ℓ_1 -norm can be shown to be the tightest convex approximation of the ℓ_0 -norm, the nuclear norm can be shown to be the tightest convex approximation of the rank (under certain conditions, see [LO15] for a counterexample).

The low-rank estimation problem also allows for factorization based techniques. Factorization based techniques are used a lot [KBV09, ZSJC12, TN11, RFGST09, CWZY15, YC11, LBA11, Alq13, MS07, SM08, LT07, RIK07, BLMK12] since they allow one to specify a strict upper bound on the rank of the estimate. Factorization based methods are fast, but they require an upper bound on the rank to be known a-priori. It is also difficult to relate the factorization based approach to the opti-

mization based approach. A third approach is the greedy search method ADMIRA [LB10] which is a low-rank matrix generalization of the CoSamp method [NT09] for sparse vectors.

Bayesian methods have also been developed for low-rank matrix reconstruction. Most such methods rely on the factorized model and promotes low-rank through block sparsity promoting priors [CWZY15, YC11, Alq13, MS07, SM08, LT07, RIK07, BLMK12]. The question arises if it is possible to construct a low-rank analogue of the Relevance Vector Machine where the low-rank matrix is *not* modeled as a product of two matrices? Here we show one approach to generalize the RVM to the low-rank problem using precision matrices to induce low-rank. We call the resulting method the *Relevance Singular Vector Machine*.

5.1 Low-rank matrix estimation

In low-rank matrix estimation, a low-rank matrix $\mathbf{X} \in \mathbb{R}^{p \times q}$ is measured through linear measurements of the matrix components as

$$\mathbf{y} = \mathcal{A}(\mathbf{X}) + \mathbf{n}, \quad (5.1)$$

where $\mathbf{y} \in \mathbb{R}^m$ is the observed measurements, $\mathbf{n} \in \mathbb{R}^m$ is additive noise and $\mathcal{A} : \mathbb{R}^{p \times q} \rightarrow \mathbb{R}^m$ is a linear operator representing the measurement process. The sensing operator can be expressed as

$$\mathcal{A}(\mathbf{X}) = \begin{bmatrix} \text{tr}(\mathbf{A}_1^\top \mathbf{X}) \\ \text{tr}(\mathbf{A}_2^\top \mathbf{X}) \\ \vdots \\ \text{tr}(\mathbf{A}_m^\top \mathbf{X}) \end{bmatrix}, \quad (5.2)$$

where

$$\text{tr}(\mathbf{A}_k^\top \mathbf{X}) = \sum_{1 \leq i \leq p, 1 \leq j \leq q} [\mathbf{A}_k]_{ij} [\mathbf{X}]_{ij}$$

gives a linear combination of the elements in \mathbf{X} with coefficients from $\mathbf{A}_k \in \mathbb{R}^{p \times q}$. The sensing operation can also be expressed as $\mathcal{A}(\mathbf{X}) = \mathbf{Avec}(\mathbf{X})$ using the vectorization operator. The vectorization representation shows the similarity between the LRMR problem and the sparse vector reconstruction problem.

A standard approach to promoting structure (such as sparsity or low-rank) is to minimize a penalty function in addition to the norm of the error. This leads to the optimization problem

$$\hat{\mathbf{X}} = \arg \min_{\mathbf{X}} \frac{\beta}{2} \|\mathbf{y} - \mathbf{Avec}(\mathbf{X})\|_2^2 + g(\mathbf{X}), \quad (5.3)$$

where $\beta > 0$ is a regularization parameter and $g(\cdot)$ is a penalty function. When the penalty function $g(\mathbf{X})$ is convex (and β is fixed), the problem is a convex

optimization problem and can be solved using standard optimization tools such as cvx [GBY08]. For non-convex penalty functions the optimization problem can be approximately solved through e.g. gradient descent. Common penalty functions for promoting low-rank are

$$g(\mathbf{X}) = \sum_{i=1}^{\min(p,q)} \sigma_i(\mathbf{X}) = \text{tr}((\mathbf{X}\mathbf{X}^\top)^{1/2}), \quad (\text{Nuclear norm})$$

$$g(\mathbf{X}) = \text{tr}((\mathbf{X}\mathbf{X}^\top)^{s/2}), \quad (\text{Schatten } s\text{-norm})$$

$$g(\mathbf{X}) = \nu \log |\mathbf{X}\mathbf{X}^\top + \epsilon \mathbf{I}_p|. \quad (\text{Log-determinant penalty})$$

The nuclear norm is a convex penalty function while the log-determinant and Schatten s -norm (for $0 < s < 1$) are non-convex. In the Bayesian framework, the minimization in (5.3) correspond to finding the maximum-a-posteriori (MAP) estimate of \mathbf{X} when the noise is Gaussian and the matrix \mathbf{X} is assigned a prior distribution $p(\mathbf{X}) \propto e^{-\frac{1}{2}g(\mathbf{X})}$. An alternative approach is to set

$$\hat{\mathbf{X}} = \arg_{\mathbf{X}} \min \|\mathbf{X}\|_*, \text{ such that } \|\mathbf{y} - \mathbf{A}\text{vec}(\mathbf{X})\|_2 \leq \delta, \quad (5.4)$$

where $\delta > 0$ is related to the (known) noise power. The choice $\delta = \beta^{-1} \sqrt{m + \sqrt{8m}}$ suggested in [CRT06] often provides good performance.

In the factorization based approach, the matrix \mathbf{X} is modeled as a product of two matrices as [KBV09, ZSJC12, TN11, RFGST09, CWZY15, YC11, LBA11, Alq13, MS07, SM08, LT07, RIK07, BLMK12]

$$\mathbf{X} = \mathbf{F}\mathbf{B}^\top,$$

where $\mathbf{F} \in \mathbb{R}^{p \times r}$, $\mathbf{B} \in \mathbb{R}^{q \times r}$ and $0 < r \leq \min(p, q)$ is a parameter. The factor matrices can be found in a deterministic manner by minimizing the residual $\|\mathbf{y} - \mathbf{A}\text{vec}(\mathbf{F}\mathbf{B}^\top)\|_2^2$ iteratively for \mathbf{F} and \mathbf{B} while keeping the other matrix fixed [TN11, ZSJC12], this is sometimes called *alternating least squares* (ALS). Since the residual is non-convex in \mathbf{F} and \mathbf{B} , it is not guaranteed to converge to a global optima. One approach to improving the performance of ALS is to incorporate dual variables and ADMM iterations [LSR⁺16].

The estimates produced by ALS are mostly full rank, i.e. $\text{rank}(\hat{\mathbf{F}}\hat{\mathbf{B}}^\top) = r$. When the rank is unknown, the method can be modified so that the final estimate has rank less than r . This can be achieved by assigning appropriate priors to the factor matrices. One approach is to minimize

$$\|\mathbf{y} - \mathbf{A}\text{vec}(\mathbf{F}\mathbf{B}^\top)\|_2^2 + \lambda (\|\mathbf{F}\|_F^2 + \|\mathbf{B}\|_F^2),$$

where $\lambda > 0$ is a regularization parameter. The penalty function is expected to promote low-rank through the relation [MS07]

$$\|\mathbf{X}\|_* = \frac{1}{2} \min_{\mathbf{F}, \mathbf{B}: \mathbf{F}\mathbf{B}^\top = \mathbf{X}} (\|\mathbf{F}\|_F^2 + \|\mathbf{B}\|_F^2).$$

Another method is to use priors which promote column sparsity in the factor matrices [LT07, BLMK12], i.e. a prior which makes many of the column vectors in \mathbf{F} and \mathbf{B} zero. This approach promotes low-rank since if \mathbf{F} has r_F non-zero columns and \mathbf{B} has r_B non-zero columns, then $\text{rank}(\hat{\mathbf{X}}) \leq \min(r_L, r_B) \leq r$. We describe this model in more detail in Chapter 7.

5.2 The one-sided precision based model

The difficulty of assigning priors to low-rank matrices is that the set of low-rank matrices is a strictly lower dimensional submanifold of the set of all matrices. It is thus difficult to assign a “hard” prior which has non-zero probability (in the sense of an induced measure) only for matrices of low-rank. We therefore want to find priors which *approximately* promotes low-rank, i.e. it has non-zero probability for all matrices, but is concentrated around the set of low-rank matrices. We do this by characterizing a random low-rank matrix as a matrix with strongly correlated column and row vectors rather than the standard notion of few non-zero singular values. We thus need a prior which makes the column vectors lie in a low-dimensional subspace (similarly for the row vectors).

The Relevance Vector Machine models the components x_i of the parameter vector $\mathbf{x} \in \mathbb{R}^n$ are modeled as

$$x_i = \gamma_i^{-1/2} u_i, \quad (5.5)$$

where $\gamma_i > 0$ is the precision of x_i and $u_i \sim \mathcal{N}(0, 1)$ is a Gaussian variable. When the inverse precision, γ^{-1} is small, x_i is small with a high probability and when the inverse precision is large, x_i is large with a high probability. In the extreme case where $\gamma_i^{-1} = 0$, we have that $x_i = 0$. However, the extreme case has zero prior probability and does therefore not occur in practice. In analogue with the sparse case, we model the low-rank matrix \mathbf{X} as a Gaussian variable with a *precision matrix* $\boldsymbol{\alpha}$ as

$$\mathbf{X} = \boldsymbol{\alpha}^{-1/2} \mathbf{U}, \quad (5.6)$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{p \times p}$ is positive definite, $\boldsymbol{\alpha}^{-1/2}$ denotes a symmetric such that $(\boldsymbol{\alpha}^{-1/2})^2 = \boldsymbol{\alpha}^{-1}$ and the elements of \mathbf{U} are i.i.d. $\mathcal{N}(0, 1)$ distributed. We notice the similarity between (5.5) and (5.6). When the inverse precision matrix, $\boldsymbol{\alpha}^{-1}$, has a few large singular values and otherwise small singular values, the random matrix \mathbf{X} has low-rank with high probability. In the extreme case where $\boldsymbol{\alpha}^{-1}$ has low-rank, the matrix \mathbf{X} also has low-rank. The approach (5.6) is equivalent to letting \mathbf{X} to be a matrix variate Gaussian variable with distribution [GN99, NSB13]

$$p(\mathbf{X}|\boldsymbol{\alpha}) = \frac{|\boldsymbol{\alpha}|^{q/2}}{(2\pi)^{pq/2}} \exp\left(-\frac{1}{2}\text{tr}(\mathbf{X}^\top \boldsymbol{\alpha} \mathbf{X})\right). \quad (5.7)$$

In order to promote low-rank, we need to assign a prior to the precision matrix $\boldsymbol{\alpha}$. In Section 5.2.1 we will show one approach to selecting the prior distribution.

5.2.1 Relation between priors and the MAP estimation

Here we show how the distribution $p(\boldsymbol{\alpha})$ is related to the penalty function $g(\mathbf{X})$. We see that the prior distribution (5.7) depends on \mathbf{X} only through $\mathbf{Z} = \mathbf{X}\mathbf{X}^\top$. Let the prior distribution of $\boldsymbol{\alpha}$ be $p(\boldsymbol{\alpha})$, we find that the marginal distribution $p(\mathbf{X})$ becomes

$$\begin{aligned} p(\mathbf{X}) &= C e^{-\frac{1}{2}g(\mathbf{X})} = \int_{\boldsymbol{\alpha} > \mathbf{0}} p(\mathbf{X}|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}) d\boldsymbol{\alpha} \\ &= \int_{\boldsymbol{\alpha} > \mathbf{0}} e^{-\frac{1}{2}\text{tr}(\boldsymbol{\alpha}\mathbf{Z})} \frac{|\boldsymbol{\alpha}|^{q/2}}{(2\pi)^{pq/2}} p(\boldsymbol{\alpha}) d\boldsymbol{\alpha} = C' e^{-\frac{1}{2}\tilde{g}(\mathbf{Z})}, \end{aligned} \quad (5.8)$$

where we used that $\text{tr}(\mathbf{X}^\top \boldsymbol{\alpha}\mathbf{X}) = \text{tr}(\boldsymbol{\alpha}\mathbf{X}\mathbf{X}^\top) = \text{tr}(\boldsymbol{\alpha}\mathbf{Z})$, $g(\cdot)$ and $\tilde{g}(\cdot)$ are some functions and C and C' are normalization constants. To relate $\tilde{g}(\mathbf{Z})$ and $p(\boldsymbol{\alpha})$, we note that $p(\mathbf{X})$ is the matrix Laplace transform of $|\boldsymbol{\alpha}|^{q/2}p(\boldsymbol{\alpha})/(2\pi)^{pq/2}$ at $\mathbf{Z}/2$ [Ter12] (we give a short introduction to the matrix Laplace transform in Section 5.5.6). This gives us that we can calculate $p(\boldsymbol{\alpha})$ from $p(\mathbf{X}) = C' e^{-\frac{1}{2}\tilde{g}(\mathbf{Z})}$ by the inverse Laplace transform [Ter12] as

$$p(\boldsymbol{\alpha}) \propto |\boldsymbol{\alpha}|^{-q/2} \int_{\text{Re } \mathbf{Z} = \boldsymbol{\alpha}_*} e^{\frac{1}{2}\text{tr}(\boldsymbol{\alpha}\mathbf{Z})} e^{-\frac{1}{2}\tilde{g}(\mathbf{Z})} d\mathbf{Z}, \quad (5.9)$$

where the integral is taken over all symmetric matrices $\mathbf{Z} \in \mathbb{C}^{p \times p}$ such that $\text{Re } \mathbf{Z} = \boldsymbol{\alpha}_*$ and $\boldsymbol{\alpha}_*$ is a real matrix such that the contour path of integration is in the region of convergence of the integrand. While the Laplace transform characterizes the exact relation between priors, the computation is non-trivial and often analytically intractable. In practice, a standard approach to evaluating integrals like (5.8) is to use the Laplace approximation [Mac03, Bis06] where typically the mode of the distribution under approximation is found first and then a Gaussian distribution is modeled around that mode. Let us write $p(\boldsymbol{\alpha})$ as $p(\boldsymbol{\alpha}) \propto e^{-\frac{1}{2}K(\boldsymbol{\alpha})}$; then the Laplace approximation becomes

$$\begin{aligned} \tilde{g}(\mathbf{Z}) &= \min_{\boldsymbol{\alpha} > \mathbf{0}} \{ \text{tr}(\boldsymbol{\alpha}\mathbf{Z}) - q \log |\boldsymbol{\alpha}| + K(\boldsymbol{\alpha}) \} \\ &\quad - \log |\mathbf{H}| + \text{constant}, \end{aligned}$$

where \mathbf{H} is the Hessian of $\text{tr}(\boldsymbol{\alpha}\mathbf{Z}) - q \log |\boldsymbol{\alpha}| + K(\boldsymbol{\alpha})$ evaluated at the minima. The derivation of the Laplace approximation is shown in Section 5.5.7.

Denoting $\tilde{K}(\boldsymbol{\alpha}) = q \log |\boldsymbol{\alpha}| - K(\boldsymbol{\alpha})$ and neglecting the Hessian we get that

$$\tilde{g}(\mathbf{Z}) = \min_{\boldsymbol{\alpha} > \mathbf{0}} \{ \text{tr}(\boldsymbol{\alpha}\mathbf{Z}) - \tilde{K}(\boldsymbol{\alpha}) \},$$

where we absorbed the constants into the normalization factor of $p(\mathbf{X})$. We find that $\tilde{g}(\mathbf{Z})$ is the concave conjugate of $\tilde{K}(\boldsymbol{\alpha})$ [BV04]. Hence, for a given $\tilde{g}(\mathbf{Z})$ we can recover $\tilde{K}(\boldsymbol{\alpha})$ as

$$\tilde{K}(\boldsymbol{\alpha}) = \min_{\mathbf{Z} > \mathbf{0}} \{ \text{tr}(\boldsymbol{\alpha}\mathbf{Z}) - \tilde{g}(\mathbf{Z}) \} \quad (5.10)$$

if $\tilde{K}(\boldsymbol{\alpha})$ is concave (which holds under the assumption that $K(\boldsymbol{\alpha})$ is convex). Further, we can find $K(\boldsymbol{\alpha})$ from $\tilde{K}(\boldsymbol{\alpha})$ and solve for the prior $p(\boldsymbol{\alpha}) \propto e^{-\frac{1}{2}K(\boldsymbol{\alpha})}$. Using the concave conjugate relation (5.10), we now find priors (functions $K(\boldsymbol{\alpha})$) corresponding to the Schatten- s norm and the log-determinant penalty.

1. *The Schatten s -norm:* The regularized Schatten s -norm based penalty function is

$$g(\mathbf{X}) = \text{tr}((\mathbf{X}\mathbf{X}^\top + \epsilon \mathbf{I}_p)^{s/2}), \quad (5.11)$$

where the constant $\epsilon > 0$ is used to bring numerical stability. For the penalty function (5.11), we find that

$$K(\boldsymbol{\alpha}) = C_s \text{tr}(\boldsymbol{\alpha}^{-\frac{s}{2-s}}) + q \log |\boldsymbol{\alpha}| + \epsilon \text{tr}(\boldsymbol{\alpha}), \quad (5.12)$$

where $C_s = \frac{2-s}{s} \left(\frac{2}{s}\right)^{-\frac{s}{2-s}}$. The derivation of (5.12) is given in Section 5.5.10. Let $\sigma_i(\mathbf{X})$ denote the i 'th singular value of \mathbf{X} , for $s = 1$ and $\epsilon = 0$ we see that $g(\mathbf{X})$ becomes the nuclear norm of \mathbf{X} as $\text{tr}((\mathbf{X}\mathbf{X}^\top)^{\frac{1}{2}}) = \sum_{i=1}^{\min(p,q)} \sigma_i(\mathbf{X})$.

2. *The Log-determinant penalty:* The log-determinant based penalty function is

$$g(\mathbf{X}) = \nu \log |\mathbf{X}\mathbf{X}^\top + \epsilon \mathbf{I}_p|, \quad (5.13)$$

where $\nu > q - 2$ is a positive number. We find that

$$K(\boldsymbol{\alpha}) = \epsilon \text{tr}(\boldsymbol{\alpha}) + (q - \nu) \log |\boldsymbol{\alpha}|. \quad (5.14)$$

The derivation of (5.14) is shown in Section 5.5.11. As $p(\boldsymbol{\alpha}) \propto e^{-\frac{1}{2}K(\boldsymbol{\alpha})}$, we find that the precision matrix $\boldsymbol{\alpha}$ is Wishart distributed [GN99, Bis06]. We also find that $p(\mathbf{X}) \propto e^{-\frac{1}{2}\tilde{g}(\mathbf{X}\mathbf{X}^\top)}$ is a matrix t-distribution [GN99].

5.2.2 Left and right-sided precision based models

For a low-rank matrix, the components of each column vector are correlated as well as the components of each row vector. In (5.6), the precision matrix $\boldsymbol{\alpha}^{-1/2}$ is used in the left side of \mathbf{U} , we refer to this as the left-sided precision based model. The model helps to make the components of each column of \mathbf{X} correlated. We can also make the row-vectors correlated by setting

$$\mathbf{X} = \mathbf{U}\boldsymbol{\alpha}^{-1/2}. \quad (5.15)$$

We refer to this model as the right-sided precision based model.

5.3 Two-sided precision based model

In this section, we propose to use precision matrices from two sides to model a random low-rank matrix, we refer to this model as the two-sided precision based model. In the two-sided model, we set

$$\mathbf{X} = \boldsymbol{\alpha}_L^{-1/2} \mathbf{U} \boldsymbol{\alpha}_R^{-1/2} \quad (5.16)$$

where $\boldsymbol{\alpha}_L \in \mathbb{R}^{p \times p}$ and $\boldsymbol{\alpha}_R \in \mathbb{R}^{q \times q}$ are positive definite random matrices. Our hypothesis is that the two-sided precision based model helps to inculcate correlations between column vectors as well as row vectors, and hence promotes low-rank in a stronger manner than the one-sided precision based model in Section 5.2. Using the relation $\text{vec}(\mathbf{X}) = (\boldsymbol{\alpha}_R^{-1/2} \otimes \boldsymbol{\alpha}_L^{-1/2}) \text{vec}(\mathbf{U}) = (\boldsymbol{\alpha}_R \otimes \boldsymbol{\alpha}_L)^{-1/2} \text{vec}(\mathbf{U})$, we find that

$$\begin{aligned} p(\mathbf{X}|\boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R) &= \frac{|\boldsymbol{\alpha}_R \otimes \boldsymbol{\alpha}_L|^{1/2}}{(2\pi)^{pq/2}} \exp(-\text{vec}(\mathbf{X})^\top (\boldsymbol{\alpha}_R \otimes \boldsymbol{\alpha}_L) \text{vec}(\mathbf{X})/2) \\ &= \frac{|\boldsymbol{\alpha}_L|^{q/2} |\boldsymbol{\alpha}_R|^{p/2}}{(2\pi)^{pq/2}} \exp(-\text{tr}(\mathbf{X}^\top \boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R)/2). \end{aligned} \quad (5.17)$$

The matrix \mathbf{X} is thus zero-mean matrix variate Gaussian distributed. To promote low-rank, we use a prior distribution $p(\boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R)$. The marginal distribution of \mathbf{X} then becomes

$$p(\mathbf{X}) = \int_{\substack{\boldsymbol{\alpha}_L > \mathbf{0} \\ \boldsymbol{\alpha}_R > \mathbf{0}}} p(\mathbf{X}|\boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R) p(\boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R) d\boldsymbol{\alpha}_R d\boldsymbol{\alpha}_L. \quad (5.18)$$

To the best of our knowledge, the integral (5.18) is analytically intractable for many relevant priors $p(\boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R)$. It is thus non-trivial to establish a direct connection between $p(\mathbf{X}|\boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R)$ of (5.17) and the MAP estimator (5.3). Instead of a direct connection we here establish an indirect connection by an approximation.

When marginalizing over $\boldsymbol{\alpha}_L$ for a fixed $\boldsymbol{\alpha}_R$ and β , we find that $p(\mathbf{X}|\boldsymbol{\alpha}_R)$ is a function of $\mathbf{X}\boldsymbol{\alpha}_R\mathbf{X}^\top$ alone. Following the discussion of section 5.2.1, we find that the corresponding MAP estimator becomes

$$\min_{\mathbf{X}} \beta \|\mathbf{y} - \mathbf{A} \text{vec}(\mathbf{X})\|_2^2 + \tilde{g}_L(\mathbf{X}\boldsymbol{\alpha}_R\mathbf{X}^\top), \quad (5.19)$$

for some function $\tilde{g}_L(\cdot)$. A similar cost function can be found for a fixed $\boldsymbol{\alpha}_L$ and β by marginalizing over $\boldsymbol{\alpha}_R$. We discuss the roles of $\boldsymbol{\alpha}_L$ and $\boldsymbol{\alpha}_R$ in the next section.

5.3.1 Interpretation of the precision matrices

For a low-rank matrix, many column vectors can be represented as a linear combination of a few columns vectors. Alternatively, we can have that the column vectors

of a low-rank matrix lie in a low-dimensional subspace. To show that the proposed model (5.17) promotes low-rank, we here establish in a qualitative sense that when the inverse precision matrices have few dominant singular values, the matrix \mathbf{X} is approximately low-rank with high probability

Let us denote (i, j) 'th component of $\boldsymbol{\alpha}_R^{-1}$ by $[\boldsymbol{\alpha}_R^{-1}]_{ij}$ and i 'th column vector of \mathbf{X} by \mathbf{x}_i , respectively. From (5.17) we find that \mathbf{x}_i is zero-mean and that the covariance matrix of \mathbf{x}_i is

$$\mathcal{E}[\mathbf{x}_i \mathbf{x}_i^\top] = [\boldsymbol{\alpha}_R^{-1}]_{ii} \boldsymbol{\alpha}_L^{-1}. \quad (5.20)$$

Let $\lambda_{L,k}$ denotes the k 'th largest eigenvalue of $\boldsymbol{\alpha}_L^{-1}$ and $\lambda_{R,k}$ denotes the k 'th largest eigenvalue of $\boldsymbol{\alpha}_R^{-1}$. The eigenvalues of $\boldsymbol{\alpha}_L^{-1}$ and $\boldsymbol{\alpha}_R^{-1}$ are real and non-negative.

Assumption 5.1. We assume that either $\boldsymbol{\alpha}_L^{-1}$ or $\boldsymbol{\alpha}_R^{-1}$ has r dominant eigenvalues. That means that $\frac{\lambda_{L,r}}{\lambda_{L,r+1}} \gg 1$ or $\frac{\lambda_{R,r}}{\lambda_{R,r+1}} \gg 1$.

Under assumption 5.1, $\boldsymbol{\alpha}_L^{-1}$ can be closely approximated by a positive semi-definite covariance matrix of rank r . This matrix approximation also holds for the covariance matrix of \mathbf{x}_i due to the relation (5.20). A natural qualitative argument is that \mathbf{x}_i approximately lies in the subspace spanned by the eigenvectors corresponding to the r dominant eigenvalues, resulting in promoting low-rank in \mathbf{X} . Let \mathcal{P}_L denotes the orthogonal projection onto the subspace spanned by the r eigenvectors of $\boldsymbol{\alpha}_L^{-1}$ corresponding to the r largest eigenvalues, and let \mathcal{P}_L^\perp denotes the orthogonal projection onto the subspace spanned by the eigenvectors corresponding to the $p - r$ smallest eigenvalues. Note that $\mathcal{P}_L^\perp = \mathbf{I}_p - \mathcal{P}_L$, and $\|\mathbf{x}_i\|_2^2 = \|\mathcal{P}_L^\perp \mathbf{x}_i\|_2^2 + \|\mathcal{P}_L \mathbf{x}_i\|_2^2$. For a scalar $0 \leq \varsigma < 1$, we want to show that the probability of the event $(\|\mathcal{P}_L^\perp \mathbf{x}_i\|_2^2 \leq \varsigma \|\mathcal{P}_L \mathbf{x}_i\|_2^2)$ is high under Assumption 5.1. First we state the following relation

$$\Pr(\|\mathcal{P}_L^\perp \mathbf{x}_i\|_2^2 \leq \varsigma \|\mathcal{P}_L \mathbf{x}_i\|_2^2) \geq 1 - \frac{B\left(\frac{r}{2}, \frac{n}{2}, \epsilon_1\right)}{B\left(\frac{r}{2}, \frac{n}{2}\right)} \quad (5.21)$$

$$\geq 1 - C_{p,r} \epsilon_1^{r/2}, \quad (5.22)$$

where $B(\cdot, \cdot, \cdot)$ is the incomplete beta function, $B(\cdot, \cdot) = B(\cdot, \cdot, 1)$ is the beta function, $C_{p,r}^{-1} = \frac{r}{2} B\left(\frac{r}{2}, \frac{n}{2}\right)$ and

$$\epsilon_1 = \frac{1}{1 + \varsigma \frac{\lambda_{L,r}}{\lambda_{L,r+1}}}.$$

The derivation of (5.21) and (5.22) is given in Section 5.5.8. Under some technical conditions and assumption 5.1, we have that $C_{p,r} \epsilon_1^{r/2} \ll 1$ and therefore $\Pr(\|\mathcal{P}_L^\perp \mathbf{x}_i\|_2^2 \leq \varsigma \|\mathcal{P}_L \mathbf{x}_i\|_2^2)$ is high. Similar arguments also can be put forward by considering a row vector of \mathbf{X} where the eigenvalues of $\boldsymbol{\alpha}_R^{-1}$ will play role under Assumption 5.1.

Using (5.17) we note that covariance of $\text{vec}(\mathbf{X})$ is $(\boldsymbol{\alpha}_R^{-1} \otimes \boldsymbol{\alpha}_L^{-1})$. Using \mathbf{X}_r to denote the best r -rank approximation of \mathbf{X} (in the sense of Frobenius norm) we find that probability of $(\|\mathbf{X} - \mathbf{X}_r\|_F^2 \leq \varsigma \|\mathbf{X}\|_F^2)$ can be bounded from below as follows

$$\begin{aligned} & \Pr(\|\mathbf{X} - \mathbf{X}_r\|_F^2 \leq \varsigma \|\mathbf{X}\|_F^2) \\ & \geq 1 - \frac{B\left(\frac{r^2}{2}, \frac{(p-r)(q-r)}{2}, \epsilon_2\right)}{B\left(\frac{r^2}{2}, \frac{(p-r)(q-r)}{2}\right)} \end{aligned} \quad (5.23)$$

$$\geq 1 - C_{p,q,r} \epsilon_2^{r^2/2}, \quad (5.24)$$

The derivation of (5.23) and (5.24) is given in Section 5.5.8. In (5.23) and (5.24), $C_{p,q,r}^{-1} = \frac{r^2}{2} B\left(\frac{r^2}{2}, \frac{(p-r)(q-r)}{2}\right)$ and

$$\epsilon_2 = \frac{1}{1 + \frac{\varsigma}{1-\varsigma} \frac{\lambda_{L,r}}{\lambda_{L,r+1}} \frac{\lambda_{R,r}}{\lambda_{R,r+1}}}.$$

Under some technical conditions and Assumption 5.1, we have that $C_{p,q,r} \epsilon_2^{r^2/2} \ll 1$, and $\Pr(\|\mathbf{X} - \mathbf{X}_r\|_F^2 \leq \varsigma \|\mathbf{X}\|_F^2)$ is high. We provide a numerical example in Figure 5.1 where $\Pr(\|\mathbf{X} - \mathbf{X}_r\|_F^2 \leq \varsigma \|\mathbf{X}\|_F^2)$ and the two lower bounds of (5.23) and (5.24) are shown. For the numerical experiments we used $p = q = 25$, $r =$

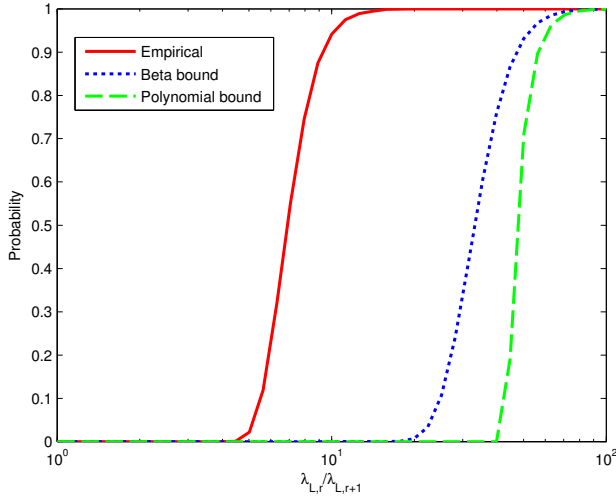


Figure 5.1: Plot of the empirical probability that $\|\mathbf{X} - \mathbf{X}_r\|_F^2 \leq 0.05 \|\mathbf{X}\|_F^2$ in red solid line, the bound (5.23) in blue dotted line and the bound (5.24) in green dashed line, versus the value of $\frac{\lambda_{L,r}}{\lambda_{L,r+1}} = \frac{\lambda_{R,r}}{\lambda_{R,r+1}}$ for $r = 3$ and $p = q = 25$.

3, $\varsigma = 0.05$ and $\frac{\lambda_{L,r}}{\lambda_{L,r+1}} = \frac{\lambda_{R,r}}{\lambda_{R,r+1}}$. In the simulation we used $\lambda_{L,i} = \lambda_{R,j}$ for $1 \leq i, j \leq r$ and $\lambda_{L,k} = \lambda_{R,l}$ for $r+1 \leq k \leq p$ and $r+1 \leq j \leq q$. We computed $\Pr(\|\mathbf{X} - \mathbf{X}_r\|_F^2 \leq \varsigma \|\mathbf{X}\|_F^2)$ as an empirical probability via Monte Carlo simulations.

5.4 Practical algorithms

We here derive the update equations for the variables in the two-sided model. The update equations for the one-sided models can be found by fixing one of the precision matrices. We denote the model parameters as $\Theta \triangleq \{\alpha_L, \alpha_R, \beta\}$. The MAP estimate of \mathbf{X} and Θ is

$$\hat{\mathbf{X}}, \hat{\Theta} = \arg \max_{\mathbf{X}, \Theta} p(\mathbf{X}, \mathbf{y}, \Theta).$$

We here assume that $p(\Theta) = p(\alpha_L) p(\alpha_R) p(\beta)$ for computational simplicity. Directly solving the MAP estimation problem is hard and often analytically intractable. Therefore various approximations are used to design practical algorithms. This section is dedicated to design estimation algorithms (type II estimators) via the expectation-maximization (EM) approach [DLR77, Bis06]. We assume that the noise precision β has a Gamma distribution as

$$p(\beta) = \text{Gamma}(\beta|a+1, b) = \frac{b^{a+1}}{\Gamma(a+1)} \beta^a e^{-b\beta},$$

with $a > -1$, $b > 0$ and $\beta \geq 0$.

The objective of EM is to solve the following problem

$$\max_{\Theta} \log p(\mathbf{y}, \Theta) = \max_{\Theta} \{\log p(\mathbf{y}|\Theta) + \log p(\Theta)\}, \quad (5.25)$$

which is the MAP estimate of Θ when \mathbf{X} is marginalized. It is thus an approximation of the MAP inference problem. EM minimizes (5.25) indirectly by minimizing the EM help function

$$\begin{aligned} Q(\Theta, \Theta') &= \mathcal{E}_{\mathbf{X}|\mathbf{y}, \Theta'}[\log p(\mathbf{y}, \mathbf{X}|\Theta)] \\ &= \int_{\mathbf{X}} p(\mathbf{X}|\mathbf{y}, \Theta') \log p(\mathbf{y}, \mathbf{X}|\Theta) d\mathbf{X}, \end{aligned}$$

where $\mathcal{E}[\cdot]$ denotes the expectation operator. The iterative formulation of EM guarantees a locally optimum solution of (5.25) through the following steps [DLR77].

1. Initialize the method with the parameter values $\Theta' = \{\alpha'_L, \alpha'_R, \beta'\}$.
2. **E-step:** Evaluate $p(\mathbf{X}|\mathbf{y}, \Theta')$ where Θ' is the values of Θ from the previous iteration. We find that

$$p(\mathbf{X}|\mathbf{y}, \Theta') = \mathcal{N}(\text{vec}(\mathbf{X}); \text{vec}(\hat{\mathbf{X}}), \Sigma'),$$

where

$$\begin{aligned} \text{vec}(\hat{\mathbf{X}}) &= \beta' \boldsymbol{\Sigma}' \mathbf{A}^\top \mathbf{y}, \\ \boldsymbol{\Sigma}' &= ((\boldsymbol{\alpha}'_R \otimes \boldsymbol{\alpha}'_L) + \beta' \mathbf{A}^\top \mathbf{A})^{-1}. \end{aligned} \quad (5.26)$$

3. **M-step:** Update $\boldsymbol{\Theta}$ as

$$\boldsymbol{\Theta} = \arg \max_{\boldsymbol{\Theta}} \{Q(\boldsymbol{\Theta}, \boldsymbol{\Theta}') + \log p(\boldsymbol{\Theta})\}, \quad (5.27)$$

where for our model

$$\begin{aligned} Q(\boldsymbol{\Theta}, \boldsymbol{\Theta}') &= \mathcal{E}_{\mathbf{X}|\mathbf{y}, \boldsymbol{\Theta}'}[\log p(\mathbf{y}, \mathbf{X}|\boldsymbol{\Theta})] = \text{constant} \\ &- \frac{\beta}{2} \|\mathbf{y} - \mathbf{A} \text{vec}(\hat{\mathbf{X}})\|_2^2 - \frac{1}{2} \text{tr}(\boldsymbol{\alpha}_L \hat{\mathbf{X}} \boldsymbol{\alpha}_R \hat{\mathbf{X}}^\top) \\ &- \frac{1}{2} \text{tr}(\boldsymbol{\Sigma}^{-1} \boldsymbol{\Sigma}') + \frac{q}{2} \log |\boldsymbol{\alpha}_L| + \frac{p}{2} \log |\boldsymbol{\alpha}_R| \\ &+ \frac{m}{2} \log \beta, \end{aligned} \quad (5.28)$$

and $\boldsymbol{\Sigma} = ((\boldsymbol{\alpha}_R \otimes \boldsymbol{\alpha}_L) + \beta \mathbf{A}^\top \mathbf{A})^{-1}$. In simulations, we initialized the variables as $\{\boldsymbol{\alpha}'_L, \boldsymbol{\alpha}'_R, \beta'\} = \{\mathbf{I}_p, \mathbf{I}_q, 1\}$.

By maximizing (5.27) for the noise precision β , we find the update equation

$$\beta = \frac{m + 2a}{\|\mathbf{y} - \mathbf{A} \text{vec}(\hat{\mathbf{X}})\|_2^2 + \text{tr}(\mathbf{A} \boldsymbol{\Sigma}' \mathbf{A}^\top) + 2b}. \quad (5.29)$$

The update equations of the precision matrices depend on their prior distribution. We here assume that both precision matrices have the same type of distribution. We find that the left and right precisions are updated as follows.

a) *The Schatten s -norm:* Using the Schatten s -norm prior (5.12) gives us the update equations

$$\begin{aligned} \boldsymbol{\alpha}_L &= c_s \left(\hat{\mathbf{X}} \boldsymbol{\alpha}'_R \hat{\mathbf{X}}^\top + \tilde{\boldsymbol{\Sigma}}_L + \epsilon \mathbf{I}_p \right)^{(s-2)/2}, \\ \boldsymbol{\alpha}_R &= c_s \left(\hat{\mathbf{X}}^\top \boldsymbol{\alpha}'_L \hat{\mathbf{X}} + \tilde{\boldsymbol{\Sigma}}_R + \epsilon \mathbf{I}_q \right)^{(s-2)/2}, \end{aligned} \quad (5.30)$$

where $c_s = (s/2)^{s/2}$, $\tilde{\boldsymbol{\Sigma}}_L \in \mathbb{R}^{p \times p}$ and $\tilde{\boldsymbol{\Sigma}}_R \in \mathbb{R}^{q \times q}$ are matrices with elements

$$\begin{aligned} [\tilde{\boldsymbol{\Sigma}}_L]_{ij} &= \text{tr}(\boldsymbol{\Sigma}'(\boldsymbol{\alpha}'_R \otimes \mathbf{E}_{ij}^{(L)})), \\ [\tilde{\boldsymbol{\Sigma}}_R]_{ij} &= \text{tr}(\boldsymbol{\Sigma}'(\mathbf{E}_{ij}^{(R)} \otimes \boldsymbol{\alpha}'_L)), \end{aligned}$$

and $\mathbf{E}_{ij}^{(L)} \in \mathbb{R}^{p \times p}$, $\mathbf{E}_{ij}^{(R)} \in \mathbb{R}^{q \times q}$ are matrices with ones in position (i, j) and zeros otherwise.

- b) *The log-determinant penalty:* For the log-determinant prior (5.14) the update equations become

$$\begin{aligned}\boldsymbol{\alpha}_L &= \nu \left(\hat{\mathbf{X}} \boldsymbol{\alpha}'_R \hat{\mathbf{X}}^\top + \tilde{\boldsymbol{\Sigma}}_L + \epsilon \mathbf{I}_p \right)^{-1}, \\ \boldsymbol{\alpha}_R &= \nu \left(\hat{\mathbf{X}}^\top \boldsymbol{\alpha}'_L \hat{\mathbf{X}} + \tilde{\boldsymbol{\Sigma}}_R + \epsilon \mathbf{I}_q \right)^{-1}.\end{aligned}\tag{5.31}$$

The derivations of (5.28) and (5.31) are given in Section 5.5.9.

4. Stop iterating if $\hat{\mathbf{X}}$ does not change significantly. Otherwise go to step 2.

5.4.1 Balancing the precisions

We have found that in practical algorithms, there is a chance that one of the two precisions becomes large and the other becomes small. A small precision results in numerical instability in the Kronecker covariance structure (5.17). To prevent the imbalance we rescale the matrix precisions in each iteration such that 1) the a-priori and a-posteriori squared Frobenius norm of \mathbf{X} are equal, i.e.

$$\begin{aligned}\mathcal{E}[\|\mathbf{X}\|_F^2 | \boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R] &= \text{tr}(\boldsymbol{\alpha}_L^{-1}) \text{tr}(\boldsymbol{\alpha}_R^{-1}) \\ &= \mathcal{E}[\|\mathbf{X}\|_F^2 | \boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R, \beta, \mathbf{y}] = \|\hat{\mathbf{X}}\|_F^2 + \text{tr}(\boldsymbol{\Sigma}),\end{aligned}$$

and 2) the contribution of each precision to the norm is equal, i.e.

$$\text{tr}(\boldsymbol{\alpha}_L^{-1}) = \text{tr}(\boldsymbol{\alpha}_R^{-1}).$$

The rescaling makes the algorithm more stable and often improves estimation performance.

5.4.2 Nomenclature

Here we explain the nomenclature used for the RSVM algorithms. We use the name left-sided RSVM for the left-sided model where $\boldsymbol{\alpha}_L$ is random and $\boldsymbol{\alpha}_R = \mathbf{I}_q$ and right-sided RSVM for the right-sided model where $\boldsymbol{\alpha}_R$ is random and $\boldsymbol{\alpha}_L = \mathbf{I}_p$. We refer to the model where both $\boldsymbol{\alpha}_L$ and $\boldsymbol{\alpha}_R$ are random as the two-sided RSVM or simply as RSVM. We assume that $\boldsymbol{\alpha}_L$ and $\boldsymbol{\alpha}_R$ are independent with distributions $p(\boldsymbol{\alpha}) \propto e^{-\frac{1}{2}K(\boldsymbol{\alpha})}$. We refer to the RSVM method with priors related to the log-determinant penalty function (5.14) as RSVM-LD and to the RSVM method priors related to the Schatten s -norm penalty function (5.12) as RSVM-SN. For example, left-sided RSVM-LD is RSVM with log-determinant prior (5.14) and a left-sided precision.

5.4.3 Complexity analysis

One motivation for the RSVM algorithm is that it can infer low-rank directly without factorizing the matrix \mathbf{X} . However, one disadvantage of this is that for given

precision matrices, the estimation of \mathbf{X} is a Gaussian inference problem with pq variables, requiring computationally demanding matrix inverses. To study how the problem size affects the runtime, we here study the computational complexity of the algorithm.

The most computationally demanding operation of the RSVM algorithm is the computation of Σ . A direct approach to computing Σ is to invert a $pq \times pq$ matrix, this requires $\mathcal{O}((pq)^3)$ multiplications. However, this does not use the fact that $\text{rank}(\mathbf{A}^\top \mathbf{A}) \leq m \ll pq$. By using the Woodbury matrix inversion formula [Bis06] we get that

$$\Sigma = (\alpha_R^{-1} \otimes \alpha_L^{-1}) - (\alpha_R^{-1} \otimes \alpha_L^{-1}) \mathbf{A}^\top \mathbf{C}_y^{-1} \mathbf{A} (\alpha_R^{-1} \otimes \alpha_L^{-1})$$

where

$$\mathbf{C}_y = \beta^{-1} \mathbf{I}_m + \mathbf{A} (\alpha_R^{-1} \otimes \alpha_L^{-1}) \mathbf{A}^\top.$$

The computation of Σ thus reduces to computing $(\alpha_R^{-1} \otimes \alpha_L^{-1})$ (requiring $\mathcal{O}(p^3 + q^3 + p^2 q^2)$ multiplications), $\mathbf{A} (\alpha_R^{-1} \otimes \alpha_L^{-1})$ and $\mathbf{A} (\alpha_R^{-1} \otimes \alpha_L^{-1}) \mathbf{A}^\top$ (requiring $\mathcal{O}(mp^2 q^2)$ multiplications) and \mathbf{C}_y^{-1} (requiring $\mathcal{O}(m^3)$ multiplications). The total complexity is therefore

$$\mathcal{O}(p^3 + q^3 + p^2 q^2 + m^3 + mp^2 q^2) \approx \mathcal{O}(mp^2 q^2),$$

where we used that $\mathcal{O}(mp^2 q^2)$ dominates over the other terms when $m \ll pq$. The use of the Woodbury matrix inversion thus helps to reduce computational complexity per iteration from $\mathcal{O}((pq)^3)$ to $\mathcal{O}(mpq^2)$. Further we note that the complexity does not depend on the rank of \mathbf{X} . This is because RSVM, in contrast to factorization based methods [BLMK12], does not use rank information as an a-priori information.

5.5 Simulation experiments

5.5.1 Simulations setup

Here we describe datasets, performance measure, experimental setup, competing algorithms and computational resources. In the experiments we used synthetic data for low-rank matrix reconstruction (LRMR) and low-rank matrix completion (LRMC). We used the MovieLens dataset [HKBR99] as a real dataset for matrix completion. To compare the algorithms for synthetic data, we use the normalized-mean-square-error

$$\text{NMSE} = \frac{\mathcal{E}[\|\hat{\mathbf{X}} - \mathbf{X}\|_F^2]}{\mathcal{E}[\|\mathbf{X}\|_F^2]}$$

as the performance measure. The NMSE was evaluated by averaging over many instances of \mathbf{n} , \mathbf{A} and \mathbf{X} using Monte-Carlo simulations. For experiments with synthetic data, we evaluated the NMSE as follows:

1. For LRMR, the random measurement matrix $\mathbf{A} \in \mathbb{R}^{m \times pq}$ was generated by independently drawing the elements from $\mathcal{N}(0, 1)$ and normalizing the column vectors to unit norm. For LRMC, we chose m elements uniformly at random from $[p] \times [q]$. We then chose \mathbf{A} such that it selected the elements from $\text{vec}(\mathbf{X})$ by letting one element in each row be one and the other elements zero.
2. Matrices $\mathbf{F} \in \mathbb{R}^{p \times r}$ and $\mathbf{B} \in \mathbb{R}^{q \times r}$ with elements drawn from $\mathcal{N}(0, 1)$ were randomly generated and the matrix \mathbf{X} was formed as $\mathbf{X} = \mathbf{FB}^\top$. Note that \mathbf{X} has rank r (with probability one).
3. Generate the measurement $\mathbf{y} = \mathbf{A}\text{vec}(\mathbf{X}) + \mathbf{n}$, where \mathbf{n} is drawn from $\mathcal{N}(\mathbf{0}, \beta^{-1}\mathbf{I}_m)$ and β^{-1} is chosen such that the signal-to-measurement-noise ratio (SNR) is fixed. The SNR is given by

$$\text{SNR} = \frac{\mathcal{E}[\|\mathbf{A}\text{vec}(\mathbf{X})\|_2^2]}{\mathcal{E}[\|\mathbf{n}\|_2^2]} = \beta \frac{r pq}{m},$$

for LRMR and $\text{SNR} = \beta r$ for LRMC.

4. Estimate $\hat{\mathbf{X}}$ using competing algorithms and calculate the error $\|\hat{\mathbf{X}} - \mathbf{X}\|_F^2$.
5. Repeat steps 2 – 4 for T_1 times.
6. Repeat steps 1 – 5 for T_2 times.
7. Then compute the NMSE by averaging.

In the simulations we chose $T_1 = T_2 = 10$, which means that the averaging was done over 100 realizations. Further, we evaluated the performance of RSVM vis-a-vis other methods for LRMR and LRMC. For LRMR, we compared with NN and the variational Bayes method of [BLMK12] (referred henceforth as VB-1). The use of VB-1 for LRMR was not addressed in [BLMK12] and hence we derive the algorithm in Section 5.5.12. Further, following (5.3), we also compare with the performance of

$$\min_{\mathbf{X}} \text{tr}((\mathbf{X}\mathbf{X}^\top)^{s/2}) \text{ s.t. } \|\mathbf{y} - \mathbf{A}\text{vec}(\mathbf{X})\|_2 \leq \delta, \quad (5.32)$$

with $s = 0.5$ and $\delta = \beta^{-1}\sqrt{m} + \sqrt{8m}$ as in (5.4), where β denotes the true noise precision. Both NN and SNA thus require knowledge of the noise power. The SNA problem is non-convex and we use a gradient search method to find the solution; to initialize the method, we used $\text{vec}(\hat{\mathbf{X}}) = \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top)^{-1}\mathbf{y}$. For LRMC, we compared with NN, VB-1, Probabilistic Matrix Factorization (PMF) [MS07], the Weighted Trace Norm (WTN) [SS10] and the variational Bayes method of [LT07] (referred to as VB-2). Finally, we mention our computational resources. For the experiments, we used a Dell Latitude E6400 laptop computer with a 3GHz processor and 8GB memory.

5.5.2 Experiments using synthetic data for LRMR

The objective of our first experiment is to compare the performance of the two-sided precision based model the one-sided precision based models. For the one-sided model, we also have two choices – left-sided and right-sided – and hence it interesting to know which choice is better for a particular setup. In this experiment, we fixed $\text{rank}(\mathbf{X}) = r = 3$, $p = 15$, $q = 30$, $\text{SNR} = 20$ dB and varied m . The results are shown in Figure 5.2 where NMSE is plotted against the normalized number of measurements $m/(pq)$. We note that for RSVM-LD, the right-sided model performs best for $m/pq < 0.6$, the left-sided model performs best for $m/pq > 0.6$ and that the two-sided RSVM-LD gives a good compromise between the left- and right-sided RSVM-LD. For RSVM-SN we find that the left-sided model performs better than both the right- and two-sided model for $m/pq \neq 0.7$. However, the two-sided model has a more consistent performance improvement with increasing m/pq while the left and right-sided models show performance degradation for $m/pq = 0.7$ and $m/pq = 0.8$, respectively. Henceforth we use the two-sided models because of their reasonable good performance. It is possible that the one-sided models are preferable in other scenarios. We mention that, for RSVM-SN, it was empirically found that $s = 0.5$ provides good performance. The same trend also repeats for LRMC, reported in section 5.5.3. Henceforth RSVM-SN with $s = 0.5$ is used unless stated otherwise.

In the second experiment we report the LRMR performance of RSVM-LD and RSVM-SN vis-a-vis NN, VB-1 and SNA. We mention that NN and SNA know the measurement noise power (see (5.4) and (5.32)), and VB-1 knows the rank of \mathbf{X} . The results are shown in Figure 5.3. With the parameters of the first experiment ($\text{rank}(\mathbf{X}) = r = 3$, $p = 15$, $q = 30$, $\text{SNR} = 20$ dB), we show NMSE vs. $m/(pq)$ in Figure 5.3 (a). We observe that RSVM-LD provides the best performance for $m/pq < 0.9$, whereas NN, SNA and RSVM-SN are close to each other with NN performing better than SNA and RSVM-SN. In Figure 5.3 (b), we report performance NMSE vs. SNR while rank $r = 3$ and $m/(pq) = 0.7$ are fixed. We find that RSVM-LD shows best performance in the middle SNR region ($15 < \text{SNR} < 35$), while NN and SNA perform best in the low and high SNR regions. Next, in Figure 5.3 (c), we report the NMSE vs. rank for $m/(pq) = 0.7$ and $\text{SNR} = 20$ dB. We find that RSVM-LD is the best, and RSVM-SN and NN are comparable. At this point it is interesting to investigate the performance at higher SNR, shown in Figure 5.4 for $\text{SNR} = 40$ dB. While NN performs better for the case of NMSE versus $m/(pq)$ in Figure 5.4 (a) where $r = 3$, we notice that RSVM-LD is promising when rank is higher as reported in Figure 5.4 (b). Finally, in Figure 5.5 we show the cpu execution times of all competing algorithms for the setup reported in Figure 5.3 (c). From Figure 5.5, we conclude that NN, implemented using the cvx toolbox [GBY08], is the fastest algorithm followed by the RSVM algorithms (implemented in Matlab).

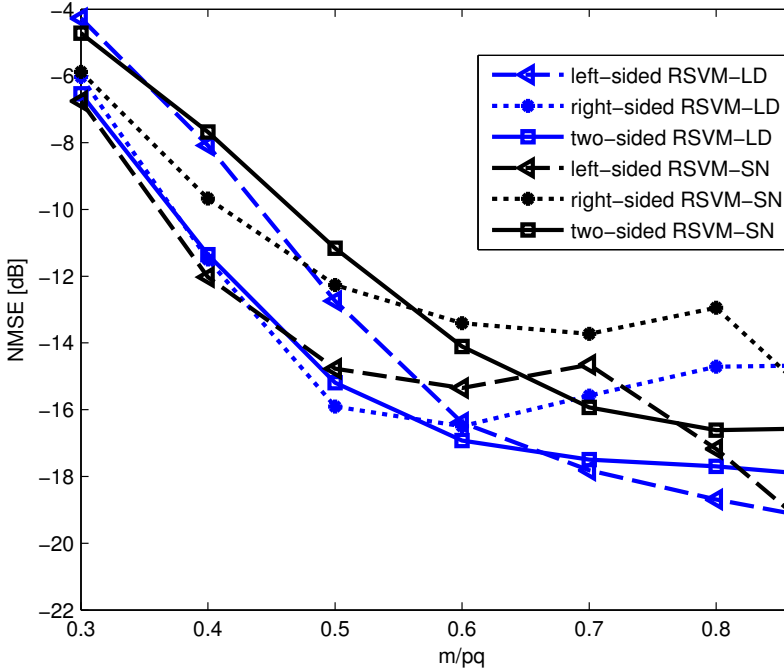


Figure 5.2: NMSE vs. $m/(pq)$ for LRMR at SNR=20 dB.

