INVERSE PROBLEMS AND REGULARIZATION IN SIGNAL PROCESSING WITH APPLICATIONS TO WIRELESS CHANNEL ESTIMATION

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

DOCTOR OF PHILOSOPHY

in Electronics and Communication Engineering

by
Ahmet ŞAHİN

December 2011
İZMİR

We approve the thesis of **Ahmet ŞAHİN**

Assist. Prof. Dr. Mustafa Aziz ALTINKAYA
Supervisor

Prof. Dr. Erdal PANAYIRCI
Committee Member

Assoc. Prof. Dr. Olcay AKAY
Committee Member

Assist. Prof. Dr. Gülden KÖKTÜRK
Committee Member

Assist. Prof. Dr. Şevket GÜMÜŞTEKİN
Committee Member

16 December 2011

Prof. Dr. F. Acar SAVACI
Head of the Department of
Electrical and Electronics Engineering

Prof. Dr. R. Tuğrul SENGER
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGMENTS

I must express my greatest gratitude to my advisor Dr. Serdar Özen for giving me an excellent topic of research and all due directions and insights to the completion of my thesis. I thank the members of my committee, Prof. Dr. Erdal Panayırcı, Dr. Olcay Akay, Dr. Gülden Köktürk and Dr. Şevket Gümüştekin for the valuable evaluation of my work and for their helpful feedback.

I am extremely grateful to Prof. Dr. Kayhan Erciyes and his wife Dr. Gülnur Erciyes for giving me the valuable opportunity to complete my thesis and for all their support and encouragement during my work with them in İzmir University.

I owe much gratitude to all my family members my father Yaşar, my mother Sadinaz, my wife Hatice, my brother Alim, my grandfather Ahmet and grandmother Gülayşe and my sons Fatih and Mehmet for all their support, encouragement and also for their patience for possible negligence which occured during my research, and I must acknowledge that without their support I would not have enough energy and motivation to complete my thesis.

I am very grateful to my co-worker Dr. Thomas Bechteler for his precious help in the LaTeX documentation of this thesis.

I do not know which words can suffice to express my extreme gratitude to my last advisor Dr. Mustafa Aziz Altınkaya who took me upon as a student in the last stages of my PhD study and guided me through its final steps and passages.

ABSTRACT

INVERSE PROBLEMS AND REGULARIZATION IN SIGNAL PROCESSING WITH APPLICATIONS TO WIRELESS CHANNEL ESTIMATION

The research presented in this thesis is on inverse problems encountered in the field of signal processing. Theory, classification and solution techniques of linear discrete inverse problems (LDIP) are investigated. LDIP are classified as underdetermined LDIP (ULDIP) and overdetermined LDIP (OLDIP). The solution methods developed for LDIP are applied to the particular problems of signal processing mainly channel estimation, equalization and compressive sampling. A new solution technique named constraint removal (CR) is presented for ULDIP type problems with sparse inputs. CR is applied to terrestrial digital TV (DTV) channel estimation. CR is also compared with subspace pursuit (SP) and linear programming. Regularization and optimum regularization parameter selection for ill-posed OLDIP type problems are discussed. Sparse channel estimation for wireless digital communications is investigated. A new channel estimation method, permuted deconvolution (PDEC), for long delay spread channels with short training sequences is proposed and compared with other methods. A review on equalization is presented. Different equalization techniques are discussed and compared. DFE is explained from an inverse problem perspective. A new non-feedback equalization technique called frequency compensated linear equalization (FC-LE) for sparse channels is presented and compared with DFE.

ÖZET

İŞARET İŞLEMEDE TERS PROBLEMLER VE DÜZENLİLEŞTİRME İLE KABLOSUZ KANAL KESTİRİMİ UYGULAMALARI

Bu tezde sunulan araştırma sinyal işleme alanında karşılaşılan ters problemler üzerinedir. Doğrusal ayrık ters problemler, teori, sınıflandırma ve çözüm teknikleri açısından incelenmiştir. Doğrusal ayrık ters problemler (DATP) az belirli doğrusal ayrık ters problemler (ADATP) ve çok belirli doğrusal ayrık ters problemler (ÇDATP) olarak iki sınıfa ayrılmıştır. DATP için geliştirilen çözüm teknikleri sinyal işleme alanında özellikle kanal kestirimi, eşitleme ve sıkıştırmalı örnekleme alanlarına uygulanmıştır. ADATP için 'kısıtlama kaldırımı' (KK) adında yeni bir yöntem önerilmiştir. Kısıtlama kaldırımı yöntemi karasal sayısal televizyon kanal kestirimi için uygulanmıştır. Kısıtlama kaldırımı, alt uzay takibi ve doğrusal programlama yöntemleriyle kıyaslanmıştır. Kötü huylu çok belirli doğrusal ayrık ters problemlerin çözümü için düzenlileştirme ve en iyi düzenlileştirme katsayısının otomatik hesaplanması tartışılmıştır. Telsiz sayısal haberleşme seyrek kanal kestirimi incelenmiştir. Kısa eğitim dizisi ile uzun telsiz sayısal haberleşme kanal kestirimi için 'değişimli ters evrişim' (DTE) isimli yeni bir yöntem önerilmiş ve diğer kanal kestirimi yöntemleri ile kıyaslanmıştır. Eşitleme konusu incelenmiş ve farklı eşitleme teknikleri tartışılarak kıyaslanmıştır. Karar geri beslemeli eşitleyici (KGBE) ters problem perspektifi ile açıklanmıştır. Seyrek kanallar için frekans dengelemeli doğrusal eşitleyici (FD-DE) adında yeni bir eşitleyici önerilmiş ve karar geri beslemeli eşitleyici ile kıyaslanmıştır.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

BER	Bit Error Rate
BLS	Blended Least Squares
BP	Basis Pursuit
CE	Channel Estimation
CFAR	Constant False Alarm Rate
CIR	Channel Impulse Response
CoSaMP	Compressive Sampling Matching Pursuit
CR	Constraint Removal
CS	Compressive Sampling or Compressed Sensing
DFE	Decision Feedback Equalizer
DP	Detector Pulse
DTV	Digital Television
HDTV	High Definition Television
FFT	Fast Fourier transform
FFF	Feed Forward Filter
FBF	Feed Back Filter
FC-LE	Frequency Compensated Linear Equalizer
FOCUSS	Focal Underdetermined System Solver
GCV	Generalized Cross Validation
HP	High Pass
IHT	Iterative Hard Thresholding
ISI	Inter Symbol Interference
LASSO	Least Absolute Shrinkage and Selection Operator
LE	Linear Equalizer
LS	Least Squares
LN	Least Norm
LDIP	Linear Discrete Inverse Problem
LP	Linear Programming or Low Pass
MP	Matching Pursuit
MPEG2	Moving Pictures Expert Group Video Compression Format 2

MIMO	Multiple Input Multiple Output
ML	Maximum Likelihood
MMSE	Minimum Mean Squared Error
MLE	Maximum Likelihood Estimation
NP	Nonpolynomial time
OLDIP	Overdetermined Linear Discrete Inverse Problem
OMP	Orthogonal Matching Pursuit
PD	Permuted Deconvolution Estimator
PDEC	Permuted Deconvolution
PMF	Pulse Matched Filter
PSF	Pulse Shaping Filter
RIC	Restricted Isometry Constant
RIP	Restricted Isometry Property
SCR	Sequential Constraint Removal
SNR	Signal to Noise Ratio
SP	Subspace Pursuit
StOMP	Stagewise Orthogonal Matching Pursuit
SVD	Singular Value Decomposition
TSVD	Truncated Singular Value Decomposition
ULDIP	Underdetermined Linear Discrete Inverse Problem
UPRE	Unbiased Predictive Risk Estimator
ZFE	Zero Forcing Equalizer

CHAPTER 1

INTRODUCTION

1.1. Inverse Problems

1.1.1. Definition

Inverse problems can safely be defined as the inverse of a direct problem. A similar definition is given in Keller (1976). A direct problem can be defined as a process which has a natural flow of actions going forward in the dimension of time. The amount of information that a direct problem produces as an effect is less than or equal to the amount of information it uses. Inverse problem is a process that flows backward in time scale against the natural flow of actions. The information that an inverse problem should produce is equal to or greater than the information it uses. This definition of inverse problem is valid for physical systems for which dimension of time exists. But there may be theoretical systems without an explicit scale of time such as inverse of a matrix. It is well known that if an input vector x is multiplied with a known matrix A, an output vector y is always produced. This is the direct problem, but the reverse may not be true. In order to get x from y, we need the inverse of matrix A. The inverse of matrix A may not exist, or x that gives y may not be unique or no x can produce y. In general, inverse problems are difficult to solve because the corresponding direct problem may not be one to one and there may be loss of information going in the forward direction. For that reason, inverse problems generally require additional constraints in order to be solved. Imposing these constraints on the inverse problem is called regularization (Engl, Hanke and Neubauer 1996, Groetsch 2007). As will be mentioned further in the thesis, two such constraints are sparsity and minimum norm constraints without which many important inverse problems can not be solved (Donoho 2006b, Donoho et al. 2006, Foucart 2010, Tikhonov and Arsenin 1977).

1.1.2. Motivation

Inverse problems are not a rare case in science. In fact, many important scientific problems are of inverse nature which are characterized by the processing of sampled data to extract information (Bertero and Boccacci 1998, Bertero and Piana 2006, Menke 1989, Munk, Worcester and Wunsch 1995, Enderle 2005). One of the motivations behind the rising interest in inverse problems is the fact that many scientific and engineering inverse problems which involved prohibitively large amount of sensory data processing and hence were considered impossible to solve within a practical time limit became increasingly solvable during the last few decades thanks to the advances in computer technology (Wang, Polydorides and Bertsekas 2009). Many high technology systems like computed tomography (CT) (Hounsfield 1973) and magnetic resonance imaging (MR) (Lauterbur 1973), seismic underground imaging (Menke 1989, Richardson and Zandt 2009) ocean acoustic tomography (Munk, Worcester and Wunsch 1995) and channel estimation depend on the solution of inverse problems. In addition, increasing computational power encouraged the scientists and engineers to explore and challenge new areas of research involving intensive computations. One such area is the compressed sensing (Donoho 2006a) or equivalently compressive sampling (Candès 2006) field which intends to change the way data is sampled and compressed with promising practical consequences. Two important inverse problems in terrestrial, mobile and wireless digital communications are the channel estimation and equalization problems (Proakis 1995) which gained more significance with the large scale deployment of mobile and wireless communication devices and very high data rate digital terrestrial television systems (DTV).

1.1.3. Linear Discrete Inverse Problems (LDIP)

Many practical inverse problems are of discrete nature since they involve processing of data sampled at discrete time instants (Bertero, Mol and Pike 1985, 1988, Hansen 2010, Menke 1989). In addition, inverse problems need to be discrete or discretized in order to be solved by powerful digital computers. Although most practical inverse problems are linear, nonlinear ones can be approximated by a linear inverse problem within a small period of time. Typically a noisy linear discrete system can be described as

$$y = Ax + \eta, \tag{1.1}$$

where x is the $n \times 1$ input vector, y is the $m \times 1$ measurement vector, η is the $m \times 1$ noise vector which includes the perturbation in A and A is the $m \times n$ measurement matrix. The linear discrete inverse problem here is to find unknown x, given y and A. If the matrix A were a square ($m = n$), full rank and well-conditioned matrix, it would be straightforward to solve the inverse problem by finding the inverse of A and multiplying it with y. If $m < n$, the system is called underdetermined and if $m > n$, the system is called overdetermined and the corresponding inverse problems can be called underdetermined linear discrete inverse problem (ULDIP) and overdetermined linear discrete inverse problem (OLDIP) respectively. Both systems are encountered in different contexts and possess different characteristics and hence their solutions require different strategies.

1.1.4. Overdetermined Linear Discrete Inverse Problems (OLDIP)

Overdetermined Linear Discrete Inverse Problems (OLDIP) are usually encountered in biomedical and seismic imaging which depend on spatiotemporal sensory data processing (Enderle 2005, Biondi 2006, Brown 2004). Such systems acquire very large amount of sensory data for a non-invasive imaging of a target. There are two main reasons for such systems to acquire large amount of data. First reason is to be able to construct a faithful three dimensional image of the target with sufficient resolution. The second reason is to overcome the effect of noise and measurement errors by using as many measurements as possible. Due to the spatiotemporal proximity of sensory data, there is usually an incremental difference between consecutive measurements and from neighboring sensors which makes the inverse system very noise sensitive and hence ill-posed. The term ill-posed will be defined in Subsection 1.1.6.