5.5.3 Experiments using synthetic data for LRMC

For LRMC, the objective of the first experiment is to empirically find a good choice of s for RSVM-SN. Like the first experiment for LRMR in section 5.5.2, we fixed $\text{rank}(\mathbf{X}) = r = 3$, $p = 15$, $q = 30$, SNR = 20 dB and varied m . The performance of RSVM-SN for different s is shown in Figure 5.6. We found that $s = 0.5$ is a good choice, and decided to use that throughout all relevant experiments.

Like the second experiment for LRMR in section 5.5.2, we conducted the second experiment here to evaluate the LRMC performance of RSVM-LD and RSVM-SN vis-a-vis NN, VB-1, VB-2, PMF and WTN. We used Matlab codes for VB-1 and PMF from their respective authors. For other methods, we used our own codes. We manually tuned the competing algorithms to improve their performance. The results are shown in Figure 5.7. Figure 5.7 (a) shows NMSE vs. $m/(pq)$ at $r = 3$, $p = 15$, $q = 30$, SNR = 20 dB. We observe that for $m/pq \leq 0.5$, RSVM-LD, PMF and VB-2 provide the best performance while for $m/pq \geq 0.6$ VB-1 performs the best. Then, in Figure 5.7 (b), we report performance NMSE versus SNR where rank $r = 3$ and $m/(pq) = 0.7$ are fixed, and find that VB-1 shows best performance closely followed by PMF and RSVM-LD. Next, in Figure 5.7 (c), we report the

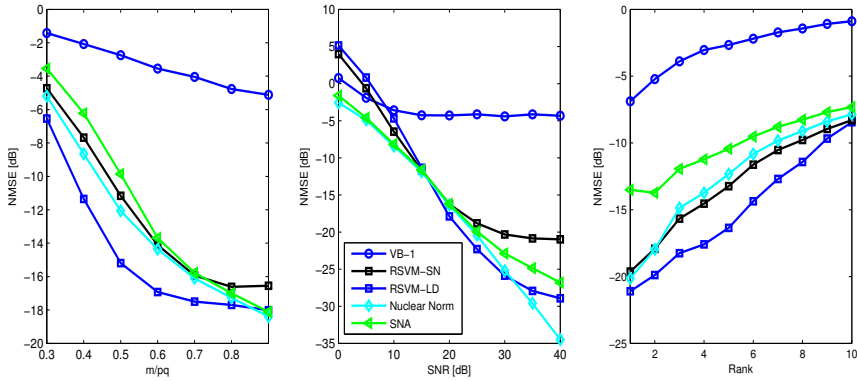


Figure 5.3: Comparison of algorithms for LRMR. (a) NMSE versus $m/(pq)$ at SNR=20 dB and $r=3$. (b) NMSE versus SNR at rank $r=3$ and $m/(pq)=0.7$. (c) NMSE versus rank r at $m/(pq)=0.7$ and SNR=20 dB.

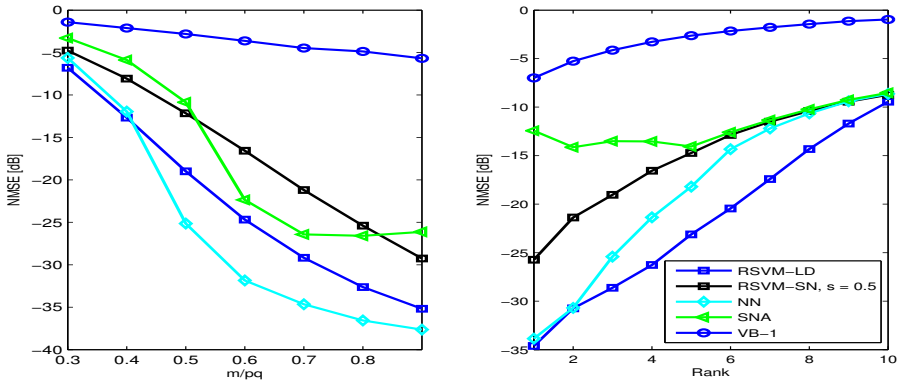


Figure 5.4: Comparison of algorithms for LRMR at SNR=40 dB. (a) NMSE versus $m/(pq)$ for $r=3$. (b) NMSE versus rank r for $m/(pq)=0.7$.

NMSE versus rank for fixed $m/(pq)=0.7$ and SNR=20 dB, we find that VB-1 gave the best performance for rank ≤ 3 but then quickly degrades in performance. For higher ranks, while PMF showed the best performance, the proposed RSVM showed good performance.

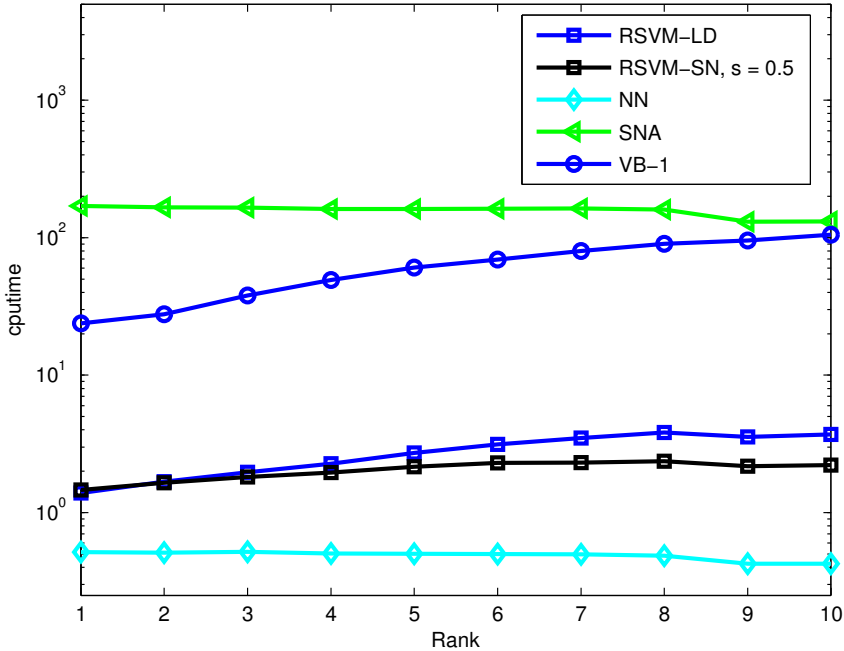


Figure 5.5: Mean cputime versus rank r for the setup in Figure 5.3 (c).

5.5.4 Experiments using MovieLens data for LRMC - Movie rating prediction

To evaluate the performance on real data we used the MovieLens 100K dataset [HKBR99]. The dataset contains user-movie pairs where in each pair, a user provides an integer rating between 1 – 5 to a movie. Each user has only rated a few movies according to the features of movies, and the underlying assumption is that the rating matrix has a factorized representation, leading to low-rank. Now let us assume that we observe few ratings of the large rating matrix randomly. The goal is to infer the missing ratings from the observed ratings and hence the problem is an LRMC problem.

In our experimental study, we used the u1 datasets (both training and testing) from the MovieLens dataset. Then we used a portion of the u1 datasets as \mathbf{X} . The dimensions of \mathbf{X} are $p = q = 100$. According to the MovieLens dataset instructions, we used $m = 307$ measurements. The measurements are collected via a pre-determined element picking matrix \mathbf{A} . Using the measurements and \mathbf{X} from the u1 training data, we learn the model parameters for all competing algorithms. Then, using the learned parameters we perform LRMC for the u1 test data. For

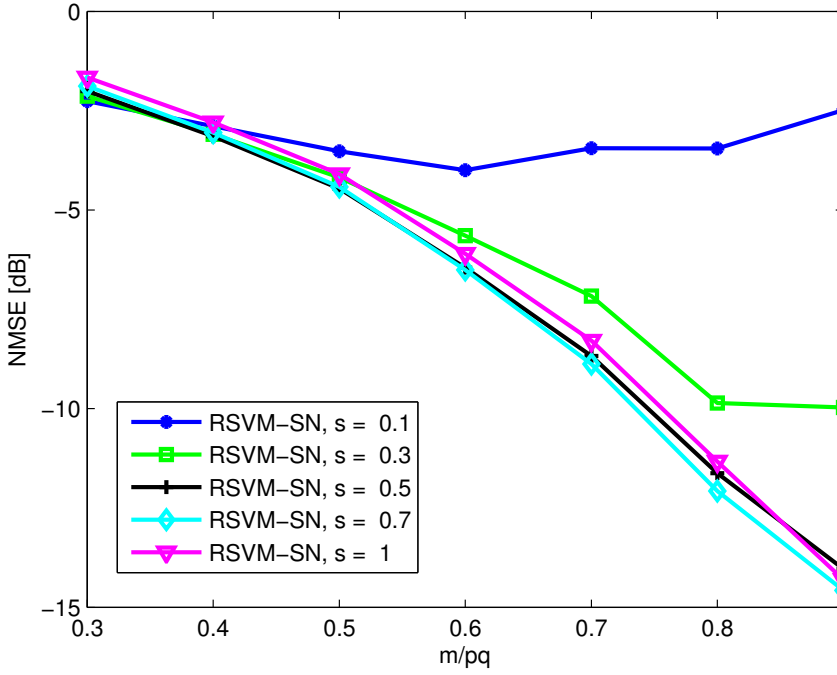


Figure 5.6: LRM performance: NMSE versus $m/(pq)$ for RSVM-SN at different choice of s .

performance comparison, we use root-mean-square-error

$$\text{RMSE} = \frac{1}{|J_{test}|} \sqrt{\sum_{(i,j) \in J_{test}} (X_{ij} - \hat{X}_{ij})^2},$$

where J_{test} denotes the set of unknown ratings, as this is a standard performance measure for movie rating prediction. The performance of all competing algorithms are shown in Table 5.1. It can be observed that RSVM algorithms provide good performance.

5.5.5 Reproducible research

In the spirit of reproducible research, we provide code necessary for reproducing the results in the website: https://github.com/MartinSundin/rsvm_simulation_code. The code can be used to reproduce the figures 5.2, 5.3, 5.6 and 5.7.

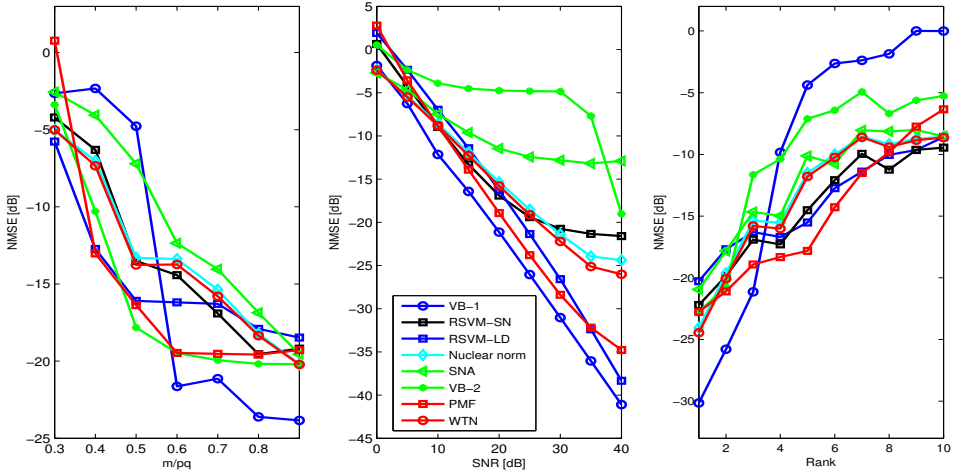


Figure 5.7: Comparison of algorithms for LRMC. (a) NMSE versus $m/(pq)$ at SNR=20 dB and $r = 3$. (b) NMSE versus SNR at rank $r = 3$ and $m/(pq) = 0.7$. (c) NMSE versus rank r at $m/(pq) = 0.7$ and SNR = 20 dB.

Table 5.1: RMSE of algorithms for MovieLens

Algorithm	RMSE
RSVM-LD	0.0410
RSVM-SN	0.0588
NN	0.0757
VB-1	0.0389
SNA	0.1255
VB-2	0.0389
PMF	0.1258
WTN	0.1249

5.5.6 The Laplace transform for positive definite matrices

We here summarize the definition of the Laplace transform for positive definite matrices. Further details can be found in [Ter12]. Let $\mathbb{S}_+^n = \{\mathbf{Z} \in \mathbb{R}^{n \times n} : \mathbf{Z} \succeq \mathbf{0}\}$ be the space of $n \times n$ positive definite matrices and let f be a real valued function on \mathbb{S}_+^n . The matrix Laplace transform of f at $\mathbf{Y} \in \mathbb{S}_+^n$ is

$$\mathcal{L}f(\mathbf{Y}) = \int_{\mathbf{Z} \succeq \mathbf{0}} f(\mathbf{Z}) e^{-\text{tr}(\mathbf{Z}\mathbf{Y})} d\mathbf{Z}, \text{ where } d\mathbf{Z} = \prod_{1 \leq i \leq j \leq n} dZ_{ij}.$$

The transform is defined for *sufficiently nice functions* [Ter12] for which it converges when $\operatorname{Re} \mathbf{Y} \succeq \mathbf{Z}_*$ for some \mathbf{Z}_* . The inverse Laplace transform can be expressed as [Ter12]

$$\begin{aligned} & \frac{1}{(2\pi i)^{n(n+1)/2}} \int_{\operatorname{Re} \mathbf{Y} = \mathbf{Z}_*} \mathcal{L}f(\mathbf{Y}) e^{\operatorname{tr}(\mathbf{Y}\mathbf{Z})} d\mathbf{Y} \\ &= \begin{cases} f(\mathbf{Z}) & , \text{ if } \mathbf{Z} \in \mathbb{S}_+^n, \\ 0, & , \text{ otherwise} \end{cases}. \end{aligned}$$

5.5.7 Derivation of the Laplace Approximation

The Laplace approximation is an approximation of the integral

$$I = \int e^{-\frac{1}{2}f(\mathbf{a})} d\mathbf{a},$$

where the integral is over $\mathbf{a} \in \mathbb{R}^n$. When the function $e^{-\frac{1}{2}f(\mathbf{a})}$ is sufficiently well behaved, the integral can be well approximated by the Laplace approximation. In the approximation, the function $f(\mathbf{a})$ is approximated by a second order polynomial around its minima \mathbf{a}_0 as

$$f(\mathbf{a}) \approx f(\mathbf{a}_0) + \frac{1}{2}(\mathbf{a} - \mathbf{a}_0)^\top \mathbf{H}(\mathbf{a} - \mathbf{a}_0),$$

where $\mathbf{H} = \nabla^2 f(\mathbf{a})|_{\mathbf{a}=\mathbf{a}_0}$ is the Hessian of $f(\mathbf{a})$ at \mathbf{a}_0 . The term linear in \mathbf{a} vanishes and $\mathbf{H} \succ \mathbf{0}$ at \mathbf{a}_0 since we expand around a minima. With this approximation, the integral becomes

$$I \approx \int e^{-\frac{1}{2}f(\mathbf{a}_0) - \frac{1}{4}(\mathbf{a} - \mathbf{a}_0)^\top \mathbf{H}(\mathbf{a} - \mathbf{a}_0)} d\mathbf{a} = \sqrt{\frac{(4\pi)^n}{|\mathbf{H}|}} e^{-\frac{1}{2}f(\mathbf{a}_0)}.$$

In (5.8), the integral is given by

$$I = \frac{1}{(2\pi)^{pq/2}} \int_{\boldsymbol{\alpha} \succ \mathbf{0}} e^{-\frac{1}{2}[\operatorname{tr}(\boldsymbol{\alpha}\mathbf{Z}) - q \log |\boldsymbol{\alpha}| + K(\boldsymbol{\alpha})]} d\boldsymbol{\alpha}.$$

Set $f(\mathbf{a}) = \operatorname{tr}(\boldsymbol{\alpha}\mathbf{Z}) - q \log |\boldsymbol{\alpha}| + K(\boldsymbol{\alpha})$, where $\mathbf{a} = \operatorname{vech}(\boldsymbol{\alpha})$ and $\operatorname{vech}(\cdot)$ is the *half-vectorization operator* for symmetric matrices, e.g.

$$\operatorname{vech} \begin{pmatrix} a & b \\ b & c \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}.$$

Let $\boldsymbol{\alpha}_0 \succ \mathbf{0}$ denote the minima of $f(\mathbf{a})$ and \mathbf{H} the Hessian at $\boldsymbol{\alpha}_0$. Assuming that $\boldsymbol{\alpha}_0$ and \mathbf{H} are “large” in the sense that the integral over $\boldsymbol{\alpha} \succ \mathbf{0}$ can be approximated

by the integral over $\mathbf{a} \in \mathbb{R}^{p \times p}$ we find that

$$\begin{aligned} I &\approx \frac{1}{(2\pi)^{pq/2}} \int e^{-\frac{1}{2}f(\mathbf{a}_0) - \frac{1}{4}(\mathbf{a} - \mathbf{a}_0)^\top \mathbf{H}(\mathbf{a} - \mathbf{a}_0)} d\mathbf{a} \\ &= \frac{(4\pi)^{p^2/2}}{(2\pi)^{pq/2} |\mathbf{H}|^{1/2}} e^{-\frac{1}{2}f(\mathbf{a}_0)}, \end{aligned}$$

where $\mathbf{a}_0 = \text{vech}(\boldsymbol{\alpha}_0)$.

5.5.8 Derivation of (5.21), (5.22), (5.23) and (5.24)

Let $P_1 \triangleq \Pr(\|\mathcal{P}_L^\perp \mathbf{x}_i\|_2^2 \leq \varsigma \|\mathcal{P}_L \mathbf{x}_i\|_2^2)$ and assume that $1 \leq r < p$ and let $n = p - r$. The random variables $\mathcal{P}_L \mathbf{x}_i$ and $\mathcal{P}_L^\perp \mathbf{x}_i$ are independent zero-mean Gaussian variables. Let $w_1, w_2, \dots, w_r, z_1, z_2, \dots, z_n$ be i.i.d. $\mathcal{N}(0, 1)$ variables. For the proofs of (5.21), (5.22), we find that

$$\begin{aligned} P_1 &= \Pr\left(\frac{\lambda_{L,r+1}z_1^2 + \lambda_{L,r+2}z_2^2 + \dots + \lambda_{L,p}z_n^2}{\lambda_{L,1}w_1^2 + \lambda_{L,2}w_2^2 + \dots + \lambda_{L,r}w_r^2} \leq \varsigma\right) \\ &\stackrel{(a)}{\geq} \Pr\left(\frac{\lambda_{L,r+1}(z_1^2 + z_2^2 + \dots + z_n^2)}{\lambda_{L,r}(w_1^2 + w_2^2 + \dots + w_r^2)} \leq \varsigma\right) \\ &= 1 - \Pr\left(\frac{\lambda_{L,r+1}(z_1^2 + z_2^2 + \dots + z_n^2)}{\lambda_{L,r}(w_1^2 + w_2^2 + \dots + w_r^2)} \geq \varsigma\right) \\ &= 1 - \Pr\left(\frac{(w_1^2 + w_2^2 + \dots + w_r^2)/r}{(z_1^2 + z_2^2 + \dots + z_n^2)/n} \leq \frac{n\lambda_{L,r+1}}{\varsigma r\lambda_{L,r}}\right) \\ &\stackrel{(b)}{\geq} 1 - \frac{B\left(\frac{r}{2}, \frac{n}{2}, \epsilon_1\right)}{B\left(\frac{r}{2}, \frac{n}{2}\right)} \\ &\stackrel{(c)}{\geq} 1 - C_{p,r} \epsilon_1^{r/2}, \end{aligned} \tag{5.33}$$

where $B(\cdot, \cdot, \cdot)$ is the incomplete beta function, $B(\cdot, \cdot) = B(\cdot, \cdot, 1)$ is the beta function and

$$C_{p,r}^{-1} = \frac{r}{2} B\left(\frac{r}{2}, \frac{n}{2}\right), \text{ and } \epsilon_1 = \frac{1}{1 + \varsigma \frac{\lambda_{L,r}}{\lambda_{L,r+1}}}.$$

In (a), $\frac{\lambda_{L,r+1}z_1^2 + \lambda_{L,r+2}z_2^2 + \dots + \lambda_{L,p}z_n^2}{\lambda_{L,1}w_1^2 + \lambda_{L,2}w_2^2 + \dots + \lambda_{L,r}w_r^2} \leq \frac{\lambda_{L,r+1}(z_1^2 + z_2^2 + \dots + z_n^2)}{\lambda_{L,r}(w_1^2 + w_2^2 + \dots + w_r^2)}$ is used. In (b) we note that $\frac{(w_1^2 + w_2^2 + \dots + w_r^2)/r}{(z_1^2 + z_2^2 + \dots + z_n^2)/n}$ is F-distributed. Then, in (c), we use the following relation

$$\begin{aligned} B\left(\frac{r}{2}, \frac{n}{2}, \epsilon_1\right) &= \int_0^{\epsilon_1} t^{r/2-1} (1-t)^{n/2-1} dt \\ &= \epsilon_1^{r/2} \int_0^1 u^{r/2-1} (1-\epsilon_1 u)^{n/2-1} du \\ &\leq \frac{2}{r} \epsilon_1^{r/2}. \end{aligned} \tag{5.34}$$

This shows (5.22).

Suppose $P_2 \triangleq \Pr(\|\mathbf{X} - \mathbf{X}_r\|_F^2 \leq \varsigma \|\mathbf{X}\|_F^2)$. To show (5.24) we introduce the projection \mathcal{P}_R onto the subspace spanned by the r eigenvectors of $\boldsymbol{\alpha}_R^{-1}$ corresponding

to the r largest eigenvalues and $\mathcal{P}_R^\perp = \mathbf{I}_q - \mathcal{P}_R$. Using the projection operators \mathcal{P}_L and \mathcal{P}_R , we have the following relation

$$\begin{aligned} \|\mathbf{X}\|_F^2 &= \|\mathcal{P}_L \mathbf{X} \mathcal{P}_R\|_F^2 + \|\mathcal{P}_L \mathbf{X} \mathcal{P}_R^\perp\|_F^2 \\ &\quad + \|\mathcal{P}_L^\perp \mathbf{X} \mathcal{P}_R\|_F^2 + \|\mathcal{P}_L^\perp \mathbf{X} \mathcal{P}_R^\perp\|_F^2. \end{aligned}$$

If \mathbf{P}_1 and \mathbf{P}_2 are two projection operators with properties $\text{rank}(\mathbf{P}_1) = p - r$ and $\text{rank}(\mathbf{P}_2) = q - r$, then we have the following relation

$$\begin{aligned} \|\mathbf{X} - \mathbf{X}_r\|_F^2 &= \min_{\mathbf{P}_1, \mathbf{P}_2} \|\mathbf{P}_1 \mathbf{X} \mathbf{P}_2\|_F^2 \\ &\leq \|\mathcal{P}_L^\perp \mathbf{X} \mathcal{P}_R^\perp\|_F^2. \end{aligned}$$

We now find the following relation

$$\begin{aligned} \mathbf{P}_2 &\triangleq \Pr(\|\mathbf{X} - \mathbf{X}_r\|_F^2 \leq \varsigma \|\mathbf{X}\|_F^2) \\ &\geq \Pr(\|\mathcal{P}_L^\perp \mathbf{X} \mathcal{P}_R^\perp\|_F^2 \leq \varsigma \|\mathbf{X}\|_F^2) \\ &\geq \Pr(\|\mathcal{P}_L^\perp \mathbf{X} \mathcal{P}_R^\perp\|_F^2 \leq \varsigma (\|\mathcal{P}_L \mathbf{X} \mathcal{P}_R\|_F^2 + \|\mathcal{P}_L^\perp \mathbf{X} \mathcal{P}_R^\perp\|_F^2)) \\ &= \Pr\left(\|\mathcal{P}_L^\perp \mathbf{X} \mathcal{P}_R^\perp\|_F^2 \leq \frac{\varsigma}{1-\varsigma} \|\mathcal{P}_L \mathbf{X} \mathcal{P}_R\|_F^2\right) \triangleq \mathbf{P}_3 \end{aligned}$$

The random variables $\mathcal{P}_L \mathbf{X} \mathcal{P}_R$ and $\mathcal{P}_L^\perp \mathbf{X} \mathcal{P}_R^\perp$ are independent zero-mean Gaussian variables. Let $\{w_{i,j}\}$, and $\{z_{k,l}\}$ be i.i.d. $\mathcal{N}(0,1)$ variables, where $1 \leq i, j \leq r$, $1 \leq k \leq p - r$ and $1 \leq l \leq q - r$. This gives us that [GN99]

$$\mathbf{P}_3 = \Pr\left(\frac{\sum_{k=1}^{p-r} \sum_{l=1}^{q-r} \lambda_{L,r+k} \lambda_{L,r+l} w_{k,l}^2}{\sum_{1 \leq i,j \leq r} \lambda_{L,i} \lambda_{R,j} z_{i,j}^2} \leq \frac{\varsigma}{1-\varsigma}\right).$$

Using that

$$\begin{aligned} \sum_{1 \leq i,j \leq r} \lambda_{L,i} \lambda_{R,j} z_{i,j}^2 &\geq \lambda_{L,r} \lambda_{R,r} \sum_{1 \leq i,j \leq r} z_{i,j}^2, \\ \sum_{k=1}^{p-r} \sum_{l=1}^{q-r} \lambda_{L,r+k} \lambda_{R,r+l} w_{k,l}^2 &\leq \lambda_{L,r+1} \lambda_{R,r+1} \sum_{k=1}^{p-r} \sum_{l=1}^{q-r} w_{k,l}^2, \end{aligned}$$

we find that

$$\mathbf{P}_3 \geq \Pr\left(\frac{\sum_{k=1}^{p-r} \sum_{l=1}^{q-r} w_{k,l}^2}{\sum_{1 \leq i,j \leq r} z_{i,j}^2} \leq \frac{\varsigma}{1-\varsigma} \frac{\lambda_{L,r} \lambda_{R,r}}{\lambda_{L,r+1} \lambda_{R,r+1}}\right).$$

Since

$$\frac{\frac{1}{(p-r)(q-r)} \sum_{k=1}^{p-r} \sum_{l=1}^{q-r} w_{k,l}^2}{\frac{1}{r^2} \sum_{1 \leq i,j \leq r} z_{i,j}^2}$$

is F-distributed, we find that

$$P_2 \geq P_3 \geq 1 - \frac{B\left(\frac{r^2}{2}, \frac{(p-r)(q-r)}{2}, \epsilon_2\right)}{B\left(\frac{r^2}{2}, \frac{(p-r)(q-r)}{2}\right)} \geq 1 - C_{r,p,q} \epsilon_2^{\frac{r^2}{2}},$$

where

$$C_{r,p,q}^{-1} = \frac{r^2}{2} B\left(\frac{r^2}{2}, \frac{(p-r)(q-r)}{2}\right),$$

$$\epsilon_2 = \frac{1}{1 + \frac{\frac{\lambda_{L,r}}{\lambda_{L,r+1}}}{1-\zeta} + \frac{\frac{\lambda_{R,r}}{\lambda_{R,r+1}}}{1-\zeta}}.$$

5.5.9 The EM help function

The EM help function $Q(\Theta, \Theta')$ is given by

$$\begin{aligned} Q(\Theta, \Theta') &= \mathcal{E}_{\mathbf{X}|\mathbf{y}, \Theta'}[\log p(\mathbf{X}|\mathbf{y}, \Theta)] = c + \frac{m}{2} \log \beta \\ &\quad - \frac{\beta}{2} \mathcal{E}[\|\mathbf{y} - \mathbf{A} \text{vec}(\mathbf{X})\|_2^2] - \frac{1}{2} \mathcal{E}[\text{tr}(\alpha_L \mathbf{X} \alpha_R \mathbf{X}^\top)] \\ &\quad + \frac{q}{2} \log |\alpha_L| + \frac{p}{2} \log |\alpha_R|, \end{aligned}$$

where c is a constant. Using that

$$\begin{aligned} \mathcal{E}[\|\mathbf{y} - \mathbf{A} \text{vec}(\mathbf{X})\|_2^2] &= \|\mathbf{y}\|_2^2 - 2\mathbf{y}^\top \mathbf{A} \text{vec}(\hat{\mathbf{X}}) \\ &\quad + \text{tr}(\mathbf{A}^\top \mathbf{A} (\text{vec}(\hat{\mathbf{X}}) \text{vec}(\hat{\mathbf{X}})^\top + \Sigma')) \\ &= \|\mathbf{y} - \mathbf{A} \text{vec}(\hat{\mathbf{X}})\|_2^2 + \text{tr}(\mathbf{A}^\top \mathbf{A} \Sigma'), \end{aligned}$$

and

$$\begin{aligned} \mathcal{E}[\text{tr}(\alpha_L \mathbf{X} \alpha_R \mathbf{X}^\top)] &= \text{tr}((\alpha_R \otimes \alpha_L) (\text{vec}(\hat{\mathbf{X}}) \text{vec}(\hat{\mathbf{X}})^\top + \Sigma')) \\ &= \text{tr}(\alpha_L \hat{\mathbf{X}} \alpha_R \hat{\mathbf{X}}^\top) + \text{tr}((\alpha_R \otimes \alpha_L) \Sigma'), \end{aligned}$$

we recover the expression (5.28) for the EM help function.

5.5.10 Details for the RSVM with the Schatten s -norm penalty

We here set $\mathbf{S} = \epsilon \mathbf{I}_q$ to keep the derivation more general. The regularized Schatten s -norm penalty is given by

$$\tilde{g}(\mathbf{Z}) = \text{tr}((\mathbf{X}^\top \mathbf{X} + \mathbf{S})^{s/2}).$$

For the concave conjugate formula (5.10) we find that the minimum over \mathbf{Z} occurs when

$$\boldsymbol{\alpha} - \frac{s}{2}(\mathbf{Z} + \mathbf{S})^{s/2-1} = \mathbf{0}.$$

Solving for \mathbf{Z} gives us that

$$\tilde{K}(\boldsymbol{\alpha}) = -\text{tr}(\boldsymbol{\alpha}\mathbf{S}) - \frac{2-s}{s} \left(\frac{2}{s}\right)^{-2/(2-s)} \text{tr}(\boldsymbol{\alpha}^{-2/(2-s)}),$$

which results in (5.12).

Using (5.28), we find that the minimum of (5.27) for the Schatten s -norm occurs when

$$\hat{\mathbf{X}}\boldsymbol{\alpha}_R\hat{\mathbf{X}}^\top + \tilde{\boldsymbol{\Sigma}}_R - \left(\frac{2}{s}\right)^{-s/(2-s)} \boldsymbol{\alpha}_L^{-2/(2-s)} = \mathbf{0}$$

Solving for $\boldsymbol{\alpha}_L$ gives us (5.30). The update equation for $\boldsymbol{\alpha}_R$ is derived in a similar manner.

5.5.11 Details for RSVM-LD

The log-determinant penalty is given by

$$g(\mathbf{X}) = \nu \log |\mathbf{Z} + \mathbf{S}|.$$

For the concave conjugate formula (5.10) we find that the minimum over \mathbf{Z} occurs when

$$\boldsymbol{\alpha} - \nu(\mathbf{Z} + \mathbf{S})^{-1} = \mathbf{0}.$$

Solving for \mathbf{Z} gives that

$$\tilde{K}(\boldsymbol{\alpha}) = -\text{tr}(\boldsymbol{\alpha}\mathbf{S}) + \nu \log |\boldsymbol{\alpha}| + \nu p - \nu \log \nu.$$

By removing the constants we recover (5.14).

Using (5.28), we find that the minimum of (5.27) with respect to $\boldsymbol{\alpha}_L$ for the log-determinant penalty occurs when

$$\hat{\mathbf{X}}\boldsymbol{\alpha}_R\hat{\mathbf{X}}^\top + \tilde{\boldsymbol{\Sigma}}_R + \mathbf{S}_L - \nu\boldsymbol{\alpha}_L^{-1} = \mathbf{0}$$

Solving for $\boldsymbol{\alpha}_L$ gives us (5.31). The derivation of the update equation for $\boldsymbol{\alpha}_R$ is found in a similar way.

5.5.12 Update equations of VB-1 for LRMR

Here we generalize the update equations of the Variational Bayes method from [BLMK12] to LRMR, referred to as VB-1 in section 5.5.1. Similar methods were used in [MS07, SM08, NSB13, LT07]. The VB-1 method factorizes the matrix $\mathbf{X} \in \mathbb{R}^{p \times q}$ as

$$\mathbf{X} = \mathbf{F}\mathbf{B}^\top,$$

were the column vectors of $\mathbf{F} = [\mathbf{f}_1 \mathbf{f}_2 \dots \mathbf{f}_r] \in \mathbb{R}^{p \times r}$, $\mathbf{B} = [\mathbf{b}_1 \mathbf{b}_2 \dots \mathbf{b}_r] \in \mathbb{R}^{q \times r}$ ($r \leq \min(p, q)$ is a user parameter) are given Gaussian priors as

$$p(\mathbf{F}|\boldsymbol{\gamma}) = \prod_{i=1}^r \mathcal{N}(\mathbf{f}_i | \mathbf{0}, \gamma_i^{-1} \mathbf{I}_p),$$

$$p(\mathbf{B}|\boldsymbol{\gamma}) = \prod_{j=1}^r \mathcal{N}(\mathbf{b}_j | \mathbf{0}, \gamma_j^{-1} \mathbf{I}_q),$$

where $\gamma_i > 0$ is the precisions of \mathbf{f}_i and \mathbf{b}_i . We usually set $r = \text{rank}(\mathbf{X})$ when the rank is known, otherwise r can be used to upper bound the rank of $\hat{\mathbf{X}} = \hat{\mathbf{F}}\hat{\mathbf{B}}^\top$. The additive noise in (5.1) is modeled as a zero-mean white Gaussian with the unknown precision $\beta > 0$. The precisions are given Gamma and Jeffreys priors as

$$p(\gamma_i) \propto \gamma_i^{a-1} \exp(-b\gamma_i),$$

$$p(\beta) \propto \beta^{-1}.$$

In the variational Bayes framework, blocks of variables are assumed to have independent posterior distributions allowing to approximate the posterior. Assume that we want to approximate a distribution $p(\mathbf{z})$ using variational Bayes. Let \mathbf{z}_I denote the variables with indices's in a set I , the variational Bayes approximates the distribution $p(\mathbf{z}_I)$ by $q(\mathbf{z}_I)$ as [Bis06]

$$\log q(\mathbf{z}_I) = \mathcal{E}_{\mathbf{z}_{I^c} | \mathbf{z}_I} [\log p(\mathbf{z}_I, \mathbf{z}_{I^c})] + \text{constant}.$$

Different choices of blocks of parameters can be made, we here chose to use independent rows in \mathbf{F} and \mathbf{B} (as in [BLMK12]) since it gives good (empirical) performance. We here use \mathbf{A}_k to denote the k 'th sensing matrix in (5.2), $[\mathbf{A}_k]_{.i}$ to denote the i 'th column vector of \mathbf{A}_k , $[\mathbf{A}_k]_{i.}$ to denote the i 'th row vector of \mathbf{A}_k and $[\mathbf{A}_k]_{ij}$ to denote the (i, j) 'th component of \mathbf{A}_k . We also set $\boldsymbol{\Gamma} = \text{diag}(\gamma_1, \gamma_2, \dots, \gamma_r)$.

Given means $\hat{\boldsymbol{\gamma}}$, $\hat{\beta}$ of $\boldsymbol{\gamma}$ and β , that $\mathbf{b}_k \sim \mathcal{N}(\hat{\mathbf{b}}_k, \boldsymbol{\Sigma}_k^{(B)})$ for all k and $\mathbf{f}_j \sim \mathcal{N}(\hat{\mathbf{f}}_j, \boldsymbol{\Sigma}_j^{(F)})$ for all $j \neq i$, we find that

$$\log q(\mathbf{f}_i) = -\frac{\hat{\beta}}{2} \left(\sum_{k=1}^m y_k^2 - 2\mathbf{f}_i^\top \hat{\mathbf{B}}^\top [\mathbf{A}_k]_{i.} y_k \right)$$

$$\begin{aligned}
& +2 \sum_{j \neq i} \mathbf{f}_i^\top \mathcal{E} [\mathbf{B}^\top [\mathbf{A}_k]_i. [\mathbf{A}_k]_j^\top \mathbf{B}] \hat{\mathbf{f}}_j \\
& + \mathbf{f}_i^\top \mathcal{E} [\mathbf{B}^\top [\mathbf{A}_k]_i. [\mathbf{A}_k]_i^\top \mathbf{B}] \mathbf{f}_i) - \frac{1}{2} \mathbf{f}_i^\top \mathbf{\Gamma} \mathbf{f}_i + \text{constant}.
\end{aligned}$$

where

$$\begin{aligned}
\mathcal{E} [\mathbf{B}^\top [\mathbf{A}_k]_i. [\mathbf{A}_k]_j^\top \mathbf{B}] & = \hat{\mathbf{B}}^\top [\mathbf{A}_k]_i. [\mathbf{A}_k]_j^\top \hat{\mathbf{B}} \\
& + \sum_c [\mathbf{A}_k]_{ic} [\mathbf{A}_k]_{jc} \Sigma_d^{(B)}.
\end{aligned}$$

This gives us that $\mathbf{f}_i \sim \mathcal{N}(\hat{\mathbf{f}}_i, \Sigma_i^{(F)})$ with

$$\begin{aligned}
\hat{\mathbf{f}}_i & = \hat{\beta} \Sigma_i^{(F)} \left(\hat{\mathbf{B}}^\top \sum_k y_k [\mathbf{A}_k]_i. \right. \\
& \quad \left. - \sum_{j \neq i, k} \mathcal{E} [\mathbf{B}^\top [\mathbf{A}_k]_i. \mathbf{A}_k]_j^\top \mathbf{B}] \hat{\mathbf{f}}_j \right), \\
\Sigma_i^{(F)} & = \left(\hat{\beta} \sum_k \mathcal{E} [\mathbf{B}^\top [\mathbf{A}_k]_i. [\mathbf{A}_k]_i^\top \mathbf{B}] + \mathbf{\Gamma} \right)^{-1}.
\end{aligned}$$

Similarly, when the distributions of the other variables are fixed, we get that $\mathbf{b}_i \sim \mathcal{N}(\hat{\mathbf{b}}_i, \Sigma_i^{(B)})$ with

$$\begin{aligned}
\hat{\mathbf{b}}_i & = \hat{\beta} \Sigma_i^{(B)} \left(\hat{\mathbf{F}}^\top \sum_k y_k [\mathbf{A}_k]_i. \right. \\
& \quad \left. - \sum_{j \neq i, k} \mathcal{E} [\mathbf{F}^\top [\mathbf{A}_k]_i. [\mathbf{A}_k]_j^\top \mathbf{F}] \hat{\mathbf{b}}_j \right), \\
\Sigma_i^{(B)} & = \left(\hat{\beta} \sum_k \mathcal{E} [\mathbf{F}^\top [\mathbf{A}_k]_i. [\mathbf{A}_k]_i^\top \mathbf{F}] + \mathbf{\Gamma} \right)^{-1},
\end{aligned}$$

where now

$$\begin{aligned}
\mathcal{E} [\mathbf{F}^\top [\mathbf{A}_k]_i. [\mathbf{A}_k]_j^\top \mathbf{F}] & = \hat{\mathbf{F}}^\top [\mathbf{A}_k]_i. [\mathbf{A}_k]_j^\top \hat{\mathbf{F}} \\
& + \sum_d [\mathbf{A}_k]_{di} [\mathbf{A}_k]_{dj} \Sigma_d^{(F)}.
\end{aligned}$$

We also find that the precisions γ_i are Gamma distributed with posterior parameters

$$\begin{aligned}
\hat{a}_i & = \frac{p + q + 2a}{2}, \\
\hat{b}_i & = \frac{1}{2} \left(\|\hat{\mathbf{f}}_i\|_2^2 + \|\hat{\mathbf{b}}_i\|_2^2 + \text{tr}(\Sigma_i^{(F)}) + \text{tr}(\Sigma_i^{(B)}) + 2b \right).
\end{aligned}$$

This gives us that the posterior mean of γ_i is $\hat{\gamma}_i = \hat{a}_i/\hat{b}_i$. Similarly we find that the posterior distribution of β is $\text{Gamma}(\hat{c}, \hat{d})$ with

$$\begin{aligned} \hat{c} &= m/2, \\ \hat{d} &= \frac{1}{2} \left[\|\mathbf{y} - \text{vec}(\hat{\mathbf{F}}\hat{\mathbf{B}}^\top)\|_2^2 + \sum_{i,k} \left(\hat{\mathbf{f}}_i^\top \mathbf{A}_k \boldsymbol{\Sigma}_i^{(B)} \mathbf{A}_k^\top \hat{\mathbf{f}}_i \right. \right. \\ &\quad \left. \left. + \hat{\mathbf{b}}_i^\top \mathbf{A}_k^\top \boldsymbol{\Sigma}_i^{(F)} \mathbf{A}_k \hat{\mathbf{b}}_i + \text{tr}(\boldsymbol{\Sigma}_i^{(F)} \mathbf{A}_k \boldsymbol{\Sigma}_i^{(B)} \mathbf{A}_k^\top) \right) \right]. \end{aligned}$$

The posterior mean of β is thus $\hat{\beta} = \hat{c}/\hat{d}$.

5.6 Conclusion

We derived a low-rank analogue of the Relevance Vector Machine, called the Relevance Singular Vector Machine (RSVM). The RSVM uses precision matrices and a hierarchical prior to promote low-rank in \mathbf{X} . For the one-sided model, the prior of the precision matrix is related to the marginal prior on \mathbf{X} through the Laplace transform and through the concave conjugate formula. However, for the MAP estimation problem, the concave conjugate formula gives an exact relation. For the two-sided model, an relation is more difficult to establish because of the prior on \mathbf{X} depends non-linearly on the precision matrices. Simulations show that the performance of the RSVM methods is similar to the performance of the nuclear norm estimator for SNR = 20 dB and slightly worse than the performance of the nuclear norm for SNR = 40 dB. The RSVM method suffers from high complexity but shows good performance. To develop the method for larger scale problems is therefore an interesting problem.

Bayesian learning for robust PCA

One of the main tasks in signal processing and machine learning is to describe data in as simple a manner as possible. This is usually done by describing the data with fewer parameters than the number of data points. In the linear model, this is done by writing the data points as linear combinations of a fixed set of atoms. Often, the atoms themselves are not known. It is then required to learn all parameters from data alone. One method for finding a simple linear description is *Principal Component Analysis* (PCA) which attempts to find a lower-dimensional subspace which best describes the data best. The mismatch between the data and the PCA description is often treated as noise.

PCA is a kind of least square estimator and works well when the noise is dense. However, similarly to the standard least squares, PCA is sensitive to sparse outlier noise. Since the outliers are sparse and the PCA estimate is low-rank, the robust PCA problem is a combination of the sparse and low-rank estimation problems. Many methods for sparse and low-rank problems have been adapted to the robust PCA problem. Here we construct a Bayesian method for robust PCA by combining the robust SD-RVM from Chapter 4 and the RSVM from Chapter 5.

6.1 Robust principal component analysis

Robust Principal Component Analysis (RPCA) is the problem of estimating a low-rank matrix \mathbf{X} from measurements

$$\mathbf{Y} = \mathbf{X} + \mathbf{S} + \mathbf{N} \in \mathbb{R}^{p \times q}, \quad (6.1)$$

where \mathbf{Y} is the observed matrix, \mathbf{N} is additive dense noise (typically isotropic Gaussian) and \mathbf{S} is a sparse matrix modeling outliers. The RPCA model (6.1) has been used in e.g. [CLMW11, WLZ13, ZT11, BLMK12, Wip12, DHC11, WYG⁺09, CSPW11, OCS14, SW12, MMG13] to model different phenomena and has applications in e.g. image processing, collaborative filtering, face recognition [CLMW11] and machine learning [WLZ13]. The system model (6.1) can also be used in a matrix completion

setup, where only some components of \mathbf{Y} are observed, by modeling the unobserved entries as outliers.

In the literature, three classes of estimation methods are typically used: greedy, convex optimization based and Bayesian. The greedy method in [ZT11] uses alternating optimization to estimate \mathbf{X} and \mathbf{S} via a least-squares principle. The greedy method is highly effective but requires rank and sparsity level to be known a-priori. The method may therefore be infeasible in many applications. The convex optimization based method in [CLMW11] is called principal component pursuit (PCP) and uses nuclear-norm and ℓ_1 -norm penalty functions to estimate \mathbf{X} and \mathbf{S} . One limitation of PCP is that it requires the power of the dense noise to be known. In absence of a-priori knowledge, Bayesian methods are a suitable choice since they can learn all necessary parameters from data. Formulating a Bayesian method for (6.1) requires low-rank and sparsity promoting priors for \mathbf{X} and \mathbf{S} , respectively. The method of [BLMK12] uses a variational Bayes (VB) approach where the low-rank prior is induced using block sparsity in a matrix factorization model. The empirical Bayes (EB) method of [Wip12] promotes low-rank by modeling the column vectors of \mathbf{X} as correlated Gaussian vectors. Further, in [Wip12] \mathbf{S} is given a sparsity promoting prior by the usual approach where the elements are Gaussian variables with gamma distributed precisions.

In this chapter we develop a new Bayesian method for RPCA. To promote low-rank, we use a model that induces correlations among the column and row vectors of \mathbf{X} . The new method is called robust RSVM (rRSVM) and the parameters are estimated using the expectation-maximization (EM) framework. Through numerical simulations, we investigate the performance for synthetic data as well as real data from the MovieLens 100K dataset [HKBR99]. The performance of rRSVM is found to be better than that of the competing algorithms PCP, EB and VB for both synthetic and real data.

6.2 Robust RSVM

To formulate a Bayesian learning method for the RPCA model (6.1), we need appropriate priors for \mathbf{X} , \mathbf{S} and \mathbf{N} . Here we use the prior for outliers from Chapter 4 to promote sparsity and the two-sided precision with log-determinant prior from Chapter 5 to promote low-rank. We first discuss the priors in Section 6.2.1 and then design the learning algorithm rRSVM in Section 6.2.2 using the EM framework.

6.2.1 Priors for low-rank and sparsity

Low-rank promoting prior

As in Chapter 5 we set the prior on \mathbf{X} to be

$$\text{vec}(\mathbf{X}) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\alpha}_R^{-1} \otimes \boldsymbol{\alpha}_L^{-1}), \quad (6.2)$$

and the prior on the precision matrices to be

$$\begin{aligned} p(\boldsymbol{\alpha}_L) &\propto |\boldsymbol{\alpha}_L|^{(\nu-q)/2} e^{-\frac{\epsilon}{2} \text{tr}(\boldsymbol{\alpha}_L)}, \\ p(\boldsymbol{\alpha}_R) &\propto |\boldsymbol{\alpha}_R|^{(\nu-p)/2} e^{-\frac{\epsilon}{2} \text{tr}(\boldsymbol{\alpha}_R)}, \end{aligned} \quad (6.3)$$

Prior for combined noise

As in Chapter 4 we model the combined noise as

$$S_{ij} + N_{ij} \sim \mathcal{N}(0, \beta_{ij}^{-1}), \quad (6.4)$$

where $\beta_{ij} > 0$ is the total noise precision of the combined noise component $S_{ij} + N_{ij}$. The motivation of using a combined model instead of an independent treatment stems from the fact that \mathbf{S} and \mathbf{N} need not be separated individually for estimation of \mathbf{X} . This approach also reduces the number of model parameters and often improves estimation performance [SCJ15b]. The combined noise is only approximately sparse and is well modeled using a sparsity promoting prior. We here use the Gamma prior

$$p(\beta_{ij}) = \text{Gamma}(\beta_{ij} | a + 1, b) = \frac{b^{a+1}}{\Gamma(a + 1)} \beta_{ij}^a e^{-b\beta_{ij}}, \quad (6.5)$$

for the noise precisions β_{ij} , where $\Gamma(\cdot)$ denotes the Gamma function [Bis06], to promote sparsity in the noise [Wip12, MVC10].

6.2.2 Bayesian learning algorithm for rRSVM

A common method for estimating the model parameters is the maximum a-posteriori method

$$\hat{\mathbf{X}}, \hat{\theta} = \arg \max_{\mathbf{X}, \theta} p(\mathbf{X}, \theta | \mathbf{Y}), \quad (6.6)$$

where $\theta = \{\boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R, \boldsymbol{\beta}\}$. The maximization of (6.6) is often hard to perform in practice and therefore needs to be approximated through e.g. evidence approximation or expectation maximization (EM) [Bis06]. To initialize EM, we make an initial choice of θ . Next, in the expectation step, EM computes the posterior distribution $p(\mathbf{X} | \mathbf{Y}, \theta')$ of \mathbf{X} given the measurements \mathbf{Y} and the latent variables θ' from the previous iteration. In the second step (the maximization step), the latent variables θ are updated by maximizing the EM help function

$$Q(\theta, \theta') = E[\log p(\mathbf{Y}, \mathbf{X} | \theta) | \mathbf{Y}, \theta'] + \log p(\theta),$$

with respect to θ . The expectation and maximization step is repeated until convergence. An advantage of EM over e.g. evidence approximation is that it has established monotone convergence properties [Bis06], i.e. in each iteration the cost in (6.6) does not increase.

For Bayesian RPCA (6.1) with the priors (5.17), (5.14) and (6.5), the posterior distribution $p(\mathbf{X}|\mathbf{Y}, \theta')$ is Gaussian with mean

$$\begin{aligned} \text{vec}(\hat{\mathbf{X}}) &= \mathbf{\Sigma} \mathbf{B}' \text{vec}(\mathbf{Y}), \\ \mathbf{\Sigma} &= ((\boldsymbol{\alpha}'_R \otimes \boldsymbol{\alpha}'_L) + \mathbf{B}')^{-1}, \end{aligned}$$

where $\mathbf{B}' = \text{diag}(\text{vec}(\boldsymbol{\beta}'))$ and $\mathbf{\Sigma}$ is the covariance matrix of $\text{vec}(\mathbf{X})$.

The EM help function for our model becomes

$$\begin{aligned} Q(\theta, \theta') &= -\frac{1}{2} \sum_{i,j} \left(\beta_{ij} (Y_{ij} - \hat{X}_{ij})^2 - \log \beta_{ij} \right) \\ &\quad - \frac{1}{2} \text{tr}(\hat{\mathbf{X}}^\top \boldsymbol{\alpha}_L \hat{\mathbf{X}} \boldsymbol{\alpha}_R) - \frac{1}{2} \text{tr}(\mathbf{\Sigma} [(\boldsymbol{\alpha}_R \otimes \boldsymbol{\alpha}_L) + \mathbf{B}]) \\ &\quad + \frac{q}{2} \log |\boldsymbol{\alpha}_L| + \frac{p}{2} \log |\boldsymbol{\alpha}_R| + \log p(\theta) + \text{constant}, \end{aligned}$$

where $\mathbf{B} = \text{diag}(\text{vec}(\boldsymbol{\beta}))$. The priors of the precisions is denoted by $p(\theta)$, i.e.

$$\log p(\theta) = \log p(\boldsymbol{\alpha}_L) + \log p(\boldsymbol{\alpha}_R) + \sum_{\substack{1 \leq i \leq p \\ 1 \leq j \leq q}} \log p(\beta_{ij}).$$

Maximizing the EM help-function with respect to θ we find the update equations

$$\beta_{ij} = \frac{1 + 2a}{(Y_{ij} - \hat{X}_{ij})^2 + [\boldsymbol{\Sigma}_\beta]_{ij} + 2b}, \quad (6.7)$$

$$\boldsymbol{\alpha}_L = \nu \left(\hat{\mathbf{X}} \boldsymbol{\alpha}_R \hat{\mathbf{X}}^\top + \boldsymbol{\Sigma}_L + \epsilon \mathbf{I}_p \right)^{-1}, \quad (6.8)$$

$$\boldsymbol{\alpha}_R = \nu \left(\hat{\mathbf{X}}^\top \boldsymbol{\alpha}_L \hat{\mathbf{X}} + \boldsymbol{\Sigma}_R + \epsilon \mathbf{I}_q \right)^{-1}, \quad (6.9)$$

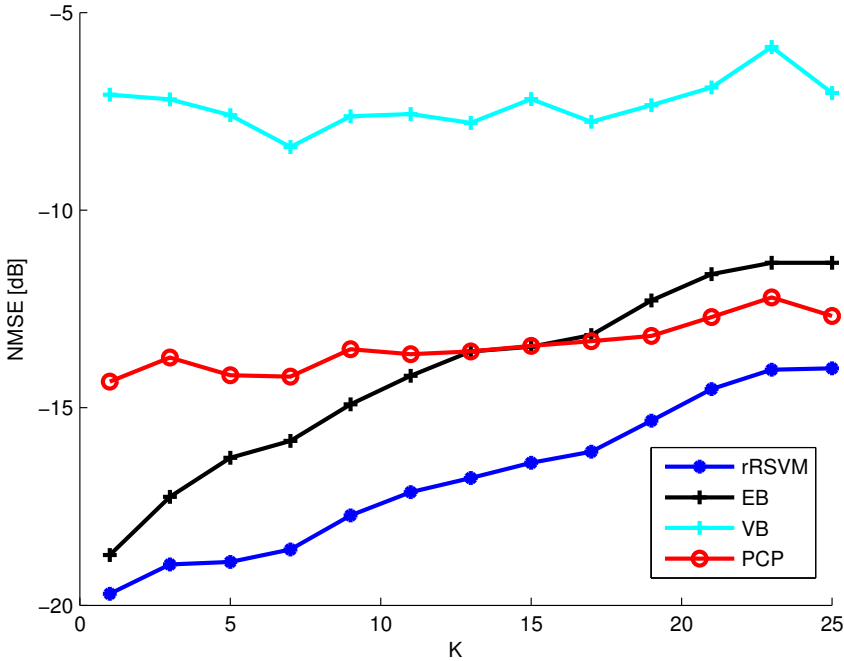
where $[\boldsymbol{\Sigma}_\beta]_{ij}$ denotes the (i, j) component of the matrix $\boldsymbol{\Sigma}_\beta$. The matrices $\boldsymbol{\Sigma}_\beta \in \mathbb{R}^{p \times q}$, $\boldsymbol{\Sigma}_L \in \mathbb{R}^{p \times p}$ and $\boldsymbol{\Sigma}_R \in \mathbb{R}^{q \times q}$ are defined by their elements

$$[\boldsymbol{\Sigma}_\beta]_{ij} = [\boldsymbol{\Sigma}]_{i+p(j-1), i+p(j-1)}, \quad (6.10)$$

$$[\boldsymbol{\Sigma}_L]_{ij} = \text{tr}(\boldsymbol{\Sigma}(\boldsymbol{\alpha}_R \otimes \mathbf{E}_{ij}^L)), \quad (6.11)$$

$$[\boldsymbol{\Sigma}_R]_{ij} = \text{tr}(\boldsymbol{\Sigma}(\mathbf{E}_{ij}^R \otimes \boldsymbol{\alpha}_R)), \quad (6.12)$$

where $\mathbf{E}_{ij}^L \in \mathbb{R}^{p \times p}$ and $\mathbf{E}_{ij}^R \in \mathbb{R}^{q \times q}$ are matrices with a 1 in position (i, j) and zeros otherwise. Typically, the regularization parameters a , b and ϵ are set to small values, e.g. 10^{-4} . In the simulations we initialized the algorithm by setting the matrix precisions to identity matrices and all noise precisions to one. We stopped iterating when the relative difference $\|\hat{\mathbf{X}} - \hat{\mathbf{X}}^{(old)}\|_F^2 / \|\hat{\mathbf{X}}^{(old)}\|_F^2$ was less than 1%.

Figure 6.1: NMSE vs. number of outliers, K .

6.3 Numerical experiments

We used numerical simulations to evaluate the performance of the algorithms PCP [CLMW11], VB [BLMK12], EB [Wip12] and rRSVM. First, we generated synthetic test data for (6.1). We estimated the low-rank matrix \mathbf{X} using the different algorithms and empirically evaluated the Normalized Mean Square Error (NMSE)

$$\text{NMSE} = \frac{\mathcal{E} \left[\|\mathbf{X} - \hat{\mathbf{X}}\|_F^2 \right]}{\mathcal{E} \left[\|\mathbf{X}\|_F^2 \right]}.$$

We considered the case where both the rank, sparsity and SNR is unknown. To make a broader comparison we also compared with the PCP algorithm for which we assumed the SNR to be known a-priori. For PCP we used $\epsilon = \sigma_n \sqrt{pq} + \sqrt{8pq}$ as suggested in [CRT06].

6.3.1 Synthetic data

To generate synthetic measurements (6.1), we generated the low rank matrix by setting $\mathbf{X} = \mathbf{A}\mathbf{B}$, where the elements of $\mathbf{A} \in \mathbb{R}^{p \times r}$ and $\mathbf{B} \in \mathbb{R}^{r \times q}$ were drawn from

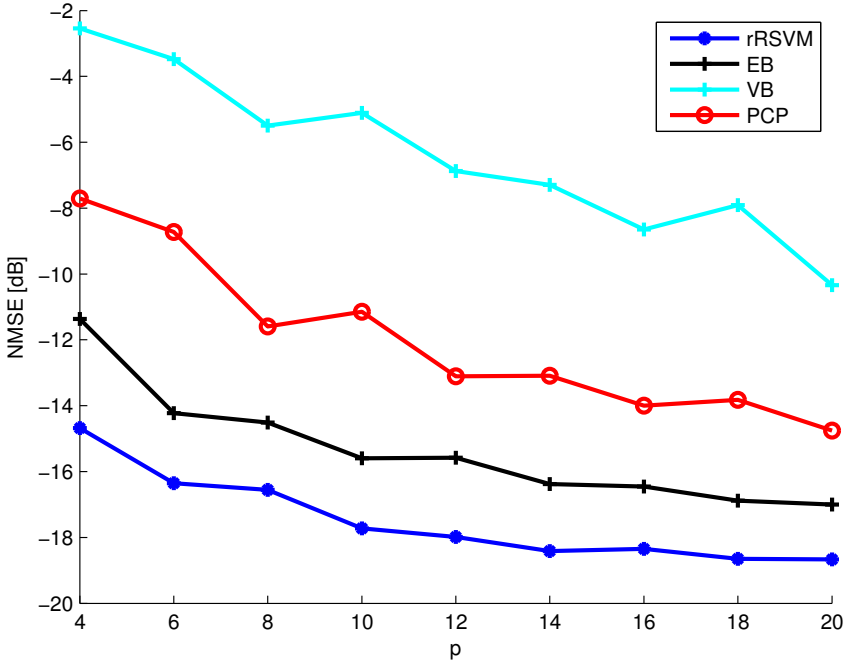


Figure 6.2: NMSE vs. p , number of rows of the matrix.

a $\mathcal{N}(0, 1)$ distribution. The sparse matrix \mathbf{S} was generated by selecting the positions of the K non-zero coefficients uniformly at random and drawing their values from $\mathcal{N}(0, 1)$. The elements of the dense noise matrix was drawn independently from $\mathcal{N}(0, \sigma_n^2)$, where σ_n^2 is chosen to fix the signal-to-noise ratio (SNR)

$$\text{SNR} = \frac{E[\|\mathbf{X} + \mathbf{S}\|_F^2]}{E[\|\mathbf{N}\|_F^2]} = \frac{rpq + K}{pq\sigma_n^2}.$$

We evaluated the NMSE over 100 realizations for each parameter value.

We measured how the number of outliers affect the algorithms by setting $p = 10$, $q = 20$, $r = 3$, $\text{SNR} = 20$ dB and varying K , the number of outliers. We found that EB and rRSVM gave a lower NMSE than PCP for $K \leq 14$. The NMSE of rRSVM was 2.6 dB lower than that of EB for $K \geq 5$. The NMSE of PCP was 6 dB lower than that of VB. The results are shown in Figure 6.1. For recovering the sparse component, VB was most efficient followed by rRSVM.

To evaluate the effect of the matrix size, we varied p , the height of the matrix, for $q = 2p$, $r = \lceil 0.15p \rceil$, $K = \lceil 0.05pq \rceil$ and $\text{SNR} = 20$ dB. We found that the NMSE of rRSVM was 1.7 to 3.3 dB lower than the NMSE of EB, the NMSE of EB was 2.4 to 5.5 dB lower than the NMSE of PCP and the NMSE of PCP was 4.4 to 6.2 dB lower than the NMSE of VB. The results are shown in Figure 6.2.

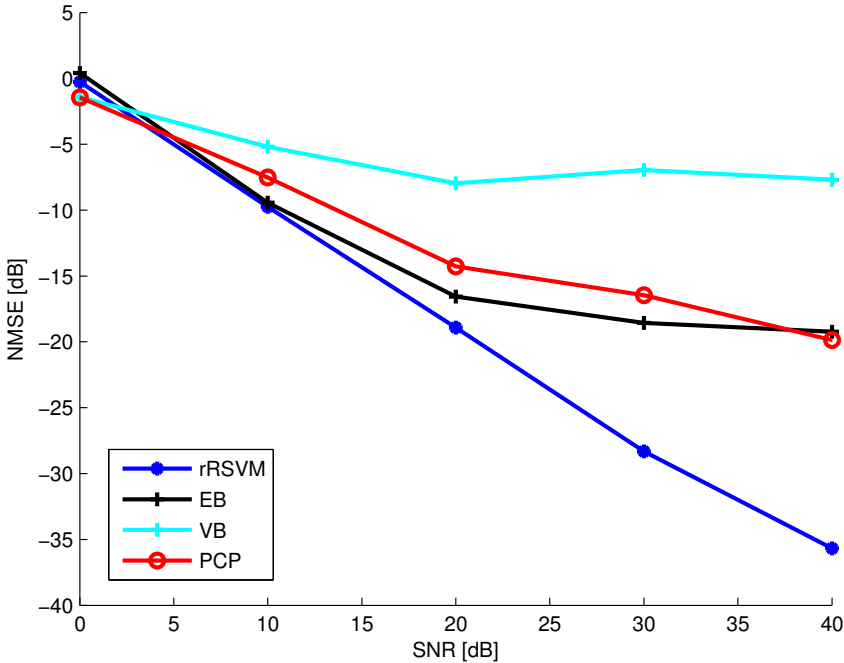


Figure 6.3: NMSE vs. SNR.

Finally, we measured the sensitivity to noise by setting $p = q = 10$, $r = 2$, $K = 5$ and varied the SNR. We found that rRSVM performed best for $\text{SNR} \geq 10$ dB. For $\text{SNR} = 30$ dB, the NMSE of rRSVM was 9.7 dB lower than the NMSE of EB while the NMSE of PCP was 9.5 dB lower than the NMSE of VB. The results are shown in Figure 6.3.

6.3.2 MovieLens dataset

The MovieLens 100K dataset [HKBR99] consists of 100 000 ratings of 1682 movies by 943 users collected in the years 1997 and 1998. Each rating is given by an integer from 1 to 5. In collaborative filtering, the movies are modeled by certain features, e.g. genre, and each user has preferences based on these features. The preferences of a user can thus be modeled as a *linear combination* of preferences for certain (unknown) features. For a low number of relevant features, the matrix of ratings is a low-rank matrix.

Some users may have unique preferences for which the low-rank model is ill-suited. There are also examples of so called *schilling attacks* in which users generate ratings in order to manipulate recommendations [CNZ05]. Ratings which are not modeled well by a low-rank matrix are often few and can thus be modeled by a sparse matrix, the recommendation problem then becomes a robust matrix

Partition	PCP	VB	EB	rRSVM
u1	65.9	78.5	78.5	22.8
u2	79.7	58.3	58.3	22.5
u3	64.9	40.7	40.7	13.8
u4	58.5	37.2	37.2	13.4
u5	37.5	37.4	37.4	12.3

Table 6.1: Error when using the first 75 rows and columns of the MovieLens 100K dataset.

completion problem.

To test the algorithms for robust matrix completion, we used the predefined partitions $u1$, $u2$, $u3$, $u4$ and $u5$ of the MovieLens dataset into training and test data. We used only part of the dataset in order to run the algorithms in reasonable time. In simulations we performed full matrix completion on the training set and calculated the (Frobenius) error over the test set, i.e.

$$\text{Error} = \sqrt{\sum_{(i,j) \in \Omega_{\text{test}}} (\hat{X}_{ij} - X_{ij}^{(\text{test})})^2}.$$

We assumed noise-free measurements for PCP.

We found that rRSVM gave a lower error than the other algorithms. The performance of EB and VB was close to identical (differing first in the 6'th decimal place). PCP gave a lower error than EB and VB only for $u1$. The errors are shown in Table 6.1.

6.4 Conclusion

In this chapter we developed a robust Relevance Singular Vector Machine for robust principal component analysis. The algorithm uses matrix precisions to promote low-rank and models the sparse and dense noise as a single noise source. Through Bayesian modeling, we are able to learn all parameters from data and can thus handle situations in which neither the rank, sparsity of outliers nor the noise power is known. Moreover, the Bayesian method provide error estimates of the estimated variables. The algorithm outperforms principal component pursuit, the empirical Bayes and the variational Bayes in numerical experiments with synthetic and real data.

Robust principal component analysis is a relevant problem that appears in many applications. Hence, it is important to develop more accurate methods. In many real world scenarios, such as the MovieLens dataset, neither the rank, sparsity of outliers nor noise power are known a-priori. Since this is a common scenario, robust Bayesian methods are important for signal processing and machine learning applications.

Bayesian Cramér-Rao bounds for low-rank matrix estimation

In the work of designing more accurate estimation algorithms, it is useful to know how close (or far) the algorithms are from being optimal. Theoretical lower bounds provide limits on the Mean-Square Error (MSE) of estimators. When the performance of an estimator reaches the lower bound, we know that the estimator is optimal and that the bound is the best possible. When the performance of the best estimator and best lower bounds does not meet, it shows that there is possible to design better estimators and/or better bounds.

The Cramér-Rao bound [Cra47, Kay93] provides a lower bound on the MSE for unbiased estimators of deterministic parameters. When the parameters to be estimated are random, the prior distributions need to be taken into account when deriving lower bounds. This is because the prior distribution brings additional information about the parameters. A strong prior gives much information about the variable while a weak prior gives less information about the parameters. For the random variables, a lower bound is given by the Bayesian Cramér-Rao Bound (BCRB) [VT04, VTB07, GL95, BS80]. In this chapter we investigate Bayesian Cramér-Rao bounds for certain Bayesian low-rank reconstruction models.

7.1 Introduction

In low-rank matrix reconstruction (LRMR), we seek to estimate a low-rank matrix $\mathbf{X} \in \mathbb{R}^{p \times q}$ from linear measurements

$$\mathbf{y} = \mathcal{A}(\mathbf{X}) + \mathbf{n} = \mathbf{A} \text{vec}(\mathbf{X}) + \mathbf{n} = \mathbf{A} \mathbf{x} + \mathbf{n}, \quad (7.1)$$

where $\mathbf{y} \in \mathbb{R}^m$ is the observed measurements, $\mathbf{n} \in \mathbb{R}^m$ is additive measurement noise and the sensing operator $\mathcal{A} : \mathbb{R}^{p \times q} \rightarrow \mathbb{R}^m$ and the sensing matrix $\mathbf{A} \in \mathbb{R}^{m \times pq}$ are two equivalent representations of the linear sensing process. For brevity we introduce $\mathbf{x} \triangleq \text{vec}(\mathbf{X})$ where $\text{vec}(\cdot)$ is the standard vectorization operator. The sensing

operator \mathcal{A} and sensing matrix \mathbf{A} are linear operators which can be represented as

$$\mathbf{A}\text{vec}(\mathbf{X}) = \mathcal{A}(\mathbf{X}) = \begin{bmatrix} \text{tr}(\mathbf{A}_1^\top \mathbf{X}) \\ \text{tr}(\mathbf{A}_2^\top \mathbf{X}) \\ \vdots \\ \text{tr}(\mathbf{A}_m^\top \mathbf{X}) \end{bmatrix},$$

where the i 'th row of \mathbf{A} is $\text{vec}(\mathbf{A}_i)^\top$ with $\mathbf{A}_i \in \mathbb{R}^{p \times q}$ and $i = 1, 2, \dots, m$. The sensing matrix \mathbf{A} (and therefore also \mathcal{A}) is assumed to be known. An important special case of LRMR is *matrix completion* where we observe individual elements of \mathbf{X} .

The LRMR problem occurs in several applications, such as system identification [CP10, Faz02, ZSJC12] and recommendation systems [CP10, CR09, CP11, RFGST09, CWZY15, YC11, KBV09, LBA11, Alq13, Suz15, HKBR99, SS10, SM08, LT07, RIK07, FRW11, MS07, LB10, BLMK12, Wip12, SRJC, TN11, CT10]. In many applications, the LRMR problem setup (7.1) is under-determined, i.e. $m < pq$.

There exists several reconstruction algorithms for LRMR, see e.g. [CR09, RFGST09, CWZY15, YC11, KBV09, LBA11, SS10, SM08, LT07, RIK07, FRW11, MS07, LB10, BLMK12, Wip12, SRJC, TN11]. In the Bayesian strategy, the low-rank property of the matrix \mathbf{X} is modeled by a prior distribution. Prominent models of prior distributions are the factorized model of [KBV09, SS10, SM08, LT07, RIK07, BLMK12] and the hierarchical model of our previous work [SRJC]. Our contribution is the theoretical derivation of Bayesian Cramer-Rao bounds (BCRB) for the mentioned prior models. We also evaluate BCRB for a direct low-rank promoting prior distribution that was not used in practical algorithms, but that is interesting for theoretical underpinning. Finally, we perform numerical simulations to compare the performance of practical Bayesian LRMR algorithms against the derived BCRB bounds.

BCRB's for sparse Bayesian models was considered in [PM13]. On the topic of deriving BCRB for LRMR, there exists no work in literature except the work [YC11] that only considered a restricted case of low-rank matrix completion and a factorized model for \mathbf{X} . Our work goes much beyond the work of [YC11]. At this point we mention that there exists bounds for deterministic scenario of LRMR, such as Cramer-Rao bounds for unstructured [TN11] and structured [ZSJC12] low-rank matrices. In the following subsections, we explain notations used in this article and provide preliminaries of BCRB.

7.1.1 Notation

We use $\mathcal{E}_{\mathbf{q}}[\cdot]$ to denote the expectation value with respect to random variables \mathbf{q} . The element-wise (Khatri-Rao) product of two matrices is denoted by \circ and the Kronecker product by \otimes . The ℓ_2 -norm and Frobenius norm are denoted by $\|\cdot\|_F$. We denote the $k \times k$ identity matrix by \mathbf{I}_k and the i 'th unit vector by \mathbf{e}_i , i.e. $\mathbf{e}_i = [0, 0, \dots, 0, 1, 0, \dots, 0]^\top$. We also use the matrices $\mathbf{E}_{ij}^L \in \mathbb{R}^{p \times p}$ and

$\mathbf{E}_{ij}^R \in \mathbb{R}^{q \times q}$ defined as

$$[\mathbf{E}_{ij}]_{kl} = \begin{cases} 1 & \text{if } (k, l) = (i, j) \text{ or } (k, l) = (j, i), \\ 0 & \text{otherwise} \end{cases}.$$

The *commutation matrix* is the matrix representation of the transpose operation, i.e. $\mathbf{K}_{p,q} \text{vec}(\mathbf{Z}) = \text{vec}(\mathbf{Z}^\top)$ for all $\mathbf{Z} \in \mathbb{R}^{p \times q}$ and can be expressed as

$$\mathbf{K}_{p,q} = \sum_{1 \leq i \leq q, 1 \leq j \leq p} \mathbf{e}_i \mathbf{e}_j^\top \otimes \mathbf{e}_j \mathbf{e}_i^\top.$$

We introduce the linear operators \mathcal{T}_1 and \mathcal{T}_2 such that $\mathcal{T}_1(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{C}^\top \otimes \mathbf{D})$ and $\mathcal{T}_2(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{C} \otimes \mathbf{D}^\top)$ for all matrices $\mathbf{C} \in \mathbb{R}^{q \times q}$ and $\mathbf{D} \in \mathbb{R}^{p \times p}$. The operators are defined for any matrix $\mathbf{W} \in \mathbb{R}^{pq \times pq}$ through linearity.

We also introduce the matrix \mathbf{D}_p such that

$$\mathbf{D}_p \text{vec}(\mathbf{Z}) = \text{vec}(\mathbf{Z} + \mathbf{Z}^\top - (\mathbf{Z} \circ \mathbf{I}_p)),$$

where $\mathbf{Z} \in \mathbb{R}^{p \times p}$. We find that $\mathbf{D}_p^\top = \mathbf{D}_p$. The matrix is useful when taking derivative with respect to a symmetric matrix. If \mathbf{Z} is symmetric, then e.g. [KvR06]

$$\begin{aligned} \frac{\partial \log |\mathbf{Z}|}{\partial \text{vec}(\mathbf{Z})} &= \text{vec}(2\mathbf{Z}^{-1} - (\mathbf{Z}^{-1} \circ \mathbf{I}_p)) = \mathbf{D}_p \text{vec}(\mathbf{Z}^{-1}), \\ \frac{\partial \text{vec}(\mathbf{Z}^{-1})}{\partial \text{vec}(\mathbf{Z})} &= -\mathbf{D}_p(\mathbf{Z}^{-1} \otimes \mathbf{Z}^{-1}), \\ \frac{\partial \text{tr}(\mathbf{A}\mathbf{Z})}{\partial \text{vec}(\mathbf{Z})} &= \mathbf{D}_p \text{vec}(\mathbf{A}). \end{aligned}$$

7.1.2 Preliminaries on BCRB

Without loss of generality, we assume that the matrix \mathbf{X} is a function of some parameters $\mathbf{w} \in \mathbb{R}^n$. We write this dependence as $\mathbf{x} = \mathbf{h}(\mathbf{w})$ where $\mathbf{h}(\cdot)$ is a known function. We model random matrices \mathbf{X} by letting \mathbf{w} be a random variable with a prior distribution $p(\mathbf{w}|\boldsymbol{\theta})$ that depends on some hyper-parameters $\boldsymbol{\theta} \in \mathbb{R}^K$. The hyper-parameters $\boldsymbol{\theta}$ can be either deterministic or random. When the prior is on \mathbf{X} directly, we set $\mathbf{x} = \mathbf{w}$. Throughout the chapter, we assume the noise is zero-mean white Gaussian distributed as

$$\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \beta^{-1} \mathbf{I}_m).$$

where $\beta > 0$ is the noise precision. When β is random, we assume that it is Gamma distributed with a prior Gamma distribution

$$p(\beta) = \text{Gamma}(\beta|c, d) = \frac{1}{\Gamma(c)} d^c \beta^{c-1} e^{-d\beta}, \quad (7.2)$$

where $\Gamma(\cdot)$ is the standard Gamma function and $c, d > 0$ are model parameters. The joint distribution of all variables is

$$p(\mathbf{y}, \mathbf{w}, \boldsymbol{\theta}, \beta) = p(\mathbf{y}|\mathbf{w}, \beta)p(\mathbf{w}|\boldsymbol{\theta})p(\boldsymbol{\theta})p(\beta).$$

For brevity of notation, we denote the model variables by

$$\mathbf{z} \triangleq [\mathbf{w}^\top \boldsymbol{\theta}^\top \beta]^\top. \quad (7.3)$$

We are often interested in estimating a variable $\boldsymbol{\eta} \triangleq \mathbf{g}(\mathbf{z})$ where $\mathbf{g}(\cdot)$ is a known function. The BCRB provides a lower bound on the mean-square-error (MSE) of an unbiased estimator $\hat{\boldsymbol{\eta}}$. In the literature, the BCRB is also known as the van-Trees inequality [VT04, VTB07] and the Borovkov-Sakhanenko inequality [VTB07, BS80]. To derive the BCRB, we need to compute the Fisher information matrix \mathbf{F} of \mathbf{z}

$$[\mathbf{F}]_{ij} = \mathcal{E}_{\mathbf{y}} \left[\frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial z_i} \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial z_j} \right],$$

where z_i denotes the i 'th element of \mathbf{z} . We denote the covariance matrix of the estimation error $\boldsymbol{\epsilon} \triangleq \hat{\boldsymbol{\eta}} - \boldsymbol{\eta}$ by

$$\mathbf{C}_\epsilon \triangleq \mathcal{E}_{\mathbf{y}, \mathbf{z}} [\boldsymbol{\epsilon} \boldsymbol{\epsilon}^\top] = \mathcal{E}_{\mathbf{y}, \mathbf{z}} [(\hat{\boldsymbol{\eta}} - \boldsymbol{\eta})(\hat{\boldsymbol{\eta}} - \boldsymbol{\eta})^\top].$$

Proposition 7.1.1 (BCRB). *Assume that $g(\mathbf{z})$ does not depend on \mathbf{A} . For an unbiased estimator $\hat{\boldsymbol{\eta}}$, the covariance \mathbf{C}_ϵ of the estimation error $\boldsymbol{\epsilon}$ is bounded as*

$$\mathbf{C}_\epsilon \succeq \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \right] (\mathcal{E}_{\mathbf{z}}[\mathbf{F}])^{-1} \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \right]^\top. \quad (7.4)$$

It also holds that

$$\mathbf{C}_\epsilon \succeq \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \frac{\partial \mathbf{g}^\top}{\partial \mathbf{z}} \right] \left(\mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \mathbf{F} \frac{\partial \mathbf{g}^\top}{\partial \mathbf{z}} \right] \right)^{-1} \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \frac{\partial \mathbf{g}^\top}{\partial \mathbf{z}} \right]. \quad (7.5)$$

When $\mathbf{g}(\mathbf{z}) = \mathbf{z}$, both bounds reduce to the bound $\mathbf{C}_\epsilon \succeq (\mathcal{E}_{\mathbf{z}}[\mathbf{F}])^{-1}$. Proof of Proposition 7.1.1 is given in Section 7.4.3. While (7.4) is easier to evaluate, the BCRB (7.5) can sometimes be more informative, for example when $\mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \right] = \mathbf{0}$. We obtain a lower bound on the MSE by taking the trace of the inequalities. Table 7.1 shows a nomenclature of various BCRBs and their associated variables.

7.2 Priors for low-rank matrices

In this section, we show three prior models for random low-rank matrices. Later we evaluate the BCRB for these priors.

Table 7.1: BCRB for different cases

	BCRB-I	BCRB-II	BCRB-III	BCRB-IV
Variable $\mathbf{x} = \mathbf{h}(\mathbf{w})$	Random	Random	Random	Random and marginalized
hyper-parameters $\boldsymbol{\theta}, \beta$	Deterministic known	Deterministic unknown	Random	Deterministic unknown
Performance measures	$\mathcal{E}_{\mathbf{y}, \mathbf{w}} \ \mathbf{x} - \hat{\mathbf{x}}\ ^2$ - -	$\mathcal{E}_{\mathbf{y}, \mathbf{w}} \ \mathbf{x} - \hat{\mathbf{x}}\ ^2$ $\mathcal{E}_{\mathbf{y}, \mathbf{w}} \ \boldsymbol{\theta} - \hat{\boldsymbol{\theta}}\ ^2$ $\mathcal{E}_{\mathbf{y}, \mathbf{w}} (\beta - \hat{\beta})^2$	$\mathcal{E}_{\mathbf{y}, \mathbf{w}, \boldsymbol{\theta}, \beta} \ \mathbf{x} - \hat{\mathbf{x}}\ ^2$ $\mathcal{E}_{\mathbf{y}, \mathbf{w}, \boldsymbol{\theta}, \beta} \ \boldsymbol{\theta} - \hat{\boldsymbol{\theta}}\ ^2$ $\mathcal{E}_{\mathbf{y}, \mathbf{w}, \boldsymbol{\theta}, \beta} (\beta - \hat{\beta})^2$	- $\mathcal{E}_{\mathbf{y}} \ \boldsymbol{\theta} - \hat{\boldsymbol{\theta}}\ ^2$ $\mathcal{E}_{\mathbf{y}} (\beta - \hat{\beta})^2$

7.2.1 Sparsity-based model

The sparsity based model induces low-rank by making many of the singular values of a matrix zero. The singular value decomposition of a matrix $\mathbf{X} \in \mathbb{R}^{p \times q}$ is the factorization $\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$ where $\mathbf{U} \in \mathbb{R}^{p \times k}$ and $\mathbf{V} \in \mathbb{R}^{q \times k}$ are matrices such that $\mathbf{U}^\top \mathbf{U} = \mathbf{V}^\top \mathbf{V} = \mathbf{I}_k$ for $k = \min(p, q)$ [HJ12]. The matrix $\boldsymbol{\Sigma} = \text{diag}(\sigma_1, \sigma_2, \dots, \sigma_k)$ is diagonal and $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k \geq 0$ are the singular values of \mathbf{X} . To emphasize the dependence on \mathbf{X} , we sometimes use the notation $\sigma_i \triangleq \sigma_i(\mathbf{X})$ and $\boldsymbol{\sigma} \triangleq \boldsymbol{\sigma}(\mathbf{X}) = [\sigma_1(\mathbf{X}), \dots, \sigma_k(\mathbf{X})]^\top$.

The rank of a matrix is the number of non-zero singular values. A suitable low-rank prior can thus be constructed by using a sparsity-based model by setting

$$p(\mathbf{X})d\mathbf{X} = C e^{-f(\boldsymbol{\sigma}(\mathbf{X}))} d\mathbf{X}, \quad (7.6)$$

where C is a normalization constant, $d\mathbf{X}$ is the integration measure and $f(\cdot)$ is a suitable function. By a change of variables, the distribution of the singular values is found to be $p(\boldsymbol{\sigma})d\boldsymbol{\sigma} = C e^{-f(\boldsymbol{\sigma})} \zeta(\boldsymbol{\sigma})d\boldsymbol{\sigma}$, where $\zeta(\boldsymbol{\sigma})$ is the Jacobian. If realizations of $\boldsymbol{\sigma} \sim p(\boldsymbol{\sigma})$ are sparse, then realizations of $p(\mathbf{X})$ are low-rank. In general, the realizations are only approximately low-rank since the distribution is continuous. For the above prior (7.6), the matrices \mathbf{U} and \mathbf{V} are uniformly distributed on their domains, i.e. their probability distributions are proportional to the Haar measure on their respective Stiefel manifolds [Mui09]. For the sparsity-based model (7.6), we have that

$$\mathbf{w} = \mathbf{x} \text{ and } \boldsymbol{\theta} = \emptyset,$$

where \emptyset denotes the null/empty set. The prior is directly on \mathbf{x} and there are no unknown hyper parameters. For the sparsity-based model, we especially consider the generalized compressible prior (GCP) [PM13, GCD12]

$$f(\boldsymbol{\sigma}) - \log \zeta(\boldsymbol{\sigma}) = \frac{S}{\tau} \sum_{i=1}^k \log(1 + \sigma_i^\tau), \quad (7.7)$$

where $\tau, s > 0$ are model parameters. The choice of the prior (7.7) is due to its sparsity promoting properties [GCD12].

7.2.2 Factorized model

In the factorized model, the low-rank matrix is written as a product of two matrices

$$\mathbf{X} = \mathbf{L}\mathbf{R}^\top, \quad (7.8)$$

where $\mathbf{L} \in \mathbb{R}^{p \times r}$, $\mathbf{R} \in \mathbb{R}^{q \times r}$, and $r < \min(p, q)$ [RFGST09, CWZY15, YC11, KBV09, LBA11, Alq13, SM08, LT07, RIK07, MS07, BLMK12, ZSJC12]. One method for producing rank lower than r is to promote block-sparsity for the columns of \mathbf{L} and \mathbf{R} . To achieve column-wise block-sparsity in \mathbf{L} and \mathbf{R} , a common approach is to use the priors [Alq13, SM08, LT07, MS07, BLMK12]

$$\begin{aligned} p(\mathbf{L}|\boldsymbol{\gamma}) &= \frac{|\boldsymbol{\Gamma}|^{p/2}}{(2\pi)^{pr/2}} \exp\left(-\frac{1}{2}\text{tr}(\mathbf{L}\boldsymbol{\Gamma}\mathbf{L}^\top)\right), \\ p(\mathbf{R}|\boldsymbol{\gamma}) &= \frac{|\boldsymbol{\Gamma}|^{q/2}}{(2\pi)^{qr/2}} \exp\left(-\frac{1}{2}\text{tr}(\mathbf{R}\boldsymbol{\Gamma}\mathbf{R}^\top)\right), \end{aligned} \quad (7.9)$$

where $\boldsymbol{\Gamma} = \text{diag}(\boldsymbol{\gamma})$ and $\boldsymbol{\gamma} = [\gamma_1, \gamma_2, \dots, \gamma_r]^\top$. In this model, γ_i is the precision of the i 'th column of \mathbf{L} and \mathbf{R} . To promote sparsity, the precisions are given Gamma distributions

$$p(\gamma_i) = \text{Gamma}(\gamma_i|a, b) \quad (7.10)$$

as priors. For the factorized model (7.8) we thus have that

$$\mathbf{w} = \begin{bmatrix} \text{vec}(\mathbf{L}) \\ \text{vec}(\mathbf{R}) \end{bmatrix} \text{ and } \boldsymbol{\theta} = \boldsymbol{\gamma}. \quad (7.11)$$

We use the above parameters since finding $p(\mathbf{X}|\boldsymbol{\gamma})$ is non-trivial for the factorized model.

At this point, we mention that the individual factor matrices \mathbf{L} and \mathbf{R} are not identifiable since $(\mathbf{L}\mathbf{Q}^\top)(\mathbf{R}\mathbf{Q}^{-1})^\top = \mathbf{L}\mathbf{R}$ for any invertible matrix $\mathbf{Q} \in \mathbb{R}^{r \times r}$. Also, the individual precisions γ_i are not identifiable since two precisions can be interchanged without altering the model. We can therefore not derive bounds for the individual factor matrices or the individual precisions, but instead the BCRB for $\mathbf{L}\mathbf{R}^\top$ and a symmetric function $s(\boldsymbol{\gamma})$ of $\boldsymbol{\gamma}$. The variable of interest is therefore

$$\boldsymbol{\eta} = \mathbf{g}(\mathbf{z}) = \mathbf{g}\left(\begin{bmatrix} \mathbf{w} \\ \boldsymbol{\theta} \\ \beta \end{bmatrix}\right) = \begin{bmatrix} \text{vec}(\mathbf{L}\mathbf{R}^\top) \\ s(\boldsymbol{\gamma}) \\ \beta \end{bmatrix}. \quad (7.12)$$

7.2.3 RSVM model

In the RSVM model [SRJC], the low-rank matrix \mathbf{X} is modeled as

$$\mathbf{X} = \boldsymbol{\alpha}_L^{-1/2} \mathbf{U} \boldsymbol{\alpha}_R^{-1/2} \quad (7.13)$$

where $\boldsymbol{\alpha}_L \in \mathbb{R}^{p \times p}$ and $\boldsymbol{\alpha}_R \in \mathbb{R}^{q \times q}$ are positive definite precision matrices and the elements of \mathbf{U} are iid Gaussian $\mathcal{N}(0, 1)$ variables. In [SRJC] we showed that the use of Wishart distributions as priors for precision matrices promote low-rank in \mathbf{X} . Through the relation $\text{vec}(\mathbf{X}) = (\boldsymbol{\alpha}_R^{-1/2} \otimes \boldsymbol{\alpha}_L^{-1/2}) \text{vec}(\mathbf{U})$, we find that

$$p(\mathbf{X} | \boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R) = \frac{|\boldsymbol{\alpha}_R \otimes \boldsymbol{\alpha}_L|^{1/2}}{(2\pi)^{pq/2}} \exp\left(-\frac{1}{2} \text{tr}(\mathbf{X}^\top \boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R)\right) \quad (7.14)$$

For the precision matrices we use the Wishart priors [SRJC]

$$p(\boldsymbol{\alpha}_L) = C_{\nu_L, p} |\boldsymbol{\alpha}_L|^{\frac{\nu_L}{2}} e^{-\frac{1}{2} \epsilon \text{tr}(\boldsymbol{\alpha}_L)}, \quad (7.15)$$

$$p(\boldsymbol{\alpha}_R) = C_{\nu_R, q} |\boldsymbol{\alpha}_R|^{\frac{\nu_R}{2}} e^{-\frac{1}{2} \epsilon \text{tr}(\boldsymbol{\alpha}_R)}, \quad (7.16)$$

where $\nu_L, \nu_R > 0$, $C_{\nu_L, p}$, $C_{\nu_R, q}$ are normalization constants and [Bis06]

$$\Gamma_p(x) = \pi^{\frac{p(p-1)}{4}} \prod_{k=1}^p \Gamma\left(x + \frac{1-k}{2}\right).$$

Thus, in the RSVM model

$$\mathbf{w} = \mathbf{x} \text{ and } \boldsymbol{\theta} = \begin{bmatrix} \text{vec}(\boldsymbol{\alpha}_L) \\ \text{vec}(\boldsymbol{\alpha}_R) \end{bmatrix}.$$

We notice that the individual precision matrices are not identifiable since the distribution $p(\mathbf{X} | \boldsymbol{\alpha}_L, \boldsymbol{\alpha}_R)$ is invariant under rescalings $\boldsymbol{\alpha}_L \rightarrow t\boldsymbol{\alpha}_L$ and $\boldsymbol{\alpha}_R \rightarrow t^{-1}\boldsymbol{\alpha}_R$ for all $t > 0$. This is because the distribution only depends on $(\boldsymbol{\alpha}_R \otimes \boldsymbol{\alpha}_L)$. The variables of interest are therefore

$$\boldsymbol{\eta} = \mathbf{g}(\mathbf{z}) = \mathbf{g}\left(\begin{bmatrix} \mathbf{w} \\ \boldsymbol{\theta} \\ \beta \end{bmatrix}\right) = \begin{bmatrix} \mathbf{x} \\ \text{vec}(\boldsymbol{\alpha}_R \otimes \boldsymbol{\alpha}_L) \\ \beta \end{bmatrix}. \quad (7.17)$$

7.3 Bayesian Cramér-Rao bounds for low-rank matrix reconstruction

In this section, we derive the BCRBs for the models described in section 7.2.

7.3.1 Bounds for the sparsity-based model

In the sparsity-based model, the distribution of \mathbf{X} depends only on the singular values of \mathbf{X} and the function $f(\cdot)$ is assumed to be known. We compute BCRB-I, BCRB-II and BCRB-III for this model.

Proposition 7.3.1. *Let $\mathbf{X} = \sum_{i=1}^k \sigma_i \mathbf{u}_i \mathbf{v}_i^\top$ be the SVD of \mathbf{X} and assume that $f(\cdot)$ is differentiable. The Fisher information matrix of the sparsity based model is given by*

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{\mathbf{x}} & \mathbf{F}_{\mathbf{x}\beta} \\ \mathbf{F}_{\mathbf{x}\beta}^\top & F_\beta \end{bmatrix},$$

where

$$\begin{aligned} \mathbf{F}_{\mathbf{x}} &= \sum_{1 \leq i, j \leq k} \frac{\partial f}{\partial \sigma_i} \frac{\partial f}{\partial \sigma_j} (\mathbf{v}_j \mathbf{v}_i^\top \otimes \mathbf{u}_j \mathbf{u}_i^\top) + \beta \mathbf{A}^\top \mathbf{A}, \\ \mathbf{F}_{\mathbf{x}\beta} &= \left(d - \frac{c-1}{\beta} \right) \sum_{i=1}^r \frac{\partial f}{\partial \sigma_i} (\mathbf{v}_i \otimes \mathbf{u}_i), \\ F_\beta &= \frac{m}{2\beta^2} + \left(d - \frac{c-1}{\beta} \right)^2. \end{aligned}$$

Setting $c = 1$ and $d = 0$ corresponds to a deterministic β .

The proof of Proposition 7.3.1 is given in Section 7.4.4.

For the sparsity based model we have that

$$\frac{\partial \mathbf{g}}{\partial \mathbf{z}} = \mathbf{I}_{pq+1}.$$

The bounds (7.4) and (7.5) thus coincide for the sparsity-based model.

BCRB-I and BCRB-II

The only hyper parameter in the sparsity based model is the noise precision β . BCRB-I and BCRB-II are the bounds for the problems where β is known and unknown, respectively.

Proposition 7.3.2. *For the sparsity based model (7.6), the BCRB-II is given by*

$$\mathbf{C}_\epsilon \succeq \begin{bmatrix} (\mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\mathbf{x}}])^{-1} & \mathbf{0} \\ \mathbf{0} & F_\beta^{-1} \end{bmatrix}$$

where we used that $\mathbf{F}_{\mathbf{x}\beta} = \mathbf{0}$. The non-zero factors are given by

$$\begin{aligned} F_\beta &= \frac{m}{2\beta^2}, \\ \mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\mathbf{x}}] &= \frac{1}{pq} \frac{N[f]}{D[f]} \mathbf{I}_{pq} + \beta \mathbf{A}^\top \mathbf{A}. \end{aligned}$$

with

$$N[f] = \int_{\Omega} h(\boldsymbol{\sigma})g(\boldsymbol{\sigma})e^{-f(\boldsymbol{\sigma})}d\boldsymbol{\sigma}, \quad (7.18)$$

$$D[f] = \int_{\Omega} g(\boldsymbol{\sigma})e^{-f(\boldsymbol{\sigma})}d\boldsymbol{\sigma}, \quad (7.19)$$

for $\Omega = \{(\sigma_1, \sigma_2, \dots, \sigma_k) : \sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k \geq 0\}$ and

$$g(\boldsymbol{\sigma}) = \prod_{1 \leq i < j \leq k} (\sigma_i^2 - \sigma_j^2) \prod_{1 \leq i \leq k} \sigma_i^{|p-q|},$$

$$h(\boldsymbol{\sigma}) = \sum_{i=1}^k \left(\frac{\partial f}{\partial \sigma_i} \right)^2.$$

The bound exists provided that the function $f(\cdot)$ is such that the integrals converge.

We also find that BCRB-I of \mathbf{x} with known β is given by

$$\mathcal{E}_{\mathbf{y}, \mathbf{x}} \left[(\hat{\mathbf{x}} - \mathbf{x})(\hat{\mathbf{x}} - \mathbf{x})^\top \right] \succeq (\mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\mathbf{x}}])^{-1}.$$

The proof of Proposition 7.3.2 is given in Section 7.4.5. We find that the integrals are non-trivial to evaluate analytically. We therefore use Monte-Carlo integration [BGJM11] to numerically compute (7.24) and (7.25).

BCRB-III

To compute BCRB-III we also need to compute expectation values with respect to β , which is now a random variable. We state the bound as a proposition.

Proposition 7.3.3. *The bound BCRB-III for the sparsity based model is given by*

$$\mathbf{C}_{\epsilon} \succeq \begin{bmatrix} (\mathcal{E}_{\mathbf{z}}[\mathbf{F}_{\mathbf{x}}])^{-1} & \mathbf{0} \\ \mathbf{0} & (\mathcal{E}_{\beta}[F_{\beta}])^{-1} \end{bmatrix}.$$

where we used that $\mathcal{E}_{\mathbf{z}}[\mathbf{F}_{\mathbf{x}\beta}] = \mathbf{0}$ and

$$\mathcal{E}_{\mathbf{z}}[\mathbf{F}_{\mathbf{x}}] = \frac{1}{pq} \frac{N[f]}{D[f]} \mathbf{I}_{pq} + \frac{c}{d} \mathbf{A}^\top \mathbf{A},$$

$$\mathcal{E}_{\mathbf{z}}[F_{\beta}] = \frac{(m + 2(c-1))d^2}{2(c-1)(c-2)}.$$

Proof. The proposition follows from parts of Proposition 7.3.2 and the fact that

$$\mathcal{E}_{\beta}[\beta] = \frac{c}{d},$$

$$\mathcal{E}_{\beta} \left[\frac{m}{2\beta^2} + \left(d - \frac{c-1}{\beta} \right)^2 \right] = \frac{md^2}{2(c-1)(c-2)} + d^2$$

$$- \frac{2d(c-1)d}{c-1} + \frac{(c-1)^2 d^2}{(c-1)(c-2)} = \frac{(m + 2(c-1))d^2}{2(c-1)(c-2)}$$

□

7.3.2 Bounds for the factorized model

Here we evaluate BCRB-I, BCRB-II and BCRB-III for the factorized model described in Section 7.2.2. The following proposition gives the Fisher information matrix.

Proposition 7.3.4 (Fisher information matrix). *Let \mathbf{l}_i and \mathbf{r}_i denote the i 'th column vector of \mathbf{L} and \mathbf{R} respectively. For the factorized model, the Fisher information matrix is given by*

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{LL} & \mathbf{F}_{LR} & \mathbf{F}_{L\gamma} & \mathbf{F}_{L\beta} \\ \mathbf{F}_{LR}^\top & \mathbf{F}_{RR} & \mathbf{F}_{R\gamma} & \mathbf{F}_{R\beta} \\ \mathbf{F}_{L\gamma}^\top & \mathbf{F}_{R\gamma}^\top & \mathbf{F}_\gamma & \mathbf{F}_{\gamma\beta} \\ \mathbf{F}_{L\beta}^\top & \mathbf{F}_{R\beta}^\top & \mathbf{F}_{\gamma\beta}^\top & F_\beta \end{bmatrix}, \quad (7.20)$$

where

$$\begin{aligned} \mathbf{F}_{LL} &= \beta(\mathbf{R} \otimes \mathbf{I}_p)^\top \mathbf{A}^\top \mathbf{A} (\mathbf{R} \otimes \mathbf{I}_p) + \text{vec}(\mathbf{L}\Gamma) \text{vec}(\mathbf{L}\Gamma)^\top, \\ \mathbf{F}_{LR} &= \beta(\mathbf{R} \otimes \mathbf{I}_p)^\top \mathbf{A}^\top \mathbf{A} (\mathbf{I}_q \otimes \mathbf{L}) \mathbf{K}_{r,q} + \text{vec}(\mathbf{L}\Gamma) \text{vec}(\mathbf{R}\Gamma)^\top, \\ \mathbf{F}_{RR} &= \beta \mathbf{K}_{q,r} (\mathbf{I}_q \otimes \mathbf{L})^\top \mathbf{A}^\top \mathbf{A} (\mathbf{I}_q \otimes \mathbf{L}) \mathbf{K}_{r,q} + \text{vec}(\mathbf{R}\Gamma) \text{vec}(\mathbf{R}\Gamma)^\top, \\ \mathbf{F}_\gamma &= \mathbf{h}\mathbf{h}^\top, \quad F_\beta = \frac{m}{2\beta^2} + \left(d - \frac{c-1}{\beta}\right)^2, \end{aligned}$$

and the components of $\mathbf{h} = [h_1, h_2, \dots, h_r]^\top$ are given by

$$h_i = \frac{p+q+2(a-1)}{2\gamma_i} - \frac{\|\mathbf{l}_i\|_2^2 + \|\mathbf{r}_i\|_2^2}{2} - b.$$

The components $\mathbf{F}_{L\gamma}$, $\mathbf{F}_{R\gamma}$, $\mathbf{F}_{L\beta}$, $\mathbf{F}_{R\beta}$ and $\mathbf{F}_{\gamma\beta}$ are given in Section 7.4.6 and are zero when γ and β are deterministic ($a = c = 1$ and $b = d = 0$) and zero-mean when the hyperparameters are random. These components do therefore not affect the BCRB bounds.

The proof of Proposition 7.3.4 is shown in Section 7.4.6. Following Proposition 7.1.1 we evaluate $\frac{\partial \mathbf{g}}{\partial \mathbf{z}}$ where $\mathbf{g}(\mathbf{z})$ is shown in (7.12). We get that

$$\begin{aligned} \frac{\partial \mathbf{g}}{\partial \mathbf{z}} &= \begin{bmatrix} \frac{\partial \text{vec}(\mathbf{L}\mathbf{R}^\top)}{\partial \mathbf{w}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \frac{\partial s}{\partial \gamma} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \frac{\partial \beta}{\partial \beta} \end{bmatrix} \\ &\stackrel{(a)}{=} \begin{bmatrix} [(\mathbf{R} \otimes \mathbf{I}_p), (\mathbf{I}_q \otimes \mathbf{L}) \mathbf{K}_{r,q}] & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & (\nabla_\gamma s)^\top & 0 \\ \mathbf{0} & 0 & 1 \end{bmatrix}, \end{aligned} \quad (7.21)$$

where

$$\frac{\partial s}{\partial \boldsymbol{\gamma}} = (\nabla_{\boldsymbol{\gamma}} s)^\top = \left[\frac{\partial s}{\partial \gamma_1}, \frac{\partial s}{\partial \gamma_2}, \dots, \frac{\partial s}{\partial \gamma_r} \right],$$

and we used (7.3) and (7.11). In equality (a) we used that

$$\text{vec}(\mathbf{LR}^\top) = (\mathbf{R} \otimes \mathbf{I}_p) \text{vec}(\mathbf{L}) = (\mathbf{I}_q \otimes \mathbf{L}) \mathbf{K}_{r,q} \text{vec}(\mathbf{R}).$$

Note that

$$\boldsymbol{\mathcal{E}}_{\mathbf{w}} \left[\frac{\partial \text{vec}(\mathbf{LR}^\top)}{\partial \mathbf{w}} \right] = \boldsymbol{\mathcal{E}}_{\mathbf{w}} [[(\mathbf{R} \otimes \mathbf{I}_p), (\mathbf{I}_q \otimes \mathbf{L}) \mathbf{K}_{r,q}]] = \mathbf{0},$$

as \mathbf{L} and \mathbf{R} are zero-mean. Hence the BCRB (7.4) is non-informative for the factorized model. We thus compute the BCRB (7.5) for the factorized model. We get that

$$\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \frac{\partial \mathbf{g}}{\partial \mathbf{z}}^\top = \begin{bmatrix} (\mathbf{RR}^\top \otimes \mathbf{I}_p) + (\mathbf{I}_q \otimes \mathbf{LL}^\top) & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \|\nabla_{\boldsymbol{\gamma}} s\|_2^2 & 0 \\ \mathbf{0} & 0 & 1 \end{bmatrix},$$

where we used (7.21) and the standard relation $\mathbf{K}_{r,q} \mathbf{K}_{r,q}^\top = \mathbf{I}_{rq}$. Taking the expectation value gives that

$$\boldsymbol{\mathcal{E}}_{\mathbf{w}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \frac{\partial \mathbf{g}}{\partial \mathbf{z}}^\top \right] = \begin{bmatrix} 2 \left(\sum_{i=1}^r \gamma_i^{-1} \right) \mathbf{I}_{pq} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \|\nabla_{\boldsymbol{\gamma}} s\|_2^2 & 0 \\ \mathbf{0} & 0 & 1 \end{bmatrix}, \quad (7.22)$$

where we used that $\boldsymbol{\mathcal{E}}_{\mathbf{w}} [(\mathbf{RR}^\top \otimes \mathbf{I}_p) + (\mathbf{I}_q \otimes \mathbf{LL}^\top)] = \sum_{i=1}^r 2\gamma_i^{-1} \mathbf{I}_{pq}$. Next, using (7.20) and (7.21) we find that

$$\mathbf{G} = \frac{\partial \mathbf{g}}{\partial \mathbf{z}} \mathbf{F} \frac{\partial \mathbf{g}}{\partial \mathbf{z}}^\top = \begin{bmatrix} \mathbf{G}_{\mathbf{w}} & \mathbf{G}_{\mathbf{w}\gamma} & \mathbf{G}_{\mathbf{w}\beta} \\ \mathbf{G}_{\mathbf{w}\gamma}^\top & G_\gamma & G_{\gamma\beta} \\ \mathbf{G}_{\mathbf{w}\beta}^\top & G_{\gamma\beta} & G_\beta \end{bmatrix},$$

where

$$\begin{aligned} \mathbf{G}_{\mathbf{w}} &= \beta(\mathbf{RR}^\top \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{RR}^\top \otimes \mathbf{I}_p) \\ &+ \beta(\mathbf{I}_q \otimes \mathbf{LL}^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{I}_q \otimes \mathbf{LL}^\top) \\ &+ \beta(\mathbf{RR}^\top \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{I}_q \otimes \mathbf{LL}^\top) \\ &+ \beta(\mathbf{I}_q \otimes \mathbf{LL}^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{RR}^\top \otimes \mathbf{I}_p), \\ &+ 4 \text{vec}(\mathbf{LFR}^\top) \text{vec}(\mathbf{LFR}^\top)^\top, \\ G_\gamma &= (\nabla_{\boldsymbol{\gamma}} s)^\top \mathbf{F}_\gamma (\nabla_{\boldsymbol{\gamma}} s) = ((\nabla_{\boldsymbol{\gamma}} s)^\top \mathbf{h})^2, \\ G_\beta &= F_\beta. \end{aligned}$$

The components $\mathbf{G}_{\mathbf{w}\gamma}$, $\mathbf{G}_{\mathbf{w}\beta}$ and $G_{\gamma\beta}$ are given in Appendix 7.4.6 and are zero when the parameters are deterministic and zero-mean when the parameters are random.