1.1.5. Underdetermined Linear Discrete Inverse Problems (ULDIP)

Underdetermined Linear Discrete Inverse Problems (ULDIP) are those in which less measurement or sensory data is available than the number of unknowns. Examples of such problems are channel estimation with very long delay spread and comparatively short training sequence (Özen, Zoltowski and Fimoff 2002) as in DTV channels and compressive sampling (Candès 2006) in which less measurements are taken for the purpose of compression. It is obvious that such problems are theoretically unsolvable unless there are some additional information or constraint to fill the gap caused by the missing data.

Currently, there exist three main tracks to solve ULDIP type problems. The first is the L_1 norm minimization formulations like basis pursuit (Chen, Donoho and Saunders 1998), LASSO (Tibshirani 1996) and Dantzig selector (Candès and Tao 2007) which can be solved by convex optimization methods like linear programming. Although they achieve high correct recovery rates, convex optimization used to solve them may require very large number of iterations. The second track is the greedy pursuit methods like CoSaMP (Needell and Tropp 2009) and SP (Dai and Milenkovic 2008) based on the initial matching pursuit (Mallat and Zhang 1993) algorithm which require less computation but slightly less performance in terms of correctness than the first track. Rigorous research is being done to improve the correctness of the greedy pursuits without compromising the speed. Practical systems which are usually consumer oriented with limited energy and computational resources require fast algorithms with acceptable level of correctness. Therefore, the second track is more of a choice to be researched for practical usage. Algorithms in the third track involves weighted least squares solutions like FOCUSS (Gorodnitsky and Rao 1997) and iteratively reweighted least squares IRLS (Daubechies et al. 2010). Although both algorithms have tractable complexity and good performances, they do not have the speed advantage of the greedy algorithms.

1.1.6. Ill-Posedness and Regularization

Well-Posed: Hadamard defined that a system is well-posed if it satisfies the following conditions (Hadamard 1902, 1923):

1. Existence : For all admissible data, a solution exists.

2. Uniqueness : For all admissible data, the solution is unique.

3. Continuity : The solution depends continuously on the data.

Ill-Posed: A system is ill-posed if it is not well-posed, that is, if it does not satisfy one or more of the above conditions. Ill-posed linear discrete systems have an ill-conditioned measurement matrix and well-posed linear discrete systems have a well-conditioned matrix. The terms ill-posed and well-posed are used for systems. For matrices that define linear discrete systems, the terms ill-conditioned and well-conditioned are more appropriate. The condition number of a matrix is given by the ratio of its largest and smallest singular values. As the condition number gets higher, the matrix becomes more ill-conditioned and hence the associated linear discrete system more ill-posed.

Physically Ill-Posed: Hadamard's definition for well-posedness is too strict in the physical sense. A system may satisfy all three conditions for well-posedness but still may be considered ill-posed physically, due to unrealistic or physically inadmissable solutions. A general definition of ill-posedness for LDIP type systems need to be made in a relaxed physical sense. A noisy LDIP type system is defined by the Equation 1.1. If the measurement matrix A does not have an inverse or pseudoinverse, this LDIP system is strictly ill-posed. Yet, if $A^T A$ is near singular and hence the measurement matrix A has an inverse or pseudoinverse with very large entries such that the solution is dominated by the amplified noise, the system is still considered to be ill-posed in the physical sense. Therefore, a relaxed physical definition of ill-posedness could be the following: "An inverse problem is physically ill-posed if noise gets amplified in its solution". Depending on the amplification level, we can categorize ill-posed problems as severely ill-posed if amplification is large, moderately ill-posed if amplification is moderate and mildly ill-posed if amplification is low. From the above discussion, it is safe to say that ULDIP type problems are always strictly ill-posed.

Most OLDIP type problems which are frequently encountered in sensory data acquisition and processing systems where large amount of spatiotemporal data is collected for subsequent processing are severely ill-posed. The spatiotemporal proximity of sensor data due to physical constraints produce an overdetermined measurement matrix A with nearly linear dependent columns. Consequently, the near singularity of $A^T A$ causes the pseudoinverse of A to have blowing entries causing the least squares solution to amplify noise. Even very small amount of noise in the measurement data y may be amplified to an extent to dominate the solution x which becomes effectively an amplified noise. The solution to noise amplification is regularization.

Regularization: It is usually not possible to get meaningful results from an ill-posed inverse problem unless some regularization method is applied (Bertero, Mol and Pike 1988, Engl, Hanke and Neubauer 1996). Regularization aims to constrain or force the solution to be in a meaningful range. It is common to minimize the L_1 or L_2 norms of the solution for regularization. The well known Tikhonov regularization (Tikhonov and Arsenin 1977) minimizes the Lagrangian sum functional

$$J(x) = \|y - Ax\|_2^2 + \lambda \|x\|_2^2. \tag{1.2}$$

Tikhonov regularization is the only regularization technique with a closed form representation. Iterative regularization methods minimizing L_1 and L_2 norms are also available with no closed form expressions.

Regularization parameter: Parameter based regularization brings about the issue of regularization parameter selection. The question is how to find the optimum λ such that the solution x which minimize the sum in (1.2) is closest to the unknown actual solution x_a. It is common to relate the regularization parameter to noise variance. Some heuristic regularization parameter selection methods are proposed (Mallows 1973, Wahba 1977, Hansen and O'Leary 1993) but the issue still remains to be developed.

Usually, there exists a range of regularization parameters which produce admissible solutions. If the regularization parameter is selected out of this range, either overregularization or underregularization may occur.

Overregularization: the solution is largely determined by regularization and regularization parameter and has negligible connection with the actual solution.

Underregularization: the solution is largely determined by noise and has negligible connection with the actual solution.

For an extensive treatment of regularization see Engl, Hanke and Neubauer (1996)

1.1.7. Rank Deficiency and Ill-Posedness

Rank deficient matrices do not have an inverse or pseudoinverse. Ill-posed (conditioned) matrices, despite their near singularity, have always inverses or pseudoinverses. This topic is treated in Section 3.6.3. using singular value decomposition.

1.2. Inverse Problem Applications in Signal Processing

1.2.1. Channel Estimation

Channel estimation or equivalently channel impulse response estimation is an inevitable part of digital wireless, terrestrial and mobile receiving systems. This is because, without the knowledge of CIR, efficient equalization of channels to remove ISI would not be possible. High error rates caused by large ISI would make efficient and reliable digital communication impossible. There are mainly 3 types of channel estimation methods currently available; training sequence based channel estimation, semi-blind channel estimation and blind channel estimation. All have advantages and disadvantages. Training sequence based channel estimation is the one currently used in wireless, terrestrial and mobile communication receivers. Figure 1.1 depicts a sample DTV channel which exhibits sparsity, a property useful in solving certain important inverse problems.

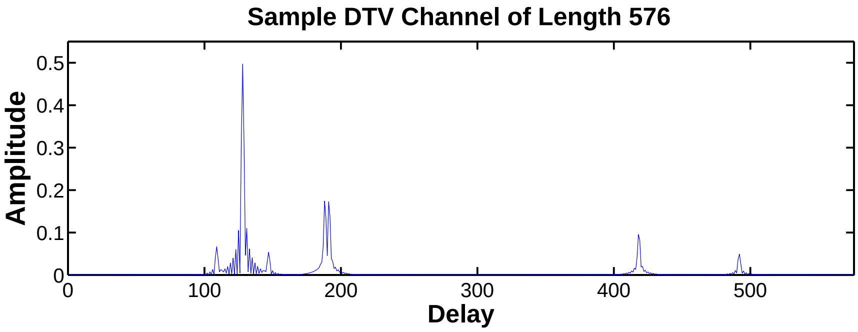


Figure 1.1. A typical DTV channel impulse response (CIR) with a delay spread (length) of 576 is plotted. DTV channels exhibit a sparse behavior with very few major taps as compared to the channel length a property which is useful in CIR estimation via sparse recovery techniques.

1.2.2. Equalization

Equalization like channel estimation is an important inverse problem in digital communications particularly in wireless terrestrial digital transmission and broadcasting. Every digital signal which is carried by a physical medium undergo some distortion during its travel. One important distortion particularly in terrestrial digital communications is the so called multi-path effect caused by the reflection of a transmitted signal from surrounding objects. Multi-path effect causes a transmitted symbol to reach its destination at different times and power levels. Therefore the sampled signal at the receiving antenna is the superposition of multiple symbols instead of a single symbol as would be in an ideal uni-path transmission. In other words, multiple symbols belonging to different time instants get summed up in one sample with different weights. This phenomenon is called inter-symbol interference (ISI). In order to detect the individual symbols from ISI corrupted samples, the receivers must counteract the effect of ISI. Since ISI causes an unequal, nonuniform response at each frequency, the receiver must try to equalize the responses at all frequencies by performing equalization. Equalization can be done in various ways. There are linear and nonlinear equalizers. Equalization is a ULDIP type inverse problem explained in Chapter 6.

1.2.3. Compressive Sampling and Sparse Recovery

Compressed sensing (CS) (Donoho 2006a) or equivalently compressive sampling (Candès 2006) aims to sample data in a compressed from so that less resources are used for processing and storage. CS aims to change the way sampling and compression have so far been done in a way by combining them in one operation and hence economizing on the computational and storage resources. However, compressive sampling must be done in such a way that the original uncompressed data can later be recovered lossless or with marginal loss. Compressive sampling itself is a direct problem not an inverse problem. Yet, recovering the original data from its sampled and compressed form is a ULDIP type inverse problem. As will be described later in Chapter 2, recovery is only possible if the original uncompressed data is sparse in some domain. The inverse of compressive sampling is therefore called sparse recovery.

Although it is a relatively new topic of research, thanks to its highly promising consequences in various fields of science and engineering particularly in data compression and acquisition, CS has swiftly became the focus of an intensive research particularly in applied mathematics and engineering disciplines.

1.2.4. Error Correction

One important inverse problem in digital communications is the detection of stream errors. In digital data transmission, signal is scrambled before being transmitted. During transmission through physical media such as air, errors may be introduced in the digital stream perhaps due to atmospheric effects. These errors are almost always sparse and sparse recovery techniques can be used to correct such errors. Suppose that x is the raw signal and $y = Ax$ is the scrambled signal to be transmitted where A is the scrambler matrix and that there exists another matrix $\tilde{A}$ such that $\tilde{A}A$ is equal to a zero matrix. Then detecting the error can be transformed into a sparse recovery problem as described below. The receiver receives the corrupted signal

$$y = Ax + \epsilon, \tag{1.3}$$

where ϵ is the sparse error. Then multiplying both sides of (1.3) with $\tilde{A}$

$$\tilde{A}y = \tilde{A}A + \tilde{A}\epsilon = \tilde{A}\epsilon, \tag{1.4}$$

setting $q = \tilde{A}y$ and rewriting (1.4) we get the sparse linear equation

$$q = \tilde{A}\epsilon, \tag{1.5}$$

where ϵ can be recovered by sparse recovery techniques.

1.2.5. Image Reconstruction

Reconstructing a distorted image is an ill-posed inverse problem (Vogel 2002, Bertero and Piana 2006). If the cause of the distortion is known, this inverse problem can be solved using regularization. There are a number of causes which impair the quality of image captured by camera sensors. The object being captured might be moving causing motion blur, the camera can be out of focus resulting in a defocused image or the camera itself can move during capture which cause the whole image to blur. Repairing impaired images using image processing techniques is called image reconstruction. The most encountered image defect is the blurring effect. Yet, the cause of blurring is not unique. There are two main reasons for blurry images and blurry objects in an image.

The first is the motion blur caused by the motion of the camera or the motion of objects in a scene. The second is the defocusing effect caused by out of focus camera lens. It is also possible that both can happen at the same time. Forward processes for both motion blur and defocusing involves convolution. Therefore, the inverse processes for both involves deconvolution. Deconvolution is an ill-posed inverse problem hence its solution requires regularization. It is still an unsolved problem to find the optimum regularization parameter automatically and usually visual feedback is necessary to determine the optimal parameter.

1.3. Organization and Contributions of the Thesis

This thesis consists of eight chapters including an introduction and a conclusion chapter. The first four chapters are on inverse problems and regularization. The ensuing three chapters are on the application of inverse problem techniques to the particular problems of sparse channel estimation and equalization.

Chapter 1 gives an introduction to inverse problems and their applications. Linear discrete inverse problems are discussed and categorized. Ill-posed inverse problems, the requirement for their regularization and regularization methods are introduced. Some important application areas of inverse problems in signal processing are exemplified.

Chapter 2 studies the ULDIP type problems with sparse solutions. It is stressed that all ULDIP are ill-posed. The impossibility of solution without additional information and constraint is described. The methods to enforce sparsity as an additional information in a ULDIP type of problem is reviewed. Sparse recovery through norm minimization and greedy pursuits are discussed.

Chapter 3 covers the OLDIP type problems and discusses the regularization of ill-posed OLDIP. It is stressed that most practical OLDIP type problems in signal processing are of ill-posed nature and need to be regularized. Regularization of OLDIP by norm minimization and Tikhonov regularization are presented. Heuristic approaches to optimal regularization parameter selection are reviewed.

Chapter 4 proposes a new sparse recovery method CR for ULDIP type problems. CR works by converting a ULDIP into an OLDIP by initially constraining all entries in the solution to zero and then removing the constraints on the largest magnitude entries in each iteration. The performance and complexity of CR is compared to other sparse recovery methods and shown to outperform them for zero-one type signals.

Chapter 5 discusses the sparse channel estimation problem. Training sequence based channel estimation strategies using long and short training sequences are discussed. Different methods to estimate sparse channels with short training sequences are reviewed and compared. A new low complexity deconvolution method named PDEC based on subsequential diversity is introduced for the estimation of sparse long delay channels with short training sequences. PDEC is shown to increase deconvolution efficiency by reducing interference from past and future symbols.

Chapter 6 treats equalization of sparse channels from an inverse problem perspective. How decision feedback transforms a heavily underdetermined inverse problem to a mildly underdetermined or overdetermined one is described. Transition from a linear equalizer to DFE by reducing the horizontal dimension of the measurement (convolution) matrix is explained with the assumption that the previous decisions are correct and part of the convolution matrix corresponding to the previous decisions can be truncated. A new double-symbol detection mechanism to combat interference is proposed as an alternative to decision feedback.

Chapter 7 applies the proposed CR method and other sparse recovery methods to the particular case of sparse DTV channel estimation. Direct estimation of physical taps using PMF-convolved training sequences to undo the effect of PMF is discussed and tested.

Chapter 8 summarizes and concludes the thesis.

CHAPTER 2

ULDIP TYPE PROBLEMS UNDER SPARSITY CONSTRAINT

2.1. Preliminaries: Norm, Support and Sparsity

Norm: For real $p \geq 1$, L_p norm of a real vector x is defined as

$$\|x\|_p = \left[\sum_{i=1}^{N} |x_i|^p \right]^{1/p}. \tag{2.1}$$

For $p \geq 1$, Equality 2.1 satisfies all axioms of a norm (Desoer 1970). But for $0 \leq p < 1$, Equality 2.1 does not define a norm since, for $0 < p < 1$, triangle inequality axiom and for $p = 0$, positive scalability (homogeneity) axioms are not satisfied. This does not mean that they do not have a useful meaning or function. For instance, for $p = 0$, Equality 2.1 gives a useful metric which counts the number of nonzero elements in x. $\|x\|_0$ is called absolute sparsity of a vector. And for $0 \leq p < 1$, Equality 2.1 defines metrics whose minimization in inverse problems induces sparsity with an increasing force as p goes to zero (Chartrand 2007, Saab and Yılmaz 2010). Despite their failure to satisfy all norm axioms, due to their useful and convenient meaning, L_ps with $0 \leq p < 1$ are informally called norms.

It would be appropriate to emphasize the meaning of L_0, L_1, L_2, L_∞ norms. L_0 norm counts the number of nonzero entries in x and hence is an absolute sparsity measure, L_1 norm is the sum of absolute values, L_2 norm is the square root of the sum of squares of the entries of x and L_∞ norm gives the maximum magnitude element.

Support: The support of a vector x denoted by $supp(x)$ is a set which contains the indices of the nonzero elements of x. Therefore, $supp(x) = \{i : x_i \neq 0\}$.

Sparsity: The sparsity of a vector x is the number of nonzero elements in x and hence is given by the L_0 norm of x, which is $\|x\|_0$. A vector x is called s sparse if $\|x\|_0 \leq s$.

Sparse, Perfect Sparse and Near-Sparse Vectors: A vector x is called a sparse vector if majority of its entries can be considered to be zero. A vector x is called perfect sparse if majority of its entries are exactly zero and is called near-sparse if majority of its entries are not exactly zero but very close to zero. In most practical situations, near-sparsity rather than perfect sparsity is valid due to external effects like noise. A simple thresholding can convert a near-sparse signal to a perfect sparse signal. Although there is not an established ratio, most sparse signals of practical interest have more than three quarters of its entries zero.

2.2. Norm Minimizations for Sparse Solutions

When solving ULDIP type problems, sparsity, in most cases, is the unique constraint and therefore has to be somehow incorporated into the solution. Norm minimization can be formulated as

$$\min_x \|x\|_p \quad \text{subject to} \quad y = Ax,$$

where x is the input vector, y is the measurement vector, and A is the measurement matrix. Since L_0 norm is the absolute measure of sparsity of a vector, minimizing L_0 norm to find the sparsest solution is straightforward. In theory, L_0 norm minimization in a ULDIP type problem will produce the best performance in terms of correctness. However, most ULDIP problems in signal processing involve near-sparse vectors instead of perfect sparse ones. Near-sparse vectors will have near-zero rather than exact zero elements. This may be due to external factors like noise or to the nature of the problem. Therefore unless there is thresholding, L_0 norm is not meaningful. Besides, a more important drawback to use L_0 norm is the nonconvexity of L_0 minimization problem which prohibits the use of convex optimization algorithms. L_0 norm minimization requires an NP-hard combinatorial search (Natarajan 1995) and is therefore practically infeasible. For these reasons, the research on ULDIP type problems has been focusing on L_1 norm minimization as a convex relaxation to the non-convex L_0 norm minimization.

2.2.1. L$_1$ Norm Minimization

L_1 norm given by $\|x\|_1 = \sum_i^n |x_i|$ is not an absolute sparsity measure as L_0 norm, yet, it is closely related to the sparsity of a vector. Suppose x_1 and x_2 are two solutions for an underdetermined system $y = Ax$, the one with smaller L_1 norm is, with high probability, to have smaller L_0 norm and hence to be sparser than the other. Candès and Tao (2005) have shown that if the measurement matrix A satisfies a certain condition, minimal L_1 norm solution is equivalent to the minimal L_0 norm solution. Thanks to the convexity of L_1 norm minimization and its tolerance for near-sparse solutions, L_1 norm has been a tractable alternative to NP-hard L_0 norm minimization. There are different L_1 norm minimization formulations like Basis Pursuit, LASSO and Dantzig selector which are solved by convex optimization methods like convex, linear or quadratic programming. They all have good performances and guarantees in terms of correct recovery. The major problem with L_1 norm minimizations is that their solutions may require very large number of iterations and generally there exists no polynomial bound for that number. Therefore for very large systems, L_1 norm minimization by convex optimization may not be practical.

2.2.2. Restricted Isometry

Restricted Isometry Condition (RIC) or Restricted Isometry Property (RIP) for underdetermined measurement matrices have been introduced by Candès and Tao (2005). If a measurement matrix A holds RIP, L_1 norm minimization is equivalent to L_0 minimization for sparse recovery (Candès, Romberg and Tao 2005, 2006, Candès and Tao 2006). A matrix A satisfies the RIC with parameters (k, δ_k) if, for all k sparse vectors x, it satisfies

$$(1 - \delta_k)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta_k)\|x\|_2^2, \tag{2.2}$$

where δ_k is the smallest parameter for which the above inequality holds for all k sparse vectors $(0 < \delta_k < 1)$. If δ_k is close to zero, k or less columns of A are nearly orthogonal which is preferable for sparse solutions. This is equivalent to saying that less measurements are needed for recovery. However as δ_k gets closer to 1, more measurements are needed for sparse solutions.

There is an important theorem by Candès and Tao (2005) about sparse recovery under RIC which states that, if the measurement matrix A satisfies RIC with parameters $(3s, 0.2)$, then every s sparse vector x can be exactly recovered from its measurements Ax as a unique solution to the convex optimization problem (L_1). This theorem is later improved by the same authors to RIC parameters $(2s, \sqrt{2} - 1)$.

It is not trivial to check if a matrix satisfies RIC, yet some natural random matrices like Gaussian, Bernoulli and partial Fourier matrices are known to satisfy RIC with very high probability (Mendelson et al. 2008). For $m \times n$ Gaussian and Bernoulli matrices, number of measurements m needed to satisfy RIC for a given sparsity of s is given approximately by $m \approx s \log(n/m)$. For $m \times n$ partial Fourier matrices $m \approx s \log^p(n)$ where p is a small integer.

Candès (2008) studies the implications of the restricted isometry property for compressed sensing.

2.3. L$_1$ Norm Minimization Formulations for Sparse Solutions

2.3.1. Basis Pursuit (BP)

The best known L_1 norm minimization method is the Basis Pursuit (BP) method proposed by Chen, Donoho and Saunders (1998). The BP method can be formulated as:

$$\min_x \|x\|_1 \quad \text{subject to} \quad y = Ax, \tag{2.3}$$

where x is the input vector, y is the measurement vector, and A is the measurement matrix. The equality constraint above may be unsuitable for the noisy case, a more general formula known as Basis Pursuit Denoising (BPDN) which takes into account possible noise can be stated as

$$\min_x \|x\|_1 \quad \text{subject to} \quad \|y - Ax\|_2 \leq \epsilon. \tag{2.4}$$

It is obvious that for $\epsilon = 0$, both representations are equivalent. It has been shown by Candès and Tao (2006) that if the underdetermined matrix A possess the restricted isometry property (RIP), the BP problem is equivalent to the L_0 norm minimization. The cost of using L_1 norm instead of L_0 norm is more restriction on the matrix A. There exists no practical method to check if a matrix has the RIP property but some type of matrices such as Gaussian, Bernoulli and partial Fourier matrices have been shown to possess RIP with very high probability. To recover a sparse vector x with length n and sparsity s, BP needs $O(s \log n)$ measurements.

BP and BPDN problem can be solved through convex optimization algorithms like linear or quadratic programming requiring, in general, very large number of iterations. Therefore, BP and BPDN are currently not considered as a practical solution to ULDIP type problems.

There also exists an unconstrained formulation for BPDN in Lagrangian form. The solution regularized by its L_1 norm with a regularization parameter λ is

$$\min_x \left(\tfrac{1}{2}\|y - Ax\|_2^2 + \lambda\|x\|_1\right).$$

This formulation requires that optimal λ be determined before solving for x.

2.3.2. Linear Programming (LP)

Linear programming is used by Chen, Donoho and Saunders (1998) to solve the basis pursuit problem. Linear programming is indeed a formulation not the solution of a linear system with linear equality and linear inequality constraints (Dantzig 1963, Dantzig and Thapa 1997). Linear programming, which is a constrained optimization problem, can be formulated as,

$$\min(c^T x) \quad \text{subject to} \quad Ax = y \quad \text{and} \quad x \geq 0, \tag{2.5}$$

where x is the unknown $n \times 1$ input vector, y is the $m \times 1$ measurement vector, A is the $m \times n$ measurement matrix and $c^T x$ is the objective function to be minimized. The basis pursuit problem (2.3) can be recast into a linear program formulation (Chen, Donoho and Saunders 1998) as

$$\min(c^T x) \quad \text{subject to} \quad \beta x = y \quad \text{and} \quad x \geq 0 \tag{2.6}$$

by making the translations

$$c = (1;1), \quad \beta = (A, -A), \quad x = (u;v), \quad u, v \in R^n, \quad c \in R^{2n}.$$

The best known algorithms to solve the BP formulation are the simplex method by Dantzig (1951) and the interior point method by Nesterov and Nemirovski (1994).