BCRB-I and II

We compute the BCRB (7.5) by taking expectation values with respect to \mathbf{w} . We state the resulting bound as the following proposition.

Proposition 7.3.5. *The BCRB-II of the factorized model is given by*

$$\mathbf{C}_\epsilon \succeq \begin{bmatrix} (2 \sum_{i=1}^r \gamma_i^{-1})^2 (\mathcal{E}_{\mathbf{w}}[\mathbf{G}_{\mathbf{w}}])^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \frac{\|\nabla_{\gamma} s\|_2^4}{\mathcal{E}_{\mathbf{w}}[G_{\gamma}]} & 0 \\ \mathbf{0} & 0 & G_{\beta}^{-1} \end{bmatrix}$$

where $G_{\beta} = F_{\beta}$ is given in Proposition 7.3.4 and

$$\begin{aligned} \mathcal{E}_{\mathbf{w}}[\mathbf{G}_{\mathbf{w}}] &= 2\beta \left(\sum_{n=1}^r \gamma_n^{-1} \right)^2 \mathbf{A}^{\top} \mathbf{A} + 4r \mathbf{I}_{pq} \\ &+ \beta \left(\sum_{n=1}^r \gamma_n^{-2} \right) (\mathcal{T}_1(\mathbf{A}^{\top} \mathbf{A}) + \mathcal{T}_2(\mathbf{A}^{\top} \mathbf{A})) \\ &+ \beta \left(\sum_{n=1}^r \gamma_n^{-2} \right) \left(\mathbf{I}_q \otimes \left(\sum_{m=1}^q (\mathbf{e}_m^{\top} \otimes \mathbf{I}_p) \mathbf{A}^{\top} \mathbf{A} (\mathbf{e}_m \otimes \mathbf{I}_p) \right) \right) \\ &+ \beta \left(\sum_{n=1}^r \gamma_n^{-2} \right) \left(\left(\sum_{m=1}^p (\mathbf{I}_q \otimes \mathbf{e}_m^{\top}) \mathbf{A}^{\top} \mathbf{A} (\mathbf{I}_q \otimes \mathbf{e}_m) \right) \otimes \mathbf{I}_p \right), \\ \mathcal{E}_{\mathbf{w}}[G_{\gamma}] &= \frac{p+q}{2} (\nabla_{\gamma} s)^{\top} \mathbf{\Gamma}^{-2} (\nabla_{\gamma} s). \end{aligned}$$

The linear operators \mathcal{T}_1 and \mathcal{T}_2 are defined in Section 7.1.1. The BCRB-I is given by

$$\mathcal{E}_{\mathbf{y}, \mathbf{w}} \left[(\hat{\mathbf{x}} - \mathbf{x}) (\hat{\mathbf{x}} - \mathbf{x})^{\top} \right] \succeq \left(2 \sum_{i=1}^r \gamma_i^{-1} \right)^2 (\mathcal{E}_{\mathbf{w}}[\mathbf{G}_{\mathbf{w}}])^{-1}.$$

The proof is given in Section 7.4.7.

BCRB-III

Computing BCRB-III requires calculating expectation values with respect to \mathbf{y} and $\mathbf{z} = [\mathbf{w}^{\top}, \gamma^{\top}, \beta]^{\top}$. We state the bound as a proposition.

Proposition 7.3.6. *Assume that $a > 2$ in (7.10) and $c > 1$ in (7.2). The BCRB-III of the factorized model is given by*

$$\mathbf{C}_\epsilon \succeq \begin{bmatrix} \left(\frac{2rb}{a-1}\right)^2 (\mathcal{E}_z[\mathbf{G}_w])^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \frac{(\mathcal{E}_z[\|\nabla_\gamma s\|^2])^2}{\mathcal{E}_z[G_\gamma]} & 0 \\ \mathbf{0} & 0 & (\mathcal{E}_z[G_\beta])^{-1} \end{bmatrix}$$

where

$$\begin{aligned} \mathcal{E}_z[\mathbf{G}_w] &= 2 \frac{c}{d} \frac{rb^2(1+ra-2r)}{(a-1)^2(a-2)} \mathbf{A}^\top \mathbf{A} + 4r \mathbf{I}_{pq} \\ &+ \frac{c}{d} \frac{rb^2}{(a-1)(a-2)} (\mathcal{T}_1(\mathbf{A}^\top \mathbf{A}) + \mathcal{T}_2(\mathbf{A}^\top \mathbf{A})) \\ &+ \frac{c}{d} \frac{rb^2}{(a-1)(a-2)} \left(\mathbf{I}_q \otimes \left(\sum_{m=1}^q (\mathbf{e}_m^\top \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{e}_m \otimes \mathbf{I}_p) \right) \right) \\ &+ \frac{c}{d} \frac{rb^2}{(a-1)(a-2)} \left(\left(\sum_{m=1}^p (\mathbf{I}_q \otimes \mathbf{e}_m^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{I}_q \otimes \mathbf{e}_m) \right) \otimes \mathbf{I}_p \right), \\ \mathcal{E}_z[G_\beta] &= \frac{(m+2(c-1))d^2}{2(c-1)(c-2)}, \\ \mathcal{E}_z[G_\gamma] &= \mathcal{E}_\gamma \left[((\nabla_\gamma s)^\top (a\gamma^{-1} - b\mathbf{1}_r))^2 \right] \\ &+ \frac{p+q+2(a-1)}{2} \mathcal{E}_\gamma \left[(\nabla_\gamma s)^\top \mathbf{F}_\gamma (\nabla_\gamma s) \right], \\ \mathcal{E}_z[G_\beta] &= \mathcal{E}_z \left[\frac{m+2(c-1)}{2\beta^2} \right] = \frac{(m+2(c-1))d^2}{2(c-1)(c-2)}, \end{aligned}$$

where $\gamma^{-1} = [\gamma_1^{-1}, \gamma_2^{-1}, \dots, \gamma_r^{-1}]^\top$ and $\mathbf{1}_r = [1, 1, \dots, 1]^\top \in \mathbb{R}^r$.

The proof is given in Section 7.4.8.

7.3.3 Bound for the RSVM model

For the RSVM model, we compute BCRB-I, BCRB-II, BCRB-III and BCRB-IV. The Fisher information matrix of the RSVM model is given by the following proposition.

Proposition 7.3.7. *The Fisher information matrix for BCRB-II and BCRB-III of the RSVM model is given by*

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{\mathbf{x}\mathbf{x}} & \mathbf{F}_{\mathbf{x}\alpha_L} & \mathbf{F}_{\mathbf{x}\alpha_R} & \mathbf{F}_{\mathbf{x}\beta} \\ \mathbf{F}_{\mathbf{x}\alpha_L}^\top & \mathbf{F}_{\alpha_L\alpha_L} & \mathbf{F}_{\alpha_L\alpha_R} & \mathbf{F}_{\alpha_L\beta} \\ \mathbf{F}_{\mathbf{x}\alpha_R}^\top & \mathbf{F}_{\alpha_L\alpha_R}^\top & \mathbf{F}_{\alpha_R\alpha_R} & \mathbf{F}_{\alpha_R\beta} \\ \mathbf{F}_{\mathbf{x}\beta}^\top & \mathbf{F}_{\alpha_L\beta}^\top & \mathbf{F}_{\alpha_R\beta} & F_{\beta\beta} \end{bmatrix},$$

where

$$\begin{aligned}\mathbf{F}_{\mathbf{xx}} &= \beta \mathbf{A}^\top \mathbf{A} + \text{vec}(\boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R) \text{vec}(\boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R)^\top, \\ \mathbf{F}_{\boldsymbol{\alpha}_L \boldsymbol{\alpha}_L} &= \mathbf{v}_L \mathbf{v}_L^\top, \quad \mathbf{F}_{\boldsymbol{\alpha}_L \boldsymbol{\alpha}_R} = \mathbf{v}_L \mathbf{v}_R^\top, \quad \mathbf{F}_{\boldsymbol{\alpha}_R \boldsymbol{\alpha}_R} = \mathbf{v}_R \mathbf{v}_R^\top, \\ F_{\beta\beta} &= \frac{m}{2\beta^2} + \left(d - \frac{c-1}{\beta} \right)^2,\end{aligned}$$

and

$$\begin{aligned}\mathbf{v}_L &= \frac{1}{2} \mathbf{D}_p \text{vec}(\mathbf{X} \boldsymbol{\alpha}_R \mathbf{X}^\top + \epsilon \mathbf{I}_p - (q + \nu_L) \boldsymbol{\alpha}_L^{-1}), \\ \mathbf{v}_R &= \frac{1}{2} \mathbf{D}_q \text{vec}(\mathbf{X}^\top \boldsymbol{\alpha}_L \mathbf{X} + \epsilon \mathbf{I}_q - (p + \nu_R) \boldsymbol{\alpha}_R^{-1}).\end{aligned}$$

For deterministic hyper parameters $\nu_L = \nu_R = \epsilon = 0$, $c = 1$ and $d = 0$. The terms $\mathbf{F}_{\mathbf{x}\boldsymbol{\alpha}_L}$ and $\mathbf{F}_{\mathbf{x}\boldsymbol{\alpha}_R}$ are zero mean with respect to \mathbf{x} and that the terms $\mathbf{F}_{\mathbf{x}\beta}$, $\mathbf{F}_{\boldsymbol{\alpha}_L\beta}$ and $\mathbf{F}_{\boldsymbol{\alpha}_R\beta}$ are zero when β is deterministic and zero-mean when β is random. The terms do therefore not contribute to the bounds. For completeness we give the terms in Section 7.4.9.

The proof of Proposition 7.3.7 is given in Section 7.4.9. The submatrix

$$\mathbf{F}_{\boldsymbol{\alpha}\boldsymbol{\alpha}} = \begin{bmatrix} \mathbf{F}_{\boldsymbol{\alpha}_L \boldsymbol{\alpha}_L} & \mathbf{F}_{\boldsymbol{\alpha}_L \boldsymbol{\alpha}_R} \\ \mathbf{F}_{\boldsymbol{\alpha}_L \boldsymbol{\alpha}_R}^\top & \mathbf{F}_{\boldsymbol{\alpha}_R \boldsymbol{\alpha}_R} \end{bmatrix} = \begin{bmatrix} \mathbf{v}_L \\ \mathbf{v}_R \end{bmatrix} \begin{bmatrix} \mathbf{v}_L^\top & \mathbf{v}_R^\top \end{bmatrix},$$

is singular for a fixed \mathbf{x} . However, $\mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\boldsymbol{\alpha}}]$ is not singular. For the RSVM model, we find that

$$\frac{\partial \mathbf{g}}{\partial \mathbf{z}} = \begin{bmatrix} \mathbf{I}_{pq} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{U}_{\boldsymbol{\alpha}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 1 \end{bmatrix},$$

where the column vectors of $\mathbf{U}_{\boldsymbol{\alpha}} = [\mathbf{U}_{\boldsymbol{\alpha}_L} \quad \mathbf{U}_{\boldsymbol{\alpha}_R}]$ are given by

$$\begin{aligned}[\mathbf{U}_{\boldsymbol{\alpha}_L}]_{:,i+(j-1)p} &= \text{vec}(\boldsymbol{\alpha}_R \otimes \mathbf{E}_{ij}^L), \\ [\mathbf{U}_{\boldsymbol{\alpha}_R}]_{:,k+(l-1)q} &= \text{vec}(\mathbf{E}_{kl}^R \otimes \boldsymbol{\alpha}_L).\end{aligned}$$

BCRB-I and BCRB-II

We compute the BCRB-II by calculating expectation values with respect to $\mathbf{x} = \text{vec}(\mathbf{X})$. We state the BCRB bound in the following proposition.

Proposition 7.3.8. *The BCRB-II of the RSVM model is given by*

$$\mathbf{C}_{\epsilon} \succeq \begin{bmatrix} (\mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\mathbf{xx}}])^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{U}_{\boldsymbol{\alpha}} (\mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\boldsymbol{\alpha}\boldsymbol{\alpha}}])^{-1} \mathbf{U}_{\boldsymbol{\alpha}}^\top & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & F_{\beta\beta}^{-1} \end{bmatrix},$$

where

$$\begin{aligned}
\mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\mathbf{x}}] &= (\boldsymbol{\alpha}_R \otimes \boldsymbol{\alpha}_L) + \beta \mathbf{A}^\top \mathbf{A}, \\
\mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\boldsymbol{\alpha}}] &= \begin{bmatrix} \mathcal{E}_{\mathbf{x}}[\mathbf{v}_L \mathbf{v}_L^\top] & \mathcal{E}_{\mathbf{x}}[\mathbf{v}_L \mathbf{v}_R^\top] \\ \mathcal{E}_{\mathbf{x}}[\mathbf{v}_R \mathbf{v}_L] & \mathcal{E}_{\mathbf{x}}[\mathbf{v}_R \mathbf{v}_R^\top] \end{bmatrix}, \\
\mathcal{E}_{\mathbf{x}}[\mathbf{v}_L \mathbf{v}_L^\top] &= \frac{1}{4} \mathbf{D}_p \text{vec}(\epsilon \mathbf{I}_p - \nu_L \boldsymbol{\alpha}_L^{-1}) \text{vec}(\epsilon \mathbf{I}_p - \nu_L \boldsymbol{\alpha}_L^{-1})^\top \mathbf{D}_p^\top \\
&\quad + \frac{q}{4} \mathbf{D}_p (\mathbf{I}_{p^2} + \mathbf{K}_{p,p}) (\boldsymbol{\alpha}_L^{-1} \otimes \boldsymbol{\alpha}_L^{-1}) \mathbf{D}_p^\top, \\
\mathcal{E}_{\mathbf{x}}[\mathbf{v}_R \mathbf{v}_R^\top] &= \frac{1}{4} \mathbf{D}_q \text{vec}(\epsilon \mathbf{I}_q - \nu_R \boldsymbol{\alpha}_R^{-1}) \text{vec}(\epsilon \mathbf{I}_q - \nu_R \boldsymbol{\alpha}_R^{-1})^\top \mathbf{D}_q^\top \\
&\quad + \frac{p}{4} \mathbf{D}_q (\mathbf{I}_{q^2} + \mathbf{K}_{q,q}) (\boldsymbol{\alpha}_R^{-1} \otimes \boldsymbol{\alpha}_R^{-1}) \mathbf{D}_q^\top, \\
\mathcal{E}_{\mathbf{x}}[\mathbf{v}_L \mathbf{v}_R^\top] &= \frac{1}{4} \mathbf{D}_p \text{vec}(\epsilon \mathbf{I}_p - \nu_L \boldsymbol{\alpha}_L^{-1}) \text{vec}(\epsilon \mathbf{I}_q - \nu_R \boldsymbol{\alpha}_R^{-1})^\top \mathbf{D}_q^\top \\
&\quad + \frac{1}{2} \mathbf{D}_p \text{vec}(\boldsymbol{\alpha}_L^{-1}) \text{vec}(\boldsymbol{\alpha}_R^{-1})^\top \mathbf{D}_q, \\
F_{\beta\beta} &= \frac{m}{2\beta^2}.
\end{aligned}$$

Also, the BCRB-I of the RSVM model is given by

$$\mathcal{E}_{\mathbf{y},\mathbf{x}} \left[(\hat{\mathbf{x}} - \mathbf{x})(\hat{\mathbf{x}} - \mathbf{x})^\top \right] \succeq (\mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\mathbf{xx}}])^{-1} = ((\boldsymbol{\alpha}_R \otimes \boldsymbol{\alpha}_L) + \beta \mathbf{A}^\top \mathbf{A})^{-1}$$

The proof of Proposition 7.3.8 is given in Section 7.4.10.

7.3.4 BCRB-III

Computation of BCRB-III requires taking expectation values with respect to the precision matrices $\boldsymbol{\alpha}_L$, $\boldsymbol{\alpha}_R$ and the noise precision β . We state the BCRB bound as a proposition.

Proposition 7.3.9. *The BCRB-III of the RSVM model is given by*

$$\mathbf{C}_\epsilon \succeq \begin{bmatrix} (\mathcal{E}_{\mathbf{z}}[\mathbf{F}_{\mathbf{xx}}])^{-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathcal{E}_{\mathbf{z}}[\mathbf{U}_{\boldsymbol{\alpha}}] (\mathcal{E}_{\mathbf{z}}[\mathbf{F}_{\boldsymbol{\alpha}\boldsymbol{\alpha}}])^{-1} \mathcal{E}_{\mathbf{z}}[\mathbf{U}_{\boldsymbol{\alpha}}]^\top & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & (\mathcal{E}_{\mathbf{z}}[F_{\beta\beta}])^{-1} \end{bmatrix},$$

where

$$\begin{aligned}
\mathcal{E}_{\mathbf{z}}[\mathbf{F}_{\mathbf{xx}}] &= (\nu_L + p + 1)(\nu_R + q + 1) \epsilon^{-2} \mathbf{I}_{pq} + \frac{1+c}{d} \mathbf{A}^\top \mathbf{A}, \\
\mathcal{E}_{\mathbf{z}}[\mathbf{v}_L \mathbf{v}_L^\top] &= \frac{\epsilon^2}{4} (\nu_L^2 c_L - 1 + 2qd_L) \text{vec}(\mathbf{I}_p) \text{vec}(\mathbf{I}_p)^\top \\
&\quad + \frac{\epsilon^2}{4} (\nu_L^2 d_L + c_L + d_L) \mathbf{D}_p^2
\end{aligned}$$

$$\begin{aligned}
& + \frac{\epsilon^2 q}{4} (c_L + d_L) \mathbf{D}_p \mathbf{K}_{p,p} \mathbf{D}_p + \frac{\epsilon^2 \nu_L^2 d_L}{4} \mathbf{D}_p \tilde{\mathbf{K}}_{p,p} \mathbf{D}_p, \\
\mathcal{E}_{\mathbf{z}} [\mathbf{v}_L \mathbf{v}_R^\top] & = \frac{\epsilon^2}{2\nu_L \nu_R} \text{vec}(\mathbf{I}_p) \text{vec}(\mathbf{I}_q)^\top, \\
\mathcal{E}_{\mathbf{z}} [\mathbf{v}_R \mathbf{v}_R^\top] & = \frac{\epsilon^2}{4} (\nu_R^2 c_R - 1 + 2pd_R) \text{vec}(\mathbf{I}_q) \text{vec}(\mathbf{I}_q)^\top \\
& + \frac{\epsilon^2}{4} (\nu_R^2 d_R + c_R + d_R) \mathbf{D}_q^2 \\
& + \frac{\epsilon^2 p}{4} (c_R + d_R) \mathbf{D}_q \mathbf{K}_{q,q} \mathbf{D}_q + \frac{\epsilon^2 \nu_R^2 d_R}{4} \mathbf{D}_q \tilde{\mathbf{K}}_{q,q} \mathbf{D}_q, \\
\mathcal{E}_{\mathbf{z}} [F_{\beta\beta}] & = \frac{(m^2 + 2c)d^2}{2c(c-1)}, \\
\mathcal{E}_{\mathbf{z}} [[\mathbf{U}_{\alpha_L}]_{:,i+(j-1)p}] & = (\nu_R + q + 1) \epsilon^{-1} \text{vec}(\mathbf{I}_p \otimes \mathbf{E}_{ij}^L), \\
\mathcal{E}_{\mathbf{z}} [[\mathbf{U}_{\alpha_R}]_{:,k+(l-1)q}] & = (\nu_L + p + 1) \epsilon^{-1} \text{vec}(\mathbf{E}_{kl}^R \otimes \mathbf{I}_q).
\end{aligned}$$

The proof of Proposition 7.3.9 is given in Section 7.4.11.

7.3.5 BCRB-IV

In the BCRB-IV, we are only interested in estimating the precisions, i.e. (deterministic and unknown) precisions matrices and the noise precision. The variable \mathbf{X} is marginalized and the measurements have the distribution

$$p(\mathbf{y} | \alpha_L, \alpha_R, \beta) = \mathcal{N}(\mathbf{y} | \mathbf{0}, \mathbf{C}),$$

where

$$\mathbf{C} = \mathbf{A}(\alpha_R^{-1} \otimes \alpha_L^{-1}) \mathbf{A}^\top + \beta^{-1} \mathbf{I}_m.$$

We want to find bounds for estimators of

$$\boldsymbol{\eta} = \mathbf{g}(\mathbf{z}) = \begin{bmatrix} \text{vec}(\alpha_R \otimes \alpha_L) \\ \beta \end{bmatrix},$$

where now

$$\frac{\partial \mathbf{g}}{\partial \mathbf{z}} = \begin{bmatrix} \mathbf{U}_{\alpha_L} & \mathbf{U}_{\alpha_R} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & 1 \end{bmatrix}.$$

The Fisher information for BCRB-IV becomes different from that of Proposition 7.3.7. We state the Fisher information matrix and the bound BCRB-IV in the following proposition.

Proposition 7.3.10. *Let*

$$\begin{aligned}\tilde{\mathbf{A}} &= \mathbf{A}(\boldsymbol{\alpha}_R^{-1} \otimes \boldsymbol{\alpha}_L^{-1}), \\ \mathbf{P} &= \tilde{\mathbf{A}}^\top \mathbf{C}^{-1} \tilde{\mathbf{A}}.\end{aligned}$$

The BCRB-IV of the RSVM model is given by

$$\mathbf{C}_\eta \succeq \frac{\partial \mathbf{g}}{\partial \mathbf{z}} \mathbf{F}^{-1} \frac{\partial \mathbf{g}^\top}{\partial \mathbf{z}}$$

where the Fisher information is

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{LL} & \mathbf{F}_{LR} & \mathbf{F}_{L\beta} \\ \mathbf{F}_{RL} & \mathbf{F}_{RR} & \mathbf{F}_{R\beta} \\ \mathbf{F}_{L\beta}^\top & \mathbf{F}_{R\beta}^\top & F_{\beta\beta} \end{bmatrix}$$

and the components of the block matrices are given by

$$\begin{aligned}[\mathbf{F}_{LL}]_{i+(j-1)p, k+(l-1)p} &= \frac{1}{4} \text{tr} \left(\mathbf{P}(\boldsymbol{\alpha}_R \otimes \mathbf{E}_{ij}^L) \mathbf{P}(\boldsymbol{\alpha}_R \otimes \mathbf{E}_{kl}^L) \right), \\ [\mathbf{F}_{LR}]_{i+(j-1)p, k+(l-1)q} &= \frac{1}{4} \text{tr} \left(\mathbf{P}(\boldsymbol{\alpha}_R \otimes \mathbf{E}_{ij}^L) \mathbf{P}(\mathbf{E}_{kl}^R \otimes \boldsymbol{\alpha}_L) \right), \\ [\mathbf{F}_{RR}]_{i+(j-1)q, k+(l-1)q} &= \frac{1}{4} \text{tr} \left(\mathbf{P}(\mathbf{E}_{ij}^R \otimes \boldsymbol{\alpha}_L) \mathbf{P}(\mathbf{E}_{kl}^R \otimes \boldsymbol{\alpha}_L) \right), \\ [\mathbf{F}_{L\beta}]_{i+(j-1)p} &= \frac{1}{4\beta^2} \text{tr} \left(\tilde{\mathbf{A}}^\top \mathbf{C}^{-2} \tilde{\mathbf{A}}(\boldsymbol{\alpha}_R \otimes \mathbf{E}_{ij}^L) \right), \\ [\mathbf{F}_{R\beta}]_{k+(l-1)q} &= \frac{1}{4\beta^2} \text{tr} \left(\tilde{\mathbf{A}}^\top \mathbf{C}^{-2} \tilde{\mathbf{A}}(\mathbf{E}_{kl}^R \otimes \boldsymbol{\alpha}_L) \right), \\ F_{\beta\beta} &= \frac{1}{4\beta^4} \text{tr} \left(\mathbf{C}^{-2} \right).\end{aligned}$$

The proof of Proposition 7.3.10 is given in Section 7.4.12.

7.4 Numerical experiments

In this section we perform numerical experiments to evaluate how low-rank realizations of the random matrix models are and also compare the BCRB bounds with existing methods.

7.4.1 Numerical rank of the random matrix models

The sparsity-based model, the factorized model and the RSVM model were outlined in Section 7.2 where we argued that the models could be used as priors to promote

low-rank. Here we show that for appropriate values of the hyperparameters, the models give realizations that are approximately low-rank. We notice that since the distributions are continuous, the probability of a matrix being exactly low-rank is zero. However, the realizations can be effectively low-rank, meaning that the matrices have few large singular values and the remaining singular values are small. This can be quantified through the numerical rank. We here use two versions of the numerical rank.

The first measure of numerical rank we use, for $\mathbf{X} \in \mathbb{R}^{p \times q}$, is

$$\text{nrank}_1(\mathbf{X}) = \frac{\|\mathbf{X}\|_*^2}{\|\mathbf{X}\|_F^2},$$

where $\|\mathbf{X}\|_* = \sum_{i=1}^k \sigma_i(\mathbf{X})$ is the nuclear norm of \mathbf{X} and $\|\mathbf{X}\|_F^2 = \sum_{i=1}^k \sigma_i(\mathbf{X})^2$ is the Frobenius norm. By the Cauchy-Schwarz inequality we get that the numerical rank is a lower bound for the true rank as $\text{nrank}_1(\mathbf{X}) \leq \text{rank}(\mathbf{X})$. The second measure of numerical rank is

$$\text{nrank}_2(\mathbf{X}) = |\{i : \sigma_i(\mathbf{X}) > \epsilon \|\mathbf{X}\|_F\}|,$$

where we set $\epsilon = 0.9/\sqrt{\min(p, q)}$. We see that $\text{nrank}_2(\cdot)$ also is a lower bound on the actual rank. We will find in simulations that both measures of numerical rank show the same qualitative behavior.

Numerical rank of the sparsity-based model

We saw in Section 7.2.1 that the effective distribution of the singular values in the sparsity based model is

$$e^{-\tilde{f}(\boldsymbol{\sigma})} = \zeta(\boldsymbol{\sigma})e^{-f(\boldsymbol{\sigma})},$$

where $\tilde{f}(\boldsymbol{\sigma}) = f(\boldsymbol{\sigma}) - \log \zeta(\boldsymbol{\sigma})$.

To evaluate the numerical rank, we generate samples from the effective and naive GCP prior using the Metropolis-Hastings algorithm [ADFDJ03]. For $p = q = 100$ and $\tau = 1$ we randomly chose s uniformly in the interval $[0, 10]$. Using the value of s we compute a realization of $\boldsymbol{\sigma}$ and compute the numerical rank of $\boldsymbol{\sigma}$. We obtained the results shown in Figure 7.1. We find that the majority of realizations of the GCP prior satisfy $\text{nrank}_1(\mathbf{X}) \leq 30$ and $\text{nrank}_2(\mathbf{X}) \approx 20$ for $1 \leq s \leq 2$. We thus see that the GCP prior gives realizations of low numerical rank.

Numerical rank of the factorized model

In the factorized model, the low rank matrix $\mathbf{X} \in \mathbb{R}^{p \times q}$ is written as the product $\mathbf{X} = \mathbf{L}\mathbf{R}^\top$ where $\mathbf{L} \in \mathbb{R}^{p \times r}$ and $\mathbf{R} \in \mathbb{R}^{q \times r}$. Clearly $\text{rank}(\mathbf{X}) \leq r$. However, the effective rank can be made lower by setting the parameter a in (7.10) to an appropriate value. We find that the parameter $b > 0$ only affects the magnitude

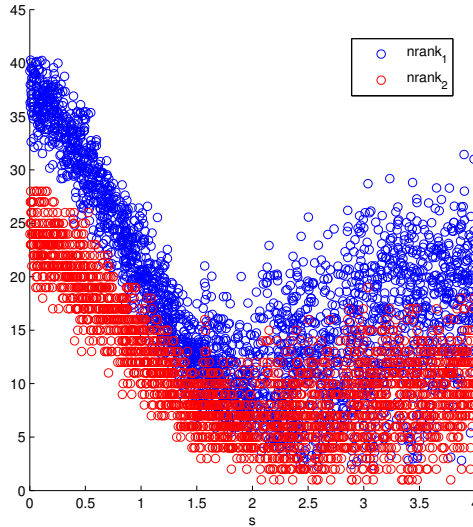


Figure 7.1: Numerical rank for realizations of the sparsity based model with generalized compressible priors for $p = q = 100$ and $\tau = 1$.

of the precisions since if $\gamma_i \sim \text{Gamma}(1 + a, 1)$, then $b\gamma_i \sim \text{Gamma}(1 + a, b)$. To evaluate how the choice of a affects the numerical rank we drew $\log(a)$ uniformly from the interval $[-10, 5]$ and for the corresponding value of a we generated \mathbf{L} and \mathbf{R} and computed the numerical rank of \mathbf{X} . In the experiment we used $p = q = 100$, $r = 50$ and $b = 1$. The results are shown in Figure 7.2 where the green line shows a local average value of the numerical rank. We find that for $10^{-10} \leq a \leq 1$, rank_1 has mean 7.8 and rank_2 has mean 6.2. Both measures of numerical rank have high variance in this region. For $a > 1$, rank_1 has mean 38 and rank_2 has mean 30. We thus see that the factorized model promotes sparsity less than r when $a \leq 1$. Unfortunately, this is also the region where the BCRB-III does not exist.

Numerical rank of the RSVM model

The parameters ν_L and ν_R control the numerical rank in the RSVM model since the parameter ϵ only controls the magnitude of the precision matrices. Here we set $\nu_L = \nu_R = \nu$ for simplicity. In the experiment we drew ν uniformly from $[\max(p, q), 2(p + q)]$. For each value of ν we drew the precision matrices α_L and α_R from the Wishart distributions and generated \mathbf{X} as in (7.13). We then computed nrank_1 and nrank_2 for the matrix. The results are shown in Figure 7.9. We find that $\text{nrank}_1(\mathbf{X}) \leq 20$ when $\nu \leq 115$ while $\text{nrank}_2(\mathbf{X}) \leq 20$ when $\nu \leq 150$. For larger ν , the numerical rank $\text{nrank}_1(\mathbf{X})$ increases towards 72 while $\text{nrank}_2(\mathbf{X})$ increases towards 45.

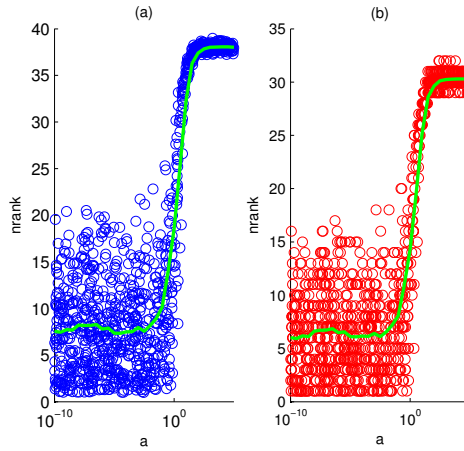


Figure 7.2: Numerical ranks for realizations of the factorized model for $p = q = 100$, $r = 50$ and $b = 1$. (a) shows $\text{nrank}_1(\mathbf{X})$ and (b) shows $\text{nrank}_2(\mathbf{X})$. The green line shows the average value of the numerical rank in an interval.

7.4.2 Evaluating the BCRB bounds

In this section we numerically evaluate the bounds and compare them to the performance of existing estimation methods. The first estimator we compare with is the nuclear norm (NN) estimator

$$\hat{\mathbf{X}} = \arg \min \|\mathbf{X}\|_*, \text{ subject to } \|\mathbf{y} - \mathbf{A}\text{vec}(\mathbf{X})\|_2 \leq \delta,$$

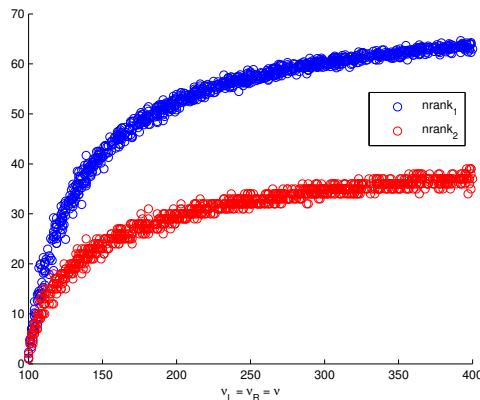


Figure 7.3: Numerical ranks of realizations of the RSVM model for $p = q = 100$ and $\epsilon = 10^{-5}$.

where $\delta = \beta^{-1}\sqrt{m + \sqrt{8m}}$ as suggested in [CRT06]. We also compare with the variational Bayesian estimator of [BLMK12, SCJR14] and the RSVM method for the log-determinant penalty from [SRJC]. Only VB estimates the precisions γ while only the RSVM method estimates the matrix precisions α_L and α_R . Note that the NN estimator requires β to be known while the other algorithms estimate β from data. We here concentrate on the matrix completion scenario where each row of the sensing matrix \mathbf{A} contains a single one and the remaining elements are zero. This means that we observe noisy measurements of m components of the matrix \mathbf{X} .

We measure the estimator performance in terms of the normalized mean square error

$$\text{NMSE} = \frac{\mathcal{E}_{\mathbf{z}} \left[\|\mathbf{X} - \hat{\mathbf{X}}\|_F^2 \right]}{\mathcal{E}_{\mathbf{z}} \left[\|\mathbf{X}\|_F^2 \right]},$$

where the expectation is taken over all random variables in the model. We chose the noise parameter d in (7.2) such that the signal to noise ratio

$$\begin{aligned} \text{SNR} &= \frac{\mathcal{E}_{\mathbf{z}} \left[\|\mathbf{A}\text{vec}(\mathbf{X})\|_2^2 \right]}{\mathcal{E}_{\mathbf{z}} \left[\|\mathbf{n}\|_2^2 \right]} \\ &= \frac{\text{tr} \left(\mathcal{E}_{\mathbf{A}} \left[\mathbf{A}^\top \mathbf{A} \right] \mathcal{E}_{\mathbf{x}} \left[\text{vec}(\mathbf{X})\text{vec}(\mathbf{X})^\top \right] \right)}{\mathcal{E}_{\beta} \left[m\beta^{-1} \right]} \\ &= \frac{\frac{m}{pq} \mathcal{E}_{\mathbf{x}} \left[\|\mathbf{X}\|_F^2 \right]}{m \frac{d}{c}} = \frac{c}{pqd} \mathcal{E}_{\mathbf{x}} \left[\|\mathbf{X}\|_F^2 \right], \end{aligned}$$

is held fixed at 20 dB.

BCRB-II and BCRB-III for the sparsity based model

The BCRB-II (and BCRB-I) and BCRB-III for \mathbf{X} of the sparsity based model is given by Proposition 7.3.2 and Proposition (7.3.3). Here we show the NMSE of the estimators and normalized BCRBs as a function of the number of measurements m in Figure 7.4. We notice a gap of about 20 dB between the estimation methods and BCRB-II and about 30 dB between the estimation methods and BCRB-III. The distribution of the numerical rank of the matrix realizations is shown in Figure 7.5.

BCRB-II and BCRB-III for the factorized model

The BCRB-II (and BCRB-I) and BCRB-III for \mathbf{X} of the factorized model is given by Proposition 7.3.5 and Proposition (7.3.6). Here we show the NMSE of the estimators and the normalized BCRBs as a function of the number of measurements m in Figure 7.6. We find that for $m/pq > 0.3$ the gap between the estimation methods and the bounds is -3 to 15 dB for BCRB-II and 20 to 30 dB for BCRB-III. The BCRB-II is above the performance of the methods for $m/pq = 0.9$. This is probably because of bias in the estimation methods. The distribution of the numerical rank of the matrix realizations is shown in Figure 7.7.

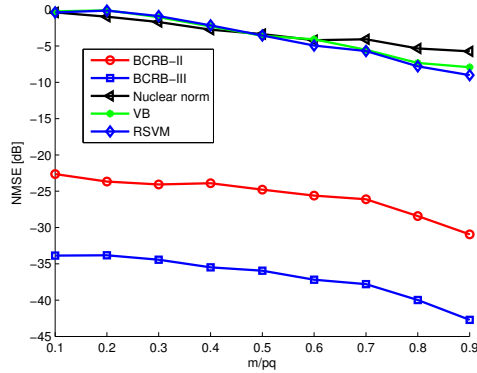


Figure 7.4: NMSE for realizations of the sparsity-based model vs. number of measurements m for matrix completion.

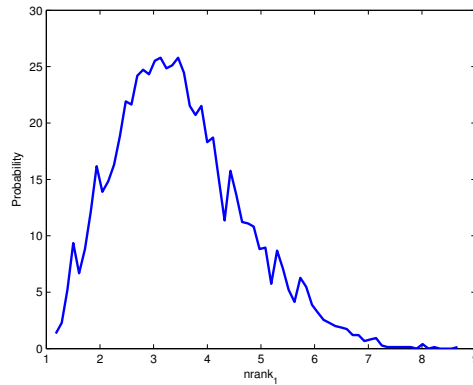


Figure 7.5: Empirical distribution of the numerical rank for the sparsity based model with $p = q = 10$, $\nu = 10$ and $\epsilon = 1$.

BCRB-II and BCRB-III for the RSVM model

The BCRB-II (and BCRB-I) and BCRB-III for \mathbf{X} of the sparsity based model is given by Proposition 7.3.8 and Proposition (7.3.9). Here we show the NMSE of the estimators and normalized BCRBs as a function of the number of measurements m in Figure 7.8. We find that the best estimation method was the nuclear norm estimator. The gap between the nuclear norm and BCRB-II was 33 to 52 dB while the gap between the nuclear norm and BCRB-III was 50 to 88 dB. Surprisingly, the RSVM method (which was designed for the model) gave the worst performance and performed worse with increasing number of measurements. The distribution of the numerical rank (rank_1) of the matrix realizations is shown in Figure 7.9. We find that the numerical rank was between 1 and 5 and often close to 1.

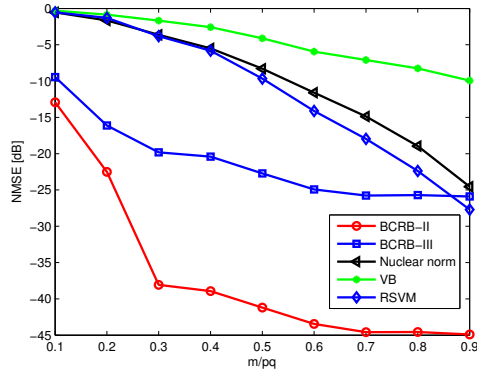


Figure 7.6: NMSE for the factorization based model vs. number of measurements m for matrix completion.

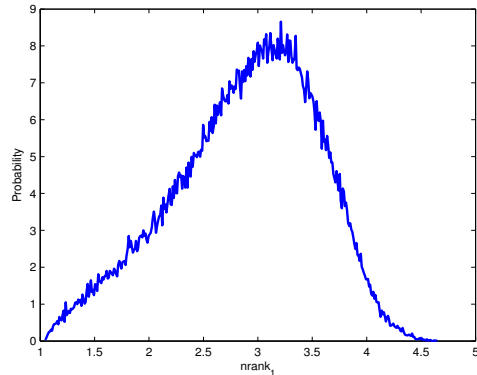


Figure 7.7: Distribution of the numerical rank for realizations of the factorization based model with $p = q = 25$, $r = 5$ and $a = b = 1 + 10^{-3}$.

7.4.3 Proof of Proposition 7.1.1

Proposition 7.1.1 gives the BCRB bounds (7.4) and (7.5). The derivation of the bounds is similar to the derivation of the deterministic CRB in [Kay93, Cra47] and was earlier given in [VT04, GL95, BS80]. For completeness we here repeat the derivation.

Let $\hat{\boldsymbol{\eta}}$ be an unbiased estimator of $\boldsymbol{\eta} = \mathbf{g}(\mathbf{z}) \in \mathbb{R}^K$ from measurements $\mathbf{y} \in \mathbb{R}^m$ and assume that the probability distribution function $p(\mathbf{y}, \mathbf{z})$ is defined for $\mathbf{z} \in \Omega \subset \mathbb{R}^n$ with $p(\mathbf{y}, \mathbf{z}) = 0$ for points on the boundary, $\mathbf{z} \in \partial\Omega$. Let also $\mathbf{a} \in \mathbb{R}^K$ be a constant vector and $\mathbf{b} = \mathbf{b}(\mathbf{z}) \in \mathbb{R}^n$ be a vector which depends on \mathbf{z} . We find that

$$\mathcal{E}_{\mathbf{y}, \mathbf{z}} \left[\mathbf{a}^\top (\hat{\boldsymbol{\eta}} - \boldsymbol{\eta}) \mathbf{b}^\top \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \mathbf{z}} \right]$$

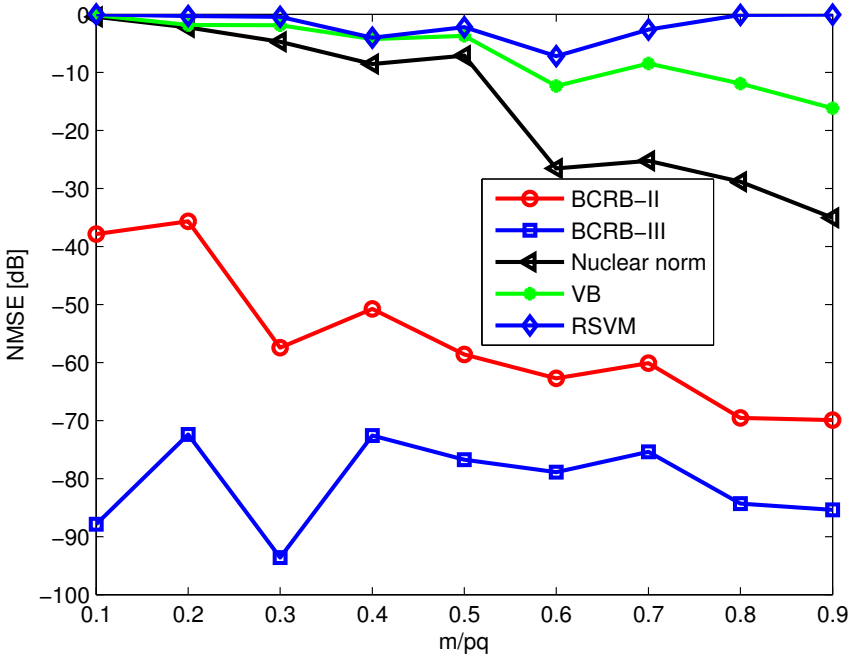


Figure 7.8: NMSE for the RSVM based model vs. number of measurements m for matrix completion with $p = q = 10$, $\nu = 10$ and SNR = 20 dB.

$$\begin{aligned}
&= \int_{\mathbb{R}^m} \int_{\Omega} \mathbf{a}^\top (\hat{\boldsymbol{\eta}} - \boldsymbol{\eta}) \mathbf{b}^\top \frac{\partial p(\mathbf{y}, \mathbf{z})}{\partial \mathbf{z}} d\mathbf{y} d\mathbf{z} \\
&= - \int_{\mathbb{R}^m} \int_{\Omega} \text{tr} \left(\frac{\partial}{\partial \mathbf{z}} (\mathbf{a}^\top (\hat{\boldsymbol{\eta}} - \boldsymbol{\eta}) \mathbf{b}) \right) p(\mathbf{y}, \mathbf{z}) d\mathbf{y} d\mathbf{z} \\
&= \mathcal{E}_{\mathbf{z}} \left[\mathbf{a}^\top \frac{\partial \mathbf{g}}{\partial \mathbf{z}} \mathbf{b} \right] - \mathcal{E}_{\mathbf{z}} \left[\mathbf{a}^\top \mathcal{E}_{\mathbf{y}} [(\hat{\boldsymbol{\eta}} - \boldsymbol{\eta})] \text{tr} \left(\frac{\partial \mathbf{b}}{\partial \mathbf{z}} \right) \right] \\
&= \mathcal{E}_{\mathbf{z}} \left[\mathbf{a}^\top \frac{\partial \mathbf{g}}{\partial \mathbf{z}} \mathbf{b} \right]
\end{aligned}$$

where we used that $\hat{\boldsymbol{\eta}}$ only depends on \mathbf{y} and that $\mathcal{E}_{\mathbf{y}} [\hat{\boldsymbol{\eta}} - \boldsymbol{\eta}] = \mathbf{0}$.

The Cauchy-Schwartz inequality gives that

$$\begin{aligned}
&\left(\mathcal{E}_{\mathbf{z}} \left[\mathbf{a}^\top \frac{\partial \mathbf{g}}{\partial \mathbf{z}} \mathbf{b} \right] \right)^2 \leq \\
&\mathcal{E}_{\mathbf{y}, \mathbf{z}} \left[((\hat{\boldsymbol{\eta}} - \boldsymbol{\eta})^\top \mathbf{a})^2 \right] \mathcal{E}_{\mathbf{y}, \mathbf{z}} \left[\left(\frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \mathbf{z}} \right)^\top \mathbf{b} \right]^2 \\
&= \mathbf{a}^\top \mathbf{C}_{\boldsymbol{\eta}} \mathbf{a} \cdot \mathcal{E}_{\mathbf{z}} [\mathbf{b}^\top \mathbf{F} \mathbf{b}]
\end{aligned} \tag{7.23}$$

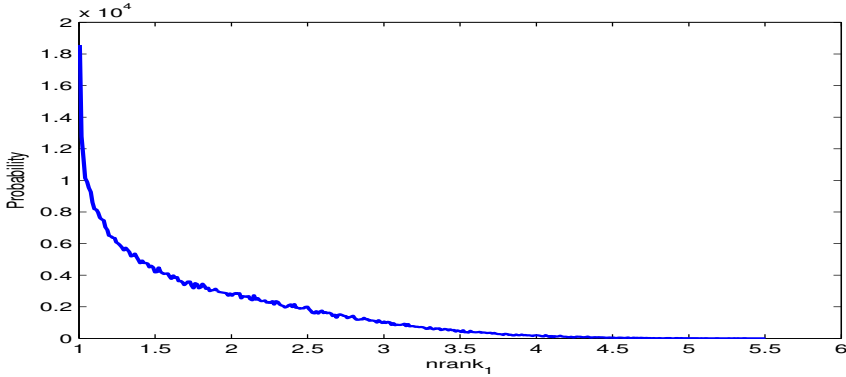


Figure 7.9: Distribution of the numerical rank for realizations of the RSVM model with $p = q = 10$ and $\nu = 10$.

where we set

$$\mathbf{C}_\epsilon = \mathcal{E}_{\mathbf{y}, \mathbf{z}} [(\hat{\boldsymbol{\eta}} - \boldsymbol{\eta})(\hat{\boldsymbol{\eta}} - \boldsymbol{\eta})^\top],$$

$$\mathbf{F} = \mathcal{E}_{\mathbf{y}} \left[\frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \mathbf{z}} \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \mathbf{z}}^\top \right].$$

From (7.23) we derive the BCRB bounds by choosing \mathbf{b} appropriately. Setting

$$\mathbf{b} = (\mathcal{E}_{\mathbf{z}}[\mathbf{F}])^{-1} \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \right]^\top \mathbf{a},$$

gives that

$$\mathbf{a}^\top \mathbf{C}_\epsilon \mathbf{a} \geq \mathbf{a}^\top \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \right] (\mathcal{E}_{\mathbf{z}}[\mathbf{F}])^{-1} \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \right]^\top \mathbf{a},$$

for all $\mathbf{a} \in \mathbb{R}^n$. It follows that

$$\mathbf{C}_\epsilon \succeq \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \right] (\mathcal{E}_{\mathbf{z}}[\mathbf{F}])^{-1} \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \right]^\top.$$

This gives us the bound (7.4). Another choice is to set

$$\mathbf{b} = \frac{\partial \mathbf{g}}{\partial \mathbf{z}}^\top \left(\mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \mathbf{F} \frac{\partial \mathbf{g}}{\partial \mathbf{z}}^\top \right] \right)^{-1} \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \frac{\partial \mathbf{g}}{\partial \mathbf{z}}^\top \right] \mathbf{a}$$

which gives us that

$$\mathbf{C}_\epsilon \succeq \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \frac{\partial \mathbf{g}}{\partial \mathbf{z}}^\top \right] \left(\mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \mathbf{F} \frac{\partial \mathbf{g}}{\partial \mathbf{z}}^\top \right] \right)^{-1} \mathcal{E}_{\mathbf{z}} \left[\frac{\partial \mathbf{g}}{\partial \mathbf{z}} \frac{\partial \mathbf{g}}{\partial \mathbf{z}}^\top \right].$$

This is the bound (7.5).

7.4.4 Proof of Proposition 7.3.1

Here we compute the Fisher information matrix for the sparsity based model. In the sparsity based model, $p(\mathbf{y}, \mathbf{z}) = p(\mathbf{y}|\mathbf{x}, \beta)p(\mathbf{x})p(\beta)$. To compute the Fisher information we need to compute the derivatives of $\log p(\mathbf{y}, \mathbf{z})$ with respect to \mathbf{x} and β . We have that

$$\begin{aligned} \log p(\mathbf{y}, \mathbf{z}) &= -\frac{\beta}{2} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \frac{m}{2} \log \beta \\ &\quad - f(\boldsymbol{\sigma}(\mathbf{X})) + (c-1) \log \beta - d\beta + \text{const} \end{aligned}$$

By varying with respect to \mathbf{x} , we get that

$$\begin{aligned} d \log p(\mathbf{y}, \mathbf{z}) &= \beta(\mathbf{y} - \mathbf{A}\mathbf{x})^\top \mathbf{A} d\mathbf{x} - \sum_{i=1}^k \frac{\partial f}{\partial \sigma_i} \mathbf{u}_i^\top d\mathbf{X}\mathbf{v}_i \\ &= \beta(\mathbf{y} - \mathbf{A}\mathbf{x})^\top \mathbf{A} d\mathbf{x} - \sum_{i=1}^k \frac{\partial f}{\partial \sigma_i} (\mathbf{v}_i^\top \otimes \mathbf{u}_i^\top) d\mathbf{x}, \end{aligned}$$

where $k = \min(p, q)$. This gives us that

$$\frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \mathbf{x}} = \beta \mathbf{A}^\top (\mathbf{y} - \mathbf{A}\mathbf{x}) - \sum_{i=1}^k \frac{\partial f}{\partial \sigma_i} (\mathbf{v}_i \otimes \mathbf{u}_i).$$

We also find that

$$\frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \beta} = -\frac{1}{2} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \frac{m + 2(c-1)}{2\beta} - d.$$

This gives us that

$$\begin{aligned} \mathbf{F}_{\mathbf{xx}} &= \sum_{i,j} \frac{\partial f}{\partial \sigma_j} \frac{\partial f}{\partial \sigma_i} (\mathbf{v}_j \mathbf{v}_i^\top \otimes \mathbf{u}_j \mathbf{u}_i^\top) \\ &\quad + \beta^2 \mathbf{A}^\top \mathcal{E}_{\mathbf{y}} [(\mathbf{y} - \mathbf{A}\mathbf{x})(\mathbf{y} - \mathbf{A}\mathbf{x})^\top] \mathbf{A}, \\ &= \sum_{i,j} \frac{\partial f}{\partial \sigma_j} \frac{\partial f}{\partial \sigma_i} (\mathbf{v}_j \mathbf{v}_i^\top \otimes \mathbf{u}_j \mathbf{u}_i^\top) + \beta \mathbf{A}^\top \mathbf{A}, \\ \mathbf{F}_{\mathbf{x}\beta} &= \left(d - \frac{c}{\beta} \right) \sum_{i=1}^k \frac{\partial f}{\partial \sigma_i} (\mathbf{v}_i \otimes \mathbf{u}_i), \\ F_{\beta\beta} &= \frac{m}{2\beta^2} + \left(d - \frac{c-1}{\beta} \right)^2. \end{aligned}$$

where we used that

$$\begin{aligned} \mathcal{E}_{\mathbf{y}} [(\mathbf{y} - \mathbf{A}\mathbf{x})] &= \mathbf{0}, \\ \mathcal{E}_{\mathbf{y}} [(\mathbf{y} - \mathbf{A}\mathbf{x})(\mathbf{y} - \mathbf{A}\mathbf{x})^\top] &= \beta^{-1} \mathbf{I}_m, \\ \mathcal{E}_{\mathbf{y}} [(\mathbf{y} - \mathbf{A}\mathbf{x}) \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2] &= \mathbf{0}. \end{aligned}$$

We note that the term $\mathbf{F}_{\mathbf{x}\beta}$ is zero when β is deterministic ($c = 1$ and $d = 0$) and zero mean when β is random. This proves Proposition 7.3.1.