In the simplex method, m linearly independent columns of A is selected and a solution is found for this set. At each step, a column is swapped with another column not in the current set to improve the objective function. Precautions are taken against cycling. Swapping is done in a way to guarantee convergence (Gill, Murray and Wright 1991). In interior point method, an initial solution to the underdetermined representation problem $y = Ax$ with $x > 0$ is found. Coefficients are modified iteratively subject to $y = Ax$, and a sparsifying transformation is applied to x. After some iterations, s significant entries will become prominent corresponding to the nonzero entries in x.

2.3.3. Least Absolute Shrinkage and Selection Operator (LASSO)

LASSO has been proposed by Tibshirani (1996). For the underdetermined noisy measurement system $y = Ax + \eta$, it aims to minimize the norm of the error $\|y - Ax\|_2$ while the norm of the solution $\|x\|_1$ is bounded by a positive constant ϵ. This is a useful formulation if we have no prior estimate of noise η but know that the sparsity of the signal is bounded. LASSO can be formulated as the constrained minimization

$$\min_x \|y - Ax\|_2 \quad \text{subject to} \quad \|x\|_1 \leq \epsilon,$$

which can be solved using quadratic programming or convex optimization methods.

2.3.4. Dantzig Selector (DS)

Candès and Tao (2007) have proposed the Dantzig selector formulation as an estimator for the solution x of an underdetermined noisy measurement system

$$y_{m\times 1} = A_{m\times n}\, x_{n\times 1} + \eta_{m\times 1} \quad (m < n),$$

where η is a Gaussian random noise with zero mean and σ^2 variance. To estimate x, DS solves the L_1 norm minimization problem

$$\min_x \|x\|_1 \quad \text{subject to} \quad \|A^T(y - Ax)\|_\infty \leq \epsilon,$$

where ϵ is a positive constant. It has been shown by Candès and Tao (2007) that if A obeys a uniform uncertainty principle with unit normed columns and if the signal vector

x is sufficiently sparse, with very high probability the norm of the error between the estimated and real input signals is upper bounded such that

$$\|x_{est} - x_{real}\|_2 \leq \Psi.$$

Dantzig selector can be cast as a linear program and can be solved by linear programming methods.

2.4. Focal Underdetermined System Solver (FOCUSS)

The focal underdetermined system solver FOCUSS has been proposed by Gorodnitsky and Rao (1997). The main idea behind the FOCUSS algorithm is that, instead of solving for x directly, a transformation of x, q is solved for where $x = Wq$ and W is the transformation matrix obtained from the previous solution for x. Therefore the new equation to be solved is $y = AWq$. The FOCUSS algorithm begins with an initial estimate x_0 of the solution which is calculated for the underdetermined system $y = Ax$ using the least norm formula $x_0 = A^T(AA^T)^{-1}y$. After finding x_0, the following steps are repeated until a sparse solution is obtained.

Step 1: $W_k = diag(x_{k-1})$

Step 2: $q_k = (AW)^{\dagger}y$ $\quad\big((AW)^{\dagger}$ is the pseudoinverse of $(AW)\big)$

Step 3: $x_k = Wq_k.$

2.5. Iteratively Reweighted Least Squares (IRLS)

IRLS has been proposed by Chartrand and Yin (2008) and Daubechies et al. (2010) for compressive sensing and sparse recovery. L_2 norm minimization is formulated as

$$\min_x \|x\|_2 \quad \text{subject to} \quad y = Ax. \tag{2.7}$$

Minimum L_2 norm solution has a convenient closed form

$$x = A^T(AA^T)^{-1}y. \tag{2.8}$$

L_1 norm minimization is

$$\min_x \|x\|_1 \quad \text{subject to} \quad y = Ax. \tag{2.9}$$

There exists no closed form solution for L_1 norm minimization. IRLS aims to solve the sparsity enforcing L_1 norm minimization problem using the convenient least squares formula. The general weighted least squares or weighted L_2 norm minimization can be written as

$$\min_x \sum_i^N (w_i x_i^2) \quad \text{subject to} \quad y = Ax, \tag{2.10}$$

where w is a weighting vector. The IRLS algorithm produces w from the previous iterate of x such that $w_i = |x_i|^{-1}$. Since $|x_i| \approx (x_i^2/|x_i|)$ the weighted least squares equation (2.10) solves an approximate L_1 norm minimization problem. An immediate problem is that if x_i is zero w_i becomes infinite. One possible solution is to regularize w_i such that

$$w_i = \left[x_i^2 + \epsilon^2\right]^{-1/2}, \tag{2.11}$$

where ϵ is a small regularization parameter. The solution to (2.10) at the nth iteration is

$$x(n) = WA^T(AWA^T)^{-1}y, \tag{2.12}$$

where W is a diagonal matrix whose diagonal elements w_i are obtained from the previous solution $x(n-1)$ using Equation 2.11.

2.6. Iterative Basis Selection: Greedy Methods for Sparse Solutions

Iterative basis selection methods also termed as "greedy methods" involve selecting one or more columns at each iteration to find an improved interim solution x. They differ from the statistical norm minimization algorithms in their speed and non-explicit imposition of sparsity. MP and its derivatives rely on the assumption that inner product of the columns of A with the measurement vector y should be high for columns which are in the active set, and should be low for columns that are not in the active set, and hence requires near orthogonality for columns of A, which also means that A^TA must not have very large non-diagonal elements relative to the diagonal ones. This is ensured by the restricted isometry condition which is also a prerequisite for statistical L_1 norm minimization algorithms.

2.6.1. Matching Pursuit (MP)

The matching pursuit (MP) algorithm for sparse recovery is proposed by Mallat and Zhang (1993) as an alternative to statistical L_1 norm minimization methods like BP and DS. Despite its very low complexity, MP had a very poor performance in terms of correct recovery. The optimized derivatives of MP have later started to challenge the L_1 norm minimization methods while still preserving the low complexity profile of the initial MP algorithm. Their incomparable speed makes them favorable to be used in practical applications and motivates researchers to improve their correctness. There exists two generations of the matching pursuit algorithms. The first generation includes the matching pursuit (MP) itself, its improved versions orthogonal matching pursuit (OMP) (Tropp and Gilbert 2007) and stagewise orthogonal matching pursuit (StOMP) (Donoho et al. 2006). The second generation, with a considerable leap in performance, includes regularized orthogonal matching pursuit (ROMP) (Needell and Vershynin 2009), compressed sampling matching pursuit CoSaMP (Needell and Tropp 2009), subspace pursuit (SP) (Dai and Milenkovic 2008) and iterative hard thresholding (IHT) (Blumensath and Davies 2009). The promises made by the second generation methods are high and are likely to be worked on and improved further. In fact, there is currently no serious alternative to them in terms of practical usage.

2.6.2. Orthogonal Matching Pursuit (OMP)

OMP (Tropp and Gilbert 2007) is the first successor to MP which encouraged researchers that there might be more potential in the MP based algorithms than the relatively discouraging performance exhibited by the basic MP. It has been shown by many researchers to have more effective results compared to the basic MP motivating the more promising second generation algorithms mentioned in the previous subsection. OMP is still included for comparison in many articles describing new methods. The basic idea behind OMP is that if entries of measurement matrix A are selected from a Gaussian distribution, its columns will be nearly orthogonal ($A^T A \approx I$). Since $A^T y = A^T Ax$, the maximum magnitude entry in $A^T y$ is expected to correspond to a nonzero entry in the sparse solution x. Therefore the first component of the sparse solution x is estimated.

Then the contribution of that component is subtracted from y and the same procedure is repeated to find the second and other components.

The OMP Algorithm:

INPUT: Measurement matrix A, measurement vector $y = Ax$, sparsity level s (number of nonzero elements in unknown vector x).

OUTPUT: Index set I.

PROCEDURE:

Initialize: $I = \emptyset$ and residual $r = y$. Iteration counter $c = 1$

Step 1: Find the index i of the maximum magnitude element in $A^T r$. Add i to the index set:

$$I \leftarrow I \cup \{i\}.$$

Step 2: Isolate the columns of A that are in the index set I to produce A_I. Using least squares solution, find a new estimate x_I for the values of x in the index set I

$$x_I = A_I^T (A_I^T A_I)^{-1} y.$$

Step 3: Update the residual: $r = y - A_I x_I$. Increment iteration counter: $c = c + 1$. If $c < s$ go to Step 1 else terminate.

End of Algorithm

2.6.3. Stagewise Orthogonal Matching Pursuit (StOMP)

StOMP (Donoho et al. 2006) differs from OMP by its index updating strategy. Instead of selecting one index at a time, StOMP selects a group of indexes for which the entries of $q = A^T r$ exceeds a predetermined threshold. At each iteration least squares solution is obtained for those entries in the index set and a residual is calculated using this least squares solution in an identical fashion to that in OMP. StOMP uses fixed number of iterations.

The StOMP Algorithm:

INPUT: Measurement matrix A, measurement vector $y = Ax$, number of iterations s.

OUTPUT: Index set I, the estimated solution x

PROCEDURE:

Initialize: $I = \emptyset$ and residual $r = y$. Iteration counter $c = 1$

Step 1: Find the index set J of the elements of $q = A^T r$ whose magnitudes are larger than $\tau_c \sigma_c$. τ_c is a threshold parameter for iteration number c and σ_c is formal noise parameter for iteration number c. Then unify the index set I with J

$$J = \{j : |j| \geq \tau_c \sigma_c\} \qquad I \leftarrow I \cup J.$$

Step 2: Isolate the columns of A that are in the index set I to produce A_I. Using least squares solution, find a new estimate x_I for the values of x in the index set I.

$$x_I = A_I^T (A_I^T A_I)^{-1} y.$$

Step 3: Update the residual: $r = y - A_I x_I$. Increment iteration counter: $c = c + 1$. If $c < s$ go to Step 1 else terminate.

End of Algorithm

2.6.4. Regularized Orthogonal Matching Pursuit (ROMP)

ROMP (Needell and Vershynin 2009) applies regularization in its index selection step. It selects the index group with maximal energy.

The ROMP Algorithm:

INPUT: Measurement matrix A, measurement vector $y = Ax$, sparsity level s.

OUTPUT: Index set I. The estimated solution x

PROCEDURE:

Initialize: $I = \emptyset$ and residual $r = y$.

Step 1: Find the index set J of the s biggest elements of $q = A^T r$ or all nonzero elements whichever set is smaller.

Step 2: Among all subsets $J_0 \subset J$ with comparable entries

$$|q(i)| \leq 2|q(j)| \quad \forall i, j \in J_0,$$

choose J_0 with maximal energy $\|q_{J_0}\|_2$.

Step 3: Unify the index set I with J_0.

$$I \leftarrow I \cup J_0.$$

Step 4: Isolate the columns of A that are in the index set I to produce A_I. Using least squares solution find a new estimate x_I for the values of x in the index set I.

$$x_I = A_I^T (A_I^T A_I)^{-1} y.$$

Step 5: Update the residual: $r = y - A_I x_I$. If $r \neq 0$ go to Step 1 else terminate.
End of Algorithm

For nearly sparse signals, the ROMP algorithm may not terminate since r may never be equal to zero. In that case, fixed number of iterations will be used.

2.6.5. Compressive Sampling Matching Pursuit (CoSaMP)

Like the ROMP method, CoSaMP (Needell and Tropp 2008) differs from OMP by selecting a group of columns in the measurement matrix A corresponding to the largest magnitude elements in x instead of a single one at each step. ROMP has the disadvantage that once it selects a column in A which does not belong to the true basis, it is not possible to remove it from the selected set. CoSaMP aims to correct this by allowing the removal of a column if necessary.

The CoSaMP Algorithm:

INPUT: Measurement matrix A, measurement vector $y = Ax$, sparsity level s (number of nonzero elements in unknown vector x).
OUTPUT: s-sparse reconstruction vector $x = a$.
PROCEDURE:
Initialize: Set $a^0 = 0, r = y, k = 0$. Repeat the following steps and increment k until the halting criterion is met.
Signal Proxy: Set $u = A^H r$, $\Omega = supp(u_{2s})$ and merge the supports $T = \Omega \text{U} supp(a^{k-1})$.
Signal Estimation: Reset all columns of A that are not in the support set T to zero to construct A_T and find the least squares solution $b = A_T^{\dagger} y$.

Prune: To obtain the next approximation, set $a^k = b_s$. (s largest elements in b are maintained and all other elements are set to zero to produce b_s).

Sample Update: Update the current samples: $r = y - Aa^k$.

End of Algorithm

2.6.6. Subspace Pursuit (SP)

SP (Dai and Milenkovic 2008) for sparse recovery has been proposed around the same time as the CoSaMP method. Like the CoSaMP method it allows the removal of a column later in iterations. This is an important improvement of both methods over the previous ones since a wrong column selection can later be corrected by removing it from the set. Before continuing to the SP algorithm steps, some definitions need to be made.

Projection: Let $y \in R^m$, and $A_I \in R^{mxI}$. Suppose $A_I^H A$ is invertible. The projection of y onto the span (A_I) is defined as

$$y_p = proj(y, A_I) := A_I A_I^\dagger y, \quad \text{where} \quad A_I^\dagger := (A_I^H A_I)^{-1} A^H$$

denotes the pseudoinverse of the matrix A_I, and A^H stands for the hermitian of the complex matrix A.

Residue: The residue vector of the projection equals

$$y_r = resid(y, A_I) := y - y_p.$$

The SP Algorithm:

INPUT: Measurement matrix A, measurement vector $y = Ax$, and K.

OUTPUT: Sparse reconstruction vector x satisfying $x_{\{1....N\}-T^l} = 0$ and $x_{T^l} = A_{T^l}^\dagger y$.

PROCEDURE:

Initialization:

1) T^0 = {The set of indices of K largest magnitude entries in $A^H y$ }.

2) y_r^0 =resid(y, A_{T^0}).

Iteration: At the lth iteration, go through the following steps.

1) $T^l = T^{l-1}$ U {The set of indices of K largest magnitude entries in $A^H y_r^{l-1}$ }.

2) Set $x_p = A_{T^l}^\dagger y$.

3) $T^l = T^{l-1}$ U {The set of indices of K largest magnitude entries in x_p }.

4) y_r^l =resid(y, A_{T^l}).

5) If $\|y_r^l\|_2 > \|y_r^{l-1}\|_2$, let $T^l = T^{l-1}$ and quit the iteration.

End of Algorithm

2.6.7. Iterative Hard Thresholding (IHT)

The Iterative Hard Thresholding algorithm for sparse recovery has been reported by Blumensath and Davies (2009) around the same time as the proposal of CoSaMP and SP algorithms. Despite its comparable efficiency, IHT has a relatively simple algorithm which only involves matrix multiplication and hard thresholding. The algorithm for the IHT method is as follows.

The IHT Algorithm:

INPUT: Measurement matrix A, measurement vector $y = Ax$, Nonlinear H_K operator which operates on vector a and resets all but K largest magnitude entries in a to zero.

OUTPUT: Sparse reconstruction vector x.

PROCEDURE:

Initialization:

Start with $x_0 = 0$.

Iteration:

$x_{n+1} = H_K(x_n + A^T(y - Ax))$.

End of Algorithm

The original IHT algorithm has later been improved by Blumensath and Davies (2010) to the normalized IHT with better performance and stability.

CHAPTER 3

OLDIP TYPE PROBLEMS AND REGULARIZATION

3.1. Introduction

Overdetermined linear discrete inverse problems (OLDIP) arise in sensory data processing applications such as in biomedical and seismic imaging. OLDIP type problems are characterized by having a measurement matrix A with more, usually much more rows than columns. However the name overdetermined can be misleading in that an overdetermined inverse problem may not have a well determined solution in many cases. Therefore overdetermined means only that the measurement matrix A has more rows (measurements) than columns (unknowns), not that the system is well defined or well determined. In fact, in many practical applications, the situation is opposite to well determinedness. There are a number of factors which make the inverse problem overdetermined yet not well determined. One major factor is that the measurements must be taken from outside the environment the subject reside in and hence sensors must be placed at a distance. As an example, seismic and biomedical imaging involve indirect and noninvasive acquisition of information about an object underground or within a body. Secondly, the number of sensors must be high to be able to see the subject in many directions. Yet possible sensor positions may be physically limited. In addition, the number of sensors must be high to overcome noise both in the sensors and in the incoming signal which may be at very low levels. These are the reasons that such systems have many more measurements than unknowns which make the system overdetermined but not necessarily well determined. It is a common characteristic of practical overdetermined systems to have a measurement matrix A whose columns are nearly linearly dependent due to the factors stated above which makes the inverse solution x very sensitive to noise. These type of systems are defined as ill-posed overdetermined systems and their direct solution perhaps by the well known least squares (LS) method yields unrealistic results usually with very large norms. Therefore a method named regularization is applied to solve ill-posed overdetermined problems.

3.2. Least Squares Solution (LS)

The linear least squares or least squares (LS) (Penrose 1955, Moore 1920) is a well known method to estimate the solution of an overdetermined linear discrete inverse problem (OLDIP) defined by

$$y_{m\times 1} = A_{m\times n}\, x_{n\times 1} + \eta_{m\times 1} \quad (m > n), \tag{3.1}$$

where x is the unknown vector, y is the measurement vector, η is the noise plus measurement error vector and A is the measurement matrix. Linear discrete overdetermined systems generally do not have an exact inverse solution, instead they have an estimate of a solution for which the error $y - Ax$ is minimized using a particular metric. If this metric is L_2 norm, the estimate is named the least squares solution. The least squares solution is formulated as

$$x_{ls} = \min_{x} \|y - Ax\|_2. \tag{3.2}$$

Linear LS solution has a closed form expression

$$x = (A^T A)^{-1} A^T y, \tag{3.3}$$

where $A^{\dagger} = (A^T A)^{-1} A^T$ is called Moore-Penrose pseudoinverse or generalized inverse of the matrix A. In linear overdetermined systems, LS solution gives a good estimate of the solution x provided that the system is noiseless or well-posed.

3.3. Ill-Posed OLDIP Type Systems and Noise

As discussed in Section 1.1.6, most practical OLDIP type problems are ill-posed due to the physical limitations stated. The measurement matrix A has columns which are nearly dependent, which cause the matrix $A^T A$ to be near singular. Therefore, the inverse of $A^T A$ has very large entries. This is not an issue as long as there is an inverse. That is, if y and A are known, the overdetermined inverse problem defined by $y = Ax$ can be solved using LS to find

$$x = (A^T A)^{-1} A^T y. \tag{3.4}$$

There seems to be no problem until the measurement vector y has noise η in it. With noise, the LS solution for x becomes

$$\begin{aligned} x &= (A^T A)^{-1}(A^T(y+\eta)) \\ &= (A^T A)^{-1}(A^T y) + (A^T A)^{-1}(A^T \eta) \\ &= x_a + x_n. \end{aligned} \tag{3.5}$$

It can be seen in the equation above that the LS solution x consists of two parts, the actual solution x_a and the noise contribution x_n. Since for an ill-posed matrix A $(A^T A)^{-1}$ has very large entries, the noise contribution $x_n = (A^T A)^{-1}(A^T \eta)$ is not a term to be omitted. Yet, in fact, it is almost always the dominant term in the solution for OLDIP type systems even when the noise is very low. This makes the solution x practically unusable.

Noise is inevitably present in all sensing systems and there is no way to remove it completely. Even a trace amount of noise may have lethal consequences for ill-posed OLDIP type systems. Regularization aims to solve this problem by striking a balance between the magnitude of the noise and magnitude of the solution without explicitly knowing neither of them.

3.4. Regularization

Regularization targets to restrict both the solutions x and the error $y - Ax$ to be within admissible limits. Here admissible limit is a subjective entity which may be determined from experiences. Regularization of OLDIP type inverse problems is not a deterministic approach and it is, indeed, not possible to regularize perfectly since it is generally not known how much to regularize. The degree of regularization is controlled by the regularization parameter and this parameter in general can not be known. Estimation of optimum regularization parameter is currently a topic of continuing research. If the regularization parameter is wrongly selected, either overregularization or underregularization may occur (see Section 1.1.6.).

3.5. Norm Based Regularization

Given the inverse problem defined by Equation 3.1, norm based regularization can be accomplished in a number of ways.

1. Regularization by solution norm minimization under a noise norm bound: The norm of the solution can be minimized for a given noise norm bound:

   $\min_x \|x\| \quad \text{subject to} \quad \|y - Ax\| \leq \epsilon^2.$

   Useful if bound for noise norm is known.

2. Regularization by noise norm minimization under a solution norm bound: The noise norm can be minimized for a given solution norm bound:

   $\min_x \quad \|y - Ax\| \quad \text{subject to} \quad \|x\| \leq \epsilon^2.$

   Useful if there is a prior knowledge of the solution norm bound.

3. Regularization by Lagrangian minimization of the sum of solution norm and noise norm:

   $\min_x(\, \|y - Ax\| + \lambda\|x\| \,).$

   There is no constraint on either of the solution and noise norms. Instead both are minimized together in a Lagrangian sum with a positive regularization parameter λ. Useful if no prior knowledge of noise and solution bounds are available but optimum regularization parameter λ needs to be determined.

3.5.1. Tikhonov Regularization

If the squared Euclidian (L_2) norm for both the solution and noise terms are used in the third item of norm based regularizations described above, this regularization is called Tikhonov regularization (Tikhonov and Arsenin 1977). For the system of equations described by Equation 3.1, the solution to the Lagrangian minimization problem

$$\min_x(\, \|y - Ax\|_2^2 + \lambda\|x\|_2^2 \,) \qquad \lambda > 0$$

is given by

$$x = (A^T A + \lambda I)^{-1} A^T y.$$

Tikhonov regularization is particularly important because it is the only norm based regularization with a closed form expression. Other regularization methods need to use mathematical optimization algorithms involving large number of iterations. The conjugate gradient and the steepest descent algorithms are common iterative optimization tools.

3.5.2. Tikhonov Regularization from a Different Perspective

There exists two different perspectives to look at the Tikhonov Regularization. The first perspective is the minimization of the error and solution norms together in a Lagrangian sum. For positive λ, equating the derivative of the functional $J(x)$ given by

$$J(x) = \|y - Ax\|_2^2 + \lambda \|x\|_2^2 \tag{3.6}$$

to zero gives the minimal solution

$$x = (A^T A + \lambda I)^{-1} A^T y.$$

The second perspective is to extend the system matrix A to A_e by adding rows that force each element in x to zero and to extend the measurement vector y to y_e by adding zeros. This is the same procedure described in the SCR method to be described in Section 4.3.for ULDIP type problems except that for ULDIP type inverse problems an underdetermined matrix A is converted to an overdetermined matrix A_e. Here an overdetermined matrix is extended to be overdetermined again. By applying an ordinary LS solution without regularization to the extended system

$$y_e = A_e x,$$

exactly the same result is obtained as that obtained by minimizing the functional in Equation 3.6 above given by

$$x_{ls} = (A_e^T A_e)^{-1} A_e^T y_e,$$

which is equivalent to

$$x_{ls} = (A^T A + \lambda I)^{-1} A^T y$$

for positive λ.

3.6. Regularization Using Singular Value Decomposition (SVD)

Although SVD based regularizations are computationally prohibitive for most practical OLDIP type problems involving very large measurement matrices, SVD is considered a theoretically important tool in explaining the characteristics of matrices including ill and well-posedness. SVD has been first introduced by the Italian mathematician Eugenio Beltrami in 1873.