7.4.5 Proof of Proposition 7.3.2

To show Proposition 7.3.2 we need to compute the expectation value of $\mathbf{F}_{\mathbf{x}\mathbf{x}}$ and $\mathbf{F}_{\mathbf{x}\beta}$ with respect to \mathbf{x} . When the distribution $p(\mathbf{X})$ only depends on the singular values of \mathbf{X} , the singular values and the left and right singular vectors are independent (provided that the matrix is not symmetric). So the singular vectors have zero mean, $\mathcal{E}_{\mathbf{x}}[\mathbf{u}_i] = \mathbf{0}$. This gives us that $\mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\mathbf{x}\beta}] = \mathbf{0}$. Moreover, conditioned on a singular vector \mathbf{u}_i , another singular vector \mathbf{u}_j (with $j \neq i$) is marginally uniformly distributed on the set $\{\mathbf{u} \in \mathbb{R}^p : \|\mathbf{u}\|_2 = 1, \mathbf{u}^\top \mathbf{u}_i = 0\}$. This gives us that when $i \neq j$

$$\mathcal{E}_{\mathbf{x}}[\mathbf{u}_j \mathbf{u}_i^\top] = \mathbf{0}.$$

Also, if $h \sim \chi^2(p)$, then $\sqrt{h}\mathbf{u}_i$ is Gaussian $\mathcal{N}(\mathbf{0}, \mathbf{I})$ and

$$\mathbf{I}_p = \mathcal{E}_{h, \mathbf{x}}[h \mathbf{u}_i \mathbf{u}_i^\top] = \mathcal{E}_h[h] \mathcal{E}_{\mathbf{x}}[\mathbf{u}_i \mathbf{u}_i^\top] = p \mathcal{E}_{\mathbf{x}}[\mathbf{u}_i \mathbf{u}_i^\top],$$

so

$$\mathcal{E}_{\mathbf{x}}[\mathbf{u}_i \mathbf{u}_i^\top] = \frac{1}{p} \mathbf{I}_p.$$

Similarly we find that

$$\mathcal{E}_{\mathbf{x}}[\mathbf{v}_i \mathbf{v}_j^\top] = \delta_{ij} \frac{1}{q} \mathbf{I}_q.$$

We thus get that

$$\mathcal{E}_{\mathbf{x}}[\mathbf{F}_{\mathbf{x}}] = \frac{1}{pq} \mathbf{I}_{pq} \sum_{i=1}^k \mathcal{E}_{\mathbf{x}} \left[\left(\frac{\partial f}{\partial \sigma_i} \right)^2 \right].$$

To compute the remaining expectation we make a change of variables corresponding to the singular value decomposition. Without loss of generality, we here assume that $p \leq q$. Under the variable transformation, the integration measure $d\mathbf{X}$ transforms as [Mui09]

$$d\mathbf{X} = \prod_{1 \leq i < j \leq k} (\sigma_i^2 - \sigma_j^2) \prod_{1 \leq l \leq k} \sigma_l^{|p-q|} d\boldsymbol{\sigma} [\mathbf{U}^\top d\mathbf{U}] [\tilde{\mathbf{V}}^\top d\tilde{\mathbf{V}}],$$

where $\boldsymbol{\sigma} \in \Omega$ and the extended orthogonal matrix $\tilde{\mathbf{V}} = [\mathbf{V} \ \mathbf{V}']$ is such that $\mathbf{V}'^\top \mathbf{V}' = \mathbf{I}_{q-k}$ and $\mathbf{V}'^\top \mathbf{V} = \mathbf{0}$. By performing the integration over the singular vectors, we get that

$$\mathcal{E}_{\mathbf{x}} \left[\left(\frac{\partial f}{\partial \sigma_i} \right)^2 \right] = \frac{N[f]}{D[f]},$$

where

$$N[f] = \int_{\Omega} h(\boldsymbol{\sigma})g(\boldsymbol{\sigma})e^{-f(\boldsymbol{\sigma})}d\boldsymbol{\sigma} \quad (7.24)$$

$$D[f] = \int_{\Omega} g(\boldsymbol{\sigma})e^{-f(\boldsymbol{\sigma})}d\boldsymbol{\sigma}, \quad (7.25)$$

$$g(\boldsymbol{\sigma}) = \prod_{q \leq i < j \leq r} (\sigma_i^2 - \sigma_j^2) \prod_{1 \leq l \leq k} \sigma_l^{|p-q|}, \quad (7.26)$$

$$h(\boldsymbol{\sigma}) = \sum_{i=1}^r \left(\frac{\partial f}{\partial \sigma_i} \right)^2. \quad (7.27)$$

7.4.6 Proof of Proposition 7.3.4

Proposition 7.3.4 gives the Fisher information matrix for the factorized model. In the factorized model, we have that

$$\begin{aligned} \log p(\mathbf{y}, \mathbf{z}) &= -\frac{\beta}{2} \|\mathbf{y} - \mathbf{A} \text{vec}(\mathbf{L}\mathbf{R}^\top)\|_2^2 + \frac{m+2(c-1)}{2} \log \beta \\ &- d\beta + \frac{p+q+2(a-1)}{2} \log |\boldsymbol{\Gamma}| - \frac{1}{2} \text{tr}(\mathbf{L}\boldsymbol{\Gamma}\mathbf{L}^\top) - \frac{1}{2} \text{tr}(\mathbf{R}\boldsymbol{\Gamma}\mathbf{R}^\top) \\ &- b \text{tr}(\boldsymbol{\Gamma}). \end{aligned}$$

We find that

$$\begin{aligned} \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \text{vec}(\mathbf{L})} &= \beta(\mathbf{R} \otimes \mathbf{I}_p)^\top \mathbf{A}^\top (\mathbf{y} - \mathbf{A} \text{vec}(\mathbf{L}\mathbf{R}^\top)) - \text{vec}(\mathbf{L}\boldsymbol{\Gamma}), \\ \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \text{vec}(\mathbf{R})} &= \beta \mathbf{K}_{q,r}(\mathbf{I}_q \otimes \mathbf{L})^\top \mathbf{A}^\top (\mathbf{y} - \mathbf{A} \text{vec}(\mathbf{L}\mathbf{R}^\top)) - \text{vec}(\mathbf{R}\boldsymbol{\Gamma}), \\ \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \gamma_i} &= \frac{p+q+2(a-1)}{2\gamma_i} - \frac{\|\mathbf{l}_i\|_2^2 + \|\mathbf{r}_i\|_2^2}{2} - b = h_i, \\ \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \beta} &= -\frac{1}{2} \|\mathbf{y} - \mathbf{A} \text{vec}(\mathbf{L}\mathbf{R}^\top)\|_2^2 + \frac{m+2(c-1)}{2\beta} - d. \end{aligned}$$

Setting $\mathbf{h} = [h_1, h_2, \dots, h_r]^\top$, we find that

$$\begin{aligned} \mathbf{F}_{LL} &= \boldsymbol{\mathcal{E}}_{\mathbf{y}} \left[\frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \text{vec}(\mathbf{L})} \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \text{vec}(\mathbf{L})}^\top \right] \\ &= \beta(\mathbf{R} \otimes \mathbf{I}_p)^\top \mathbf{A}^\top \mathbf{A}(\mathbf{R} \otimes \mathbf{I}_p) + \text{vec}(\mathbf{L}\boldsymbol{\Gamma})\text{vec}(\mathbf{L}\boldsymbol{\Gamma})^\top, \\ \mathbf{F}_{LR} &= \beta(\mathbf{R} \otimes \mathbf{I}_p)^\top \mathbf{A}^\top \mathbf{A}(\mathbf{I}_q \otimes \mathbf{L})\mathbf{K}_{r,q} + \text{vec}(\mathbf{L}\boldsymbol{\Gamma})\text{vec}(\mathbf{R}\boldsymbol{\Gamma})^\top, \\ \mathbf{F}_{RR} &= \beta \mathbf{K}_{q,r}(\mathbf{I}_q \otimes \mathbf{L})^\top \mathbf{A}^\top \mathbf{A}(\mathbf{I}_q \otimes \mathbf{L})\mathbf{K}_{r,q} + \text{vec}(\mathbf{R}\boldsymbol{\Gamma})\text{vec}(\mathbf{R}\boldsymbol{\Gamma})^\top, \\ \mathbf{F}_{L\gamma} &= -\text{vec}(\mathbf{L}\boldsymbol{\Gamma})\mathbf{h}^\top, \quad \mathbf{F}_{R\gamma} = -\text{vec}(\mathbf{R}\boldsymbol{\Gamma})\mathbf{h}^\top, \end{aligned}$$

$$\begin{aligned}
\mathbf{F}_{\gamma\gamma} &= \mathbf{h}\mathbf{h}^\top, \quad \mathbf{F}_{L\beta} = \left(d - \frac{c-1}{\beta}\right) \text{vec}(\mathbf{L}\boldsymbol{\Gamma}), \\
\mathbf{F}_{R\beta} &= \left(d - \frac{c-1}{\beta}\right) \text{vec}(\mathbf{R}\boldsymbol{\Gamma}), \quad \mathbf{F}_{\gamma\beta} = \left(d - \frac{c-1}{\beta}\right) \mathbf{h}, \\
F_{\beta\beta} &= \frac{m}{2\beta^2} + \left(d - \frac{c-1}{\beta}\right)^2.
\end{aligned}$$

We find that $\mathbf{F}_{L\gamma}$, $\mathbf{F}_{R\gamma}$, $\mathbf{F}_{L\beta}$, $\mathbf{R}\beta$ and $\mathbf{F}_{\gamma\beta}$ are zero when γ and β are deterministic ($a = c = 1$ and $b = d = 0$) and zero-mean when γ and β are random variables.

Using the expression for \mathbf{F} and (7.21) we get that

$$\begin{aligned}
\mathbf{G}_{\mathbf{w}\mathbf{w}} &= \beta(\mathbf{R}\mathbf{R}^\top \otimes \mathbf{I}_p)\mathbf{A}^\top\mathbf{A}(\mathbf{R}\mathbf{R}^\top \otimes \mathbf{I}_p) \\
&+ \beta(\mathbf{I}_q \otimes \mathbf{L}\mathbf{L}^\top)\mathbf{A}^\top\mathbf{A}(\mathbf{I}_q \otimes \mathbf{L}\mathbf{L}^\top) \\
&+ \beta(\mathbf{R}\mathbf{R}^\top \otimes \mathbf{I}_p)\mathbf{A}^\top\mathbf{A}(\mathbf{I}_q \otimes \mathbf{L}\mathbf{L}^\top) \\
&+ \beta(\mathbf{I}_q \otimes \mathbf{L}\mathbf{L}^\top)\mathbf{A}^\top\mathbf{A}(\mathbf{R}\mathbf{R}^\top \otimes \mathbf{I}_p), \\
&+ 4\text{vec}(\mathbf{L}\mathbf{R}\mathbf{R}^\top)\text{vec}(\mathbf{L}\mathbf{R}\mathbf{R}^\top)^\top, \\
\mathbf{G}_{\mathbf{w}\gamma} &= -2((\nabla_\gamma s)^\top \mathbf{h})\text{vec}(\mathbf{L}\mathbf{R}\mathbf{R}^\top), \\
\mathbf{G}_{\mathbf{w}\beta} &= 2\left(d - \frac{c-1}{\beta}\right)\text{vec}(\mathbf{L}\mathbf{R}\mathbf{R}^\top), \\
G_{\gamma\gamma} &= (\nabla_\gamma s)^\top \mathbf{F}_\gamma (\nabla_\gamma s) = ((\nabla_\gamma s)^\top \mathbf{h})^2, \\
G_{\gamma\beta} &= \left(d - \frac{c-1}{\beta}\right)((\nabla_\gamma s)^\top \mathbf{h}), \\
G_{\beta\beta} &= F_\beta.
\end{aligned} \tag{7.28}$$

We find that the parameters $\mathbf{G}_{\mathbf{w}\gamma}$, $\mathbf{G}_{\mathbf{w}\beta}$ and $G_{\gamma\beta}$ are zero when γ and β are deterministic and zero mean when the parameters are random because of the respective elements of the Fisher information matrix.

7.4.7 Proof of Proposition 7.3.5

Proposition 7.3.5 gives the bounds BCRB-I and BCRB-II of the factorized model. To derive the BCRB-II for the factorized model, we need to compute expectation values with respect to \mathbf{w} .

To compute $\mathcal{E}_{\mathbf{w}}[\mathbf{G}_{\mathbf{w}\mathbf{w}}]$ we use that

$$\begin{aligned}
&\mathcal{E}_{\mathbf{w}}[\text{vec}(\mathbf{L}\mathbf{R}\mathbf{R}^\top)\text{vec}(\mathbf{L}\mathbf{R}\mathbf{R}^\top)^\top] \\
&= \mathcal{E}_{\mathbf{w}}\left[\sum_{i,j=1}^r \text{vec}(\gamma_i \mathbf{l}_i \mathbf{r}_i^\top) \text{vec}(\gamma_j \mathbf{l}_j \mathbf{r}_j^\top)^\top\right] \\
&= \mathcal{E}_{\mathbf{w}}\left[\sum_{i,j=1}^r \gamma_i \gamma_j (\mathbf{r}_i \otimes \mathbf{l}_i)(\mathbf{r}_j \otimes \mathbf{l}_j)^\top\right]
\end{aligned}$$

$$\begin{aligned}
&= \sum_{i,j=1}^r \gamma_i \gamma_j (\mathcal{E}_{\mathbf{w}} [\mathbf{r}_i \mathbf{r}_j^\top] \otimes \mathcal{E}_{\mathbf{w}} [\mathbf{l}_i \mathbf{l}_j^\top]) \\
&= \sum_{i,j=1}^r \delta_{ij} \gamma_i \gamma_j (\gamma_i^{-1} \mathbf{I}_q \otimes \gamma_j^{-1} \mathbf{I}_p) = r \mathbf{I}_{pq}, \\
\mathcal{E}_{\mathbf{w}} \left[\beta (\mathbf{R} \mathbf{R}^\top \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{I}_q \otimes \mathbf{L} \mathbf{L}^\top) \right] &= \\
\beta (\mathcal{E}_{\mathbf{w}} [\mathbf{R} \mathbf{R}^\top] \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{I}_q \otimes \mathcal{E}_{\mathbf{w}} [\mathbf{L} \mathbf{L}^\top]) &= \\
\beta \left(\sum_{i=1}^r \gamma_i^{-1} \right)^2 (\mathbf{I}_q \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{I}_q \otimes \mathbf{I}_p). &
\end{aligned}$$

Expectations such as $\mathcal{E}_{\mathbf{w}} [\beta (\mathbf{R} \mathbf{R}^\top \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{R} \mathbf{R}^\top \otimes \mathbf{I}_p)]$ are more challenging to calculate. To compute the expectation, we note that

$$\begin{aligned}
& [(\mathbf{R} \mathbf{R}^\top \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{R} \mathbf{R}^\top \otimes \mathbf{I}_p)]_{i+(k-1)p, j+(l-1)p} \\
&= (\mathbf{e}_k \otimes \mathbf{e}_i)^\top (\mathbf{R} \mathbf{R}^\top \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{R} \mathbf{R}^\top \otimes \mathbf{I}_p) (\mathbf{e}_l \otimes \mathbf{e}_j) \\
&= (\mathbf{e}_k^\top \mathbf{R} \mathbf{R}^\top \otimes \mathbf{e}_i^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{R} \mathbf{R}^\top \mathbf{e}_l \otimes \mathbf{e}_j) \\
&= \text{tr} ((\mathbf{R} \mathbf{R}^\top \mathbf{e}_l \mathbf{e}_k^\top \mathbf{R} \mathbf{R}^\top \otimes \mathbf{e}_j \mathbf{e}_i^\top) \mathbf{A}^\top \mathbf{A}).
\end{aligned}$$

The i 'th row vector of \mathbf{R} is $\mathbf{R}^\top \mathbf{e}_i$, this gives us that

$$\begin{aligned}
& \mathcal{E}_{\mathbf{w}} [\mathbf{e}_m^\top \mathbf{R} \mathbf{R}^\top \mathbf{e}_l \mathbf{e}_k^\top \mathbf{R} \mathbf{R}^\top \mathbf{e}_n] \\
&= \mathcal{E}_{\mathbf{w}} [(\mathbf{R}^\top \mathbf{e}_m)^\top (\mathbf{R}^\top \mathbf{e}_l) (\mathbf{R}^\top \mathbf{e}_k)^\top (\mathbf{R}^\top \mathbf{e}_n)] \\
&= \left(\sum_{i=1}^r \gamma_i^{-1} \right)^2 \delta_{ml} \delta_{kn} + \left(\sum_{i=1}^r \gamma_i^{-2} \right) (\delta_{lk} \delta_{mn} + \delta_{mk} \delta_{ln}) \\
&= \mathbf{e}_m^\top \left[\left(\sum_{i=1}^r \gamma_i^{-1} \right)^2 \mathbf{e}_l \mathbf{e}_k^\top + \left(\sum_{i=1}^r \gamma_i^{-2} \right) (\delta_{lk} \mathbf{I}_q + \mathbf{e}_k \mathbf{e}_l^\top) \right] \mathbf{e}_n.
\end{aligned}$$

By using the above, we get that

$$\begin{aligned}
& \mathcal{E}_{\mathbf{w}} [(\mathbf{R} \mathbf{R}^\top \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{R} \mathbf{R}^\top \otimes \mathbf{I}_p)]_{i+(k-1)p, j+(l-1)p} \\
&= \left(\sum_{i=1}^r \gamma_i^{-1} \right)^2 (\mathbf{e}_k^\top \otimes \mathbf{e}_i^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{e}_l \otimes \mathbf{e}_j) \\
&+ \left(\sum_{i=1}^r \gamma_i^{-2} \right) (\mathbf{e}_l^\top \otimes \mathbf{e}_i^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{e}_k \otimes \mathbf{e}_j) \\
&+ \left(\sum_{i=1}^r \gamma_i^{-2} \right) \delta_{lk} \text{tr} [(\mathbf{I}_q \otimes \mathbf{e}_i^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{I}_q \otimes \mathbf{e}_j)].
\end{aligned}$$

Let \mathcal{T}_1 be the operator defined in section 7.1.1, we find that

$$\begin{aligned} (\mathbf{e}_k^\top \otimes \mathbf{e}_i^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{e}_l \otimes \mathbf{e}_j) &= [\mathbf{A}^\top \mathbf{A}]_{i+(k-1)p, j+(l-1)p}, \\ (\mathbf{e}_l^\top \otimes \mathbf{e}_i^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{e}_k \otimes \mathbf{e}_j) &= [\mathcal{T}_1(\mathbf{A}^\top \mathbf{A})]_{i+(k-1)p, j+(l-1)p}, \end{aligned}$$

Using that $(\mathbf{e}_n \otimes \mathbf{I}_p) \mathbf{e}_j = (\mathbf{e}_n \otimes \mathbf{I}_p)(1 \otimes \mathbf{e}_j) = (\mathbf{e}_n \otimes \mathbf{e}_j)$, we get that

$$\begin{aligned} &\delta_{lk} \text{tr} \left[(\mathbf{I}_q \otimes \mathbf{e}_i^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{I}_q \otimes \mathbf{e}_j) \right] \\ &= \mathbf{e}_l^\top \mathbf{e}_k \text{tr} \left[(\mathbf{I}_q \otimes \mathbf{e}_j \mathbf{e}_i^\top) \mathbf{A}^\top \mathbf{A} \right] \\ &= \mathbf{e}_l^\top \mathbf{e}_k \sum_{n=1}^r (\mathbf{e}_n^\top \otimes \mathbf{e}_i^\top) \mathbf{A}^\top \mathbf{A} (\mathbf{e}_n \otimes \mathbf{e}_j) \\ &= \left[\left(\mathbf{I}_q \otimes \left(\sum_{n=1}^r (\mathbf{e}_n^\top \otimes \mathbf{I}_p) \mathbf{A}^\top \mathbf{A} (\mathbf{e}_n \otimes \mathbf{I}_p) \right) \right) \right]_{i+(k-1)p, j+(l-1)p}. \end{aligned}$$

A similar computation can be made for \mathbf{L} .

To compute

$$G_{\gamma\gamma} = \mathcal{E}_{\mathbf{w}} [((\nabla_{\gamma s})^\top \mathbf{h})^2] = (\nabla_{\gamma s})^\top \mathcal{E}_{\mathbf{w}} [\mathbf{h} \mathbf{h}^\top] (\nabla_{\gamma s})$$

we use that

$$\mathcal{E}_{\mathbf{w}} [\mathbf{h} \mathbf{h}^\top] = \mathcal{E}_{\mathbf{w}} [\mathbf{h}] \mathcal{E}_{\mathbf{w}} [\mathbf{h}]^\top + \text{Cov}(\mathbf{h}),$$

where the covariance is diagonal since the precisions are independent. We find (after a somewhat lengthy calculation) that

$$\begin{aligned} \mathcal{E}_{\mathbf{w}} [h_i] &= \frac{a-1}{\gamma_i} - b, \\ \mathcal{E}_{\mathbf{w}} [(h_i - \mathcal{E}_{\mathbf{z}}[h_i])^2] &= \frac{p+q}{2\gamma_i^2}. \end{aligned}$$

We see that $\mathcal{E}_{\mathbf{w}} [h_i] = 0$ when γ is deterministic. So

$$\begin{aligned} G_{\gamma\gamma} &= ((\nabla_{\gamma s})^\top (a\gamma^{-1} - b\mathbf{1}_r))^2 + \frac{p+q+2(a-1)}{2} (\nabla_{\gamma s})^\top \mathbf{\Gamma}^{-2} (\nabla_{\gamma s}) \\ &= \frac{p+q}{2} (\nabla_{\gamma s})^\top \mathbf{\Gamma}^{-2} (\nabla_{\gamma s}). \end{aligned}$$

7.4.8 Proof of Proposition 7.3.6

The bound BCRB-III can be computed from BCRB-II by taking the appropriate expectation values with respect to γ and β . Using that

$$\begin{aligned} \mathcal{E}_{\beta} [\beta] &= \frac{c}{d}, \quad \mathcal{E}_{\beta} [\beta^{-2}] = \frac{d^2}{(c-1)(c-2)}, \\ \mathcal{E}_{\gamma} [\gamma_i^{-k}] &= b^k \frac{\Gamma(a-k)}{\Gamma(a)} = \frac{b^k}{a \dots (a-k)}, \quad \text{for } k \geq 1, \end{aligned}$$

we are able to compute the respective expectation values.

7.4.9 Proof of Proposition 7.3.7

Proposition 7.3.7 gives the Fisher information matrix for the RSVM model. We have that

$$\begin{aligned} \log p(\mathbf{y}, \mathbf{z}) &= -\frac{\beta}{2} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \frac{m + 2(c-1)}{2} \log \beta \\ &\quad - d\beta - \frac{1}{2} \text{tr}(\boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R \mathbf{X}^\top) + \frac{q + \nu_L}{2} \log |\boldsymbol{\alpha}_L| \\ &\quad + \frac{p + \nu_R}{2} \log |\boldsymbol{\alpha}_R| - \frac{\epsilon}{2} \text{tr}(\boldsymbol{\alpha}_L) - \frac{\epsilon}{2} \text{tr}(\boldsymbol{\alpha}_R) + \text{const}, \end{aligned}$$

where $\mathbf{x} = \text{vec}(\mathbf{X})$. We find that

$$\begin{aligned} \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \mathbf{x}} &= \beta \mathbf{A}^\top (\mathbf{y} - \mathbf{A}\mathbf{x}) - \text{vec}(\boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R), \\ \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \text{vec}(\boldsymbol{\alpha}_L)} &= -\frac{1}{2} \mathbf{D}_p \text{vec}(\mathbf{X} \boldsymbol{\alpha}_R \mathbf{X}^\top) + \\ &\quad \frac{q + \nu_L}{2} \mathbf{D}_p \text{vec}(\boldsymbol{\alpha}_L^{-1}) - \frac{\epsilon}{2} \mathbf{D}_p \text{vec}(\mathbf{I}_p) = -\mathbf{v}_L, \\ \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \text{vec}(\boldsymbol{\alpha}_R)} &= -\frac{1}{2} \mathbf{D}_q \text{vec}(\mathbf{X}^\top \boldsymbol{\alpha}_L \mathbf{X}) + \\ &\quad \frac{p + \nu_R}{2} \mathbf{D}_q \text{vec}(\boldsymbol{\alpha}_R^{-1}) - \frac{\epsilon}{2} \mathbf{D}_q \text{vec}(\mathbf{I}_q) = -\mathbf{v}_R, \\ \frac{\partial \log p(\mathbf{y}, \mathbf{z})}{\partial \beta} &= -\frac{1}{2} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2^2 + \frac{m + 2(c-1)}{2\beta} - d. \end{aligned}$$

This gives us that the Fisher information is

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{\mathbf{xx}} & \mathbf{F}_{\mathbf{x}\boldsymbol{\alpha}_L} & \mathbf{F}_{\mathbf{x}\boldsymbol{\alpha}_R} & \mathbf{F}_{\mathbf{x}\beta} \\ \mathbf{F}_{\mathbf{x}\boldsymbol{\alpha}_L}^\top & \mathbf{F}_{\boldsymbol{\alpha}_L\boldsymbol{\alpha}_L} & \mathbf{F}_{\boldsymbol{\alpha}_L\boldsymbol{\alpha}_R} & \mathbf{F}_{\boldsymbol{\alpha}_L\beta} \\ \mathbf{F}_{\mathbf{x}\boldsymbol{\alpha}_R}^\top & \mathbf{F}_{\boldsymbol{\alpha}_L\boldsymbol{\alpha}_R}^\top & \mathbf{F}_{\boldsymbol{\alpha}_R\boldsymbol{\alpha}_R} & \mathbf{F}_{\boldsymbol{\alpha}_R\beta} \\ \mathbf{F}_{\mathbf{x}\beta}^\top & \mathbf{F}_{\boldsymbol{\alpha}_L\beta}^\top & \mathbf{F}_{\boldsymbol{\alpha}_R\beta} & F_{\beta\beta} \end{bmatrix},$$

where

$$\begin{aligned} \mathbf{F}_{\mathbf{xx}} &= \beta \mathbf{A}^\top \mathbf{A} + \text{vec}(\boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R) \text{vec}(\boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R)^\top, \\ \mathbf{F}_{\mathbf{x}\boldsymbol{\alpha}_L} &= \text{vec}(\boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R) \mathbf{v}_L^\top, \quad \mathbf{F}_{\mathbf{x}\boldsymbol{\alpha}_R} = \text{vec}(\boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R) \mathbf{v}_R^\top, \\ \mathbf{F}_{\boldsymbol{\alpha}_L\boldsymbol{\alpha}_L} &= \mathbf{v}_L \mathbf{v}_L^\top, \quad \mathbf{F}_{\boldsymbol{\alpha}_L\boldsymbol{\alpha}_R} = \mathbf{v}_L \mathbf{v}_R^\top, \quad \mathbf{F}_{\boldsymbol{\alpha}_R\boldsymbol{\alpha}_R} = \mathbf{v}_R \mathbf{v}_R^\top, \\ \mathbf{F}_{\mathbf{x}\beta} &= \left(d - \frac{c-1}{\beta} \right) \text{vec}(\boldsymbol{\alpha}_L \mathbf{X} \boldsymbol{\alpha}_R), \\ \mathbf{F}_{\boldsymbol{\alpha}_L\beta} &= \left(d - \frac{c-1}{\beta} \right) \mathbf{v}_L, \quad \mathbf{F}_{\boldsymbol{\alpha}_R\beta} = \left(d - \frac{c-1}{\beta} \right) \mathbf{v}_R, \\ F_{\beta\beta} &= \frac{m}{2\beta^2} + \left(d - \frac{c-1}{\beta} \right)^2. \end{aligned}$$

We find that the terms $\mathbf{F}_{\mathbf{x}\alpha_L}$ and $\mathbf{F}_{\mathbf{x}\alpha_R}$ zero mean with respect to \mathbf{x} and that the terms $\mathbf{F}_{\mathbf{x}\beta}$, $\mathbf{F}_{\alpha_L\beta}$ and $\mathbf{F}_{\alpha_R\beta}$ are zero when β is deterministic and zero-mean when β is random. The terms do therefore not contribute to the bounds.

7.4.10 Proof of Proposition 7.3.8

Proposition 7.3.8 gives the BCRB-I and BCRB-II for the RSVM model.

We get that

$$\begin{aligned} \mathcal{E}_{\mathbf{x}} [\text{vec}(\alpha_L \mathbf{X} \alpha_R) \text{vec}(\alpha_L \mathbf{X} \alpha_R)^\top] = \\ (\alpha_R \otimes \alpha_L)(\alpha_R \otimes \alpha_L)^{-1}(\alpha_R \otimes \alpha_L) = (\alpha_R \otimes \alpha_L). \end{aligned}$$

Using the Einstein summation convention (repeated indices are summed over) and denoting $\alpha_{R,ab} = [\alpha_R]_{ab}$, $\alpha_{R,ab}^{-1} = [\alpha_R^{-1}]_{ab}$ for brevity (similarly for α_L) we find that

$$\begin{aligned} \mathcal{E}_{\mathbf{x}} [\mathbf{X} \alpha_R \mathbf{X}^\top] = \mathbf{e}_i \mathbf{e}_j^\top \alpha_{R,ab} \mathcal{E}_{\mathbf{x}} [X_{ia} X_{jb}] \\ = \mathbf{e}_i \mathbf{e}_j^\top \alpha_{R,ab} \alpha_{L,ij}^{-1} \alpha_{R,ab}^{-1} = \mathbf{e}_i \mathbf{e}_j^\top \delta_{aa} \alpha_{L,ij}^{-1} = q \alpha_L^{-1}, \end{aligned}$$

and similarly $\mathcal{E}_{\mathbf{x}} [\mathbf{X}^\top \alpha_L \mathbf{X}] = p \alpha_R^{-1}$. We also find that

$$\begin{aligned} \mathcal{E}_{\mathbf{x}} [\text{vec}(\mathbf{X} \alpha_R \mathbf{X}^\top) \text{vec}(\mathbf{X} \alpha_R \mathbf{X}^\top)^\top] = \\ (\mathbf{e}_i \otimes \mathbf{e}_j)(\mathbf{e}_k \otimes \mathbf{e}_l)^\top \alpha_{R,ab} \alpha_{R,cd} \mathcal{E}_{\mathbf{x}} [X_{ia} X_{jb} X_{kc} X_{ld}] = \\ (\mathbf{e}_i \otimes \mathbf{e}_j)(\mathbf{e}_k \otimes \mathbf{e}_l)^\top \alpha_{R,ab} \alpha_{R,cd} \left(\alpha_{L,ij}^{-1} \alpha_{R,ab}^{-1} \alpha_{L,kl}^{-1} \alpha_{R,cd}^{-1} \right. \\ \left. + \alpha_{L,ik}^{-1} \alpha_{R,ac}^{-1} \alpha_{L,jl}^{-1} \alpha_{R,bd}^{-1} + \alpha_{L,il}^{-1} \alpha_{R,ad}^{-1} \alpha_{L,jk}^{-1} \alpha_{R,bc}^{-1} \right) = \\ (\mathbf{e}_i \otimes \mathbf{e}_j)(\mathbf{e}_k \otimes \mathbf{e}_l)^\top \left(q^2 \alpha_{L,ij}^{-1} \alpha_{L,kl}^{-1} + q \alpha_{L,ik}^{-1} \alpha_{L,jl}^{-1} \right. \\ \left. + q \alpha_{L,il}^{-1} \alpha_{L,jk}^{-1} \right) = q^2 \text{vec}(\alpha_L^{-1}) \text{vec}(\alpha_L^{-1})^\top \\ + q(\alpha_L^{-1} \otimes \alpha_L^{-1}) + q \mathbf{K}_{p,p}(\alpha_L^{-1} \otimes \alpha_L^{-1}), \end{aligned}$$

In the same fashion one can show that

$$\begin{aligned} \mathcal{E}_{\mathbf{x}} [\text{vec}(\mathbf{X} \alpha_R \mathbf{X}^\top) \text{vec}(\mathbf{X}^\top \alpha_L \mathbf{X})^\top] = \\ (pq + 2) \text{vec}(\alpha_L^{-1}) \text{vec}(\alpha_R^{-1}), \\ \mathcal{E}_{\mathbf{x}} [\text{vec}(\mathbf{X}^\top \alpha_L \mathbf{X}) \text{vec}(\mathbf{X}^\top \alpha_L \mathbf{X})^\top] = p^2 \text{vec}(\alpha_R^{-1}) \text{vec}(\alpha_R^{-1})^\top \\ + p(\alpha_R^{-1} \otimes \alpha_R^{-1}) + p \mathbf{K}_{q,q}(\alpha_R^{-1} \otimes \alpha_R^{-1}) \end{aligned}$$

We thus get that

$$\mathcal{E}_{\mathbf{x}} [\mathbf{F}_{\mathbf{xx}}] = (\alpha_R \otimes \alpha_L) + \beta \mathbf{A}^\top \mathbf{A},$$

and

$$\begin{aligned}
\mathcal{E}_{\mathbf{x}} [\mathbf{v}_L \mathbf{v}_L^\top] &= \frac{1}{4} \mathbf{D}_p \text{vec}(\epsilon \mathbf{I}_p - \nu_L \boldsymbol{\alpha}_L^{-1}) \text{vec}(\epsilon \mathbf{I}_p - \nu_L \boldsymbol{\alpha}_L^{-1})^\top \mathbf{D}_p \\
&\quad + \frac{q}{4} \mathbf{D}_p (\mathbf{I}_{p^2} + \mathbf{K}_{p,p}) (\boldsymbol{\alpha}_L^{-1} \otimes \boldsymbol{\alpha}_L^{-1}) \mathbf{D}_p, \\
\mathcal{E}_{\mathbf{x}} [\mathbf{v}_R \mathbf{v}_R^\top] &= \frac{1}{4} \mathbf{D}_q \text{vec}(\epsilon \mathbf{I}_q - \nu_R \boldsymbol{\alpha}_R^{-1}) \text{vec}(\epsilon \mathbf{I}_q - \nu_R \boldsymbol{\alpha}_R^{-1})^\top \mathbf{D}_q^\top \\
&\quad + \frac{p}{4} \mathbf{D}_q (\mathbf{I}_{q^2} + \mathbf{K}_{q,q}) (\boldsymbol{\alpha}_R^{-1} \otimes \boldsymbol{\alpha}_R^{-1}) \mathbf{D}_q, \\
\mathcal{E}_{\mathbf{x}} [\mathbf{v}_L \mathbf{v}_R^\top] &= \frac{1}{4} \mathbf{D}_p \text{vec}(\epsilon \mathbf{I}_p - \nu_L \boldsymbol{\alpha}_L^{-1}) \text{vec}(\epsilon \mathbf{I}_q - \nu_R \boldsymbol{\alpha}_R^{-1})^\top \mathbf{D}_q \\
&\quad + \frac{1}{2} \mathbf{D}_p \text{vec}(\boldsymbol{\alpha}_L^{-1}) \text{vec}(\boldsymbol{\alpha}_R^{-1})^\top \mathbf{D}_q.
\end{aligned}$$

We also find that $\mathcal{E}_{\mathbf{x}} [\mathbf{F}_{\mathbf{x}\boldsymbol{\alpha}_L}] = \mathbf{0}$ and $\mathcal{E}_{\mathbf{x}} [\mathbf{F}_{\mathbf{x}\boldsymbol{\alpha}_R}] = \mathbf{0}$ since \mathbf{x} has zero mean. The terms $\mathbf{F}_{\mathbf{x}\beta}$, $\mathbf{F}_{\boldsymbol{\alpha}_L\beta}$ and $\mathbf{F}_{\boldsymbol{\alpha}_R\beta}$ are zero since β is deterministic.

7.4.11 Proof of Proposition 7.3.9

We have that

$$\begin{aligned}
\mathcal{E}_{\mathbf{z}} [\boldsymbol{\alpha}_L] &= (\nu_L + p + 1) \epsilon^{-1} \mathbf{I}_q, \\
\mathcal{E}_{\mathbf{z}} [\boldsymbol{\alpha}_L^{-1}] &= \nu_L^{-1} \epsilon \mathbf{I}_q.
\end{aligned}$$

From [vR88, KvR06] we get that

$$\begin{aligned}
\mathcal{E}_{\mathbf{z}} [(\boldsymbol{\alpha}_L^{-1} \otimes \boldsymbol{\alpha}_L^{-1})] &= c_L \epsilon^2 \mathbf{I}_{p^2} \\
&\quad + d_L \epsilon^2 \text{vec}(\mathbf{I}_p) \text{vec}(\mathbf{I}_p)^\top + d_L \epsilon^2 \mathbf{K}_{p,p}, \\
\mathcal{E}_{\mathbf{z}} [\text{vec}(\boldsymbol{\alpha}_L^{-1}) \text{vec}(\boldsymbol{\alpha}_L^{-1})^\top] &= c_L \epsilon^2 \text{vec}(\mathbf{I}_p) \text{vec}(\mathbf{I}_p)^\top \\
&\quad + d_L \epsilon^2 \mathbf{I}_{p^2} + d_L \epsilon^2 \tilde{\mathbf{K}}_{p,p}
\end{aligned}$$

where $c_L = (\nu_L - 1)d_L$, $d_L = 1/((\nu_L + 1)\nu_L(\nu_L - 2))$. Similar expressions can easily be found for $\boldsymbol{\alpha}_R$.

This gives us that

$$\mathcal{E}_{\mathbf{z}} [\mathbf{F}_{\mathbf{xx}}] = (\nu_L + p + 1)(\nu_R + q + 1) \epsilon^{-2} \mathbf{I}_{pq} + \frac{1+c}{d} \mathbf{A}^\top \mathbf{A},$$

and

$$\begin{aligned}
\mathcal{E}_{\mathbf{z}} [\mathbf{v}_L \mathbf{v}_L^\top] &= \frac{1}{4} \mathbf{D}_p ((\nu_L^2 c_L - 1) \text{vec}(\mathbf{I}_p) \text{vec}(\mathbf{I}_p)^\top \\
&\quad + \nu_L^2 \epsilon^2 d_L \mathbf{I}_{p^2} + \nu_L^2 \epsilon^2 \tilde{\mathbf{K}}_{p,p}) \mathbf{D}_p + \\
&\quad \frac{q}{4} \mathbf{D}_p (\mathbf{I}_{p^2} + \mathbf{K}_{p,p}) (c_L \epsilon^2 \mathbf{I}_{p^2} + d_L \epsilon^2 \text{vec}(\mathbf{I}_p) \text{vec}(\mathbf{I}_p)^\top
\end{aligned}$$

$$\begin{aligned}
& +d_L\epsilon^2\mathbf{K}_{p,p})\mathbf{D}_p = \frac{\epsilon^2}{4}(\nu_L^2c_L - 1 + 2qd_L)\text{vec}(\mathbf{I}_p)\text{vec}(\mathbf{I}_p)^\top \\
& + \frac{\epsilon^2}{4}(\nu_L^2d_L + c_L + d_L)\mathbf{D}_p^2 \\
& + \frac{\epsilon^2q}{4}(c_L + d_L)\mathbf{D}_p\mathbf{K}_{p,p}\mathbf{D}_p + \frac{\epsilon^2\nu_L^2d_L}{4}\mathbf{D}_p\tilde{\mathbf{K}}_{p,p}\mathbf{D}_p, \\
\mathcal{E}_z[\mathbf{v}_L\mathbf{v}_R^\top] & = \frac{\epsilon^2}{2\nu_L\nu_R}\text{vec}(\mathbf{I}_p)\text{vec}(\mathbf{I}_q)^\top,
\end{aligned}$$

and similarly for $\mathcal{E}[\mathbf{v}_R\mathbf{v}_R^\top]$.

7.4.12 Proof of Proposition 7.3.10

In the BCRB-IV bound, the variable \mathbf{X} is marginalized and the precisions α_L , α_R and β are deterministic. The only random variable is thus \mathbf{y} . We find that

$$\begin{aligned}
\log p & = \log p(\mathbf{y}|\alpha_L, \alpha_R, \beta) \\
& = -\frac{1}{2}\log|\mathbf{C}| - \frac{1}{2}\mathbf{y}^\top\mathbf{C}^{-1}\mathbf{y} + \text{const},
\end{aligned}$$

where

$$\mathbf{C} = \mathbf{A}(\alpha_R^{-1} \otimes \alpha_L^{-1})\mathbf{A}^\top + \beta^{-1}\mathbf{I}_m.$$

Recalling the definition of \mathbf{E}_{ij}^L and \mathbf{E}_{kl}^R from Section 7.1.1 and setting $\tilde{\mathbf{A}} = \mathbf{A}(\alpha_R^{-1} \otimes \alpha_L^{-1})$, we get that

$$\begin{aligned}
\frac{\partial \log p}{\partial[\alpha_L]_{ij}} & = \frac{1}{2}\text{tr}(\mathbf{C}^{-1}\tilde{\mathbf{A}}(\alpha_R \otimes \mathbf{E}_{ij}^L)\tilde{\mathbf{A}}^\top) \\
& \quad - \frac{1}{2}\mathbf{y}^\top\mathbf{C}^{-1}\tilde{\mathbf{A}}(\alpha_R \otimes \mathbf{E}_{ij}^L)\tilde{\mathbf{A}}^\top\mathbf{C}^{-1}\mathbf{y}, \\
\frac{\partial \log p}{\partial[\alpha_R]_{kl}} & = \frac{1}{2}\text{tr}(\mathbf{C}^{-1}\tilde{\mathbf{A}}(\mathbf{E}_{kl}^R \otimes \alpha_L)\tilde{\mathbf{A}}^\top) \\
& \quad - \frac{1}{2}\mathbf{y}^\top\mathbf{C}^{-1}\tilde{\mathbf{A}}(\mathbf{E}_{kl}^R \otimes \alpha_L)\tilde{\mathbf{A}}^\top\mathbf{C}^{-1}\mathbf{y}, \\
\frac{\partial \log p}{\partial \beta} & = \frac{1}{2\beta^2}\text{tr}(\mathbf{C}^{-1}) - \frac{1}{2\beta^2}\mathbf{y}^\top\mathbf{C}^{-2}\mathbf{y}.
\end{aligned}$$

A useful identity when computing the Fisher information matrix is that if $\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$ and \mathbf{A} and \mathbf{B} are symmetric matrices, then

$$\begin{aligned}
\mathcal{E}_y[\mathbf{y}^\top\mathbf{A}\mathbf{y}] & = \text{tr}(\mathbf{A}\mathcal{E}_y[\mathbf{y}\mathbf{y}^\top]) = \text{tr}(\mathbf{A}\mathbf{C}), \\
\mathcal{E}_y[\mathbf{y}^\top\mathbf{A}\mathbf{y}\mathbf{y}^\top\mathbf{B}\mathbf{y}] & = \text{tr}(\mathbf{A}\mathbf{C})\text{tr}(\mathbf{B}\mathbf{C}) + 2\text{tr}(\mathbf{A}\mathbf{C}\mathbf{B}\mathbf{C}).
\end{aligned}$$

Setting and $\mathbf{P} = \tilde{\mathbf{A}}^\top \mathbf{C}^{-1} \tilde{\mathbf{A}}$, we find that the Fisher information matrix becomes

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{\alpha_L \alpha_L} & \mathbf{F}_{\alpha_L \alpha_R} & \mathbf{F}_{\alpha_L \beta} \\ \mathbf{F}_{\alpha_L \alpha_R}^\top & \mathbf{F}_{\alpha_R \alpha_R} & \mathbf{F}_{\alpha_R \beta} \\ \mathbf{F}_{\alpha_L \beta}^\top & \mathbf{F}_{\alpha_R \beta}^\top & F_{\beta\beta} \end{bmatrix},$$

where

$$\begin{aligned} [\mathbf{F}_{\alpha_L \alpha_L}]_{i+(j-1)p, k+(l-1)p} &= \mathcal{E}_{\mathbf{y}} \left[\frac{\partial \log p}{\partial [\alpha_L]_{ij}} \frac{\partial \log p}{\partial [\alpha_L]_{kl}} \right] \\ &= -\frac{1}{4} \text{tr}(\mathbf{P}(\alpha_R \otimes \mathbf{E}_{ij}^L)) \text{tr}(\mathbf{P}(\alpha_R \otimes \mathbf{E}_{kl}^L)) \\ &\quad + \frac{1}{4} \mathcal{E}_{\mathbf{y}} \left[\mathbf{y}^\top \mathbf{C}^{-1} \tilde{\mathbf{A}}(\alpha_R \otimes \mathbf{E}_{ij}^L) \tilde{\mathbf{A}}^\top \mathbf{C}^{-1} \mathbf{y} \cdot \right. \\ &\quad \left. \mathbf{y}^\top \mathbf{C}^{-1} \tilde{\mathbf{A}}(\alpha_R \otimes \mathbf{E}_{kl}^L) \tilde{\mathbf{A}}^\top \mathbf{C}^{-1} \mathbf{y} \right] \\ &= \frac{1}{4} \text{tr}(\mathbf{P}(\alpha_R \otimes \mathbf{E}_{ij}^L) \mathbf{P}(\alpha_R \otimes \mathbf{E}_{kl}^L)). \end{aligned}$$

In a similar way we find that

$$\begin{aligned} [\mathbf{F}_{\alpha_L \alpha_R}]_{i+(j-1)p, k+(l-1)q} &= \mathcal{E}_{\mathbf{y}} \left[\frac{\partial \log p}{\partial [\alpha_L]_{ij}} \frac{\partial \log p}{\partial [\alpha_R]_{kl}} \right] \\ &= \frac{1}{4} \text{tr}(\mathbf{P}(\alpha_R \otimes \mathbf{E}_{ij}^L) \mathbf{P}(\mathbf{E}_{kl}^R \otimes \alpha_L)), \\ [\mathbf{F}_{\alpha_R \alpha_R}]_{i+(j-1)q, k+(l-1)q} &= \mathcal{E}_{\mathbf{y}} \left[\frac{\partial \log p}{\partial [\alpha_R]_{ij}} \frac{\partial \log p}{\partial [\alpha_R]_{kl}} \right] \\ &= \frac{1}{4} \text{tr}(\mathbf{P}(\mathbf{E}_{ij}^R \otimes \alpha_L) \mathbf{P}(\mathbf{E}_{kl}^R \otimes \alpha_L)), \\ [\mathbf{F}_{L\beta}]_{i+(j-1)p} &= \mathcal{E}_{\mathbf{y}} \left[\frac{\partial \log p}{\partial [\alpha_L]_{ij}} \frac{\partial \log p}{\partial \beta} \right] \\ &= \frac{1}{4\beta^2} \text{tr}(\tilde{\mathbf{A}}^\top \mathbf{C}^{-2} \tilde{\mathbf{A}}(\alpha_R \otimes \mathbf{E}_{ij}^L)), \\ [\mathbf{F}_{\alpha_R \beta}]_{k+(l-1)q} &= \mathcal{E}_{\mathbf{y}} \left[\frac{\partial \log p}{\partial [\alpha_R]_{kl}} \frac{\partial \log p}{\partial \beta} \right] \\ &= \frac{1}{4\beta^2} \text{tr}(\tilde{\mathbf{A}}^\top \mathbf{C}^{-2} \tilde{\mathbf{A}}(\mathbf{E}_{kl}^R \otimes \alpha_L)), \\ F_{\beta\beta} &= \mathcal{E}_{\mathbf{y}} \left[\frac{\partial \log p}{\partial \beta} \frac{\partial \log p}{\partial \beta} \right] \\ &= \frac{1}{4\beta^4} \text{tr}(\mathbf{C}^{-2}). \end{aligned}$$

These terms gives us the BCRB-IV for the RSVM model.

7.5 Conclusion

In this chapter we derived Bayesian Cramér-Rao bounds for low-rank matrix completion. We considered a sparsity-based model, a factorized model and the hierarchical RSVM model. We compute the BCRB bounds for all models and simulated them numerically. We found that the random low-rank matrix models does give realizations of low numerical rank for appropriate parameter values. We also found that there exists a considerable gap between the lower bounds of the BCRB's and the performance of existing algorithms. This indicates that there is room for designing better estimation algorithms and/or tighter theoretical bounds.

Low-rank phase retrieval

When measurements are non-linear, it is often much more difficult to estimate parameters. There are, however, some problems which are *almost linear*. This means that the non-linear problem can be transformed into a linear problem. Phase retrieval is one problem which can be transformed to a linear problem, the transformation of the problem is often called PhaseLift. In phase retrieval we seek to estimate the parameters of a vector by only measuring the amplitudes of measurements. PhaseLift has been modified to exploit sparsity in the parameter vector. However, it is unknown how to exploit low-rank properties of the parameters. In this chapter we show how the phase retrieval problem can be adapted to non-linear measurements of low-rank matrices and how the low-rank matrices can be recovered using convex optimization techniques.

8.1 Introduction

In the phase retrieval model, a vector $\mathbf{x} \in \mathbb{C}^n$ (or $\mathbf{x} \in \mathbb{R}^n$) is measured as

$$y_i = |\mathbf{a}_i^\top \mathbf{x}|^2 + n_i \quad (8.1)$$

where $\mathbf{a}_i \in \mathbb{C}^n$ (or $\mathbf{a}_i \in \mathbb{R}^n$) represents the measurement process, $n_i \in \mathbb{R}$ is additive noise and $i = 1, 2, \dots, m$. The process (8.1) only register the amplitude of the measurements and not the phase. Recovering \mathbf{x} is equivalent to recovering the phase's of the measurements, the problem is therefore often called *phase retrieval*. As the problem with complex coefficients can be expressed using real variables, we will from now on only consider the real scenario where $\mathbf{x} \in \mathbb{R}^n$. The problem can be found in many applications, e.g. X-ray crystallography [Fie78, Har93, Mil90], speckle imaging [RCLV13] and blind channel estimation [RCLV13] where the phase is lost during the measurement process.

8.1.1 Prior work

One of the origins of phase retrieval is X-ray crystallography [Har93]. In X-ray crystallography, a crystal sample is exposed to X-ray radiation. When the radiation hits the sample, the radiation is split up in different directions after hitting the crystal. By measuring the intensity of the resulting beams (and not the phase), one can register the diffraction pattern. From the diffraction pattern one can then reconstruct the electron density map of the sample and thus the crystal structure. The classical methods for reconstructing the crystal are the Gerchberg-Saxton [Ger72] and the Fienup methods [Fie78] which iteratively estimates the amplitudes and the phases.

Recently much work has been done on reformulating the phase retrieval problem as a rank minimization problem [CESV15, RCLV13, EM14, OE14, CCG15]. The method *lifts* the non-linearity in the problem by noting that

$$|\mathbf{a}_i^\top \mathbf{x}|^2 = \text{tr}(\mathbf{a}_i \mathbf{a}_i^* \mathbf{x} \mathbf{x}^*) = \text{tr}(\mathbf{a}_i \mathbf{a}_i^* \mathbf{Z}) = \mathcal{A}(\mathbf{Z})_i,$$

where $\mathbf{Z} = \mathbf{x} \mathbf{x}^*$, $\mathcal{A} : \mathbb{C}^{n \times n} \rightarrow \mathbb{R}^m$ is a known linear operator and $\mathcal{A}(\mathbf{Z})_i$ is the i 'th element of $\mathcal{A}(\mathbf{Z}) \in \mathbb{R}^m$. Changing variable from \mathbf{x} to \mathbf{Z} transforms the problem from a non-linear problem of finding an n -dimensional vector to finding an $n \times n$ positive definite matrix \mathbf{Z} of rank one. Given that the noise is bounded as $\|\mathbf{n}\|_1 \leq \eta$, the phase retrieval problem can be written as

$$\begin{aligned} \min \quad & \text{rank}(\mathbf{Z}), \\ \text{subject to} \quad & \|\mathbf{y} - \mathcal{A}(\mathbf{Z})\|_1 \leq \eta, \mathbf{Z} \succeq \mathbf{0}, \end{aligned} \quad (8.2)$$

A correct estimate is then such that $\text{rank}(\hat{\mathbf{Z}}) = 1$. Since the rank function is hard to minimize, a common approach is to make a convex relaxation of the problem, the rank is replaced by the trace [CESV15, CCG15]. The PhaseLift program is thus

$$\begin{aligned} \min \quad & \text{tr}(\mathbf{Z}), \\ \text{subject to} \quad & \|\mathbf{y} - \mathcal{A}(\mathbf{Z})\|_1 \leq \eta, \mathbf{Z} \succeq \mathbf{0} \end{aligned} \quad (\text{PhaseLift})$$

Another method based on convex relaxation is *PhaseCut* [WdM15] for which the unknown phases transforms into a new optimization variable. PhaseLift has the advantage over PhaseCut that it is easily modified to sparse vectors. This can be done by using ℓ_1 -norm regularization [OYDS11] to make \mathbf{Z} more sparse. The approach uses the fact that

$$\|\mathbf{x} \mathbf{x}^*\|_0 = \|\mathbf{x}\|_0^2, \quad \|\mathbf{x} \mathbf{x}^*\|_1 = \|\mathbf{x}\|_1^2.$$

Uniqueness conditions for the phase retrieval problem have been investigated in [LV11, RCLV13, EM14, OE14, BCMN14, CESV15] while recovery conditions for the ℓ_1 -penalized PhaseLift have been established in e.g. [CCG15, OE14].