3.6.1. Singular Value Decomposition

An overdetermined $m \times n$ matrix A has singular value decomposition

$$A = U \Sigma V^T = \sum_i^n u_i \sigma_i v_i^T, \tag{3.7}$$

where U is an $m \times p$ matrix, Σ is a $p \times p$ diagonal matrix with positive decreasing singular values $\sigma_i \geq 0$, and V is an $n \times p$ matrix such that

$$U^T U \ = \ V^T V = I_{pxp}. \tag{3.8}$$

The inverse of A is given in terms of SVD as

$$A^{-1} = V \Sigma U^T = \sum_i^n v_i \sigma_i^{-1} u_i^T. \tag{3.9}$$

For LS solution of OLDIP type problems, the matrix Q will represent $A^T A$ which is a $n \times n$ square symmetric matrix. Multiplying both sides of the measurement equation $y = Ax + \eta$ (Equation 3.1) by A^T, the equation becomes

$$A^T y = (A^T A)x + A^T \eta. \tag{3.10}$$

Setting $Q = A^T A$, $Y = A^T y$ and $N = A^T \eta$, the above equation becomes

$$Y = Qx + N. \tag{3.11}$$

Since $Q = A^T A$, Q is symmetric and positive semi-definite, it can be shown that the singular values σ_i are the eigenvalues of Q and $U = V$ with orthonormal columns. Therefore,

$$Q = U\Sigma U^T, \tag{3.12}$$

$$Q^{-1} = (U^T)^{-1}\Sigma^{-1}U^{-1} = U\Sigma^{-1}U^T. \tag{3.13}$$

From Equation 3.13, it is obvious that if the singular values or eigenvalues in this particular case are close to zero, Σ^{-1} hence the inverse of Q will have very large entries. This will cause x to blow up. The solution x can be extracted by multiplying both sides of Equation 3.11 with Q^{-1},

$$Q^{-1}Y = Q^{-1}Qx + Q^{-1}N$$

$$Q^{-1}Y = x + Q^{-1}N$$

$$x = Q^{-1}Y - Q^{-1}N$$

$$x = x_a - x_n$$

Here, x_a is the actual solution that is sought after and x_n is the noise component. Since $x_n = Q^{-1}N$ and if Q^{-1} has very large entries due to very small singular (eigen) values, x_n is very likely to have very large entries dominating the solution.

3.6.2. Singular Value Decomposition and Ill-Posed Matrices

Ill-posed matrices have some common properties revealed by singular value decomposition.

1- Their singular values decrease steadily towards zero with no intervals.

2- Their left and right singular vectors for small singular values become more oscillatory

3.6.3. Rank Deficient and Ill-Posed Matrices

There is an obvious distinction between an ill-posed and a rank deficient matrix in terms of their singular values. In rank deficient matrices, there is a sharp boundary between nonzero and near-zero singular values. This means nonzero singular values are prominent and near-zero singular values are very close to zero. Ill-posed matrices do not have such a sharp boundary, instead their singular values gradually decay to zero. No distinct grouping is possible. Having a sharp boundary allows rank deficient matrices to have approximate inverses through singular value truncation. Lack of such a sharp border in ill-posed matrices makes it difficult for them to have approximate inverses through truncation because truncation point is not well defined. If too high a threshold is selected below which is to be truncated, significant errors may result, if too low a point is selected, the result may be unstable.

3.6.4. Regularization by Truncated Singular Value Decomposition (TSVD)

Truncated singular value decomposition (Hansen 1987) is a method for regularization which sets zero or near-zero singular (eigen) values in Σ to infinity. Then Σ^{-1} will have zeros in place of those entries hence truncation takes place. Therefore the new Q^{-1} which is equal to $U\Sigma^{-1}U^T$ will not have very large elements. Multiplying both sides of Equation 3.11 with the modified Q^{-1}, x is obtained as

$$x = Q^{-1}Y - Q^{-1}N. \tag{3.14}$$

The first term $Q^{-1}Y$ does not correspond to the actual solution x_a any more, hence, there is an error in x_a term due to missing singular values but noise amplification is largely prevented. TSVD can be parameterized by setting a positive threshold α such that singular values with magnitude below α are set to infinity in Σ. α becomes the regularization parameter for TSVD.

3.6.5. Regularization by Filtered Singular Value Decomposition

Instead of truncation, a filtering function is used such that singular values in Σ are replaced by a new value given by

$$\sigma \leftarrow \sigma + \frac{\lambda}{\sigma} \quad \lambda > 0. \tag{3.15}$$

It can be noted that if the singular value σ is very small, its new value will be very large and if σ is large, its new value will be close to σ. Therefore the entries σ^{-1} in Σ^{-1} will not be very large for small singular values. Here, the positive λ is the regularization parameter.

This regularization is indeed identical to the Tikhonov regularization with the same positive regularization parameter λ.

3.7. Regularization Parameter Selection

An important consideration in regularization is the selection of positive regularization parameter λ. Parameter λ sets a balance between the noise norm and the solution norm. If λ is too small the noise norm will be small but the solution norm will be unrealistically high. If λ is too high the solution norm will be low yet the noise norm will be very high. In both cases, the solution will not be useful for any practical purposes. Regularization parameter selection depends on the type of regularization applied and on the estimates of the noise and solution norms.

In norm based regularization, norm bounds ϵ^2 are the regularization parameter. If a prior estimate of ϵ^2 is available, minimizations can be done for this ϵ^2.

In TSVD regularization, positive threshold α for the singular values to be truncated is the regularization parameter.

In filtered SVD, positive filter parameter λ is the regularization parameter.

In Tikhonov regularization, positive λ is the regularization parameter.

Finding the optimum parameter automatically is still a subject of continuing research.

3.7.1. Regularization Parameter for Iterative Computations

In norm based regularization, except for the Tikhonov regularization, iterative methods are needed to solve the bounded regularization problems as there is no closed form solutions for them. All iterative methods rely on error gradient to find the solution. The error functional is given as

$$N(x) = \frac{1}{2}\|y - Ax\|_2^2 \tag{3.16}$$

and its gradient is given by

$$grad(N(x)) = A^T(y - Ax). \tag{3.17}$$

Then, iteration can be done using the gradient vector to update the solution

$$x_{i+1} = x_i - \alpha\, grad(N(x_i)) \tag{3.18}$$

This is the common form of iterative regularization methods. Landweber iteration has a constant α for all iterations. The steepest descent method updates α at each iteration. α is chosen to find the minimum solution x.

If bound for error norm or solution norm or both is known beforehand, iterations can be continued until one of these criteria is met. If there is no prior knowledge of the bounds, iteration count must be bounded because iterations tend to go until there remains no noise error $N(x)$ which result in a false solution x since noise is not generally zero. Therefore, iteration count is indeed the regularization parameter which depends on the bounds for error norm and solution norm.

3.7.2. The UPRE and GCV Methods for Regularization Parameter Selection

Both the Unbiased Predictive Risk Estimator (UPRE) (Mallows 1973) and the generalized cross validation (GCV) (Wahba 1977) methods are based on predictive risk estimation. The linear measurement system in Equation 3.1 is rewritten as

$$y = Ax + \eta, \tag{3.19}$$

where y is the $m \times 1$ measurement vector, A is the $m \times n$ measurement matrix, x is the $n \times 1$ unknown input and η is the $m \times 1$ random noise. The estimate for x using the regularization parameter e is given by x_e and the actual solution by x_a. The estimation error is then $\epsilon = x_e - x_a$. It is not feasible to compute the estimation error ϵ without knowing x_a. Yet, it is feasible to estimate the predictive error p_e defined by

$$p_e = A(x_e - x_a). \tag{3.20}$$

UPRE:

The Unbiased Predictive Risk Estimator (UPRE) method for linear regression problems is also used for regularization parameter estimation. In this method, regularization parameter which minimizes a metric called UPRE is chosen as the optimal parameter. UPRE is described below in accordance with the description in Vogel (2002).
Predictive risk is defined as the mean square of the predictive error and is given by

$$\text{predictive risk} = \frac{1}{m}\|p_e\|_2^2 = \frac{1}{m}\|A(x_e - x_a)\|_2^2. \tag{3.21}$$

The estimated solution x_e will be assumed to have a linear dependence on the measured data such that

$$x_e = H_e y. \tag{3.22}$$

H_e is an $n \times m$ matrix. The influence matrix β is defined as

$$\beta_e = AH_e. \tag{3.23}$$

Then the predictive error p_e is

$$p_e = (\beta_e - I)Ax_a + \beta_e \eta. \tag{3.24}$$

Expected value of the predictive risk is

$$E(\frac{1}{m}\|p_e\|_2^2) = \frac{1}{m}\|(\beta_e - I)Ax_a\|_2^2 + \frac{\sigma^2}{m} trace(\beta_e^H \beta_e), \tag{3.25}$$

where σ^2 is the variance of the noise η. The residual r_e for the estimate x_e is defined as

$$r_e = Ax_e - y = (\beta_e - I)y = \Big[(\beta_e - I)Ax_a + (\beta_e - I)\eta\Big]. \tag{3.26}$$

Expected value of the mean square of the residual is given by

$$E(\frac{1}{m}\|r_e\|_2^2) = \frac{1}{m}\|(\beta_e - I)Ax_a\|_2^2 + \frac{\sigma^2}{m}trace(\beta_e^H \beta_e) - \frac{-2\sigma^2}{m}trace(\beta_e) + \sigma^2. \quad (3.27)$$

Rewriting Equation 3.25 using Equation 3.27 produces

$$E(\frac{1}{m}\|p_e\|_2^2) = E(\frac{1}{m}\|r_e\|_2^2) + \frac{2\sigma^2}{m}trace(\beta_e) - \sigma^2. \quad (3.28)$$

The UPRE functional $U(e)$ is defined as

$$U(e) = \frac{1}{m}\|r_e\|_2^2 + \frac{2\sigma^2}{m}trace(\beta_e) - \sigma^2. \quad (3.29)$$

Since the expected value of $U(e)$ is equal to the expected value of the predictive risk, $U(e)$ is an unbiased estimator for it. The regularization parameter e which minimizes UPRE is chosen as the optimal regularization parameter. UPRE needs prior noise variance information.

GCV:

With the same terminology as in the description of UPRE, the generalized cross validation (GCV) chooses the regularization parameter e which minimizes the functional

$$GCV(e) = \frac{\frac{1}{m}\|r_e\|^2}{\left[\frac{1}{m}trace(I - \beta_e)\right]^2} \quad (3.30)$$

In contrast to UPRE, GCV method does not need a prior noise variance estimate. Both UPRE and GCV methods assume that η is discrete white noise. If prior noise variance information is available, UPRE performs better than GCV, GCV should otherwise be preferred.

If the SVD of A is available, it is easy to implement both UPRE and GCV methods, otherwise approximate methods need to be used.

Vogel (2002) treats some implementation issues regarding UPRE and GCV and makes a comparison between them.

3.7.3. L-Curve Method for Regularization Parameter Selection

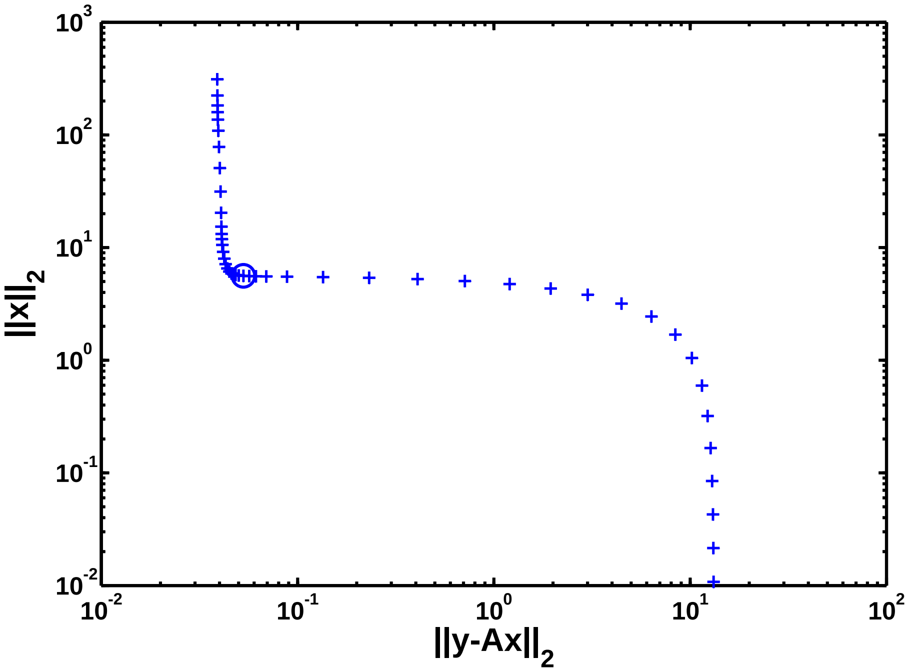


Figure 3.1. L-Curve for the Tikhonov regularization of the shaw experiment (Hansen 2007) with 47 dB SNR. L-Curve is a log-log plot of solution norm $||x||_2$ on y axis versus noise error norm $||y - Ax||_2$ on x axis for varying values of regularization parameter λ. The curve has its name from the L like shape of the plot for ill-posed systems. The center of the circle indicates the minimum error point. Note that the minimum error point occurs not exactly at the corner of the curve but it is slightly to the right of the corner. The minimum point is determined by using the actual x which is not known in the real context of the problem.

L-Curve method has been introduced by Hansen and O'Leary (1993) for optimal regularization parameter selection. In this method, the norm of the estimated solution $||x||_2$ is plotted against the norm of the estimated noise error $||y - Ax||_2$ for the increasing values of the positive regularization parameter λ in a log-log plot. The horizontal axis is $\log_{10}||y - Ax||_2$ and the vertical axis is $\log_{10}||x||_2$. For an ill-posed OLDIP type problem, the shape of the curve resembles that of the letter L, hence the name L-Curve. For very small values of λ, the regularization is very weak and the estimated solution norm is very high due to near singularity of matrix A^TA. Therefore, the plot begins at the upper left corner of the axes. As λ increases, regularization sets in and the norm of the solution decreases sharply while the norm of noise error increases slightly. At some value of λ, decrease in the solution norm slows down and increase in noise error norm speeds up which corresponds to the corner of the log-log plot. L-Curve method argues that the optimal regularization parameter lies around this corner since here both the solution and the noise error norms are at their lowest levels. At other parts of the curve, either the solution norm or the noise error norm is very high. L-Curve method can be used for any parametric regularization method but it is commonly used for Tikhonov regularization in which regularization parameter λ can be varied continuously. Hence the optimum parameter is found by finding the point of maximum curvature at the corner. There are some points to consider about the L-Curve method;

For the L-Curve method to work properly, the plot must have the L shape otherwise there will be no corner to detect. Not all ill-posed problems have proper L shaped plot.

There is no theoretical basis for the maximum curvature point to be the optimum λ although logically the optimum λ is expected in or around the corner.

L-Curve method does not presume a noise or solution norm. This is advantageous if there is no prior estimate available about solution and noise norm. If noise or solution estimate is available, L-Curve method, by not making use of this, is disadvantageous.

L-Curve method is not convergent. It does not provide the actual solution in no noise case.

3.7.4. Relation between Optimum Regularization Parameter and Noise Variance

It is often asked if noise variance is known beforehand, does this information help to estimate optimum regularization parameter? It is possible to test this by using a constant variance Gaussian noise and a known input. The result of this test is tabulated below for various noise variances for the shaw experiment.

From Table 3.1 it can be observed that for the same noise variance σ^2 and for the same input x, the optimum regularization parameter λ_{opt} fluctuates largely between 0.000050 and 0.000929. This indicates that there is not a very tight relationship between the noise variance and the optimum regularization parameter. There is another point in choosing λ_{opt}. What if ten times or one tenth of the λ_{opt} is chosen? Would the actual error $\|x_2 - x\|_2$ will change drastically? This can be tested using the last row. If $10 \times \lambda_{opt}$ is chosen, the actual error $\|x_2 - x\|_2$ is calculated to be 0.7115 and if $\lambda_{opt}/10$ is chosen, the actual error $\|x_2 - x\|_2$ is calculated to be 0.6621. Both results give almost double the error for λ_{opt}. Knowing that the norm of the actual solution $\|x\|_2$ is equal to 5.6467, the results are not a drastic yet a significant deviation.

Table 3.1. Optimum regularization parameter for constant noise variance.

Noise Variance	Optimum Reg. Param.	Actual Error
σ^2	λ_{opt}	$\|x_2 - x\|_2$
0.00011	0.000373	0.475199
0.00011	0.000072	0.400888
0.00011	0.000645	0.599660
0.00011	0.000448	0.618440
0.00011	0.000050	0.460500
0.00011	0.000311	0.493619
0.00011	0.000774	0.359588
0.00011	0.000180	0.581743
0.00011	0.000774	0.276563
0.00011	0.000538	0.422033
0.00011	0.000645	0.544617
0.00011	0.000373	0.406304
0.00011	0.000311	0.534923
0.00011	0.000373	0.326379
0.00011	0.000216	0.454709
0.00011	0.000125	0.550521
0.00011	0.000180	0.405888
0.00011	0.000072	0.326378
0.00011	0.000929	0.812508
0.00011	0.000538	0.292526

It can be concluded from the above discussion and Table 3.1 that precise estimation of λ_{opt} is not possible even if exact noise variance value σ^2 is available. Therefore optimum regularization parameter estimation methods which rely on noise variance values can not produce an exact estimation. And in many cases even a good estimate of noise variance is not available.

CHAPTER 4

CONSTRAINT REMOVAL (CR) FOR SOLVING SPARSE ULDIP TYPE PROBLEMS

4.1. Introduction

This chapter introduces a new algorithm named constraint removal (Şahin and Özen 2012) for solving sparse ULDIP type problems. As explained in Chapter 2, ULDIP type inverse problems are unsolvable unless there exists sufficient additional information or constraint about how the solution should be. Sparseness of the solution is an important and in most cases the unique additional information. Using sparseness of solution to solve a ULDIP type problem is therefore called sparse recovery. The constraint removal algorithm is preceded by the sequential constraint removal algorithm (SCR). SCR differs from CR by removing the constraints one by one instead of selecting a group. Both algorithms SCR and CR will be explained in this chapter.

4.1.1. Sparsity and Sparse Recovery

A vector is K-sparse if the number of its nonzero entries is equal to or less than the positive integer K. A vector is sparse if the majority of its entries are zero. Sparse recovery refers to the problem of reconstructing a sparse signal x from a number of linear measurements y in which the number of measurements m is smaller than the number n of the entries in x. This can be formulated as an underdetermined system of equations:

$$y^{m\times 1} = A^{m\times n}x^{n\times 1} \quad (m < n). \tag{4.1}$$

Compressive sampling (Donoho 2006a, Candès 2006) aims to sample data in compressed form by finding the sparsest solution to (4.1). The L_0 norm of a vector is equal to the number of its nonzero entries and therefore to its sparsity. Due to the absence of scaling property, L_0 norm is not mathematically a norm, although it became a usual practice to call it a norm. The straightforward way to find the sparsest solution to (4.1) is thus by minimizing the L_0 norm of the solution x:

$$min\|x\|_0 \quad \text{subject to} \quad y = Ax. \tag{4.2}$$

The solution to the nonconvex problem in (4.2) requires a combinatorial search which is NP-hard and therefore practically infeasible. It has been shown in Donoho (2006b), Candès and Tao (2005) that if the matrix A in (4.1) satisfies certain properties, it is equivalent to use the L_1 norm instead of the L_0 norm for minimization. The L_1 norm minimization problem is called *basis pursuit* (Chen, Donoho and Saunders 1998) and is solved by linear programming methods. This is frequently referred to as BP-LP or simply LP, and is written as

$$\text{Basis Pursuit:} \quad min\|x\|_1 \quad \text{subject to} \quad y = Ax. \tag{4.3}$$

Both L_0 and L_1 norm minimizations are computationally complex which have been the motivation to search for alternative greedy pursuit algorithms with much lower complexities and with comparable performances. The first is the matching pursuit (MP) (Mallat and Zhang 1993) algorithm. The MP algorithm has been followed by many derivatives in which the latest two algorithms stand out. They are the subspace pursuit (SP) (Dai and Milenkovic 2008) and the compressive sampling matching pursuit (CoSaMP) (Needell and Tropp 2009) algorithms. They both represent a leap in the recovery performance of the MP based algorithms while still preserving the low complexity profile of the basic MP algorithm. The constraint removal (CR) algorithm introduced in this presentation is the first greedy algorithm with nearly the same performance of LP. The CR algorithm has a different motivation and different initial steps than all other MP based greedy pursuits as shown in the following section.

4.1.2. Converting Sparsity into Rows

In LDIP type systems, which are represented by the equation $y = Ax$, every linearly independent row in system matrix A is a constraint necessary to produce a unique solution. If the number of constraints (linearly independent rows) is equal to the number of unknowns in x, the system has a single solution. Whereas in ULDIP type systems, the system has missing rows (constraints) and hence has infinitely many solutions. One remedy to this problem could be to fill the missing rows somehow using some additional information. Sparsity in fact is the only such criterion in many ULDIP type problems. Therefore the question is how to convert the sparsity information into rows. One idea is to force all elements in input x to zero. The justification is that if the input x is known to be sparse, for example 70 percent of its elements are zero, forcing all of them to zero is 70 percent correct and 30 percent false. This means correctness is more than falsity and solution of such a system should give an approximate result and this should be the case as long as the number of zeros in input is more than 50 percent.

4.2. Motivation for SCR and CR

Since the system in (4.1) is underdetermined, a direct solution is not possible and infinitely many solutions exist. One remedy is to force the solution to be sparse. The support of a vector x denoted by $supp(x)$ is a set which contains the indexes of the nonzero elements, i.e. $supp(x) = \{i : x_i \neq 0\}$. Since both the support and sparsity level of the signal x are not known, the only intuitive thing to be done is to press all entries in x to zero. Doing so is more correct than false if x is indeed sparse. Pressing all entries to zero is done by vertically concatenating the underdetermined matrix A with an $n \times n$ identity matrix below and elongating the measurement vector y by padding it with n zeros downwards as shown in (4.5). The new system of equations become

$$y_{new}^{(m+n)\times 1} = A_{new}^{(m+n)\times n} x^{n\times 1}, \tag{4.4}$$

where

$$A_{new}^{(m+n)\times n} = \begin{bmatrix} A^{m\times n} \\ I^{n\times n} \end{bmatrix}, \quad \text{and} \quad y_{new}^{(m+n)\times 1} = \begin{bmatrix} y^{m\times 1} \\ 0^{n\times 1} \end{bmatrix}. \tag{4.5}$$

Equation 4.4 represents an overdetermined system with an upper and a lower part. The upper part is the initial system for which a sparse solution is sought, the lower part is an identity matrix pressing all entries in x towards zero. Therefore, both parts will compromise on the solution. Multiplying both sides of Equation 4.4 with A_{new}^T, the equation transforms to

$$A_{new}^T y_{new} = A_{new}^T A_{new} x. \tag{4.6}$$

The identities in (4.5) imply that $A_{new}^T A_{new} = (A^T A + I)$ and $A_{new}^T y_{new} = A^T y$. Therefore, Equation 4.6 is equivalent to

$$A^T y = (A^T A + I)x. \tag{4.7}$$

Multiplying both sides with $(A^T A + I)^{-1}$ the initial solution $x = x_{init}$ is

$$x_{init} = (A^T A + I)^{-1} A^T y, \tag{4.8}$$

where x_{init} is the initial solution for the underdetermined Equation 4.1, and where all entries in x are equally pressed towards zero within an overdetermined context. After finding x_{init}, an initial estimate of the support can be obtained. Assuming that the maximum expected sparsity is K, the indexes of the largest K entries in x_{init} is the first estimate of the support. This means the pressure on the entries in the support can be released by removing the corresponding K rows in Equation 4.4 to obtain the new overdetermined equation

$$y_{new}^{(m+n-K)\times 1} = A_{new}^{(m+n-K)\times n} x^{n\times 1}. \tag{4.9}$$

Applying the same procedure described in the Equations 4.6 through 4.8, to Equation 4.9, the new estimate of the signal x is given by

$$x = (A^T A + D)^{-1} A^T y. \tag{4.10}$$

This time, the solution for x involves a diagonal matrix $D^{n\times n}$ in place of the identity matrix $I^{n\times n}$ in Equation 4.8 due to the removed constraints on the estimated support. The diagonal matrix D is obtained by setting the elements of the identity matrix I to zero corresponding to the current support estimate. This procedure is repeated until the support does not change. The CR algorithm consists of two interleaved parts in each iteration, estimating the solution x for a given support and estimating the new support from the solution estimate x. Equations 4.4 through 4.9 serve only to describe the motivation behind CR. Equation 4.10 is the main equation to be iterated.

4.3. Sequential Constraint Removal (SCR)

4.3.1. The SCR Algorithm

The term inverted support need to be defined before describing the SCR algorithm. Inverted support of a vector is obtained by logically inverting its support.

Algorithm 4.1: The Sequential Constraint Removal Algorithm

Input: Measurement matrix $A^{m \times n}$, measurement vector $y^{m \times 1}$ and maximum expected sparsity $K = floor(m/2)$.