To construct PhaseLift for low-rank matrices, the rank and the nuclear norm need to be lifted in a similar way as sparsity and the ℓ_1 -norm. However, unlike

sparsity, which is a component wise property, rank is harder to lift since if $\mathbf{x} = \text{vec}(\mathbf{X})$ for $\mathbf{X} \in \mathbb{C}^{p \times q}$, then

$$\begin{aligned} \text{rank}(\text{vec}(\mathbf{X})\text{vec}(\mathbf{X})^H) &\neq \text{rank}(\mathbf{X})^2, \\ \|\text{vec}(\mathbf{X})\text{vec}(\mathbf{X})^H\|_* &\neq \|\mathbf{X}\|_*^2. \end{aligned}$$

8.2 The low-rank phase retrieval problem

In the original phase retrieval problem, nothing is assumed about the parameter vector \mathbf{x} . In X-ray crystallography, this amounts to modeling the crystal as an unknown density of atoms. Since a crystal largely consists of empty space, sparsity can be used to recover the crystal from less measurements. The sparsity based approach models the crystal as a small collection of points. However, both the standard and sparse phase retrieval methods ignore the fact that the atoms in a crystal are arranged in a crystal structure. To further improve the phase retrieval methods for X-ray crystallography, it is desired to utilize the crystal structure. We here concentrate on two-dimensional crystals and discuss the extension to higher dimensions in Section 8.4.3.

Figure 8.1 shows a honeycomb lattice in a 64×81 sparse matrix. The non-zero elements equal one and are marked by black dots. The matrix has 420 non-zero elements out of 4560, i.e. the sparsity is $\approx 8.1\%$. The rank of the matrix is 2, i.e. the rank-to-size ratio is $2/\min(81, 64) \approx 3.1\%$. When the crystal size grows to infinity, the sparsity converges to 5.5% while the rank-to-size ratio converges to zero.

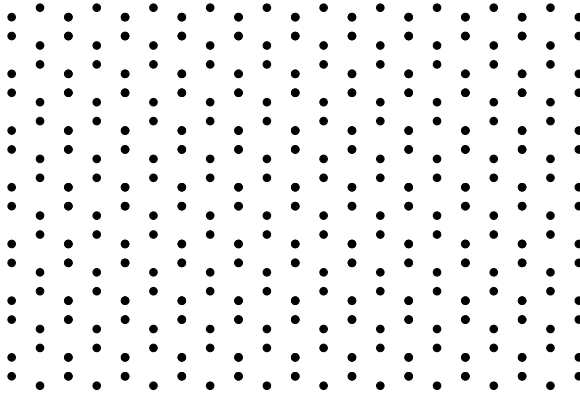


Figure 8.1: Sparse 64×81 matrix representing a honeycomb lattice. The dots mark the position of ones. The matrix has 210 non-zero entries (out of 4560) and rank 2.

In the low rank phase retrieval problem we consider the case where $\mathbf{x} = \text{vec}(\mathbf{X})$,

$\mathbf{X} \in \mathbb{R}^{p \times q}$ and $\text{rank}(\mathbf{X}) \ll \min(p, q)$. The low-rank phase retrieval problem is thus

$$\begin{aligned} & \min \text{rank}(\mathbf{X}) \\ & \text{subject to } \|\mathbf{y} - |\mathbf{A}\text{vec}(\mathbf{X})|^2\|_1 \leq \eta \end{aligned} \quad (8.3)$$

We notice the similarity between (8.3) and (8.2). However, in (8.2) the measurements are linear in \mathbf{Z} while in (8.3) they are non-linear in \mathbf{X} . The question arises of when the solution to (8.3) is unique. By extending the results of [EM14] we find the following probabilistic uniqueness condition.

Proposition 8.2.1. *Assume that the measurements are noise free ($\eta = 0$), that the components of $\mathbf{A} \in \mathbb{R}^{m \times pq}$ are i.i.d. zero-mean Gaussian distributed and that $\text{rank}(\mathbf{X}) = r \leq \min(p, q)/2$. Then the solution to the low-rank phase retrieval problem (8.3) is unique (and equals $\pm \mathbf{X}$) with probability at least $1 - 2 \exp(-cu^2r(p + q - 2r))$, if*

$$m \geq Cu^{3/2}r(p + q - 2r),$$

where c, C and u are positive constants.

The proof of Proposition 8.2.1 is given in Section 8.7.1. The proposition gives us that uniqueness can be guaranteed with high probability when the number of measurements is approximately proportional to the number of degrees of freedom

$$\text{degrees of freedom} = r(p + q - r).$$

We now turn to extending PhaseLift to use the low-rank property in (8.3).

8.3 Low-rank PhaseLift

To promote low-rank we need to construct a penalty which acts on the *lifted* variable \mathbf{Z} while promoting low-rank in \mathbf{X} . It turns out that one approach can be found using the theory of Kronecker product approximation.

8.3.1 Lifting the rank

In the Kronecker approximation problem we search to approximate a matrix $\mathbf{B} \in \mathbb{R}^{p_1 p_2 \times q_1 q_2}$ by the Kronecker product $(\mathbf{C} \otimes \mathbf{D})$ of two matrices, $\mathbf{C} \in \mathbb{R}^{p_1 \times q_1}$ and $\mathbf{D} \in \mathbb{R}^{p_2 \times q_2}$. The least square approximation problem is

$$\min_{(\mathbf{C}, \mathbf{D})} \|\mathbf{B} - (\mathbf{C} \otimes \mathbf{D})\|_F. \quad (8.4)$$

The problem has an algebraic solution which uses a linear transformation [VLP93] $\mathcal{R} : \mathbb{R}^{p_1 p_2 \times q_1 q_2} \rightarrow \mathbb{R}^{p_1 q_1 \times p_2 q_2}$ such that

$$\mathcal{R}(\mathbf{C} \otimes \mathbf{D}) = \text{vec}(\mathbf{C})\text{vec}(\mathbf{D})^\top.$$

The linear transformation \mathcal{R} is invertible, giving us that if $\tilde{\mathbf{B}}$ is the best rank-1 approximation of $\mathcal{R}(\mathbf{B})$ then the solution to (8.4) is

$$(\hat{\mathbf{C}} \otimes \hat{\mathbf{D}}) = \mathcal{R}^{-1}(\tilde{\mathbf{B}}).$$

The inverse transformation, \mathcal{R}^{-1} , is the key to lifting the rank property. If $\mathbf{Z} = \text{vec}(\mathbf{X})\text{vec}(\mathbf{X})^\top$, then $\mathcal{R}^{-1}(\mathbf{Z}) = (\mathbf{X} \otimes \mathbf{X})$ and

$$\text{rank}(\mathcal{R}^{-1}(\mathbf{Z})) = \text{rank}(\mathbf{X})^2, \quad \|\mathcal{R}^{-1}(\mathbf{Z})\|_* = \|\mathbf{X}\|_*^2.$$

By using the transformation $\mathcal{R}^{-1}(\cdot)$, we can write the low rank phase retrieval problem (8.3) in the variable $\mathbf{Z} = \text{vec}(\mathbf{X})\text{vec}(\mathbf{X})^\top$ as

$$\begin{aligned} & \min \text{rank}(\mathcal{R}^{-1}(\mathbf{Z})), \\ & \text{subject to } \|\mathbf{y} - \mathcal{A}(\mathbf{Z})\|_1 \leq \eta, \quad \mathbf{Z} \succeq \mathbf{0}, \quad \text{rank}(\mathbf{Z}) = 1. \end{aligned}$$

By relaxing the rank functions, the Low-Rank PhaseLift program becomes

$$\begin{aligned} \hat{\mathbf{Z}} &= \arg \min \text{tr}(\mathbf{Z}) + \lambda \|\mathcal{R}^{-1}(\mathbf{Z})\|_*, \\ & \text{subject to } \|\mathbf{y} - \mathcal{A}(\mathbf{Z})\|_1 \leq \eta, \quad \mathbf{Z} \succeq \mathbf{0}, \end{aligned} \tag{8.5}$$

where $\lambda > 0$ is a regularization parameter. We notice that in order to give good recovery, the regularization parameter needs to be chosen such that $\text{rank}(\hat{\mathbf{Z}}) = 1$. In Section 8.3.3 we will discuss a method for selecting λ . We now turn to establishing error bounds for (8.5).

8.3.2 Error bounds for low-rank phase lift

We here establish probabilistic error bounds for (8.5) when the entries of \mathbf{a}_i are random variables. For simplicity we here concentrate on Gaussian measurements although the extension to sub-Gaussian variables is straight forward. We state the main results in this section and give the proofs in Section 9.5. We consider two cases: (a) $\lambda \in \left[\frac{1}{\sqrt{r}}, \frac{\|\mathbf{X}\|_F}{\sqrt{r}\|\mathbf{X}_r\|_*} \right]$ (Theorem 8.1) and (b) $\lambda \rightarrow \infty$ (Theorem 8.2). The case $\lambda = 0$ corresponds to the standard PhaseLift method. The theorems upper bound the error $\|\mathbf{Z} - \hat{\mathbf{Z}}\|_F^2$. The inequality

$$\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 \cdot \|\mathbf{x} + \hat{\mathbf{x}}\|_2^2 \leq 2\|\mathbf{x}\mathbf{x}^\top - \hat{\mathbf{x}}\hat{\mathbf{x}}^\top\|_F^2 = 2\|\mathbf{Z} - \hat{\mathbf{Z}}\|_F^2, \tag{8.6}$$

can be used to give the corresponding error bounds in \mathbf{x} . The inequality (8.6) holds when \mathbf{x} and $\hat{\mathbf{x}}$ are real. For completeness we give a proof of (8.6) in Section 8.7.2. We now give the error bound for λ in a closed interval.

Theorem 8.1. *Assume that the entries of \mathbf{a}_i are i.i.d. Gaussian distributed, $\|\mathbf{n}\|_1 \leq \eta$ and $\mathbf{Z} = \text{vec}(\mathbf{X})\text{vec}(\mathbf{X})^\top = \mathbf{x}\mathbf{x}^\top$. Let \mathbf{X}_r denote the best rank- r approximation of \mathbf{X} and $\mathbf{x}_r = \text{vec}(\mathbf{X}_r)$. If*

$$m > C' \min \{ (p^2 + q^2 + 1)r^2, 2pq + 1 \},$$

then the solution $\hat{\mathbf{Z}}$ of (8.5) with $\lambda \in \left[\frac{1}{\sqrt{r}}, \frac{\|\mathbf{x}_r\|_F}{\sqrt{r}\|\mathbf{x}_r\|_*} \right]$ satisfies

$$\|\mathbf{Z} - \hat{\mathbf{Z}}\|_F \leq C \left(\|\mathbf{xx}^\top - \mathbf{x}_r \mathbf{x}_r^\top\|_* + \lambda \|\mathcal{R}^{-1}(\mathbf{xx}^\top - \mathbf{x}_r \mathbf{x}_r^\top)\|_* + \frac{\eta}{m} \right),$$

with probability at least $1 - C_0 \exp(-c_0 m)$ for some positive constants C, C', C_0, c_0 which only depend on r .

Next we give the error bound for $\lambda \rightarrow \infty$, i.e. for the estimator which only penalize the nuclear norm of \mathbf{X} (through \mathbf{Z}) and not the trace of \mathbf{Z} .

Theorem 8.2. *Assume that the entries of \mathbf{a}_i are i.i.d. zero-mean Gaussian, $\|\mathbf{n}\|_1 \leq \eta$ and let $\mathcal{R}^{-1}(\mathbf{Z})_s$ denote the best rank- s approximation of $\mathcal{R}^{-1}(\mathbf{Z})$. If*

$$m > C(p^2 + q^2 + 1)s,$$

then the solution $\hat{\mathbf{Z}}$ of (8.5) in the limit $\lambda \rightarrow \infty$ satisfies

$$\|\mathbf{Z} - \hat{\mathbf{Z}}\|_F \leq C_1 \frac{\|\mathcal{R}^{-1}(\mathbf{Z}) - \mathcal{R}^{-1}(\mathbf{Z})_s\|_*}{\sqrt{s}} + C_2 \frac{\eta}{m},$$

with probability at least $1 - C_0 \exp(-c_0 m)$ for some positive constants C_0, c_0, C_1, C_2, C which only depends on s .

For our scenario $\text{rank}(\mathcal{R}^{-1}(\mathbf{Z})) = r^2$, so the sufficient number of measurements are $m > C(p^2 + q^2 + 1)r^2$ to obtain a “noise-only” error bound. For $p = q$, the theorems therefore imply that the estimators with finite λ and $\lambda \rightarrow \infty$ requires less measurements in order to obtain a noise-only error bound. The bounds are, however, expected to be quite loose compared to the actual performance of the estimators. In Section 8.5, we compare the empirical performance of the estimators through numerical simulations.

8.3.3 Regularization path for λ

As “the world is full of obvious things which nobody by any chance ever observes” [DF02] we note that the Low-Rank PhaseLift estimate (8.5) depends on the parameter λ . Selecting a proper value of λ is important for finding a good estimate. We therefore write the estimate (8.5) as $\hat{\mathbf{Z}}(\lambda)$ to emphasize the dependence on λ . The value of λ should be such that $\text{rank}(\hat{\mathbf{Z}}(\lambda)) = 1$ and $\text{rank}(\mathcal{R}^{-1}(\hat{\mathbf{Z}}(\lambda)))$ is minimized. This means that λ should be small enough to ensure that $\text{rank}(\hat{\mathbf{Z}}(\lambda)) = 1$ while still maximally penalizing the rank of $\mathcal{R}^{-1}(\hat{\mathbf{Z}}(\lambda))$. By this argument we hypothesize that the optimal value of λ is

$$\lambda_{\text{optimal}} = \max \lambda, \text{ subject to } \text{rank}(\hat{\mathbf{Z}}(\lambda)) = 1.$$

One problem with the parameter λ is that we possibly need to search over arbitrarily large $\lambda \in [0, \infty)$. For this reason we instead consider the equivalent estimator

$$\begin{aligned} \mathbf{Z}_*(t) = \arg \min \quad & \|\mathcal{R}^{-1}(\mathbf{Z})\|_* \\ \text{subject to} \quad & \|\mathbf{y} - \mathcal{A}(\mathbf{Z})\|_1 \leq \eta, \quad \mathbf{Z} \succeq \mathbf{0}, \\ & \text{tr}(\mathbf{Z}) \leq t. \end{aligned} \quad (8.7)$$

The advantage of the estimator (8.7) is that the parameter t lies in the bounded interval

$$\text{tr}(\hat{\mathbf{Z}}(0)) = t_0 \leq t \leq t_1 = \text{tr}(\hat{\mathbf{Z}}(\infty)). \quad (8.8)$$

The goal is thus to find the maximal value of t that minimize $\text{rank}(\mathbf{Z}_*(t))$ for t in the interval (8.8). As minimizing the rank is numerically difficult, we instead minimize the continuous function

$$f(t) = \text{tr}(\mathbf{Z}_*(t)) - \|\mathbf{Z}_*(t)\|_F. \quad (8.9)$$

Using that $\text{tr}(\mathbf{Z}_*(t)) \geq \|\mathbf{Z}_*(t)\|_F$ and the Cauchy-Schwarz inequality, we get that

$$0 \leq f(t) \leq \left(\sqrt{\text{rank}(\mathbf{Z}_*(t))} - 1 \right) \|\mathbf{Z}_*(t)\|_F.$$

So for t such that $\mathbf{Z}_*(t) \neq \mathbf{0}$, $f(t) = 0$ if and only if $\text{rank}(\mathbf{Z}_*(t)) = 1$.

Several methods can be used to minimize (8.9). Here we use the secant method which updates t based on earlier values t_n and t_{n-1} as

$$\begin{aligned} \tilde{t}_{n+1} &= \frac{t_{n-1}f(t_n) - t_n f(t_{n-1})}{f(t_n) - f(t_{n-1})}, \\ t_{n+1} &= \min(t_{max}, \max(t_{min}, \tilde{t}_{n+1})), \end{aligned}$$

where we use the min-max operation to ensure that t lies in the interval. One example of how $f(t)$ and the NMSE of \mathbf{Z} depend on t is shown in Figure 8.2. We now discuss possible extensions of the Low-Rank Phase Retrieval problem.

8.4 Extensions of low-rank phase retrieval

In this section we discuss the extension of low-rank phase retrieval to other commonly discussed scenarios.

8.4.1 Robust low-rank phase retrieval

As in robust PCA, the low-rank matrix is in some scenarios corrupted by sparse noise. To separate the low-rank and sparse component, low-rank and sparse penalties are used.

Let $\mathbf{X}, \mathbf{E} \in \mathbb{R}^{p \times q}$ where \mathbf{X} is low-rank and \mathbf{E} is sparse, set also $\mathbf{x} = \text{vec}(\mathbf{X})$ and $\mathbf{e} = \text{vec}(\mathbf{E})$. We find that

$$\begin{aligned} \mathbf{X} + \mathbf{E} &= [\mathbf{I}_p \ \mathbf{I}_p] \begin{bmatrix} \mathbf{X} \\ \mathbf{E} \end{bmatrix}, \\ |\mathbf{a}_i^\top \text{vec}(\mathbf{X} + \mathbf{E})|^2 &= \left| \mathbf{a}_i^\top (\mathbf{I}_{2q} \otimes [\mathbf{I}_p \ \mathbf{I}_p]) \begin{bmatrix} \mathbf{x} \\ \mathbf{e} \end{bmatrix} \right|^2 \\ &= \text{tr} \left(\bar{\mathbf{a}}_i \bar{\mathbf{a}}_i^\top \begin{bmatrix} \mathbf{x} \\ \mathbf{e} \end{bmatrix} \begin{bmatrix} \mathbf{x}^* & \mathbf{e}^* \end{bmatrix} \right) = \text{tr}(\bar{\mathbf{a}}_i \bar{\mathbf{a}}_i^\top \mathbf{Z}) \end{aligned}$$

where $\bar{\mathbf{a}}_i^\top = \mathbf{a}_i^\top (\mathbf{I}_{2q} \otimes [\mathbf{I}_p \ \mathbf{I}_p])$ and now

$$\mathbf{Z} = \begin{bmatrix} \mathbf{x}\mathbf{x}^* & \mathbf{x}\mathbf{e}^* \\ \mathbf{e}\mathbf{x}^* & \mathbf{e}\mathbf{e}^* \end{bmatrix} = \begin{bmatrix} \mathbf{Z}_{11} & \mathbf{Z}_{12} \\ \mathbf{Z}_{21} & \mathbf{Z}_{22} \end{bmatrix}.$$

We thus find that the robust low-rank phase retrieval problem can be solved by changing the objective function in (8.5) to

$$\text{tr}(\mathbf{Z}) + \lambda_1 \|\mathcal{R}^{-1}(\mathbf{Z}_{11})\|_* + \lambda_2 \|\mathbf{Z}_{22}\|_1.$$

8.4.2 Low-rank and sparse matrices

In the example of X-ray crystallography and the Figure 8.1, the matrix \mathbf{X} is both sparse and low-rank. For such problems, it is beneficial to penalize both the rank

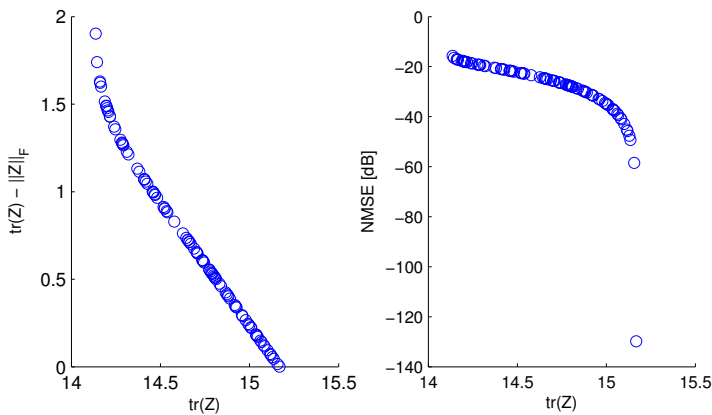


Figure 8.2: The rank substitute function $f(t) = \text{tr}(\mathbf{Z}_*(t)) - \|\mathbf{Z}_*(t)\|_F$ and the NMSE $\|\mathbf{Z}_*(t) - \mathbf{Z}\|_F^2 / \|\mathbf{Z}\|_F^2$ vs. $t = \text{tr}(\mathbf{Z}_*(t))$ for $p = q = 5$, $r = 1$ and $m = 50$ Gaussian measurements.

and the sparsity of the solution. The natural modification to the estimator (8.5) is therefore

$$\begin{aligned} \hat{\mathbf{Z}} &= \arg \min \operatorname{tr}(\mathbf{Z}) + \lambda_1 \|\mathcal{R}^{-1}(\mathbf{Z})\|_* + \lambda_2 \|\mathbf{Z}\|_1, \\ &\text{subject to } \|\mathbf{y} - \mathcal{A}(\mathbf{Z})\|_1 \leq \eta, \mathbf{Z} \succeq \mathbf{0} \end{aligned} \quad (8.10)$$

where $\lambda_1, \lambda_2 \geq 0$ are regularization parameters. For the estimator (8.10) we find the following error bounds.

Theorem 8.3. *Assume that the entries of \mathbf{a}_i are i.i.d. Gaussian distributed, $\|\mathbf{n}\|_1 \leq \eta$ and $\mathbf{Z} = \operatorname{vec}(\mathbf{X})\operatorname{vec}(\mathbf{X})^\top = \mathbf{xx}^\top$. Let*

$$\mathbf{X}_{r,k} = \arg \min_{\substack{\operatorname{rank}(\mathbf{X}') \leq r \\ \|\mathbf{X}'\|_1 \leq k}} \|\mathbf{X}' - \mathbf{X}\|_F$$

denote the best k -sparse and rank- r approximation of \mathbf{X} and $\mathbf{x}_{r,k} = \operatorname{vec}(\mathbf{X}_{r,k})$. If

$$m > C' k^2 r^2 \log(pq/k^2),$$

then the solution $\hat{\mathbf{Z}}$ of (8.10) with $\lambda_1 \in \left[\frac{1}{\sqrt{r}}, \frac{\|\mathbf{X}_{r,k}\|_F}{\sqrt{r}\|\mathbf{X}_{r,k}\|_*} \right]$ and $\lambda_2 \in \left[\frac{1}{\sqrt{k}}, \frac{\|\mathbf{X}_{r,k}\|_F}{\sqrt{k}\|\mathbf{X}_{r,k}\|_1} \right]$ satisfies

$$\begin{aligned} \|\mathbf{Z} - \hat{\mathbf{Z}}\|_F &\leq C \left(\|\mathbf{xx}^\top - \mathbf{x}_{r,k}\mathbf{x}_{r,k}^\top\|_* + \lambda_1 \|\mathcal{R}^{-1}(\mathbf{xx}^\top - \mathbf{x}_{r,k}\mathbf{x}_{r,k}^\top)\|_* \right. \\ &\quad \left. + \lambda_2 \|\mathbf{xx}^\top - \mathbf{x}_{r,k}\mathbf{x}_{r,k}^\top\|_1 + \frac{\eta}{m} \right), \end{aligned}$$

with probability at least $1 - C_0 \exp(-c_0 m)$ for some positive constants C, C', C_0, c_0 which only depends on k and r .

The proof of Theorem 8.3 is similar to the proof of Theorem 8.1 and is therefore omitted.

8.4.3 Recovery of low-rank tensors

So far we have only discussed the recovery of matrices, i.e. two-dimensional structures which describe e.g. two dimensional crystals. As most crystals are three dimensional, it is interesting to consider how the low-rank phase retrieval becomes modified for higher order structures. The higher order generalization of a matrix is the tensor [KB09]. As the components of a vector is written with one index x_i (a first order tensor), the component of a matrix with two indices X_{ij} (a second order tensor), the components of a k 'th order tensor is written with k indices $X_{i_1 i_2 \dots i_k}$. Since we considered X-ray crystallography as an important application we, for simplicity, focus on rank minimization for third order tensors.

Unlike matrices, several low-rank decompositions exist for third order tensors [BL10, KB09]. The two most common decompositions are the Tucker or higher order

SVD (HOSVD) and the CP decomposition. The Tucker decomposition of a tensor with components X_{ijk} is

$$X_{ijk} = \sum_{m,n,p} G_{mnp} U_{mi} V_{nj} W_{pk},$$

while the CP decomposition is

$$X_{ijk} = \sum_{q=1}^r A_{qi} B_{qj} C_{qk}.$$

We see that the CP decomposition is a special case of the HOSVD decomposition as it corresponds to setting

$$G_{mnp} = \begin{cases} 1, & m = n = p \\ 0, & \text{otherwise} \end{cases}.$$

The rank of a tensor is often defined as the minimal number of terms, r , in the CP decomposition. Another definition of rank is through the tensor unfolding where the elements of the tensor is mapped to matrices. Let $\mathbf{X} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$ be a third order tensor of size $P \times Q \times R$, the mode- n unfolding $\mathbf{X}_{(n)}$ of the tensor [YHS13, GRY11] is a matrix where the component (i_1, i_2, i_3) is mapped to the component (i_n, j) for

$$j = 1 + N_1 N_2 N_3 \sum_{k=1, k \neq n}^3 \frac{i_k - 1}{N_k}.$$

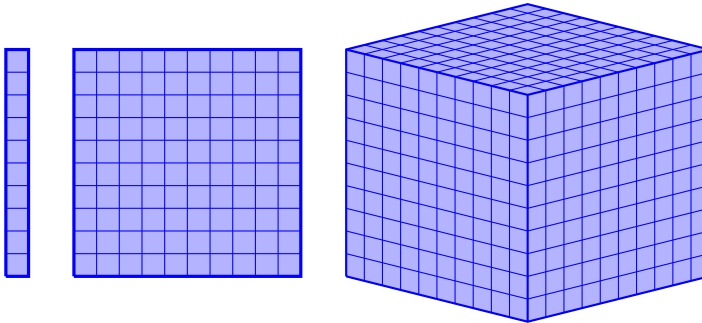


Figure 8.3: Illustration of first, second and a third order tensor. A first order tensor is a vector and a second order tensor is a matrix. A k 'th order tensor can be represented as a k -dimensional array of numbers.

The unfolding of a tensor is sometimes called *matricization*. It is also possible to define the *vectorization* of a tensor in the same way as for matrices. The *3-rank* of a third order tensor is the tuple

$$(r_1, r_2, r_3) = (\text{rank}(\mathbf{X}_{(1)}), \text{rank}(\mathbf{X}_{(2)}), \text{rank}(\mathbf{X}_{(3)})).$$

A common approach to tensor rank minimization is to minimize the sum of the 3-ranks [YHS13]. The corresponding convex penalty is then to replace the matrix ranks by the nuclear norm.

In the phase retrieval problem, the lifted variable is $\mathbf{Z} = \text{vec}(\mathbf{X})\text{vec}(\mathbf{X})^\top$. To promote low tensor rank in \mathbf{X} we note that $\text{vec}(\mathbf{X})$ is related to the tensor unfoldings by a linear transformation. Thus we can write

$$\begin{aligned} \text{vec}(\mathbf{X}) &= \text{vec}_1(\mathbf{X}_{(1)}) = \mathbf{P}_1 \text{vec}(\mathbf{X}_{(1)}), \\ \text{vec}(\mathbf{X}) &= \text{vec}_2(\mathbf{X}_{(2)}) = \mathbf{P}_2 \text{vec}(\mathbf{X}_{(2)}), \\ \text{vec}(\mathbf{X}) &= \text{vec}_3(\mathbf{X}_{(3)}) = \mathbf{P}_3 \text{vec}(\mathbf{X}_{(3)}), \end{aligned}$$

where $\text{vec}_k(\cdot)$ is the vectorization operators related to the k 'th tensor unfolding and $\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3$ are permutation matrices. By using the permutation matrices, we find that the nuclear norm tensor rank penalty becomes

$$\|\mathcal{R}^{-1}(\mathbf{P}_1^{-1}\mathbf{Z}(\mathbf{P}_1^\top)^{-1})\|_* + \|\mathcal{R}^{-1}(\mathbf{P}_2^{-1}\mathbf{Z}(\mathbf{P}_2^\top)^{-1})\|_* + \|\mathcal{R}^{-1}(\mathbf{P}_3^{-1}\mathbf{Z}(\mathbf{P}_3^\top)^{-1})\|_*.$$

This formulation is easily extended to higher order tensors.

8.5 Experiments with random measurement matrices

Here we perform numerical experiments to investigate the empirical performance of low-rank matrix phase retrieval (LRPR). We compare the performance of Low-Rank PhaseLift (LR-PhaseLift) (8.7), with the parameter t set by the procedure described in 8.3.3, with the standard PhaseLift. The methods were implemented using the `cvx` toolbox [GBY08]. Since the solution is only unique up to a global phase, we use the error measure

$$\min_{\theta} \|\mathbf{X} - e^{i\theta} \hat{\mathbf{X}}\|_F^2 = \|\mathbf{X}\|_F^2 + \|\hat{\mathbf{X}}\|_F^2 - 2|\text{tr}(\mathbf{X}^\top \hat{\mathbf{X}})|.$$

We consider the scenario where the elements of the sensing matrix in (8.1) \mathbf{A} are drawn from an $\mathcal{N}(0, 1)$ distribution. We generated the low-rank matrix by setting $\mathbf{X} = \mathbf{L}\mathbf{R}^\top$ where $\mathbf{L} \in \mathbb{R}^{p \times r}$, $\mathbf{R} \in \mathbb{R}^{q \times r}$ and the elements from \mathbf{L} and \mathbf{R} are independently drawn from a $\mathcal{N}(0, 1)$ distribution.

8.5.1 Noise-free measurements

In the first experiment we considered noise free measurements. We used $p = 7$, $q = 7$, $r = 2$ and varied the number of measurements m . The results are shown

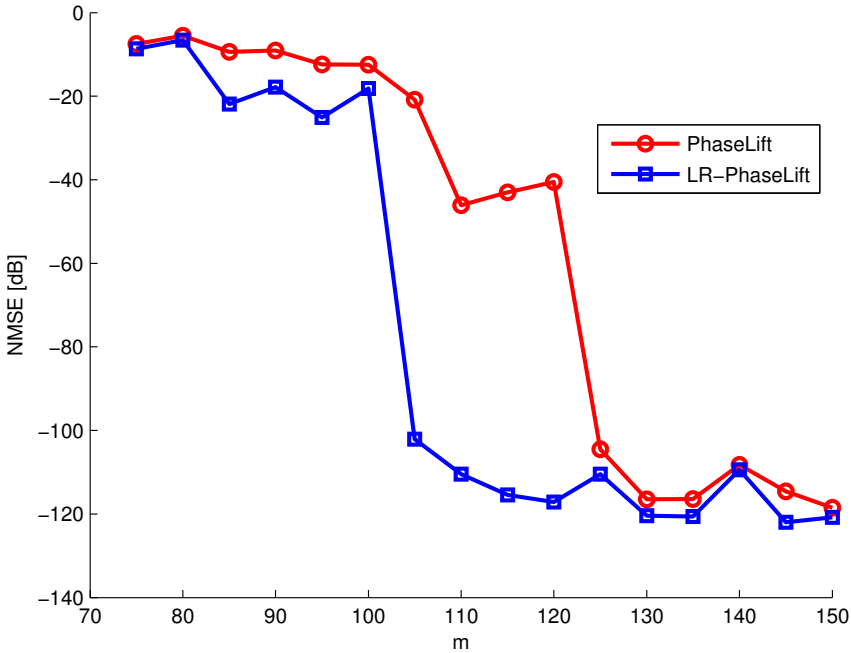


Figure 8.4: NMSE for reconstruction from Gaussian measurements for $p = q = 5$ and $\text{rank}(\mathbf{X}) = 1$. (Change to other figure when simulation is done)

in Figure 8.4. We find that LR-PhaseLift is able to reconstruct the matrix (up to numerical precision) for $m \geq 105$ measurements while PhaseLift requires $m \geq 125$ measurements to reconstruct the matrix.

8.5.2 Noisy measurements

In the second experiment we examined how noise affects the final estimates. In the experiments, we generated the noise from a $\mathcal{N}(\mathbf{0}, \sigma_n^2 \mathbf{I}_m)$ distribution. The noise variance, σ_n^2 was chosen so that the Signal-to-Noise-Ratio (SNR)

$$\text{SNR} = \frac{\mathcal{E}[\|\mathbf{A}\mathbf{x}\|_2^2]}{\mathcal{E}[\|\mathbf{n}\|_2^2]} = \frac{r^2 m (pq)^2 + 2r^2 mpq}{m\sigma_n^2},$$

was equal to 20 dB.

To estimate \mathbf{X} we used the standard PhaseLift and LR-PhaseLift with

$$\eta = \sigma_n m,$$

as in [CCG15].

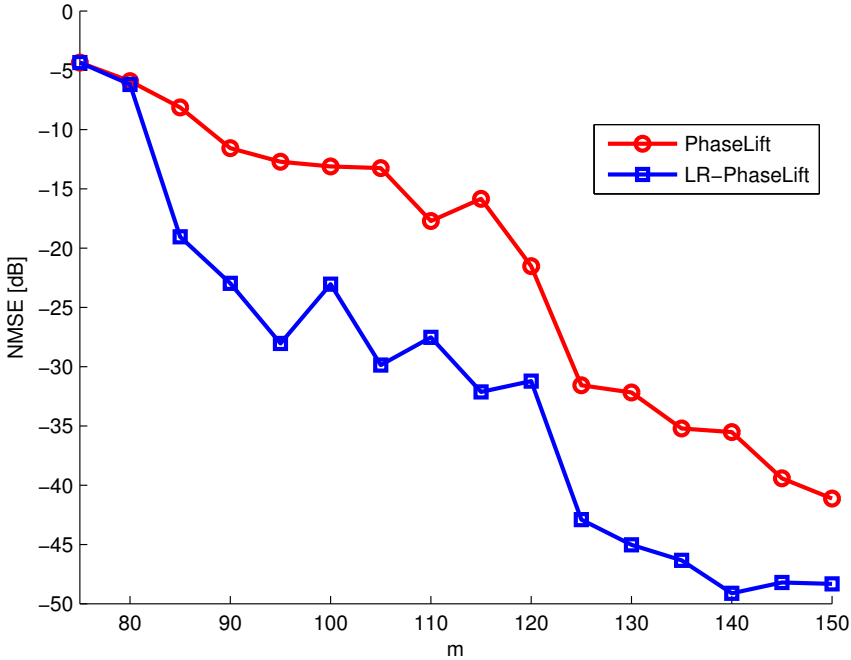


Figure 8.5: NMSE for reconstruction from noisy Gaussian measurements for $p = q = 7$, $r = 2$ and SNR = 20 dB.

The results are shown in Figure 8.5. We find that the estimation methods gave larger errors when noise was present. This is to be expected since the use of $\eta > 0$ makes the estimates biased towards zero. LR-PhaseLift gave an NMSE which was 10 to 7 dB lower than the NMSE of PhaseLift for the values of m shown in the Figure.

8.6 Conclusion

In this chapter we showed how the PhaseLift method for phase retrieval can be adapted to promote low-rank in the estimate. In phase retrieval we only measure the magnitudes of measurements, the sign or phase of the measurements are therefore lost. This occurs in e.g. X-ray crystallography where X-rays are scattered against a crystal. The approach uses the theory of approximation with Kronecker products and leads to a convex penalty in the lifted variable. Using methods for convex optimization bounds, we derived error bounds for the estimation methods. The low-rank phase retrieval method was also extended to robust low-rank matrix reconstruction, low-rank and sparse matrix reconstruction and the recovery of higher order tensors. Lastly we evaluated the empirical performance of the algorithms us-

ing numerical experiments. We found that the proposed method could reconstruct a low-rank matrix from fewer number of measurements than the standard PhaseLift algorithm.

8.7 Derivations and proofs

8.7.1 Details for uniqueness bound

The proof of Proposition 8.2.1 relies on Theorem 2.4. in [EM14].

A random variable $\mathbf{a} \in \mathbb{R}^n$ is *isotropic* if

$$\mathcal{E}[|\mathbf{a}^\top \mathbf{t}|^2] = \|\mathbf{t}\|_2^2$$

for all $\mathbf{t} \in \mathbb{R}^n$ and L -subgaussian if

$$\Pr(|\mathbf{a}^\top \mathbf{t}| \geq Lu(\mathcal{E}[|\mathbf{a}^\top \mathbf{t}|^2])^{1/2}) \leq 2 \exp(-u^2/2),$$

for all $\mathbf{t} \in \mathbb{R}^n$. Let $T \subset \mathbb{R}^n$ be a set and let

$$T_- = \left\{ \frac{\mathbf{t} - \mathbf{s}}{\|\mathbf{t} - \mathbf{s}\|_2}, \mathbf{t}, \mathbf{s} \in T, \mathbf{t} \neq \mathbf{s} \right\},$$

$$T_+ = \left\{ \frac{\mathbf{t} + \mathbf{s}}{\|\mathbf{t} + \mathbf{s}\|_2}, \mathbf{t}, \mathbf{s} \in T, \mathbf{t} \neq \mathbf{s} \right\}.$$

Denote

$$E = \max \left\{ \mathcal{E}_{\mathbf{g}} \left[\sup_{\mathbf{v} \in T_-} \sum_{i=1}^n g_i v_i \right], \mathcal{E}_{\mathbf{g}} \left[\sup_{\mathbf{v} \in T_+} \sum_{i=1}^n g_i v_i \right] \right\},$$

$$\rho_{T,m} = \frac{E}{\sqrt{m}} + \frac{E^2}{m}$$

and

$$\kappa(\mathbf{v}, \mathbf{w}) = \mathcal{E} \left[\mathbf{a}^\top \mathbf{v} / \|\mathbf{v}\|_2 \mathbf{a}^\top \mathbf{w} / \|\mathbf{w}\|_2 \right].$$

Theorem 8.4 (Theorem 2.4 from [EM14]). *Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be a matrix with i.i.d. random isotropic L -subgaussian row vectors. For every $L \geq 1$ there exists constants c_1 , c_2 and c_3 that depends only on L such that for $u \geq c_1$*

$$\left| \|\mathbf{A}\mathbf{s}\|^2 - \|\mathbf{A}\mathbf{t}\|^2 \right|_1 \geq D \|\mathbf{s} - \mathbf{t}\|_2 \|\mathbf{s} + \mathbf{t}\|_2,$$

$$D = \kappa(\mathbf{s} - \mathbf{t}, \mathbf{s} + \mathbf{t}) - c_3 u^3 \rho_{T,m}$$

holds for all $\mathbf{s}, \mathbf{t} \in T$ with probability at least

$$1 - 2 \exp(-c_2 u^2 \min\{m, E^2\}).$$

The quantity $\kappa(\mathbf{v}, \mathbf{w})$ can be shown to be bounded from below as $\kappa(\mathbf{v}, \mathbf{w}) \geq c_1$ under some conditions (e.g. small-ball or Paley Sygmund arguments) [EM14]. We find that it is necessary for m to be large enough such that

$$c_1 - c_3 u^3 \rho_{T,m} > 0$$

to ensure a unique solution.

Proof of Proposition 8.2.1. For simplicity we only consider Gaussian measurement matrices. The extension to sub-Gaussian measurement matrices is straightforward. Let $T_r = \{\mathbf{X} \in \mathbb{R}^{p \times q} : \text{rank}(\mathbf{X}) \leq r\}$. Each row vector of \mathbf{A} can be represented by a $p \times q$ matrix. Let \mathbf{G} denote a reshaped row vector of \mathbf{A} and let $U_r = \{\mathbf{X} \in \mathbb{R}^{p \times q} : \text{rank}(\mathbf{X}) \leq r, \|\mathbf{X}\|_F = 1\}$, we find that [CRPW12]

$$\begin{aligned} E &= \mathbb{E}_{\mathbf{G}} \left[\sup_{\mathbf{W} \in U_{2r}} \text{tr}(\mathbf{G}^\top \mathbf{W}) \right] = \mathbb{E}_{\mathbf{G}} \left[\left(\sum_{i=1}^{2r} \sigma_i(\mathbf{G})^2 \right)^{1/2} \right] \\ &\leq \sqrt{6r(p+q-2r)} = E' \end{aligned}$$

Provided that $\inf_{v,w} \kappa(v, w) \geq c_1$, Theorem 8.4 now gives that the solution to

$$\mathbf{y} = |\mathbf{A} \text{vec}(\mathbf{X})|^2, \text{rank}(\mathbf{X}) \leq r$$

is unique with probability at least

$$1 - 2 \exp(-c_2 u^2 E'^2) = 1 - 2 \exp(-6c_2 u^2 r(p+q-2r)),$$

when $r \leq \min(p, q)/2$ and

$$\begin{aligned} c_1 - c_3 u^3 \rho_{T,m} &\geq c_1 - c_3 u^3 \left(\frac{E'}{\sqrt{m}} + \frac{E'^2}{m} \right) \\ &\geq c_1 - c_3 u^3 \left[\frac{E'}{\sqrt{m}} + \left(\frac{E'}{\sqrt{m}} \right)^2 \right] > 0 \\ \Leftrightarrow m &> \frac{E'}{\sqrt{\frac{c_1}{c_3 u^3} + \frac{1}{4} - \frac{1}{2}}} \\ \Leftrightarrow m &> \sqrt{\frac{c_3 u^3}{c_1}} \sqrt{6r(p+q-2r)}. \end{aligned}$$

□

8.7.2 Proof of (8.6)

We find that

$$\begin{aligned}
& 2\|\mathbf{xx}^\top - \hat{\mathbf{x}}\hat{\mathbf{x}}^\top\|_F^2 - \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 \cdot \|\mathbf{x} + \hat{\mathbf{x}}\|_2^2 \\
&= 2\text{tr}(\mathbf{xx}^\top \mathbf{xx}^\top) + 2\text{tr}(\hat{\mathbf{x}}\hat{\mathbf{x}}^\top \hat{\mathbf{x}}\hat{\mathbf{x}}^\top) - 4\text{tr}(\mathbf{xx}^\top \hat{\mathbf{x}}\hat{\mathbf{x}}^\top) \\
&\quad - (\|\mathbf{x}\|_2^2 + \|\hat{\mathbf{x}}\|_2^2 - 2\mathbf{x}^\top \hat{\mathbf{x}})(\|\mathbf{x}\|_2^2 + \|\hat{\mathbf{x}}\|_2^2 + 2\mathbf{x}^\top \hat{\mathbf{x}}) \\
&= 2\|\mathbf{x}\|_2^4 + 2\|\hat{\mathbf{x}}\|_2^4 - 4(\mathbf{x}^\top \hat{\mathbf{x}})^2 - \|\mathbf{x}\|_2^4 - \|\hat{\mathbf{x}}\|_2^4 - 2\|\mathbf{x}\|_2^2 \|\hat{\mathbf{x}}\|_2^2 + 4(\mathbf{x}^\top \hat{\mathbf{x}})^2 \\
&= \|\mathbf{x}\|_2^4 + \|\hat{\mathbf{x}}\|_2^4 - 2\|\mathbf{x}\|_2^2 \|\hat{\mathbf{x}}\|_2^2 \\
&= (\|\mathbf{x}\|_2^2 - \|\hat{\mathbf{x}}\|_2^2)^2 \geq 0.
\end{aligned}$$

This proves (8.6).

8.7.3 Recovery bounds under RIP-conditions

Theorem 8.1 and 8.2 require the bounds on the RIP constants given in Corollary 8.7.1. The proof of Theorem 8.2 is similar to the proof of Theorem 1 in [CCG15] when Corollary 8.7.1 is given, the proofs are therefore not repeated here. Here we first show the proof of Theorem 8.1 since it differs from the proof in [CCG15] and later also prove Corollary 8.7.1.

The lifted random sensing operator $\mathcal{A}(\cdot) = \Phi \text{vec}(\cdot)$ does not satisfy the RIP since the individual components have non zero-means. However, in Proposition 8.7.2 and Corollary 8.7.1 in Section 8.7.5 we show that the operator \mathcal{B} defined as

$$\mathcal{B}(\mathbf{Z})_i = \Phi(\mathbf{Z})_{2i-1} - \Phi(\mathbf{Z})_{2i},$$

does satisfy the following RIP with high probability.

Definition 8.1. A linear operator \mathcal{B} satisfies the \mathcal{M}_{K_1, K_2} ℓ_2/ℓ_1 -RIP property if

$$(1 - \gamma_{K_1, K_2}^{lb}) \|\mathbf{W}\|_F \leq \frac{1}{m} \|\mathcal{B}(\mathbf{W})\|_1 \leq (1 + \gamma_{K_1, K_2}^{ub}) \|\mathbf{W}\|_F,$$

for all matrices $\mathbf{W} \in \mathcal{M}_{K_1, K_2}$ where

$$\mathcal{M}_{k,r} = \left\{ \mathbf{W} \in \mathbb{R}^{p^2 \times q^2} : \text{rank}(\mathbf{W}) \leq r^2, \text{rank}(\mathcal{R}(\mathbf{W})) \leq k, \mathcal{R}^{-1}(\mathbf{W}) \succeq \mathbf{0} \right\}.$$

Assuming that the RIP condition holds, we prove Theorem 8.1 with the help of the following proposition.

Proposition 8.7.1. *Assume that $\mathbf{Z} = \text{vec}(\mathbf{X})\text{vec}(\mathbf{X})^\top = \mathbf{xx}^\top$ and let \mathbf{X}_r be the best rank- r approximation of \mathbf{X} and $\mathbf{x}_r = \text{vec}(\mathbf{X}_r)$. If there exists K_1, K_2 such that \mathcal{B} satisfies the \mathcal{M}_{K_1, K_2} ℓ_2/ℓ_1 -RIP property, with*

$$\frac{(1 + \delta_{K_1, K_2}^{ub})\sqrt{3}}{(1 - \delta_{2K_1, 2K_2}^{lb})\sqrt{K_1}} + \frac{1 + \delta_{K_1, K_2}^{ub}}{(1 - \delta_{K_1, K_2}^{lb})\sqrt{K_1}} < \frac{1}{1 + \sqrt{2}}$$

then for $\sqrt{\frac{K_1}{K_2}} \leq \lambda \leq \frac{\|\mathbf{X}\|_F}{\sqrt{r}\|\mathbf{X}_r\|_*}$ there exists a constant C such that

$$\|\mathbf{Z} - \hat{\mathbf{Z}}\|_F \leq C \left(\|\mathbf{xx}^\top - \mathbf{x}_r \mathbf{x}_r^\top\|_* + \lambda \|\mathcal{R}^{-1}(\mathbf{xx}^\top - \mathbf{x}_r \mathbf{x}_r^\top)\|_* + \frac{\eta}{m} \right).$$

Proof. Let $\hat{\mathbf{Z}}$ be the minimizer of (8.5) and let $\mathbf{Z} = \mathbf{xx}^\top$, then $\hat{\mathbf{Z}} = \mathbf{xx}^\top + \mathbf{H}$. Set $\mathbf{Z}_r = \mathbf{x}_r \mathbf{x}_r^\top$ and $\mathbf{Z}_c = \mathbf{Z} - \mathbf{x}_r \mathbf{x}_r^\top$. Let $\mathbf{X}_r = \mathbf{U}_r \boldsymbol{\Sigma}_r \mathbf{V}_r^\top$ be the SVD of \mathbf{X}_r and set $\mathbf{u} = \text{vec}(\mathbf{X}_r / \|\mathbf{X}_r\|_F)$. A subdifferential of $\|\cdot\|_* + \lambda \|\mathcal{R}^{-1}(\cdot)\|_*$ at \mathbf{Z}_r is

$$\begin{aligned} & \mathbf{uu}^\top + \mathbf{Y} + \lambda(\mathcal{R}^{-1})^*(\mathbf{V}_r \mathbf{V}_r^\top \otimes \mathbf{U}_r \mathbf{U}_r^\top) + \lambda \mathbf{W} \\ & = \mathbf{uu}^\top + \mathbf{Y} + \lambda \text{vec}(\mathbf{U}_r \mathbf{V}_r^\top) \text{vec}(\mathbf{U}_r \mathbf{V}_r^\top)^\top + \lambda \mathbf{W}, \end{aligned}$$

where $\|\mathbf{Y}\| \leq 1$, $\mathbf{u}^\top \mathbf{Y} = \mathbf{0}$ and \mathbf{W} is such that $\mathbf{U}_r^\top \mathbf{W} = \mathbf{0}$, $\mathbf{W} \mathbf{V}_r = \mathbf{0}$ and $\|\mathbf{W}\| \leq 1$. Define the tangent spaces T and S by

$$\begin{aligned} T &= T_{\mathbf{Z}_r} \{\mathbf{Z} : \text{rank}(\mathbf{Z}) \leq 1\} = \{\mathbf{u}_r \mathbf{z}^\top + \mathbf{z} \mathbf{u}_r^\top, \mathbf{z} \in \mathbb{R}^{pq}\} \\ S &= T_{\mathbf{Z}_r} \{\mathbf{Z} : \text{rank}(\mathcal{R}^{-1}(\mathbf{Z})) \leq r^2\}, \\ &= \{(\mathbf{I}_q \otimes \mathbf{U}_r) \mathbf{B} (\mathbf{I}_q \otimes \mathbf{U}_r)^\top + (\mathbf{V}_r \otimes \mathbf{I}_p) \mathbf{A} (\mathbf{V}_r^\top \otimes \mathbf{I}_p) : \mathbf{A} \in \mathbb{R}^{rp \times rp}, \mathbf{B} \in \mathbb{R}^{rq \times rq}\}. \end{aligned}$$

We find that the projections onto the tangent spaces are given by

$$\begin{aligned} \mathcal{P}_T(\mathbf{Z}) &= \mathbf{uu}^\top \mathbf{Z} + (\mathbf{I} - \mathbf{uu}^\top) \mathbf{Z} \mathbf{uu}^\top, \\ \mathcal{P}_S(\mathbf{Z}) &= (\mathbf{V}_r \mathbf{V}_r^\top \otimes \mathbf{U}_r \mathbf{U}_r^\top) \mathbf{Z} + (\mathbf{I}_{pq} - (\mathbf{V}_r \mathbf{V}_r^\top \otimes \mathbf{U}_r \mathbf{U}_r^\top)) \mathbf{Z} (\mathbf{V}_r \mathbf{V}_r^\top \otimes \mathbf{U}_r \mathbf{U}_r^\top). \end{aligned}$$

This gives us that $\mathcal{P}_S \mathcal{P}_T = \mathcal{P}_T \mathcal{P}_S = \mathcal{P}_T$, so $T \subset S$.

We can chose \mathbf{Y} and \mathbf{W} such that $\langle \mathbf{Y}, \mathbf{H} \rangle = \|\mathbf{H}_{T^\perp \cap S}\|_*$ and $\langle \mathbf{W}, \mathbf{H} \rangle = \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp})\|_*$. We also set $\mathbf{v} = \text{vec}(\mathbf{U}_r \mathbf{V}_r^\top)$ for brevity.

We find that

$$\begin{aligned} 0 &\geq \text{tr}(\mathbf{Z} + \mathbf{H}) + \lambda \|\mathcal{R}^{-1}(\mathbf{Z} + \mathbf{H})\|_* - \text{tr}(\mathbf{Z}) \\ &\quad - \lambda \|\mathcal{R}^{-1}(\mathbf{Z})\|_* = \|\mathbf{Z} + \mathbf{H}\|_* + \lambda \|\mathcal{R}^{-1}(\mathbf{Z} + \mathbf{H})\|_* \\ &\quad - \|\mathbf{Z}\|_* - \lambda \|\mathcal{R}^{-1}(\mathbf{Z})\|_* \\ &\geq \|\mathbf{Z}_r + \mathbf{H}\|_* + \lambda \|\mathcal{R}^{-1}(\mathbf{Z}_r + \mathbf{H})\|_* - 2\|\mathbf{Z}_c\|_* \\ &\quad - \|\mathbf{Z}_r\|_* - \lambda \|\mathcal{R}^{-1}(\mathbf{Z}_r)\|_* - 2\lambda \|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_* \\ &\geq \langle \mathbf{uu}^\top + \mathbf{Y} + \lambda \mathbf{vv}^\top + \lambda \mathbf{W}, \mathbf{H} \rangle - 2\lambda \|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_* \\ &\quad - 2\|\mathbf{Z}_c\|_* = \langle \mathbf{uu}^\top + \lambda \mathcal{P}_T(\mathbf{vv}^\top), \mathbf{H}_T \rangle + \|\mathbf{H}_{T^\perp \cap S}\|_* \\ &\quad + \lambda \langle \mathcal{P}_{T^\perp}(\mathbf{vv}^\top), \mathbf{H}_{T^\perp} \rangle + \lambda \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp})\|_* \\ &\quad - 2\lambda \|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_* - 2\|\mathbf{Z}_c\|_*. \end{aligned}$$

We further have that $\mathbf{H}_{T^\perp} \succeq \mathbf{0}$ because $\mathbf{Z} + \mathbf{H} \succeq \mathbf{0}$. This gives us that

$$\langle \mathcal{P}_{T^\perp}(\mathbf{vv}^\top), \mathbf{H}_{T^\perp} \rangle = \langle \mathbf{vv}^\top, \mathbf{H}_{T^\perp} \rangle = \mathbf{v}^\top \mathbf{H}_{T^\perp} \mathbf{v} \geq 0.$$

Putting this together we find that

$$\begin{aligned} & \|\mathbf{H}_{T^\perp \cap S}\|_* + \lambda \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp})\|_* \\ & \leq -\langle \mathbf{u}\mathbf{u}^\top + \lambda \mathcal{P}_T(\mathbf{v}\mathbf{v}^\top), \mathbf{H}_T \rangle + 2\|\mathbf{Z}_c\|_* + 2\lambda \|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_* \\ & \leq \|\mathbf{H}_T\| + \lambda \|\mathcal{P}_T(\mathbf{v}\mathbf{v}^\top)\|_* \|\mathbf{H}_T\| + 2\|\mathbf{Z}_c\|_* + 2\lambda \|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_* \end{aligned}$$

Also, since $\lambda \leq \frac{\|\mathbf{X}_r\|_F}{\sqrt{r}\|\mathbf{X}_r\|_*}$ we get that

$$\begin{aligned} & \|\mathcal{P}_T(\mathbf{v}\mathbf{v}^\top)\|_*^2 \leq 2\|\mathcal{P}_T(\mathbf{v}\mathbf{v}^\top)\|_F^2 \\ & = 4(\mathbf{u}^\top \mathbf{v})^2 \|\mathbf{v}\|_2^2 \|\mathbf{u}\|_2^2 - 2(\mathbf{u}^\top \mathbf{v})^4 \\ & = 4r \frac{\|\mathbf{X}_r\|_*^2}{\|\mathbf{X}_r\|_F^2} - 2 \frac{\|\mathbf{X}_r\|_*^4}{\|\mathbf{X}_r\|_F^4} \leq 2r \frac{\|\mathbf{X}_r\|_*^2}{\|\mathbf{X}_r\|_F^2} \leq \frac{2}{\lambda^2}. \end{aligned}$$

This gives us that

$$\begin{aligned} & \|\mathbf{H}_{T^\perp \cap S}\|_* + \lambda \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp})\|_* \\ & \leq (1 + \sqrt{2})\|\mathbf{H}_T\| + 2\|\mathbf{Z}_c\|_* + 2\lambda \|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_*. \end{aligned}$$

Next, decompose $\mathbf{H}_{S^\perp \cap T}$ and \mathbf{H}_{S^\perp} into mutually orthogonal matrices

$$\begin{aligned} \mathbf{H}_{T^\perp \cap S} &= \sum_{i=1}^{M_1} \mathbf{H}_{T^\perp \cap S}^{(i)}, \\ \mathbf{H}_{S^\perp} &= \sum_{i=1}^{M_2} \mathbf{H}_{S^\perp}^{(i)}, \end{aligned}$$

such that

$$\begin{aligned} & \sigma_{\min}(\mathbf{H}_{T^\perp \cap S}^{(i)}) \geq \sigma_{\max}(\mathbf{H}_{T^\perp \cap S}^{(i+1)}), \\ & \sigma_{\min}(\mathcal{R}^{-1}(\mathbf{H}_{S^\perp}^{(i)})) \geq \sigma_{\max}(\mathcal{R}^{-1}(\mathbf{H}_{S^\perp}^{(i+1)})), \\ & \text{rank}(\mathbf{H}_{T^\perp \cap S}^{(i)}) = K_1, \quad 1 \leq i \leq M_1 - 1 \\ & \|\mathbf{H}_{T^\perp \cap S}\|_* = \sum_{i=1}^{M_1} \|\mathbf{H}_{T^\perp \cap S}^{(i)}\|_*, \\ & \text{rank}(\mathcal{R}^{-1}(\mathbf{H}_{S^\perp}^{(i)})) = K_2, \quad 1 \leq i \leq M_2 - 1, \\ & \text{rank}(\mathbf{H}_{S^\perp}^{(i)}) = K_1, \quad 1 \leq i \leq M_2 - 1, \\ & \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp})\|_* = \sum_{i=1}^{M_2} \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp}^{(i)})\|_*, \end{aligned}$$

where $\sigma_{\min}(\cdot)$ denotes the smallest non-zero singular value.

Combined with the RIP-property, this gives us that

$$\begin{aligned}
\sum_{i=2}^{M_1} \frac{1}{m} \|\mathcal{B}(\mathbf{H}_{T^\perp \cap S}^{(i)})\|_1 &\leq (1 + \delta_{K_1, K_2}^{ub}) \sum_{i=2}^{M_1} \|\mathbf{H}_{T^\perp \cap S}^{(i)}\|_F \\
&\leq \frac{(1 + \delta_{K_1, K_2}^{ub})}{\sqrt{K_1}} \|\mathbf{H}_{T^\perp \cap S}\|_*, \\
\sum_{i=2}^{M_2} \frac{1}{m} \|\mathcal{B}(\mathbf{H}_{S^\perp}^{(i)})\|_1 &\leq (1 + \delta_{K_1, K_2}^{ub}) \sum_{i=2}^{M_2} \|\mathbf{H}_{S^\perp}^{(i)}\|_F \\
&= (1 + \delta_{K_1, K_2}^{ub}) \sum_{i=2}^{M_2} \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp}^{(i)})\|_F \\
&\leq \frac{(1 + \delta_{K_1, K_2}^{ub})}{\sqrt{K_2}} \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp})\|_*.
\end{aligned}$$

Because of the constraint, we get that

$$\frac{1}{m} \|\mathcal{B}(\mathbf{H})\|_1 \leq \frac{1}{m} \|\Phi(\mathbf{H})\|_1 \leq \frac{1}{m} \left(\|\mathbf{y} - \Phi(\hat{\mathbf{Z}})\|_1 + \|\mathbf{y} - \Phi(\mathbf{Z})\|_1 \right) \leq \frac{2\eta}{m}. \quad (8.11)$$

We now take K_1 and K_2 such that $\sqrt{\frac{K_1}{K_2}} \leq \lambda$. Using this, we find that

$$\begin{aligned}
\frac{2\eta}{m} &\geq \frac{1}{m} \|\mathcal{B}(\mathbf{H})\|_1 \geq \frac{1}{m} \left\| \mathcal{B} \left(\mathbf{H}_T + \mathbf{H}_{T^\perp \cap S}^{(1)} + \mathbf{H}_{S^\perp}^{(1)} \right) \right\|_1 \\
&\quad - \sum_{i=2}^{M_1} \frac{1}{m} \|\mathcal{B}(\mathbf{H}_{T^\perp \cap S}^{(i)})\|_1 - \sum_{i=2}^{M_2} \frac{1}{m} \|\mathcal{B}(\mathbf{H}_{S^\perp}^{(i)})\|_1 \\
&\geq (1 - \delta_{2K_1, 2K_2}^{lb}) \|\mathbf{H}_T + \mathbf{H}_{T^\perp \cap S}^{(1)} + \mathbf{H}_{S^\perp}^{(1)}\|_F \\
&\quad - \frac{(1 + \delta_{K_1, K_2}^{ub})}{\sqrt{K_1}} \|\mathbf{H}_{T^\perp \cap S}\|_* - \frac{(1 + \delta_{K_1, K_2}^{ub})}{\sqrt{K_2}} \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp})\|_* \\
&\geq \frac{1 - \delta_{2K_1, 2K_2}^{lb}}{\sqrt{3}} \left(\|\mathbf{H}_T\|_F + \|\mathbf{H}_{T^\perp \cap S}^{(1)}\|_F + \|\mathbf{H}_{S^\perp}^{(1)}\|_F \right) \\
&\quad - \frac{(1 + \delta_{K_1, K_2}^{ub})}{\sqrt{K_1}} \left(\|\mathbf{H}_{T^\perp \cap S}\|_* + \lambda \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp})\|_* \right) \\
&\geq \frac{1 - \delta_{2K_1, 2K_2}^{lb}}{\sqrt{3}} \left(\|\mathbf{H}_T\|_F + \|\mathbf{H}_{T^\perp \cap S}^{(1)}\|_F + \|\mathbf{H}_{S^\perp}^{(1)}\|_F \right) - \\
&\quad \frac{(1 + \delta_{K_1, K_2}^{ub})}{\sqrt{K_1}} \left((1 + \sqrt{2}) \|\mathbf{H}_T\|_* + 2\lambda \|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_* + 2\|\mathbf{Z}_c\|_* \right).
\end{aligned}$$

This gives us that

$$\begin{aligned} & \|\mathbf{H}_T\|_F + \|\mathbf{H}_{T^\perp \cap S}^{(1)}\|_F + \|\mathbf{H}_{S^\perp}^{(1)}\|_F \\ & \leq C_1 \left((1 + \sqrt{2})\|\mathbf{H}_T\| + 2\|\mathbf{Z}_c\|_* + 2\lambda\|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_* \right) + \frac{\sqrt{3}}{(1 - \delta_{2K_1, 2K_2}^{lb})} \frac{2\eta}{m}, \end{aligned}$$

where

$$C_1 = \frac{(1 + \delta_{K_1, K_2}^{ub})\sqrt{3}}{(1 - \delta_{2K_1, 2K_2}^{lb})\sqrt{K_1}}.$$

For the remaining terms we find that

$$\begin{aligned} & \sum_{i=2}^{M_1} \|\mathbf{H}_{T^\perp \cap S}^{(i)}\|_F + \sum_{i=2}^{M_2} \|\mathbf{H}_{S^\perp}^{(i)}\|_F \\ & \leq \frac{1}{1 - \delta_{K_1, K_2}^{lb}} \left(\sum_{i=2}^{M_1} \frac{1}{m} \|\mathcal{B}(\mathbf{H}_{T^\perp \cap S}^{(i)})\|_1 + \sum_{i=2}^{M_2} \frac{1}{m} \|\mathcal{B}(\mathbf{H}_{S^\perp}^{(i)})\|_1 \right) \\ & \leq \frac{1 + \delta_{K_1, K_2}^{ub}}{(1 - \delta_{K_1, K_2}^{lb})\sqrt{K_1}} \|\mathbf{H}_{T^\perp \cap S}\|_* + \frac{1 + \delta_{K_1, K_2}^{ub}}{(1 - \delta_{K_1, K_2}^{lb})\sqrt{K_2}} \|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp})\|_* \\ & \leq \frac{1 + \delta_{K_1, K_2}^{ub}}{(1 - \delta_{K_1, K_2}^{lb})\sqrt{K_1}} (\|\mathbf{H}_{T^\perp \cap S}\|_* + \lambda\|\mathcal{R}^{-1}(\mathbf{H}_{S^\perp})\|_*) \\ & \leq \frac{1 + \delta_{K_1, K_2}^{ub}}{(1 - \delta_{K_1, K_2}^{lb})\sqrt{K_1}} \left((1 + \sqrt{2})\|\mathbf{H}_T\| + 2\|\mathbf{Z}_c\|_* + 2\lambda\|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_* \right) \end{aligned}$$

Putting this together gives us that

$$\begin{aligned} \|\mathbf{H}\|_F & = \|\mathbf{H}_T + \mathbf{H}_{T^\perp \cap S} + \mathbf{H}_{S^\perp}\|_F \\ & \leq \|\mathbf{H}_T\|_F + \|\mathbf{H}_{T^\perp \cap S}^{(1)}\|_F + \|\mathbf{H}_{S^\perp}^{(1)}\|_F + \sum_{i=2}^{M_1} \|\mathbf{H}_{T^\perp \cap S}^{(i)}\|_F + \sum_{i=2}^{M_2} \|\mathbf{H}_{S^\perp}^{(i)}\|_F \\ & \leq \gamma \left((1 + \sqrt{2})\|\mathbf{H}_T\| + 2\|\mathbf{Z}_c\|_* + 2\lambda\|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_* \right) + \frac{\sqrt{3}}{(1 - \delta_{2K_1, 2K_2}^{lb})} \frac{2\eta}{m}, \end{aligned}$$

where

$$\gamma = \frac{(1 + \delta_{K_1, K_2}^{ub})\sqrt{3}}{(1 - \delta_{2K_1, 2K_2}^{lb})\sqrt{K_1}} + \frac{1 + \delta_{K_1, K_2}^{ub}}{(1 - \delta_{K_1, K_2}^{lb})\sqrt{K_1}}.$$

We thus have that

$$\begin{aligned} \|\mathbf{H}\|_F & \leq \frac{2\gamma}{1 - (1 + \sqrt{2})\gamma} (\|\mathbf{Z}_c\|_* + \lambda\|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_*) + \frac{1}{1 - (1 + \sqrt{2})\gamma} \frac{\sqrt{3}}{(1 - \delta_{2K_1, 2K_2}^{lb})} \frac{2\eta}{m} \\ & \leq C \left(\|\mathbf{Z}_c\|_* + \lambda\|\mathcal{R}^{-1}(\mathbf{Z}_c)\|_* + \frac{\eta}{m} \right) \end{aligned}$$

provided that $\gamma < 1/(1 + \sqrt{2})$. \square

We notice that the proof provides loose recovery bounds since the proof implies that λ should be chosen as small as possible. Theorem 8.2 follows directly from 8.7.1 using e.g. the techniques in the proof of Theorem 1 in [CCG15] and is therefore omitted.

8.7.4 Proof of Theorem 8.2

Proof: We note that

$$\text{tr}(\mathbf{a}_i \mathbf{a}_i^\top \mathcal{R}^{-1}(\mathbf{Z})) = \text{tr}((\mathbf{A}_i \otimes \mathbf{A}_i)^\top \mathbf{W}),$$

where $\mathbf{W} = \mathcal{R}^{-1}(\mathbf{Z})$. We can thus express the measurements as $\mathbf{y} = \mathcal{A}_K(\mathbf{W}) + \mathbf{n}$. The program (8.1) in the limit $\lambda \rightarrow \infty$ can be written as

$$\begin{aligned} & \min \|\mathbf{W}\|_*, \\ & \text{such that } \|\mathbf{y} - \mathcal{A}_K(\mathbf{W})\|_1 \leq \eta, \mathcal{R}(\mathbf{W}) \succeq \mathbf{0}, \end{aligned} \quad (8.12)$$

where $\mathcal{A}_K(\mathbf{W})_i = \text{tr}((\mathbf{A}_i \otimes \mathbf{A}_i)^\top \mathbf{W})$. Let $\hat{\mathbf{W}} = \mathbf{W} + \mathbf{H} \in \mathbb{R}^{p^2 \times q^2}$ be the minimizer of (8.12). Let \mathbf{W}_s be the best rank- s approximation of \mathbf{W} and set $\mathbf{W}_c = \mathbf{W} - \mathbf{W}_s$. Let $\mathbf{W}_s = \mathbf{U}_s \Sigma_s \mathbf{V}_s^\top$ be the SVD decomposition of \mathbf{W}_s . We find that

$$\begin{aligned} \|\mathbf{W}_T\|_* + \|\mathbf{W}_{T^\perp}\|_* & \geq \|\mathbf{W}\|_* \geq \|\mathbf{W} + \mathbf{H}\|_* \geq \|\mathbf{W}_T + \mathbf{H}\|_* - \|\mathbf{W}_{T^\perp}\|_* \\ & \geq \|\mathbf{W}_T\|_* - \|\mathbf{H}_T\|_* + \|\mathbf{H}_{T^\perp}\|_* - \|\mathbf{W}_{T^\perp}\|_*. \end{aligned}$$

This gives that

$$\|\mathbf{H}_{T^\perp}\|_* \leq 2\|\mathbf{W}_{T^\perp}\|_* + \|\mathbf{H}_T\|_*. \quad (8.13)$$

Decompose \mathbf{H}_{T^\perp} into orthogonal matrices

$$\mathbf{H}_{T^\perp} = \sum_{i=1}^M \mathbf{H}_{T^\perp}^{(i)},$$

such that

$$\begin{aligned} \sigma_{\min}(\mathbf{H}_{T^\perp}^{(i)}) & \geq \sigma_{\max}(\mathbf{H}_{T^\perp}^{(i+1)}), \\ \text{rank}(\mathbf{H}_{T^\perp}^{(i)}) & = K_1, \text{ for } i = 1, 2, \dots, M-1. \end{aligned}$$

We find that

$$\sum_{i \geq 2} \|\mathbf{H}_{T^\perp}^{(i)}\|_F \leq \frac{1}{\sqrt{K_1}} \sum_{i \geq 1} \|\mathbf{H}_{T^\perp}^{(i)}\|_* = \frac{1}{\sqrt{K_1}} \|\mathbf{H}_{T^\perp}\|_* \quad (8.14)$$

$$\leq \frac{1}{\sqrt{K_1}} (2\|\mathbf{W}_c\|_* + \|\mathbf{H}_T\|_*) \leq \frac{1}{\sqrt{K_1}} (2\|\mathbf{W}_c\|_* + \sqrt{s}\|\mathbf{H}_T\|_F). \quad (8.15)$$

This gives us that

$$\begin{aligned} \sum_{i=2}^M \frac{1}{m} \|\mathcal{B}_K(\mathbf{H}_{T^\perp}^{(i)})\|_1 &\leq (1 + \delta_{K_1}^{(ub)}) \sum_{i=2}^M \frac{1}{m} \|\mathbf{H}_{T^\perp}^{(i)}\|_F \\ &\leq \frac{1 + \delta_{K_1}^{(ub)}}{\sqrt{K_1}} \|\mathbf{H}_{T^\perp}\|_*. \end{aligned}$$

Similar to (8.11), we find that

$$\frac{1}{m} \|\mathcal{B}_K(\mathbf{H})\|_1 \leq \frac{2\eta}{m}.$$

This gives us that

$$\begin{aligned} \frac{2\eta}{m} &\geq \frac{1}{m} \|\mathcal{B}_K(\mathbf{H}_T + \mathbf{H}_{T^\perp}^{(1)})\|_1 - \sum_{i=2}^M \frac{1}{m} \|\mathcal{B}_K(\mathbf{H}_{T^\perp}^{(i)})\|_1 \geq \\ &(1 - \delta_{2K_1}^{(lb)}) \|\mathbf{H}_T + \mathbf{H}_{T^\perp}^{(1)}\|_F - \frac{1 + \delta_{K_1}^{(ub)}}{\sqrt{K_1}} \|\mathbf{H}_{T^\perp}\|_* \geq \\ &\frac{1 - \delta_{2K_1}^{(lb)}}{\sqrt{2}} \left(\|\mathbf{H}_T\|_F + \|\mathbf{H}_{T^\perp}^{(1)}\|_F \right) - \frac{1 + \delta_{K_1}^{(ub)}}{\sqrt{K_1}} \|\mathbf{H}_{T^\perp}\|_* \\ &\geq \frac{1 - \delta_{2K_1}^{(lb)}}{\sqrt{2}} \left(\|\mathbf{H}_T\|_F + \|\mathbf{H}_{T^\perp}^{(1)}\|_F \right) \\ &\quad - \frac{1 + \delta_{K_1}^{(ub)}}{\sqrt{K_1}} (2\|\mathbf{W}_c\|_* + \sqrt{s}\|\mathbf{H}_T\|_F). \end{aligned}$$

Rearranging the terms we get that

$$\begin{aligned} &\left(\frac{1 - \delta_{2K_1}^{(lb)}}{\sqrt{2}} - \sqrt{s} \frac{1 + \delta_{K_1}^{(ub)}}{\sqrt{K_1}} \right) \|\mathbf{H}_T\|_F + \frac{1 - \delta_{2K_1}^{(lb)}}{\sqrt{2}} \|\mathbf{H}_{T^\perp}^{(1)}\|_F \\ &\leq 2 \frac{1 + \delta_{K_1}^{(ub)}}{\sqrt{K_1}} \|\mathbf{W}_c\|_* + \frac{2\eta}{m}. \end{aligned}$$

We thus find that

$$\|\mathbf{H}_T\|_F + \|\mathbf{H}_{T^\perp}^{(1)}\|_F \leq \frac{2}{\beta} \left(\frac{1 + \delta_{K_1}^{(ub)}}{\sqrt{K_1}} \|\mathbf{W}_c\|_* + \frac{\eta}{m} \right). \quad (8.16)$$

Combining (8.14), (8.13) and (8.16) we finally get that

$$\begin{aligned} \|\mathbf{H}\|_F &\leq \|\mathbf{H}_T\|_F + \|\mathbf{H}_{T^\perp}^{(1)}\|_F + \sum_{i \geq 2} \|\mathbf{H}_{T^\perp}^{(i)}\|_F \\ &\leq \|\mathbf{H}_T\|_F + \|\mathbf{H}_{T^\perp}^{(1)}\|_F + \frac{1}{\sqrt{K_1}} (2\|\mathbf{W}_c\|_* + \sqrt{s}\|\mathbf{H}_T\|_F) \\ &\leq \left(\frac{C_1}{\beta} + C_2\right) \frac{\|\mathbf{W}_c\|_*}{\sqrt{K_2}} + \frac{C_3}{\beta} \frac{\eta}{m}. \end{aligned}$$

This establishes the bound.

8.7.5 Bounds for RIP conditions

Here we establish the probabilities and required number of measurements for the RIP to hold for the different cases. We first show the following proposition.

Proposition 8.7.2. *Let $\bar{\mathbf{B}}$ be the linear operator with components*

$$\bar{\mathbf{B}}(\mathbf{W})_i = \text{tr} \left(\bar{\mathbf{B}}_i^\top \mathbf{W} \right)$$

where \mathbf{W} is a $p^2 \times q^2$ matrix,

$$\bar{\mathbf{B}}_i = (\mathbf{A}_{2i} \otimes \mathbf{A}_{2i}) - (\mathbf{A}_{2i-1} \otimes \mathbf{A}_{2i-1})$$

and the elements of \mathbf{A}_k are i.i.d. sub-Gaussian variables. Then for every \mathbf{W} there exists positive constants c_1, c_2, c_3 such that

$$c_1 \|\mathbf{W}\|_F \leq \frac{1}{m} \|\bar{\mathbf{B}}(\mathbf{W})\|_1 \leq c_2 \|\mathbf{W}\|_F,$$

holds with probability at least $1 - \exp(-c_3 m)$.

Proof. To prove the proposition, we introduce the *commutation matrices* \mathbf{K}_p which are matrices such that, $\mathbf{K}_p \text{vec}(\mathbf{X}) = \text{vec}(\mathbf{X}^\top)$ for all $\mathbf{X} \in \mathbb{R}^{p \times p}$. It follows that $\mathbf{K}_p(\mathbf{X} \otimes \mathbf{Y})\mathbf{K}_q = (\mathbf{Y} \otimes \mathbf{X})$ when $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^{p \times q}$ [MN95]. Another useful operator is

$$\mathcal{T}(\mathbf{W}) = \mathcal{R}^{-1}(\text{diag}(\mathcal{R}(\mathbf{W}))),$$

which projects out certain components of the $p^2 \times q^2$ matrix \mathbf{W} . Here $\text{diag}(\cdot)$ denotes projection onto the closest diagonal matrix. The operator $\mathcal{T}(\cdot)$ can also be expressed using index-notation as

$$\mathcal{T}(\mathbf{W})_{i+(k-1)p, j+(l-1)q} = \delta_{i,k} \delta_{j,l} W_{i+(k-1)p, j+(l-1)q},$$

for $1 \leq i, k \leq p$ and $1 \leq j, l \leq q$.

From Lemma 7 (the Hanson-Wright Inequality) of [CCG15] it follows that $\bar{\mathbf{B}}(\mathbf{W})_i$ is a sub-exponential random variable with

$$\mathcal{E}[|\bar{\mathbf{B}}(\mathbf{W})_i|] \leq c'_1 \|\mathbf{W}\|_F^2. \quad (8.17)$$

To establish a lower bound on $\mathcal{E}[|\bar{\mathbf{B}}(\mathbf{W})_i|]$ we use that [CCG15]

$$\mathcal{E}[|\bar{\mathbf{B}}(\mathbf{W})_i|] \geq \sqrt{\frac{(\mathcal{E}[|\bar{\mathbf{B}}(\mathbf{W})_i|^2])^3}{c'_3 \|\mathbf{W}\|_F^2}}.$$

We have that

$$\begin{aligned} \mathcal{E}[\mathbf{A}_i \otimes \mathbf{A}_i] &= \mathcal{T}(\mathbf{1}\mathbf{1}^\top), \\ \mathcal{E}[(\mathbf{A}_i \otimes \mathbf{A}_i) \text{tr}((\mathbf{A}_i \otimes \mathbf{A}_i)^\top \mathbf{W})] &= \mathbf{W} + \mathbf{K}_{p^2} \mathbf{W} \mathbf{K}_{q^2} \\ &\quad + (\mathbf{1}^\top \mathcal{T}(\mathbf{W}) \mathbf{1}) \cdot \mathcal{T}(\mathbf{1}\mathbf{1}^\top) + (\mu_4 - 3) \mathcal{T}(\mathbf{W}). \end{aligned}$$

We thereby find that for all $1 \leq i \leq m$

$$\mathcal{E}[\bar{\mathbf{B}}_i \text{tr}(\bar{\mathbf{B}}_i^\top \mathbf{W})] = 4\mathbf{W} + 2(\mu_4 - 3) \mathcal{T}(\mathbf{W}).$$

From which it follows that

$$\mathcal{E}[|\bar{\mathbf{B}}(\mathbf{W})_i|^2] = E[\text{tr}(\mathbf{W}^\top \mathbf{B}_i \text{tr}(\mathbf{B}_i^\top \mathbf{W}))] \quad (8.18)$$

$$\begin{aligned} &= 4\|\mathbf{W}\|_F^2 + 2(\mu_4 - 3)\|\mathcal{T}(\mathbf{W})\|_F^2 \\ &\geq \min(4, 2(\mu_4 - 1))\|\mathbf{W}\|_F^2 = c_4 \|\mathbf{W}\|_F^2 \end{aligned} \quad (8.19)$$

So

$$\mathcal{E}[|\bar{\mathbf{B}}(\mathbf{W})_i|^2] \geq \sqrt{\frac{c_4^3}{c'_3}} \|\mathbf{W}\|_F = c'_2 \|\mathbf{W}\|_F. \quad (8.20)$$

Lemma 8 of [CCG15] (see also [Ver12]) gives us that with probability at least $1 - 2 \exp(-cm\epsilon)$

$$\left| \frac{1}{m} \|\bar{\mathbf{B}}(\mathbf{W})\|_1 - \frac{1}{m} \mathcal{E}[\|\bar{\mathbf{B}}(\mathbf{W})\|_1] \right| \leq \epsilon \|\mathbf{W}\|_F.$$

Together with (8.17), we get that

$$\begin{aligned} \frac{1}{m} \|\bar{\mathbf{B}}(\mathbf{W})\|_1 &\leq \frac{1}{m} \mathcal{E}[\|\bar{\mathbf{B}}(\mathbf{W})\|_1] + \epsilon \|\mathbf{W}\|_F \\ &\leq (c'_1 + \epsilon) \|\mathbf{W}\|_F = c_1 \|\mathbf{W}\|_F \end{aligned}$$

and together with (8.20), we have that

$$\begin{aligned} \frac{1}{m} \|\bar{\mathbf{B}}(\mathbf{W})\|_1 &\geq \frac{1}{m} \mathcal{E}[\|\bar{\mathbf{B}}(\mathbf{W})\|_1] - \epsilon \|\mathbf{W}\|_F \\ &\geq (c'_2 - \epsilon) \|\mathbf{W}\|_F = c_2 \|\mathbf{W}\|_F, \end{aligned}$$

with probability at least $1 - 2 \exp(-c_3 m)$ for $c_3 = c\epsilon$. \square

We want to derive the number of necessary measurements for the RIP to hold for matrices in the set

$$\mathcal{M}_{k,r} = \left\{ \mathbf{W} \in \mathbb{R}^{p^2 \times q^2} : \text{rank}(\mathbf{W}) \leq r^2, \right. \\ \left. \text{rank}(\mathcal{R}(\mathbf{W})) \leq k, \mathcal{R}^{-1}(\mathbf{W}) \succeq \mathbf{0} \right\}.$$

By using standard covering arguments as in [CP11], we find the following corollary.

Corollary 8.7.1. *For the sub-Gaussian sensing model, there exists positive constants C, c_1, c_2, c_3 such that*

$$1 - \delta_{k,r}^{lb} \geq c_1/2, \quad 1 + \delta_{k,r}^{ub} \leq 2c_2,$$

with probability at least $1 - \exp(-c_3 m)$ provided that

$$m > C \min \{ (p^2 + q^2 + 1)r^2, (2pq + 1)k \}.$$

Fast solution of the ℓ_1 -norm

Traditionally, the biggest problem to making informed decisions was to obtain data through observations and experiments. Now, with more inexpensive sensors and hardware, data can be collected at a much lower cost. The main problem now is instead to transmit, store and process the large amounts of data. The problem of handling and analyzing large amounts of data is often referred to as the *data deluge* and *Big Data* [Bar11]. In Big Data problems, machine learning methods are used to process large amounts of data to extract useful information. The large amounts of data requires that the algorithms are fast and memory efficient to be useful in practice. Sparse representations are useful for Big Data problems since they allow us to describe data in an *economical* way. To find sparse representations requires that our dictionary contains many atoms. For the standard sparse representation problem, the big data problem occurs when: (1) the number of atoms is very large, (2) the number of measurements is very large or (3) both the number of measurements and number of atoms is very large. Typically (2) can be solved using e.g. batched gradient descent methods while (1) and (3) require new approaches. In this chapter we will consider case (1) where the number of atoms is so large that the entire dictionary cannot be stored in the RAM memory of a computer. This means that the dictionary needs to be stored in the hard drive of the computer and only be accessed parts at the time. It is possible to adapt greedy algorithms such as OMP to the Big Data scenario using distributed computational methods [Sun14]. Here we show how the convex method Basis Pursuit (BP) can be applied to the Big Data problem by designing an algorithm which is fast and only needs to access parts of the dictionary in each iteration.

9.1 Introduction

A sparse representation is a special solution to a set of linear equations. Let $\mathbf{x} \in \mathbb{R}^n$ be a vector which solves the linear system of equations

$$\mathbf{y} = \mathbf{Ax}, \tag{9.1}$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a known sensing matrix and $\mathbf{y} \in \mathbb{R}^m$ is a vector containing the observed measurements. When \mathbf{x} has many components that are zero, the solution is a *sparse representation* of the measurement \mathbf{y} . Typically we need that $m \ll n$ for a sparse solution to be possible. The Basis Pursuit (BP) method finds a sparse solution to (9.1) through the convex program

$$\begin{aligned} \hat{\mathbf{x}}_{BP} = \arg \min \|\mathbf{x}\|_1 \\ \text{subject to } \mathbf{y} = \mathbf{A}\mathbf{x} \end{aligned} \quad (9.2)$$

The Basis Pursuit method can be shown to recover the maximally sparse solution under certain technical conditions [CT05, BDE09, Ela]. The problem of calculating the Basis Pursuit solution is an important problem in compressed sensing and many “off-the-shelf” methods exist that can solve the minimization problem. Common methods in the literature are interior-point methods and the simplex method [BV04, Dan98]. The advantage of these methods is that they can solve many different problems and have well established convergence properties. Efficient implementations such as CVX [GBY08] and Matlab’s `linprog` [MAT03] are also available. The disadvantage of these methods is that they are constructed for general optimization problems and therefore do not exploit the special structure of the Basis Pursuit problem. This means that the algorithms need to keep all problem variables in the memory at all times. This makes the standard algorithms unable to handle problems where the number of variables is very large.

As an example, consider the 128×128 Barbara image shown in Figure 9.1. In grayscale representation, the image is represented by 16384 pixels values. Images are usually compressed using source coding techniques such as JPEG which uses a truncated Discrete Cosine Transform (DCT). The image compression technique can be modified by e.g. representing the image in a wavelet dictionary. The question arises of which wavelet dictionary gives the best compression. In Figure 9.2 we show



Figure 9.1: The 128×128 Barbara image.



Figure 9.2: Thresholding compression of the image from Figure 9.1 for different dictionaries and compression ratios. The first column shows the component-wise representation, the second column shows the discrete cosine transform (DCT) representation, the third column shows the Haar wavelet representation and the fourth column shows the discrete Meyer (dmey) wavelet representation. The first rows show a truncation giving a 2 : 1 compression (50% of coefficients zero), the second row a 4 : 1 compression (75% of coefficients zero) and the third row a 10 : 1 compression (90% of coefficients zero).

how the image from Figure 9.1 changes with different compression ratios for four different dictionaries.

Ideally, we would like to compress the image in *all* dictionaries at the same time. The problem is that many good mathematical properties of wavelets are not preserved when combining different wavelet dictionaries, so the representation becomes more difficult to compute. Another problem is that the dictionary becomes very large when many wavelets are used. There are at least 8 wavelet families in Matlab [MMOP96] (the component basis and discrete cosine transform are usually not counted as wavelets, but we refer to them as wavelets for simplicity). The wavelet families and their wavelets are shown in Table 9.1.

To compress the Barbara image using the wavelets in Table 9.1 requires working with a 16384×901120 dictionary. The *undersampling ratio* of the problem is $\frac{1}{55} \approx 0.018 \approx 2\%$. The problem size makes it infeasible to use standard optimization methods. Also for the subsampled 64×64 image, which uses an 4096×225280 dictionary, is too large for standard methods. New approaches are therefore required to perform Basis Pursuit.

Table 9.1: Wavelet families in Matlab and their wavelets.

Wavelet family	Wavelets
Component basis	identity basis
Discrete Cosine Transform	dct
Daubechies	db1 (haar), db2, db3, . . . , db45
Coiflets	coif1, coif2, coif3, coif4, coif5
Symplets	sym2,sym3, . . . sym45
Discrete Meyer	dmey
Biorthogonal	bior1.1, bior1.3, bior1.5, bior2.2, bior2.4, bior2.6, bior2.8, bior3.1, bior3.3, bior3.5, bior 3.7, bior3.9, bior4.4, bior5.5, bior6.8
Reverse biorthogonal	rbio1.1, rbio1.3, rbio1.5, rbio2.2, rbio2.4, rbio2.6, rbio2.8, rbio3.1, rbio3.3, rbio3.5, rbio3.7, rbio3.9, rbio4.4, rbio5.5, rbio6.8

9.1.1 Prior work

Minimization of the ℓ_1 -norm is a problem of considerable importance in Compressed Sensing. Basis Pursuit and the LASSO are some of the most common algorithms for recovery of sparse solutions. This is because (or because of this) many efficient implementations exist allowing us to solve the minimization problem. Most often, these methods have been developed for the scenario of noisy measurements [DDDM04, CR, CW05, FNW07, HYZ07, BV04]. Methods have also been developed for large scale problems [KKL⁺07, BT09, BPC⁺11, MXAP12] using first order methods to lower the computational complexity or distributed algorithms to solve the problem using several computational nodes. Another approach is to lower the effective number of variables using *screening* principles [DP12, BERG14].

While convex methods are developed for solving the convex optimization problem of basis-pursuit and the LASSO, the methods are often slower than greedy search methods [MZ93, TG07, DM09, NT09]. Greedy methods solve the sparse recovery problem

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}} \|\mathbf{y} - \mathbf{A}\mathbf{x}\|_2, \quad \text{subject to } \|\mathbf{x}\|_0 \leq K,$$

in a greedy fashion by typically minimizing some objective in each iteration. Common greedy algorithms are matching pursuit (MP) [MZ93], orthogonal matching pursuit (OMP) [TG07], subspace pursuit (SP) [DM09] and Compressive Sampling matching pursuit (CoSamp) [NT09]. We note that the mentioned greedy algorithms require the sparsity level K to be known. Greedy search methods have the advantage that they can efficiently be implemented for very large scale problems using e.g. hierarchical search architectures [Sun14] since they do not require to keep all variables in memory. They can therefore (in theory) handle problems with arbitrarily

large number of variables.

To develop efficient methods for very large scale problems where the sparsity is unknown, it is desirable to combine the complexity of greedy search algorithms with the efficiency of convex optimization based methods. We now consider the geometry of the Basis Pursuit problem. This gives us the optimality conditions necessary to construct a more efficient method for large scale Basis Pursuit problems.

9.2 The geometry of basis pursuit

We assume throughout the chapter that the column vectors of \mathbf{A} have unit norm. Given an at most m -sparse vector \mathbf{x} such that $\text{supp}(\mathbf{x}) = J$, $|J| \leq m$ and $\mathbf{A}_J \mathbf{x}_J = \mathbf{y}$. When $\text{rank}(\mathbf{A}_J) = |J|$, we can enlarge the set J to a set I such that $J \subset I$, $|I| = m$ and \mathbf{A}_I is full rank (since \mathbf{A} is full rank). Thus $\mathbf{x}_I = \mathbf{A}_I^{-1} \mathbf{y}$. Since we can always enlarge the support set and the Basis Pursuit solution always has at most m non-zero components [Ela], Basis Pursuit (9.2) can equivalently be expressed as

$$I = \arg \min_{|I'|=m} \|\mathbf{A}_{I'}^{-1} \mathbf{y}\|_1, \text{ s.t. } \mathbf{A}_{I'} \text{ is invertible} \quad (9.3)$$

$$(\hat{\mathbf{x}}_{BP})_I = \mathbf{A}_I^{-1} \mathbf{y}, \quad (\hat{\mathbf{x}}_{BP})_{I^c} = \mathbf{0}.$$

We find that (9.3) is an exhaustive search over all subsets $I \subset [n]$ of size $|I| = m$. However, because of the geometry of the Basis Pursuit problem, many subsets can be eliminated from the search.

Let $\mathcal{C}(I, \mathbf{s})$ denote the *convex cone*

$$\mathcal{C}(I, \mathbf{s}) = \left\{ \mathbf{A}_I \mathbf{r} \mid \mathbf{r} \in \mathbb{R}^{|I|}, r_i s_i \geq 0 \forall i \in I \right\}$$

of \mathbf{A} , where $|I| \leq m$ and $\mathbf{s} = (s_1, s_2, \dots, s_{|I|})$ with $s_i = \pm 1$ for $1 \leq i \leq m$. We say that a cone $\mathcal{C}(I, \mathbf{s})$ is *minimal* (in \mathbf{A}) if there is no $j \notin I$ such that the column vector \mathbf{a}_j of \mathbf{A} lies in $\mathcal{C}(I, \mathbf{s})$, i.e. there is no solution to $\mathbf{a}_j = \mathbf{A}_I \mathbf{x}'_I$ with $\text{sign}(\mathbf{x}'_I) = \mathbf{s}$. An important property of Basis Pursuit is that the solution is always contained in a minimal cone. We formulate this as a proposition.

Proposition 9.2.1. *The support set of $\hat{\mathbf{x}}_{BP}$ is contained in a minimal cone $\mathcal{C}(I, \mathbf{s})$, where $\text{supp}(\hat{\mathbf{x}}_{BP}) = J \subset I$ and $\text{sign}(\hat{x}_{j_k}) = s_k$ for $k = 1, 2, \dots, |J|$.*

The proof is given in Section 9.5. An illustration in two dimensions is given in Figure 9.3. In two dimensions, the Basis Pursuit solution is given by the unique minimal cone while in higher dimensions, the minimal cone need not be unique. When \mathbf{y} has a sparse representation, then \mathbf{y} lies on the boundary of several minimal cones. Proposition 9.2.1 implies that it is sufficient to search over all minimal cones in (9.3). It also implies that if Basis Pursuit recovers an m -sparse vector \mathbf{x} from measurements $\mathbf{A}\mathbf{x}$, then Basis Pursuit also recovers any other vector \mathbf{x}' with $\text{supp}(\mathbf{x}') \subset \text{supp}(\mathbf{x})$ and $\text{sign}(\mathbf{x}'_J) = \text{sign}(\mathbf{x}_J)$, i.e. (9.2) and (9.3) gives a solution with $\text{supp}(\hat{\mathbf{x}}_{BP}) \subset I$ and $(\hat{\mathbf{x}}_{BP})_i s_i \geq 0$ for all $i \in I$ and \mathbf{y} in $\mathcal{C}(I, \mathbf{s})$.

9.3 Greedy l_1 -minimization

Since Basis Pursuit gives a solution with the same support set and sign-pattern for all measurements inside the cone containing the Basis Pursuit solution, we cannot interchange a vector in the support set for a vector in the complement to lower the l_1 -norm. This can be verified without explicitly computing the new solutions, as the following theorem explains.

Theorem 9.3.1. *Let \mathbf{x} be an m -sparse solution to $\mathbf{y} = \mathbf{A}\mathbf{x}$ with $I = \text{supp}(\mathbf{x})$ and $\mathbf{s} = \text{sign}(\mathbf{x}_I)$. Then the l_1 -norm of the solution cannot be lowered by replacing the k 'th column vector of \mathbf{A}_I for a vector \mathbf{a}_j ($j \notin I$) if*

$$1 \geq \text{sign}(s_k z_k)(\mathbf{s}^\top \mathbf{z}), \quad (9.4)$$

where $\mathbf{z} = \mathbf{A}_I^{-1} \mathbf{a}_j$. Furthermore, if (9.4) is satisfied with strict inequality for all k and \mathbf{a}_j ($j \in I^c$), then $|\mathbf{s}^\top \mathbf{z}| < 1$ and $\mathbf{x} = \hat{\mathbf{x}}_{BP}$.

The proof of Theorem 9.3.1 is given in Section 9.5. Noting that we can write

$$\mathbf{s}^\top \mathbf{z} = \mathbf{s}^\top \mathbf{A}_I^{-1} \mathbf{a}_j = \mathbf{h}^\top \mathbf{a}_j$$

where $\mathbf{h} = (\mathbf{A}_I^\top)^{-1} \mathbf{s}$, we find that if $|\mathbf{h}^\top \mathbf{a}_j| \leq 1$, then no column vector in \mathbf{A}_I can be replaced by \mathbf{a}_j to lower the l_1 -norm. By Theorem 9.3.1, if $|\mathbf{h}^\top \mathbf{a}_j| < 1$ for all $j \in I^c$, then $\mathbf{x} = \hat{\mathbf{x}}_{BP}$.

9.3.1 The GL1 algorithm

Using (9.4) we construct the *greedy algorithm for l_1 -minimization*, GL1. The algorithm starts with an initial *active set* I and use I to construct an *intermediate certificate* \mathbf{h} . The algorithm searches for candidate vectors in the complement I^c that satisfy $|\mathbf{h}^\top \mathbf{a}_j| > 1$. The candidate vectors are tested (for all k such that (9.4)

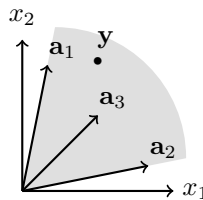


Figure 9.3: The cone $\mathcal{C}(\{1, 2\}, (1, 1)^\top)$ is not minimal since it contains \mathbf{a}_3 . Basis-pursuit gives a solution with support set $\{1, 3\}$ since it is the only minimal cone containing \mathbf{y} .

is violated) if replacing \mathbf{a}_{i_k} by the candidate vector lowers the l_1 -norm, i.e. if

$$\|\mathbf{x}'\|_1 = \|\mathbf{A}_{I'}^{-1} \mathbf{A}_I \hat{\mathbf{x}}\|_1 = \left\| \hat{\mathbf{x}} + (\mathbf{e}_k - \mathbf{z}) \frac{\hat{x}_k}{z_k} \right\|_1 < \|\hat{\mathbf{x}}\|_1, \quad (9.5)$$

where \mathbf{e}_k denotes the k 'th basis vector in the coordinate basis. When no candidate vector lowers the l_1 -norm, the algorithm terminates.

The algorithm may get stuck in a local optima (of the algorithm) if it encounters a sparse solution. If this happens, we slightly perturb \mathbf{y} to ensure that it lies in the interior of a cone. When the algorithm terminates, \mathbf{y} is restored and the solution is recomputed. The GL1 algorithm can be summarized as follows.

1. Input: $\mathbf{y}, \mathbf{A}, I, \Delta, \epsilon$.
2. Initialization: $\mathbf{y}' = \mathbf{y}$.
3. Repeat:
4. $\hat{\mathbf{x}} = \mathbf{A}_I^{-1} \mathbf{y}'$, $\mathbf{s} = \text{sign}(\hat{\mathbf{x}})$, $l_{min} = \|\hat{\mathbf{x}}\|_1$, $\mathbf{h} = (\mathbf{A}_I^\top)^{-1} \mathbf{s}$.
5. If $\|\hat{\mathbf{x}}\|_0 < m$: perturb $\mathbf{y}' = \mathbf{y} + \Delta \cdot \mathbf{A}_I \mathbf{1}$, go to 4.
6. For all $j \in I^c$ such that $|\mathbf{h}^\top \mathbf{a}_j| > 1$:
 - a) Compute $\mathbf{z} = \mathbf{A}_I^{-1} \mathbf{a}_j$ and $\mathbf{t} = \text{sign}(\mathbf{s} \circ \mathbf{z})(\mathbf{s}^\top \mathbf{z})$.
 - b) For all k such that $t_k > 1$ and $|z_k| > \epsilon$:
 - i. If $\|\hat{\mathbf{x}} + (\mathbf{e}_k - \mathbf{z}) \frac{\hat{x}_k}{z_k}\|_1 < l_{min}$:
 - A. $I \rightarrow (I \cup \{j\}) \setminus \{i_k\}$,
 - B. $I^c \rightarrow (I^c \cup \{i_k\}) \setminus \{j\}$, go to 4.
7. If $\|\hat{\mathbf{x}}\|_1 = l_{min}$: break.
8. Output: $I, \hat{\mathbf{x}}_I = \mathbf{A}_I^{-1} \mathbf{y}$.

Notes on the algorithm:

- $\Delta, \epsilon > 0$ are small constants (e.g. 10^{-5}).
- Usually, $\hat{\mathbf{x}}$ is not exactly sparse due to numerical errors. One can then perturb the measurements as

$$\mathbf{y}' = \mathbf{y} + \Delta \cdot \mathbf{A} \mathbf{s},$$

where $\mathbf{s} = \text{sign}(\hat{\mathbf{x}})$.

- In implementation, matrix inverses are replaced by solving the set of linear equations, to increase numerical accuracy and speed.

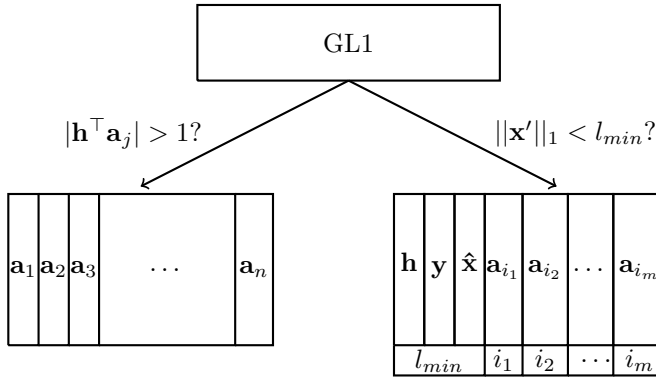


Figure 9.4: Schematic illustration of the GL1 algorithm. The algorithm reads from the complement set and computes using the active set.

- The runtime can be decreased by selecting the initial set I in a good way. In simulations we chose I to be the m vectors which have largest inner products $|\mathbf{a}_i^\top \mathbf{y}|$. Another possibility is to find the initial I using a greedy algorithm.
- When determining if the candidate vectors can lower the l_1 -norm, it is beneficial to start with the vector with largest inner product $|\mathbf{h}^\top \mathbf{a}_j|$, then proceed to the one with next largest inner product and so on.

We note that the algorithm consists of two parts, determining if $|\mathbf{h}^\top \mathbf{a}_j| > 1$ and if replacing a column vector with \mathbf{a}_j lowers the l_1 -norm, see Figure 9.4. Both of these operations can be parallelized, making the algorithm suitable for large scale problems.

9.4 Numerical comparison

In this section we numerically compare different solution methods for Basis Pursuit. We compare GL1 to the two main approaches, the interior point method and the simplex method. Lastly we consider the problem of compressing an image in several wavelet dictionaries.

9.4.1 Compressed sensing

In the first simulation we investigated the speed of different implementations of Basis Pursuit. We compared GL1 to three other solvers for Basis Pursuit, the simplex method [Dan98], l_1 -magic [CR] and Iterative Reweighted Least Squares (IRLS) [DDFG10].

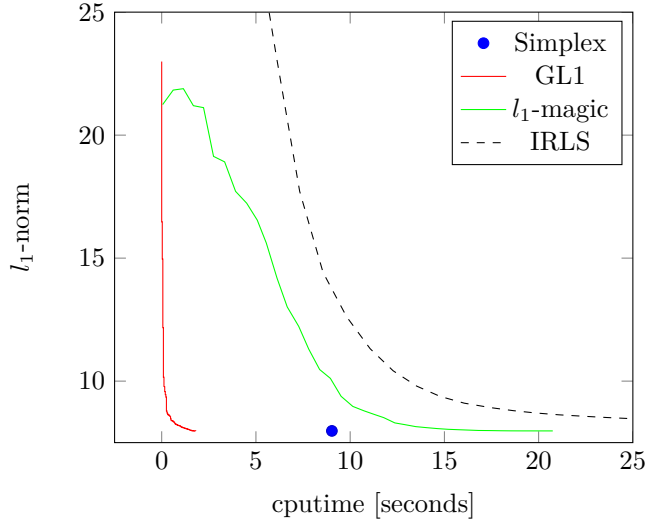


Figure 9.5: l_1 -norm vs. cputime for one problem realization for the different methods when $n = 8000$ and $m = 50$.

The Simplex method formulates (9.2) as

$$\begin{aligned} \min \quad & \mathbf{1}^\top (\mathbf{x}_+ + \mathbf{x}_-), \\ \text{s.t.} \quad & \mathbf{y} = \mathbf{A}(\mathbf{x}_+ - \mathbf{x}_-) \\ & \mathbf{x}_+, \mathbf{x}_- \geq \mathbf{0} \end{aligned}$$

where $\mathbf{1} \in \mathbb{R}^n$ is a vector of ones and we used Matlab's `linprog` to run the simplex algorithm. The method l_1 -magic formulates (9.2) as

$$\begin{aligned} \min \quad & \mathbf{1}^\top \mathbf{t} \\ \text{s.t.} \quad & \mathbf{y} = \mathbf{A}\mathbf{x} \\ & -t_i \leq x_i \leq t_i, \text{ for } i = 1, 2, \dots, n \end{aligned}$$

and solves the optimization problem using a primal-dual interior point method [CR]. The IRLS algorithm approximates the l_1 -norm by a weighted l_2 -norm which is updated iteratively [DDFG10].

In the simulation we used $n = 8000$ and varied the number of measurements m . The measurements were generated by drawing the elements of \mathbf{A} from a $\mathcal{N}(0, 1)$ distribution and normalizing the column vectors, the vector \mathbf{x} was generated as a $[0.25m]$ -sparse vector with its non-zero elements drawn from a $\mathcal{N}(0, 1)$ distribution. Finally we computed $\mathbf{y} = \mathbf{A}\mathbf{x}$ and compute the solution using the different algorithms. Because \mathbf{A} is Gaussian, $\hat{\mathbf{x}}_{BP}$ in (9.2) is unique with probability 1.

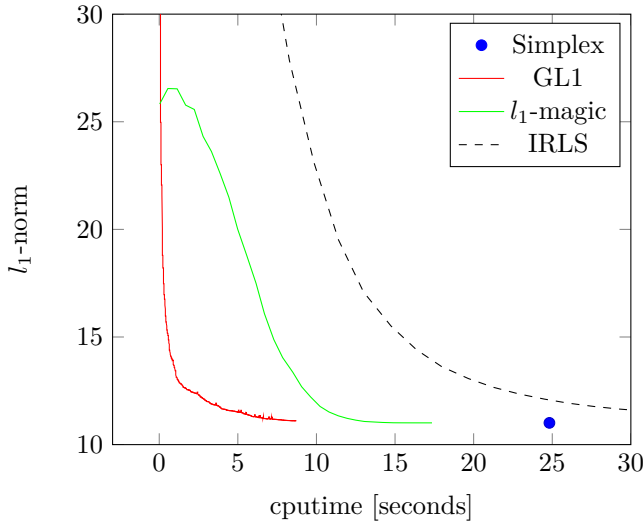


Figure 9.6: ℓ_1 -norm vs. cputime for one problem realization for the different methods when $n = 8000$ and $m = 100$.

Because the algorithms have different complexity per iteration, we measure the total cputime and the ℓ_1 -norm in each iteration rather than the number of iterations. We were not able to access the intermediate times and ℓ_1 -norms of `linprog` and therefore only display the final norm and cputime of the method. We show the convergence for one problem realizations with $m = 50$ in Figure 9.5 and with $m = 100$ in Figure 9.6. We find that the simplex method converged faster than ℓ_1 -magic for $m = 50$, but slower for $m = 100$. In both realizations, GL1 was the fastest algorithm. In Figure 9.7 we show the average cputime for different values of m averaged over 10 realizations of \mathbf{A} and \mathbf{x} . We see that ℓ_1 -magic is slower than the simplex method for $m \leq 80$ and slower than GL1 for $m \leq 140$. The simplex method was about 3 times slower than GL1 for all values of m .

9.4.2 Image compression in wavelet dictionaries

Next we demonstrate that GL1 can solve large scale problems with many parameters. We decompose an image in a dictionary that is too big to store in memory by performing a decomposition of the Barbara image in the wavelets mentioned in Table 9.1. To make the simulation run faster, we subsampled the image to the 64×64 image shown in Figure 9.8. When representing the image in the separate wavelet dictionaries we find the norms in Table 9.2. We find that the discrete Meyer basis gives the highest ℓ_1 -norm and that the component basis also gives a high ℓ_1 -norm. The db1 (Haar), bior1.1 and rbio1.1 wavelets gives a low ℓ_1 -norm and the DCT basis gives the lowest ℓ_1 -norm of all wavelets. This is to be expected since image

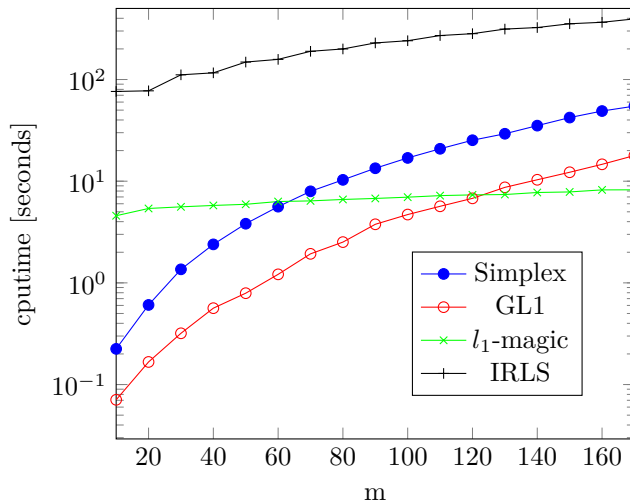


Figure 9.7: Average cputime vs. m for the different methods when $n = 8000$.

compression methods often use a truncated DCT transforms.

We initialized the algorithm by choosing the initial dictionary to be the DCT basis. In each iteration we computed the certificate and chose a wavelet dictionary at uniformly at random. We then used GL1 to attempt to lower the ℓ_1 -norm using atoms from the chosen dictionary. If the ℓ_1 -norm did not change after 470 iterations, we slightly perturbed the original image and continued to run the algorithm. By waiting 470 iterations, we ensure that the probability that not all wavelet bases are tested to be less than 1%. In the iterations of the GL1 algorithm, the ℓ_1 -norm decreases as shown in Figure 9.9.

After 32320 iterations, the GL1 representation has an ℓ_1 -norm of 59750, i.e. 15.4% lower than the ℓ_1 -norm of the representation in the DCT basis. As the algorithm has not yet converged, we expect the norm to decrease further after further iterations. The GL1 representation has 61 coefficients in the component basis (1.5%) and 1459 coefficients in the DCT basis (35.6%). The individual representations in the different wavelet families are shown in Figure 9.10. We find that most wavelets are in the original DCT basis. However, the number of wavelets in the DCT basis decreases in each iteration, it is therefore reasonable to believe that the number of wavelets in the DCT basis will decrease further when the algorithm makes more iterations. We find that the GL1 representation has the same sparsity as the DCT representation with 12% of the ℓ_1 -norm contained in the largest component, 2% in the second largest component and 1% in the third largest component.

The simulation shows that the GL1 algorithm is able to perform large scale Basis Pursuit when the dictionary is too large to fit into memory. Since the wavelet bases are related to fast transforms, we could compute the inner products without

Table 9.2: The ℓ_1 -norm of representations of the 64×64 Barbara image in the wavelet bases in Table 9.1 (ordered by magnitude). The norms have been rounded to nearest integer.

Basis	ℓ_1 -norm	Basis	ℓ_1 -norm	Basis	ℓ_1 -norm	Basis	ℓ_1 -norm
dmey	1693960	db8	399327	bior1.5	344778	bior1.3	307053
coif5	556687	sym8	392325	rbio2.4	341635	coif1	300100
component	486667	bior3.7	385719	rbio4.4	337371	rbio1.3	298401
coif4	485687	rbio2.6	380308	rbio3.3	336769	db3	297926
rbio3.9	450395	db7	376290	db5	334824	sym3	297926
db10	444615	sym7	372536	rbio1.5	334175	bior2.2	294587
bior3.9	425547	bior2.6	370869	sym5	334105	db2	283100
db9	422238	rbio3.5	369614	rbio3.1	332651	sym2	283100
rbio2.8	420781	db6	354507	bior4.4	331900	bior3.1	274027
coif3	416167	bior5.5	354391	bior2.4	331009	db1	271132
rbio6.8	415456	coif2	354373	sym4	316144	bior1.1	271132
bior6.8	410962	rbio5.5	353642	db4	315180	rbio1.1	271132
bior2.8	410550	sym6	352550	rbio2.2	310414	DCT	70642
rbio3.7	408653	bior3.5	346140	bior3.3	308735	GL1	59750

explicitly computing the wavelet dictionaries. With increasing problem sizes, it is expected that special algorithms, like GL1, will be more important and relevant.



Figure 9.8: The 64×64 subsampled Barbara image.

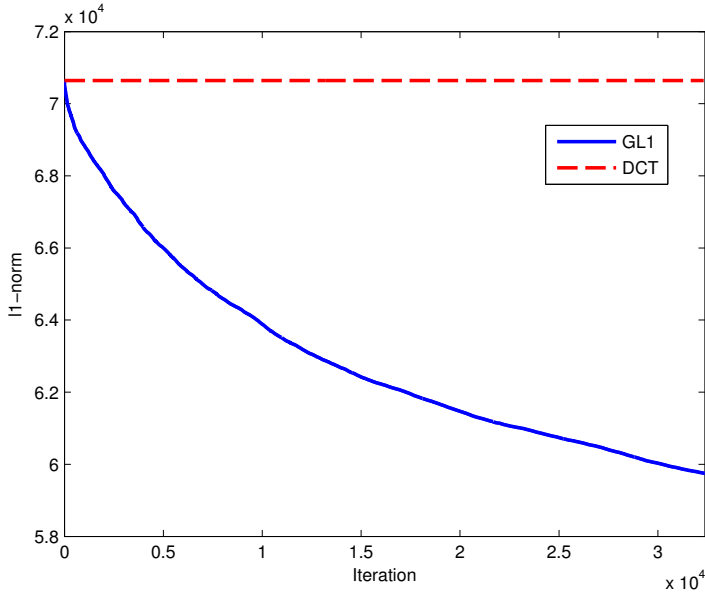


Figure 9.9: The ℓ_1 -norm of the GL1 representation of the 64×64 Barbara image in different iterations. The ℓ_1 -norm of the representation in the DCT basis is shown as a dashed line.

9.5 Derivations and proofs

Proof of proposition 1. Let I be a set with m elements and let $I' = (I \setminus \{k\}) \cup \{j\}$. We need to show that if $\mathbf{y} \in \mathcal{C}(I', \mathbf{s}') \subset \mathcal{C}(I, \mathbf{s})$, then $\|\hat{\mathbf{x}}_{I'}\|_1 < \|\hat{\mathbf{x}}_I\|_1$. Set

$$\mathbf{y} = \sum_{i \in I} x_i \mathbf{a}_i, \quad (9.6)$$

$$\mathbf{a}_j = \sum_{i \in I} z_i \mathbf{a}_i, \quad (9.7)$$

$$\mathbf{y} = x'_j \mathbf{a}_j + \sum_{i \in I, i \neq k} x'_i \mathbf{a}_i. \quad (9.8)$$

Without loss of generality we can assume that $x_i, z_i, x'_i \geq 0$ for all $i \in I \cup \{j\}$. By inserting (9.7) into (9.8) we find that

$$x'_j \mathbf{a}_j + \sum_{i \in I, i \neq k} x'_i \mathbf{a}_i = x'_j z_k \mathbf{a}_k + \sum_{i \in I, i \neq k} (x'_i + x'_j z_i) \mathbf{a}_i = \sum_{i \in I} x_i \mathbf{a}_i.$$

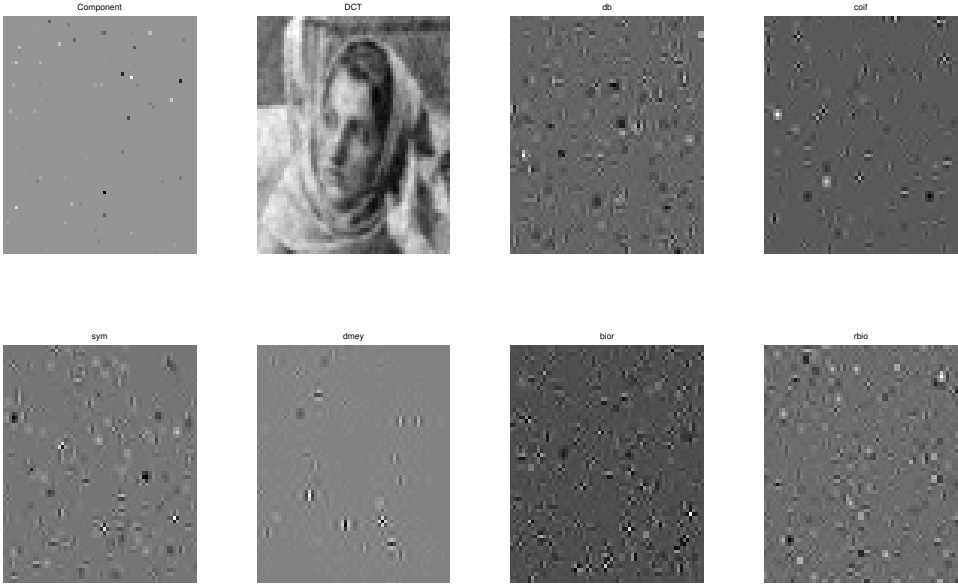


Figure 9.10: The GL1 representation of the 64×64 Barbara image after 32320 iterations in the wavelet families from Table 9.1.

We find that when \mathbf{A}_I is full rank, then $x_k = x'_j z_k$ and $x_i = x'_i + x'_j z_i$ for $i \neq k$. Using that

$$1 = \|\mathbf{a}_j\|_2 < \sum_{i \in I} |z_i| \cdot \|\mathbf{a}_i\|_2 = \|\mathbf{z}\|_1,$$

where we have strict inequality because the column vectors in \mathbf{A}_I are not parallel, we find that

$$\begin{aligned} \|\mathbf{x}\|_1 &= \sum_{i \in I} x_i = x'_j z_k + \sum_{i \in I, i \neq k} (x'_i + x'_j z_i) \\ &= x'_j \sum_{i \in I} z_i + \sum_{i \in I, i \neq k} x'_i \\ &= x'_j (\|\mathbf{z}\|_1 - 1) + \|\mathbf{x}'\|_1 > \|\mathbf{x}'\|_1, \end{aligned}$$

provided that $x'_j > 0$. □

Proof of Theorem 1. Assume that we cannot interchange the k 'th column vector of \mathbf{A}_I for a vector \mathbf{a}_j to get a solution with lower l_1 -norm. Then all \mathbf{y} in $\mathcal{C}(I, \mathbf{s})$ give Basis Pursuit solutions with the same support set and sign-pattern, i.e. for all $\mathbf{w} \in \mathbb{R}^m$, $\mathbf{w} \geq \mathbf{0}$

$$\|\mathbf{w}\|_1 \leq \|\mathbf{A}_I^{-1} \mathbf{A}_I \mathbf{S} \mathbf{w}\|_1, \quad (9.9)$$

where $\mathbf{S} = \text{diag}(\mathbf{s})$, $I' = (I \cup \{j\}) \setminus \{i_k\}$ and we assumed that $\mathbf{A}_{I'}$ is invertible.

Replacing \mathbf{a}_{i_k} for \mathbf{a}_j corresponds to making a rank-1 update of \mathbf{A}_I , i.e.

$$\mathbf{A}_{I'} = \mathbf{A}_I + (\mathbf{a}_j - \mathbf{a}_{i_k})\mathbf{e}_k^\top.$$

Setting $\mathbf{z} = \mathbf{A}_I^{-1}\mathbf{a}_j$, we get that

$$\begin{aligned} \mathbf{A}_{I'}^{-1}\mathbf{A}_I\mathbf{S}\mathbf{w} &= \left(\mathbf{A}_I^{-1} - \frac{\mathbf{A}_I^{-1}(\mathbf{a}_j - \mathbf{a}_{i_k})\mathbf{e}_k^\top\mathbf{A}_I^{-1}}{1 + \mathbf{e}_k^\top\mathbf{A}_I^{-1}(\mathbf{a}_j - \mathbf{a}_{i_k})} \right) \mathbf{A}_I\mathbf{S}\mathbf{w} \\ &= \mathbf{S}\mathbf{w} - \frac{(\mathbf{A}_I^{-1}\mathbf{a}_j - \mathbf{e}_k)\mathbf{e}_k^\top\mathbf{S}\mathbf{w}}{\mathbf{e}_k^\top\mathbf{A}_I^{-1}\mathbf{a}_j} = \mathbf{S}\mathbf{w} + (\mathbf{e}_k - \mathbf{z})\frac{s_k w_k}{z_k}. \end{aligned}$$

So if (9.9) holds, then

$$\begin{aligned} 1 &\leq \min_{\substack{\mathbf{w} \geq \mathbf{0} \\ \mathbf{1}^\top \mathbf{w} = 1}} \left\| \mathbf{S}\mathbf{w} + (\mathbf{e}_k - \mathbf{z})\frac{s_k w_k}{z_k} \right\|_1 \\ &= \min_{\substack{\mathbf{w} \geq \mathbf{0} \\ \mathbf{1}^\top \mathbf{w} = 1}} \frac{w_k}{|z_k|} + \sum_{l \in I \setminus \{i_k\}} \left| s_l w_l - \frac{z_l}{z_k} s_k w_k \right|. \end{aligned}$$

Using that

$$\left| s_l w_l - \frac{z_l}{z_k} s_k w_k \right| \geq \begin{cases} 0 & , \text{ if } z_l s_l z_k s_k > 0 \\ \frac{|z_l|}{|z_k|} w_k & , \text{ else} \end{cases} ,$$

we find that (9.9) holds for all \mathbf{w} if

$$\begin{aligned} \min_{\substack{\mathbf{w} \geq \mathbf{0} \\ \mathbf{1}^\top \mathbf{w} = 1}} \frac{w_k}{|z_k|} + \sum_{l \in I \setminus \{i_k\}} \left| s_l w_l - \frac{z_l}{z_k} s_k w_k \right| &= \\ \min_{\substack{w_i \geq 0 \\ w_k / |z_k| (|z_k| + \sum_{l \in J_+} |z_l|) = 1}} \frac{w_i}{|z_i|} \left(1 + \sum_{l \in J_-} |z_l| \right) & \\ = \frac{1 + \sum_{l \in J_-} |z_l|}{|z_k| + \sum_{l \in J_+} |z_l|} \geq 1, & \tag{9.10} \end{aligned}$$

where

$$\begin{aligned} J_+ &= \{l \mid l \in I \setminus \{i_k\}, z_l s_l z_k s_k > 0\}, \\ J_- &= \{l \mid l \in I \setminus \{i_k\}, z_l s_l z_k s_k \leq 0\}. \end{aligned}$$

Rewriting (9.10) as

$$\begin{aligned} \sum_{l: z_l s_l z_k s_k > 0} |z_l| &\leq 1 + \sum_{l: z_l s_l z_k s_k \leq 0} |z_l| \\ \Leftrightarrow 1 &\geq \sum_l \text{sign}(z_l s_l z_k s_k) |z_l| \\ &= \text{sign}(z_k s_k) \sum_l s_l z_l = \text{sign}(z_k s_k) (\mathbf{s}_I^\top \mathbf{z}), \end{aligned}$$

we recover the optimality condition (9.4).

To show that strict inequality in (9.4) implies that $|\mathbf{s}^\top \mathbf{z}| < 1$, assume that $|\mathbf{s}^\top \mathbf{z}| \geq 1$, but $\text{sign}(s_k z_k) (\mathbf{s}^\top \mathbf{z}) < 1$ for all k . This implies that $\text{sign}(s_k z_k) = -\text{sign}(\mathbf{s}^\top \mathbf{z})$ for all k . However, if all terms $s_k z_k$ have the same sign, then $\text{sign}(s_k z_k) = \text{sign}(\mathbf{s}^\top \mathbf{z})$, giving a contradiction. Thus, strict inequality in (9.4) implies that $|\mathbf{s}^\top \mathbf{z}| < 1$.

We find that

$$\mathbf{s}^\top \mathbf{z} = \mathbf{s}^\top \mathbf{A}_I^{-1} \mathbf{a}_j = \mathbf{h}^\top \mathbf{a}_j$$

where $\mathbf{h} = (\mathbf{A}_I^\top)^{-1} \mathbf{s}$. This gives that if $|\mathbf{h}^\top \mathbf{a}_j| < 1$ for all $j \in I^c$, then $\mathbf{A}_I^\top \mathbf{h} = \mathbf{s}$ and $\|\mathbf{A}_{I^c}^\top \mathbf{h}\|_\infty < 1$. The vector \mathbf{h} is thus the *dual certificate* of the Basis Pursuit solution [ZYC15], giving us that $\mathbf{x} = \hat{\mathbf{x}}_{BP}$. \square

9.6 Conclusion

In this chapter we considered the problem of solving the Basis Pursuit problem when the number of parameters is very large. Standard methods are efficient for solving the Basis Pursuit problem, but do not scale well when the number of parameters becomes very large. By considering the geometry of the problem we were able to derive optimality conditions for the Basis Pursuit solution. The optimality conditions were then used to construct the greedy minimization method GL1. The algorithm has the advantage that it does not need to keep all variables in memory, but only works with a subset of the parameters in each iteration. Through numerical simulations we showed that the algorithm is faster than the standard methods when the number of parameters is large. We also showed that the algorithm is able to perform the wavelet decomposition of an 64×64 image in several wavelet dictionaries.

Conclusion

As the amount of data and measurements increases, so does also the assistance we can receive from it to make well informed decisions. To make good use of the data we need good algorithms to help us process the data to provide us with the information we seek. Such algorithms needs to be fast, efficient and accurate. To meet such standards it is useful to exploit the internal structures of the problem to increase the performance and usability of the algorithms. Two of the most studied structures in compressed sensing is sparse vectors and low-rank matrices. These are the structures we have studied in this thesis. We used three approaches to designing estimation algorithms, greedy search algorithms, Bayesian methods and convex optimization based methods.

In Chapter 3 we considered greedy pursuit algorithms and how the way the algorithms detect non-zero components can be improved. We used two approaches, array processing and Bayesian filtering. In array processing, it is customary to cancel the contributions from sidelobes occurring in the problem to increase the resolution of the system. This is done using a beamformer, i.e. a linear filter which filters out the contribution of the sidelobes. With this methodology we were able to construct beamformers for the worst-case and average case detection scenario. Bayesian filtering often rely on linear filters to decrease the interference and noise. The Wiener filter is the optimal filter in the mean square sense and is heavily used in signal processing. By designing a Wiener filter for random sparse vectors conditioned on if a component is zero or not, we developed the Conditional prior OMP algorithm.

In Chapters 4, 5 and 6 we investigated Bayesian methods based on the Relevance Vector Machine (RVM). Bayesian methods use prior distributions to model the structure of the parameters. The difficulty is to model the structure in an appropriate way as the distribution should be continuous to give a tractable model while the structure often is non-continuous in nature. In Chapter 4 we showed how the RVM can be adapted to measurements with sparse noise without treating the noise as an additional estimation parameter. We showed that this increase the speed and accuracy of the algorithm. In Chapter 5 we used precision matrices to construct a

low-rank analogue of the RVM and related the prior distribution of the precision matrices to low-rank promoting penalty functions. In Chapter 6 we combined the methods of Chapter 4 and 5 to construct a Bayesian method for robust principal component analysis. We showed that the method can outperform standard methods but that this comes at the price of higher complexity.

In Chapter 8 we considered how low-rank can be incorporated in the phase retrieval problem. We showed that by using the theory of Kronecker product approximation, it is possible to promote rank in the underlying variable. Through numerical simulations we showed that by properly selecting the regularization parameter, it is possible to recover low-rank at fewer measurements than the with the standard method.

Lastly, in Chapter 9 we considered the *big data* problem of constructing a sparse representation when the number of parameters is very large. As standard optimization methods require all parameters to be kept in memory, the large number of parameters in the problems we consider make such methods infeasible. By deriving optimality conditions for the solution, we construct a greedy method for solving the ℓ_1 -norm minimization problem. The proposed method is faster than standard methods when the number of measurements is not too large and is able to perform wavelet decomposition on a 128×128 image in all wavelets in Matlab.

The methods used for compressed sensing are often divided into their respective classes, greedy, Bayesian and convex methods. In this thesis we have combined approaches to construct new methods. Chapter 3 combined greedy and Bayesian methods while Chapter 9 combines greedy and convex methods. The multitude of methods show that sparse and low-rank problems is a fascinating research area where methods from computer science, signal processing, machine learning and regression can be combined to give new insights into structured estimation problems. As every new solution gives rise to new problems to be explored. It is quite certain that different approaches can be further connected and that one method can solve the challenges presented by another method. As estimation methods and signal processing algorithms continue to improve, we have new possibilities to further develop new technology and new science for the future.

Bibliography

- [ADFDJ03] C. Andrieu, N. De Freitas, A. Doucet, and M. I. Jordan, “An introduction to mcmc for machine learning,” *Machine learning*, vol. 50, no. 1-2, pp. 5–43, 2003.
- [Alq13] P. Alquier, “Bayesian methods for low-rank matrix estimation: Short survey and theoretical study,” in *Algorithmic Learning Theory*. Springer, 2013, pp. 309–323.
- [AN07] A. Asuncion and D. Newman, “UCI machine learning repository,” 2007.
- [Bar11] R. G. Baraniuk, “More is less: signal processing and the data deluge,” *Science*, vol. 331, no. 6018, pp. 717–719, 2011.
- [BCDH10] R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde, “Model-based compressive sensing,” *Information Theory, IEEE Transactions on*, vol. 56, no. 4, pp. 1982–2001, 2010.
- [BCM14] A. S. Bandeira, J. Cahill, D. G. Mixon, and A. A. Nelson, “Saving phase: Injectivity and stability for phase retrieval,” *Applied and Computational Harmonic Analysis*, vol. 37, no. 1, pp. 106–125, 2014.
- [BD08] T. Blumensath and M. E. Davies, “In greedy pursuit of new directions:(nearly) orthogonal matching pursuit by directional optimisation,” in *Proc. European Signal Processing Conference (EUSIPCO)*, 2008.
- [BDE09] 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.
- [BERG14] A. Bonnefoy, V. Emiya, L. Ralaivola, and R. Gribonval, “A dynamic screening principle for the lasso,” in *Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European*. IEEE, 2014, pp. 6–10.
- [BGJM11] S. Brooks, A. Gelman, G. Jones, and X.-L. Meng, *Handbook of Markov Chain Monte Carlo*. CRC press, 2011.

- [Bis06] C. M. Bishop, *Pattern recognition and machine learning*. Springer New York, 2006, vol. 4, no. 4.
- [BL10] G. Bergqvist and E. G. Larsson, “The higher-order singular value decomposition: theory and an application [lecture notes],” *IEEE Signal Processing Magazine*, vol. 27, no. 3, pp. 151–154, 2010.
- [BLMK12] S. D. Babacan, M. Luessi, R. Molina, and A. K. Katsaggelos, “Sparse bayesian methods for low-rank matrix estimation,” *Signal Processing, IEEE Transactions on*, vol. 60, no. 8, pp. 3964–3977, 2012.
- [BPC⁺11] 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.
- [BS80] A. Borovkov and A. Sakhanienko, “On estimates of the expected quadratic risk,” *Probab. Math. Statist.*, vol. 1, pp. 185–195, 1980.
- [BT09] A. Beck and M. Teboulle, “A fast iterative shrinkage-thresholding algorithm for linear inverse problems,” *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.
- [BV04] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
- [CCG15] Y. Chen, Y. Chi, and A. J. Goldsmith, “Exact and stable covariance estimation from quadratic sampling via convex programming,” *IEEE Transactions on Information Theory*, vol. 61, no. 7, pp. 4034–4059, 2015.
- [CDS01] S. S. Chen, D. L. Donoho, and M. A. Saunders, “Atomic decomposition by basis pursuit,” *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.
- [CESV15] E. J. Candès, Y. C. Eldar, T. Strohmer, and V. Voroninski, “Phase retrieval via matrix completion,” *SIAM Review*, vol. 57, no. 2, pp. 225–251, 2015.
- [CLMW11] E. J. Candès, X. Li, Y. Ma, and J. Wright, “Robust principal component analysis?” *Journal of the ACM*, vol. 58, no. 3, p. 11, 2011.
- [CNZ05] P.-A. Chirita, W. Nejdl, and C. Zamfir, “Preventing shilling attacks in online recommender systems,” in *Proceedings of the 7th annual ACM international workshop on Web information and data management*. ACM, 2005, pp. 67–74.
- [CP10] E. J. Candès and Y. Plan, “Matrix completion with noise,” *Proceedings of the IEEE*, vol. 98, no. 6, pp. 925–936, 2010.

- [CP11] —, “Tight oracle inequalities for low-rank matrix recovery from a minimal number of noisy random measurements,” *Information Theory, IEEE Transactions on*, vol. 57, no. 4, pp. 2342–2359, 2011.
- [CR] E. Candès and J. Romberg, “l1-magic: Recovery of sparse signals via convex programming,” *URL: www.acm.caltech.edu/l1magic/downloads/l1magic.pdf*.
- [CR09] E. J. Candès and B. Recht, “Exact matrix completion via convex optimization,” *Foundations of Computational mathematics*, vol. 9, no. 6, pp. 717–772, 2009.
- [Cra47] H. Cramér, “Mathematical methods of statistics,” 1947.
- [CRPW12] V. Chandrasekaran, B. Recht, P. A. Parrilo, and A. S. Willsky, “The convex geometry of linear inverse problems,” *Foundations of Computational mathematics*, vol. 12, no. 6, pp. 805–849, 2012.
- [CRT06] E. J. Candès, 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.
- [CSGC15] A. Casamitjana, M. Sundin, P. Ghosh, and S. Chatterjee, “Bayesian learning for time-varying linear prediction of speech,” in *Signal Processing Conference (EUSIPCO), 2015 23rd European*. IEEE, 2015, pp. 325–329.
- [CSPW11] V. Chandrasekaran, S. Sanghavi, P. A. Parrilo, and A. S. Willsky, “Rank-sparsity incoherence for matrix decomposition,” *SIAM Journal on Optimization*, vol. 21, no. 2, pp. 572–596, 2011.
- [CSS11] S. Chatterjee, D. Sundman, and M. Skoglund, “Robust matching pursuit for recovery of gaussian sparse signal,” in *Digital Signal Processing Workshop and IEEE Signal Processing Education Workshop (DSP/SPE), 2011 IEEE*. IEEE, 2011, pp. 420–424.
- [CSVS11] S. Chatterjee, D. Sundman, M. Vehkaperä, and M. Skoglund, “Hybrid greedy pursuit,” in *Signal Processing Conference, 2011 19th European*. IEEE, 2011, pp. 343–347.
- [CSVS12] S. Chatterjee, D. Sundman, M. Vehkaperä, and M. Skoglund, “Projection-based and look-ahead strategies for atom selection,” *Signal Processing, IEEE Transactions on*, vol. 60, no. 2, pp. 634–647, 2012.
- [CT05] E. J. Candès and T. Tao, “Decoding by linear programming,” *IEEE transactions on information theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

- [CT10] —, “The power of convex relaxation: Near-optimal matrix completion,” *Information Theory, IEEE Transactions on*, vol. 56, no. 5, pp. 2053–2080, 2010.
- [CW05] P. L. Combettes and V. R. Wajs, “Signal recovery by proximal forward-backward splitting,” *Multiscale Modeling & Simulation*, vol. 4, no. 4, pp. 1168–1200, 2005.
- [CW08] E. J. Candès and M. B. Wakin, “An introduction to compressive sampling,” *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 21–30, 2008.
- [CWZY15] P. Chen, N. Wang, N. L. Zhang, and D.-Y. Yeung, “Bayesian adaptive matrix factorization with automatic model selection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1284–1292.
- [Dan98] G. B. Dantzig, *Linear programming and extensions*. Princeton university press, 1998.
- [DDDM04] I. Daubechies, M. Defrise, and C. De Mol, “An iterative thresholding algorithm for linear inverse problems with a sparsity constraint,” *Communications on pure and applied mathematics*, vol. 57, no. 11, pp. 1413–1457, 2004.
- [DDFG10] I. Daubechies, R. DeVore, M. Fornasier, and C. S. Güntürk, “Iteratively reweighted least squares minimization for sparse recovery,” *Communications on Pure and Applied Mathematics*, vol. 63, no. 1, pp. 1–38, 2010.
- [DF02] A. C. Doyle and C. Frayling, *The hound of the Baskervilles*. Penguin UK, 1902.
- [DHC11] X. Ding, L. He, and L. Carin, “Bayesian robust principal component analysis,” *IEEE Transactions on Image Processing*, vol. 20, no. 12, pp. 3419–3430, 2011.
- [DLR77] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the em algorithm,” *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.
- [DM09] W. Dai and O. Milenkovic, “Subspace pursuit for compressive sensing signal reconstruction,” *Information Theory, IEEE Transactions on*, vol. 55, no. 5, pp. 2230–2249, 2009.
- [Doy94] A. C. Doyle, *A scandal in Bohemia*. Springer, 1994.

- [DP12] L. Dai and K. Pelckmans, “An ellipsoid based, two-stage screening test for bpdn,” in *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*. IEEE, 2012, pp. 654–658.
- [EA06] M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” *Image Processing, IEEE Transactions on*, vol. 15, no. 12, pp. 3736–3745, 2006.
- [Ela] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*.
- [EM14] Y. C. Eldar and S. Mendelson, “Phase retrieval: Stability and recovery guarantees,” *Applied and Computational Harmonic Analysis*, vol. 36, no. 3, pp. 473–494, 2014.
- [EY09] M. Elad and I. Yavneh, “A plurality of sparse representations is better than the sparsest one alone,” *Information Theory, IEEE Transactions on*, vol. 55, no. 10, pp. 4701–4714, 2009.
- [Faz02] M. Fazel, “Matrix rank minimization with applications,” Ph.D. dissertation, PhD thesis, Stanford University, 2002.
- [Fie78] J. R. Fienup, “Reconstruction of an object from the modulus of its fourier transform,” *Optics letters*, vol. 3, no. 1, pp. 27–29, 1978.
- [FNW07] M. A. Figueiredo, R. D. Nowak, and S. J. Wright, “Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems,” *Selected Topics in Signal Processing, IEEE Journal of*, vol. 1, no. 4, pp. 586–597, 2007.
- [FRW11] M. Fornasier, H. Rauhut, and R. Ward, “Low-rank matrix recovery via iteratively reweighted least squares minimization,” *SIAM Journal on Optimization*, vol. 21, no. 4, pp. 1614–1640, 2011.
- [GAH98] M. Gharavi-Alkhansari and T. S. Huang, “A fast orthogonal matching pursuit algorithm,” in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 3. IEEE, 1998, pp. 1389–1392.
- [GBY08] M. Grant, S. Boyd, and Y. Ye, “CVX: Matlab software for disciplined convex programming,” 2008.
- [GCD12] R. Gribonval, V. Cevher, and M. E. Davies, “Compressible distributions for high-dimensional statistics,” *Information Theory, IEEE Transactions on*, vol. 58, no. 8, pp. 5016–5034, 2012.
- [Ger72] R. W. Gerchberg, “A practical algorithm for the determination of phase from image and diffraction plane pictures,” *Optik*, vol. 35, p. 237, 1972.

- [GL95] R. D. Gill and B. Y. Levit, “Applications of the van Trees inequality: a Bayesian Cramér-Rao bound,” *Bernoulli*, pp. 59–79, 1995.
- [GN99] A. K. Gupta and D. K. Nagar, *Matrix variate distributions*. CRC Press, 1999, vol. 104.
- [GRY11] S. Gandy, B. Recht, and I. Yamada, “Tensor completion and low-n-rank tensor recovery via convex optimization,” *Inverse Problems*, vol. 27, no. 2, p. 025010, 2011.
- [Har93] R. W. Harrison, “Phase problem in crystallography,” *JOSA A*, vol. 10, no. 5, pp. 1046–1055, 1993.
- [Har97] D. A. Harville, *Matrix algebra from a statistician’s perspective*. Springer, 1997, vol. 1.
- [HJ12] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.
- [HKBR99] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, “An algorithmic framework for performing collaborative filtering,” in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 230–237.
- [HR78] D. Harrison and D. L. Rubinfeld, “Hedonic housing prices and the demand for clean air,” *Journal of environmental economics and management*, vol. 5, no. 1, pp. 81–102, 1978.
- [HYZ07] E. T. Hale, W. Yin, and Y. Zhang, “A fixed-point continuation method for l_1 -regularized minimization with applications to compressed sensing,” *CAAM TR07-07, Rice University*, vol. 43, p. 44, 2007.
- [JR10] Y. Jin and B. D. Rao, “Algorithms for robust linear regression by exploiting the connection to sparse signal recovery,” in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 3830–3833.
- [JXC08] S. Ji, Y. Xue, and L. Carin, “Bayesian compressive sensing,” *Signal Processing, IEEE Transactions on*, vol. 56, no. 6, pp. 2346–2356, 2008.
- [Kay93] S. M. Kay, “Fundamentals of statistical signal processing, volume I: Estimation theory,” 1993.
- [Kay98] —, “Fundamentals of statistical signal processing, volume II: Detection theory,” *Signal Processing. Upper Saddle River, NJ: Prentice Hall*, 1998.
- [KB09] T. G. Kolda and B. W. Bader, “Tensor decompositions and applications,” *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

- [KBV09] Y. Koren, R. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” *Computer*, no. 8, pp. 30–37, 2009.
- [KKL⁺07] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, “An interior-point method for large-scale l_1 -regularized least squares,” *Selected Topics in Signal Processing, IEEE Journal of*, vol. 1, no. 4, pp. 606–617, 2007.
- [KSH00] T. Kailath, A. H. Sayed, and B. Hassibi, *Linear estimation*. Prentice Hall Upper Saddle River, NJ, 2000, vol. 1.
- [KvR06] T. Kollo and D. von Rosen, *Advanced multivariate statistics with matrices*. Springer Science & Business Media, 2006, vol. 579.
- [LB10] K. Lee and Y. Bresler, “Admira: Atomic decomposition for minimum rank approximation,” *Information Theory, IEEE Transactions on*, vol. 56, no. 9, pp. 4402–4416, 2010.
- [LBA11] B. Lakshminarayanan, G. Bouchard, and C. Archambeau, “Robust bayesian matrix factorisation,” in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 425–433.
- [LCB98] Y. LeCun, C. Cortes, and C. J. Burges, “The mnist database of handwritten digits,” 1998.
- [LDB⁺09] J. N. Laska, M. Davenport, R. G. Baraniuk *et al.*, “Exact signal recovery from sparsely corrupted measurements through the pursuit of justice,” in *Signals, Systems and Computers, 2009 Conference Record of the Forty-Third Asilomar Conference on*. IEEE, 2009, pp. 1556–1560.
- [LO15] V. Larsson and C. Olsson, “Convex envelopes for low rank approximation,” in *Energy Minimization Methods in Computer Vision and Pattern Recognition*. Springer, 2015, pp. 1–14.
- [LS07] E. G. Larsson and Y. Selen, “Linear regression with a sparse parameter vector,” *Signal Processing, IEEE Transactions on*, vol. 55, no. 2, pp. 451–460, 2007.
- [LSR⁺16] K. Li, M. Sundin, C. R. Rojas, S. Chatterjee, and M. Jansson, “Alternating strategies with internal admm for low-rank matrix reconstruction,” *Signal Processing*, vol. 121, pp. 153–159, 2016.
- [LT07] Y. J. Lim and Y. W. Teh, “Variational bayesian approach to movie rating prediction,” in *Proceedings of KDD Cup and Workshop*, vol. 7. Citeseer, 2007, pp. 15–21.

- [LV11] Y. M. Lu and M. Vetterli, “Sparse spectral factorization: Unicity and reconstruction algorithms,” in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 5976–5979.
- [Mac92] D. J. MacKay, “Bayesian interpolation,” *Neural computation*, vol. 4, no. 3, pp. 415–447, 1992.
- [Mac03] —, *Information theory, inference and learning algorithms*. Cambridge university press, 2003.
- [MAT03] D. MATLAB, “Optimization toolbox user’s guide,” 2003.
- [MES08] J. Mairal, M. Elad, and G. Sapiro, “Sparse representation for color image restoration,” *Image Processing, IEEE Transactions on*, vol. 17, no. 1, pp. 53–69, 2008.
- [Mil90] R. P. Millane, “Phase retrieval in crystallography and optics,” *JOSA A*, vol. 7, no. 3, pp. 394–411, 1990.
- [MMG13] M. Mardani, G. Mateos, and G. B. Giannakis, “Recovery of low-rank plus compressed sparse matrices with application to unveiling traffic anomalies,” *IEEE Transactions on Information Theory*, vol. 59, no. 8, pp. 5186–5205, 2013.
- [MMOP96] M. Misiti, Y. Misiti, G. Oppenheim, and J.-M. Poggi, “Wavelet toolbox,” *The MathWorks Inc., Natick, MA*, 1996.
- [MN95] J. R. Magnus and H. Neudecker, “Matrix differential calculus with applications in statistics and econometrics,” 1995.
- [MS07] A. Mnih and R. Salakhutdinov, “Probabilistic matrix factorization,” in *Advances in neural information processing systems*, 2007, pp. 1257–1264.
- [Mui09] R. J. Muirhead, *Aspects of multivariate statistical theory*. John Wiley & Sons, 2009, vol. 197.
- [MVC10] K. Mitra, A. Veeraraghavan, and R. Chellappa, “Robust rvm regression using sparse outlier model,” in *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010, pp. 1887–1894.
- [MXAP12] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Püschel, “Distributed basis pursuit,” *Signal Processing, IEEE Transactions on*, vol. 60, no. 4, pp. 1942–1956, 2012.
- [MZ93] S. G. Mallat and Z. Zhang, “Matching pursuits with time-frequency dictionaries,” *Signal Processing, IEEE Transactions on*, vol. 41, no. 12, pp. 3397–3415, 1993.

- [NSB13] S. Nakajima, M. Sugiyama, and S. D. Babacan, "Variational bayesian sparse additive matrix factorization," *Machine learning*, vol. 92, no. 2-3, pp. 319–347, 2013.
- [NT09] 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.
- [OCS14] R. Otazo, E. Candès, and D. K. Sodickson, "Low-rank plus sparse matrix decomposition for accelerated dynamic mri with separation of background and dynamic components," *Magnetic Resonance in Medicine*, 2014.
- [OE14] H. Ohlsson and Y. C. Eldar, "On conditions for uniqueness in sparse phase retrieval," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 1841–1845.
- [OT03] W. Ogryczak and A. Tamir, "Minimizing the sum of the k largest functions in linear time," *Information Processing Letters*, vol. 85, no. 3, pp. 117–122, 2003.
- [OYDS11] H. Ohlsson, A. Y. Yang, R. Dong, and S. S. Sastry, "Compressive phase retrieval from squared output measurements via semidefinite programming," *arXiv preprint arXiv*, vol. 1111, 2011.
- [PM13] R. Prasad and C. R. Murthy, "Cramér-rao-type bounds for sparse bayesian learning," *Signal Processing, IEEE Transactions on*, vol. 61, no. 3, pp. 622–632, 2013.
- [RCLV13] J. Ranieri, A. Chebira, Y. M. Lu, and M. Vetterli, "Phase retrieval for sparse signals: Uniqueness conditions," *arXiv preprint arXiv:1308.3058*, 2013.
- [RFGST09] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. AUAI Press, 2009, pp. 452–461.
- [RG09] G. Rath and C. Guillemot, "Sparse approximation with an orthogonal complementary matching pursuit algorithm," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. IEEE, 2009, pp. 3325–3328.
- [RIK07] T. Raiko, A. Ilin, and J. Karhunen, "Principal component analysis for large scale problems with lots of missing values," in *Machine Learning: ECML 2007*. Springer, 2007, pp. 691–698.

- [RKH13] C. R. Rojas, D. Katselis, and H. Hjalmarsson, “A note on the spice method,” *Signal Processing, IEEE Transactions on*, vol. 61, no. 18, pp. 4545–4551, 2013.
- [RNL02] L. Rebollo-Neira and D. Lowe, “Optimized orthogonal matching pursuit approach,” *Signal Processing Letters, IEEE*, vol. 9, no. 4, pp. 137–140, 2002.
- [RZE08] R. Rubinstein, M. Zibulevsky, and M. Elad, “Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit,” *CS Technion*, vol. 40, no. 8, pp. 1–15, 2008.
- [SACH14] P. B. Swamy, S. K. Ambat, S. Chatterjee, and K. Hari, “Reduced look ahead orthogonal matching pursuit,” in *Communications (NCC), 2014 Twentieth National Conference on*. IEEE, 2014, pp. 1–6.
- [SCJa] M. Sundin, S. Chatterjee, and M. Jansson, “Bayesian Cramér-Rao bounds for factorized model based low rank matrix reconstruction,” in *Signal Processing Conference (EUSIPCO), 2016 Proceedings of the 23rd European*.
- [SCJb] —, “Bayesian Cramér-Rao bounds for low-rank matrix reconstruction,” *In preparation*.
- [SCJc] —, “Convex recovery for low-rank phase retrieval,” *In preparation*.
- [SCJ14] —, “Combined modeling of sparse and dense noise improves bayesian rvm,” in *European Signal Processing Conference (EUSIPCO)*. Eurasip, 2014.
- [SCJ15a] —, “Bayesian learning for robust principal component analysis,” in *Signal Processing Conference (EUSIPCO), 2015 23rd European*, 2015, pp. 2361–2365.
- [SCJ15b] —, “Combined modeling of sparse and dense noise for improvement of relevance vector machine,” *Submitted paper. arXiv preprint arXiv:1501.02579*, 2015.
- [SCJ15c] —, “Greedy minimization of l1-norm with high empirical success,” in *40th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.
- [SCJR14] M. Sundin, S. Chatterjee, M. Jansson, and C. Rojas, “Relevance singular vector machine for low-rank matrix sensing,” in *2014 International Conference on Signal Processing and Communications (SPCOM)*, July 2014, pp. 1–5.

- [SCS12] D. Sundman, S. Chatterjee, and M. Skoglund, "A greedy pursuit algorithm for distributed compressed sensing," in *2012 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. IEEE, 2012, pp. 2729–2732.
- [SGIH13] C. Soussen, R. Gribonval, J. Idier, and C. Herzet, "Joint k-step analysis of orthogonal matching pursuit and orthogonal least squares," *Information Theory, IEEE Transactions on*, vol. 59, no. 5, pp. 3158–3174, 2013.
- [SJC13] M. Sundin, M. Jansson, and S. Chatterjee, "Conditional prior based lmmse estimation of sparse signals," in *Proceedings of the 21st European Signal Processing Conference, EUSIPCO 2013*, 2013, pp. 1–5.
- [SL74] L. A. Shepp and B. F. Logan, "The fourier reconstruction of a head section," *Nuclear Science, IEEE Transactions on*, vol. 21, no. 3, pp. 21–43, 1974.
- [SM05] P. Stoica and R. L. Moses, *Spectral analysis of signals*. Pearson/Prentice Hall Upper Saddle River, NJ, 2005.
- [SM08] R. Salakhutdinov and A. Mnih, "Bayesian probabilistic matrix factorization using markov chain monte carlo," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 880–887.
- [SM15] S. K. Sahoo and A. Makur, "Signal recovery from random measurements via extended orthogonal matching pursuit," *Signal Processing, IEEE Transactions on*, vol. 63, no. 10, pp. 2572–2581, 2015.
- [SPH09] M. Stojnic, F. Parvaresh, and B. Hassibi, "On the reconstruction of block-sparse signals with an optimal number of measurements," *IEEE Transactions on Signal Processing*, vol. 57, no. 8, pp. 3075–3085, Aug 2009.
- [SRJC] M. Sundin, C. R. Rojas, M. Jansson, and S. Chatterjee, "Relevance singular vector machine for low-rank matrix reconstruction," *Accepted for publication in IEEE Transactions on Signal Processing*.
- [SS10] N. Srebro and R. R. Salakhutdinov, "Collaborative filtering in a non-uniform world: Learning with the weighted trace norm," in *Advances in Neural Information Processing Systems*, 2010, pp. 2056–2064.
- [SSJ13] M. Sundin, D. Sundman, and M. Jansson, "Beamformers for sparse recovery," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 5920–5924.

- [Stu11] B. L. Sturm, “Sparse vector distributions and recovery from compressed sensing,” *arXiv preprint arXiv:1103.6246*, 2011.
- [Sun14] D. Sundman, “Greedy algorithms for distributed compressed sensing,” *PhD Thesis*, 2014.
- [Suz15] T. Suzuki, “Convergence rate of bayesian tensor estimator and its min-max optimality,” in *Proceedings of the 32nd International Conference on Machine Learning (Lille, 2015)*, 2015, pp. 1273–1282.
- [SV08] K. Schnass and P. Vandergheynst, “Dictionary preconditioning for greedy algorithms,” *Signal Processing, IEEE Transactions on*, vol. 56, no. 5, pp. 1994–2002, 2008.
- [SVJC] M. Sundin, A. Venkitaraman, M. Jansson, and S. Chatterjee, “A convex constraint for graph connectedness,” in *preparation*.
- [SW12] X. Shen and Y. Wu, “A unified approach to salient object detection via low rank matrix recovery,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012, pp. 853–860.
- [T⁺04] J. Tropp *et al.*, “Greed is good: Algorithmic results for sparse approximation,” *Information Theory, IEEE Transactions on*, vol. 50, no. 10, pp. 2231–2242, 2004.
- [TBI97] L. N. Trefethen and D. Bau III, *Numerical linear algebra*. Siam, 1997, vol. 50.
- [Ter12] A. Terras, *Harmonic analysis on symmetric spaces and applications II*. Springer Science & Business Media, 2012.
- [TFM06] H. Takeda, S. Farsiu, and P. Milanfar, “Robust kernel regression for restoration and reconstruction of images from sparse noisy data,” in *Image Processing, 2006 IEEE International Conference on*. IEEE, 2006, pp. 1257–1260.
- [TG07] J. A. Tropp and A. C. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” *Information Theory, IEEE Transactions on*, vol. 53, no. 12, pp. 4655–4666, 2007.
- [Tib96] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.
- [Tip01] M. E. Tipping, “Sparse bayesian learning and the relevance vector machine,” *Journal of Machine Learning Research*, vol. 1, pp. 211–244, 2001.

- [TN11] G. Tang and A. Nehorai, "Lower bounds on the mean-squared error of low-rank matrix reconstruction," *Signal Processing, IEEE Transactions on*, vol. 59, no. 10, pp. 4559–4571, 2011.
- [Ver12] R. Vershynin, "Introduction to the non-asymptotic analysis of random matrices," in *Compressed Sensing*, Y. C. Eldar and G. Kutyniok, Eds. Cambridge University Press, 2012, pp. 210–268, Cambridge Books Online. [Online]. Available: <http://dx.doi.org/10.1017/CBO9780511794308.006>
- [VKC13] M. Vehkaperä, Y. Kabashima, and S. Chatterjee, "Statistical mechanics approach to sparse noise denoising," in *Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 21st European*. IEEE, 2013, pp. 1–5.
- [VLP93] C. F. Van Loan and N. Pitsianis, *Approximation with Kronecker products*. Springer, 1993.
- [vR88] D. von Rosen, "Moments for the inverted wishart distribution," *Scandinavian Journal of Statistics*, pp. 97–109, 1988.
- [VT04] H. L. Van Trees, *Detection, estimation, and modulation theory*. John Wiley & Sons, 2004.
- [VTB07] H. L. Van Trees and K. L. Bell, "Bayesian bounds for parameter estimation and nonlinear filtering/tracking," *AMC*, vol. 10, p. 12, 2007.
- [WdM15] I. Waldspurger, A. d'Aspremont, and S. Mallat, "Phase recovery, max-cut and complex semidefinite programming," *Mathematical Programming*, vol. 149, no. 1-2, pp. 47–81, 2015.
- [Wip12] D. Wipf, "Non-convex rank minimization via an empirical bayesian approach," *arXiv preprint arXiv:1207.2440*, 2012.
- [WKW02] W. Wright, F. C. Kelly, and O. Wright, *Miracle at Kitty Hawk: The Letters of Wilbur and Orville Wright*. Da Capo Press, 2002.
- [WLZ13] S. Wang, D. Liu, and Z. Zhang, "Nonconvex relaxation approaches to robust matrix recovery," in *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*. AAAI Press, 2013, pp. 1764–1770.
- [WPR04] D. Wipf, J. Palmer, and B. Rao, "Perspectives on sparse bayesian learning," *Computer Engineering*, vol. 16, no. 1, p. 249, 2004.
- [WR04] 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.

- [WS11] J. Wang and B. Shim, “Exact reconstruction of sparse signals via generalized orthogonal matching pursuit,” in *Signals, Systems and Computers (ASILOMAR), 2011 Conference Record of the Forty Fifth Asilomar Conference on*. IEEE, 2011, pp. 1139–1142.
- [WS12] —, “On the recovery limit of sparse signals using orthogonal matching pursuit,” *Signal Processing, IEEE Transactions on*, vol. 60, no. 9, pp. 4973–4976, 2012.
- [WYG⁺09] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, “Robust face recognition via sparse representation,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 2, pp. 210–227, 2009.
- [YC11] J. Yoo and S. Choi, “Bayesian matrix co-factorization: Variational algorithm and cramér-rao bound,” in *Machine Learning and Knowledge Discovery in Databases*. Springer, 2011, pp. 537–552.
- [YdH15] M. Yang and F. de Hoog, “Orthogonal matching pursuit with thresholding and its application in compressive sensing,” 2015.
- [YHS13] L. Yang, Z. H. Huang, and X. Shi, “A fixed point iterative method for low n-rank tensor pursuit,” *IEEE Transactions on Signal Processing*, vol. 61, no. 11, pp. 2952–2962, June 2013.
- [ZCJ12] D. Zachariah, S. Chatterjee, and M. Jansson, “Dynamic iterative pursuit,” *Signal Processing, IEEE Transactions on*, vol. 60, no. 9, pp. 4967–4972, 2012.
- [ZR11] Z. Zhang and B. D. Rao, “Sparse signal recovery with temporally correlated source vectors using sparse bayesian learning,” *Selected Topics in Signal Processing, IEEE Journal of*, vol. 5, no. 5, pp. 912–926, 2011.
- [ZR13] Z. Zhang and B. Rao, “Extension of sbl algorithms for the recovery of block sparse signals with intra-block correlation,” *Signal Processing, IEEE Transactions on*, vol. 61, no. 8, pp. 2009–2015, 2013.
- [ZSJC12] D. Zachariah, M. Sundin, M. Jansson, and S. Chatterjee, “Alternating least-squares for low-rank matrix reconstruction,” *Signal Processing Letters, IEEE*, vol. 19, no. 4, pp. 231–234, 2012.
- [ZT11] T. Zhou and D. Tao, “Godec: Randomized low-rank & sparse matrix decomposition in noisy case,” in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 33–40.
- [ZYC15] H. Zhang, W. Yin, and L. Cheng, “Necessary and sufficient conditions of solution uniqueness in 1-norm minimization,” *Journal of Optimization Theory and Applications*, vol. 164, no. 1, pp. 109–122, 2015.