Output: signal $x_{out}^{n \times 1}$.

Initialize: $x_{init} = (A^T A + I)^{-1} A^T y$ and $x_{new} = x_{init}$. $s' = ones^{n \times 1}$.

Step-1: Set $x_{old} = x_{new}$. Set the entry in inverted support vector s' whose index correspond to the next largest magnitude entry in x_{new} to zero.

Step-2: Produce a diagonal matrix $D = diag(s')$ from the vector s'.

Step-3: Solve $x_{new} = (A^T A + D)^{-1} A^T y$.

Step-4: If $x_{new} = x_{old}$ end iterations, set $x_{out} = x_{new}$ and terminate; else go to Step 1.

End of Algorithm 4.1.

Algorithm 4.1 describes the sequential constraint removal algorithm. At the end of the iterations, x_{out} will be our sparse solution, and the diagonal s' of the matrix D will contain the inverted support for x_{out}. At each iteration the next largest entry is selected until the remaining entries are all zeros. The third step involves least squares evaluation which can be solved by Gaussian elimination.

4.3.2. An Example to Demonstrate the SCR Algorithm (no noise)

The original system: $y = Ax$ with the known input x. Throughout the example solution, x will be assumed to be unknown.

$$\begin{bmatrix} 1 \\ 1 \\ -3 \\ 3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & -1 & -1 & 1 & 1 & 1 & -1 \\ 1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 \end{bmatrix} \cdot \begin{bmatrix} -1 \\ 0 \\ 0 \\ 0 \\ 2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

STEP1: Extend the original system $y = Ax$ to the extended system: $y_e = A_e x$ $(\lambda = 1)$

$$\begin{bmatrix} 1 \\ 1 \\ -3 \\ 3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & -1 & -1 & 1 & 1 & 1 & -1 \\ 1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \\ \\ x \\ \\ \\ \end{bmatrix}$$

STEP2: Least squares solution to the extended system:

$$x_{ls} = \begin{bmatrix} -0.2351, & -0.7553, & -0.4002, & -0.0218, & 0.9204, & 0.4002, & 0.0218, & -0.1433 \end{bmatrix}^T$$

STEP3 and STEP4: The maximum magnitude component is at the 5th index position. Then zero the 4 + 5 = 9th row in A_e.

$$\begin{bmatrix} 1 \\ 1 \\ -3 \\ 3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & -1 & -1 & 1 & 1 & 1 & -1 \\ 1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \\ \\ \\ x \\ \\ \\ \end{bmatrix}$$

STEP2: (Second Iteration) Least squares solution to the extended system:

$$x_{ls} = \begin{bmatrix} -0.4612, & -0.2533, & -0.2079, & 0.0756, & 1.7543, & 0.2079, & -0.0756, & 0.1777 \end{bmatrix}^T$$

STEP3 and STEP4: The second maximum magnitude component is at the first index position. Then zero the 4 + 1 = 5th row in A_e.

$$\begin{bmatrix} 1 \\ 1 \\ -3 \\ 3 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & -1 & -1 & 1 & 1 & 1 & -1 \\ 1 & -1 & 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & 1 & 1 & 1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \\ \\ \\ x \\ \\ \\ \end{bmatrix}$$

STEP2: (Third Iteration) Least squares solution to the extended system:

$$x_{ls} = \begin{bmatrix} -1.0000, & 0.0000, & 0.0000, & -0.0000, & 2.0000, & 0.0000, & 0.0000, & 0.0000 \end{bmatrix}^T$$

This is the actual sparse solution, continuing iteration will not change the result and next maxima will be zero. There were only two iterations for this problem since there were only two nonzero elemnts in x. In no noise case, the iteration count will be equal to the number of nonzero elemnts in x.

4.4. Constraint Removal (CR)

The SCR algorithm proceeds one by one each time selecting the next maximum until the value of next maximum element drops below a predetermined threshold as the stopping criterion. This has the disadvantage that iteration count must be equal to the number of significant magnitude elements which is computationally demanding. In constraint removal (CR), instead of selecting one element at each iteration, a group of largest K elements is selected until the group does not change. K is the maximum expected number of nonzero elements. CR in general needs much less iterations than SCR and hence is computationally more efficient. Another advantage of CR over SCR is that once a wrong element is selected in SCR it generally remains so but in CR a wrongly selected element may be eliminated in ensuing iterations.

4.4.1. The CR Algorithm

Algorithm 4.2: The Constraint Removal Algorithm

Input: Measurement matrix $A^{m \times n}$, measurement vector $y^{m \times 1}$ and maximum expected sparsity $K = floor(m/2)$.

Output: signal $x_{out}^{n \times 1}$.

Initialize : $x_{init} = (A^TA + I)^{-1}A^Ty$ and $x_{new} = x_{init}$.

Step-1: Set $x_{old} = x_{new}$. By setting the largest K entries in x_{new} to zeros and all others to ones, produce an inverted support vector s'.

Step-2: Produce a diagonal matrix $D = diag(s')$ from the vector s'.

Step-3: Solve $x_{new} = (A^TA + D)^{-1}A^Ty$.

Step-4: If $x_{new} = x_{old}$ end iterations, set $x_{out} = x_{new}$ and terminate; else go to Step 1.

End of Algorithm 4.2.

The term inverted support need to be defined before describing the CR algorithm. Inverted support of a vector is obtained by logically inverting its support.

Algorithm 4.2 describes the constraint removal algorithm. At the end of the iterations, x_{out} will be our sparse solution, and the diagonal s' of the matrix D will contain the inverted support for x_{out}. At each iteration a group of the largest K entries is selected until the group does not change. A group element may stay in the group or may be eliminated from the group at each iteration prohibiting fixed false elements. The third step involves least squares evaluation which can be solved by Gaussian elimination.

4.4.2. Simulations and Comparison

Simulation for Zero-One Signals

Figure 4.1 below shows the simulation results for zero-one signals. Zero-one signal has its elements from the set $\{0,1,-1\}$ The CR algorithm is simulated and compared to the SP and LP algorithms. The SP method is chosen as a representative algorithm for MP based methods since it has a superior performance to its alternatives and is very similar to CoSaMP with a slightly better performance. The LP algorithm is a statistical procedure for sparse recovery with no polynomial bound for computation yet in terms of correctness it has the best performance and for that reason is used as a benchmark.

Simulation for Gaussian Random Signals:

Figure 4.2 shows the simulations and comparison of the same methods for Gaussian random signals. It can be observed that both the SP and CR algorithms have sharp jumps beginning from sparsity 30. This phenomenon can be explained such that this time both the values and the positions of the elements of the input vector x is Gaussian along with the elements of the system matrix A adding an extra randomness to the inverse system which decrease the predictability of the output. However, generally speaking, both algorithms have similar performances slightly above LP with CR being slightly superior to SP.

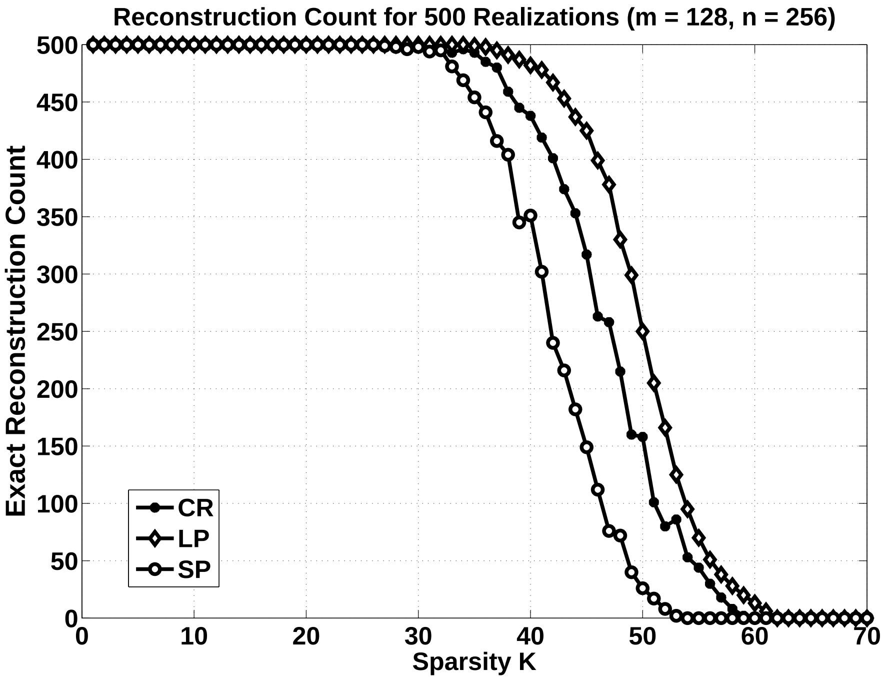


Figure 4.1. The CR algorithm has been simulated and compared to the SP and LP methods for zero-one signals. The SP method is chosen as a representative method for the MP based sparse recovery algorithms since it is very similar to the CoSaMP algorithm with a slightly better performance. The LP method is included as a benchmark.

4.5. Convergence

The convergence of the CR algorithm is straightforward. It is assumed that x_a is the actual K-sparse solution so that $y = Ax_a$. Referring to the main iterated equation in Step 3 of the CR algorithm and replacing y with Ax_a, the estimated solution x_{new} becomes

$$x_{new} = (A^TA + D)^{-1}A^TAx_a, \tag{4.11}$$

$$x_{new} = (A^TA + D)^{-1}(A^TA + D - D)x_a, \tag{4.12}$$

$$x_{new} = x_a - \underbrace{(A^TA + D)^{-1}Dx_a}_{error}. \tag{4.13}$$

Equation 4.13 indicates that if D contains in its diagonal the inverted support of the actual solution x_a so that $Dx_a = 0$, the solution x_{new} converges to the actual solution x_a. Finally, it needs to be shown that the diagonal of matrix D approaches the inverted support of x_a

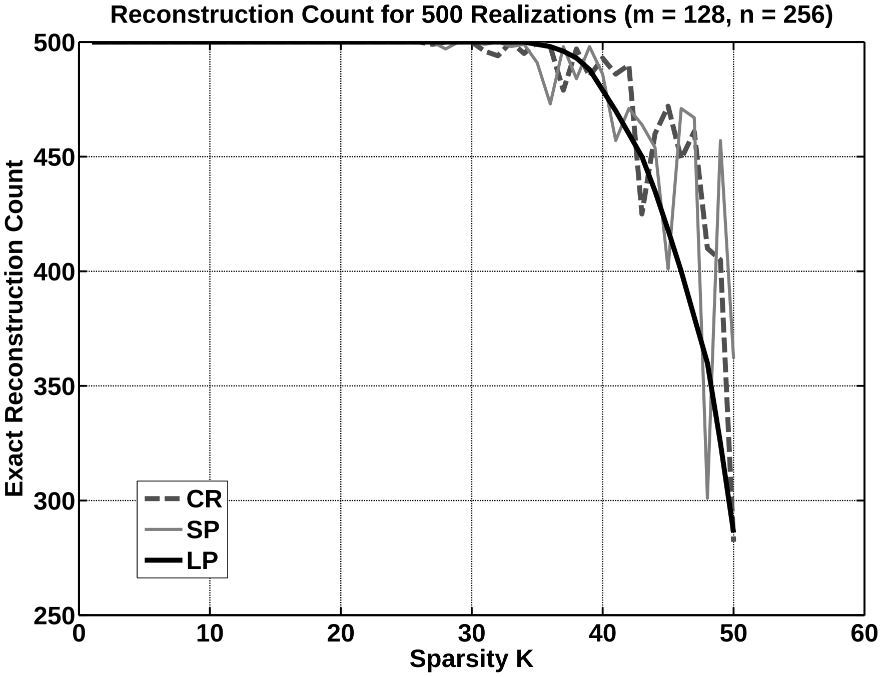


Figure 4.2. The CR algorithm has been simulated and compared to the SP and LP methods for Gaussian random signals. The SP method is chosen as a representative method for MP based sparse recovery algorithms. The LP method again is included as a benchmark. The thick black line represents the LP results, the dashed gray line represents the CR and the thin gray line represents the SP results.

beginning from the first iteration where $D = I$. In the first iteration of the CR algorithm

$$x_{new}[1] = x_a - (A^T A + I)^{-1} x_a, \tag{4.14}$$

the number of columns used in $(A^T A + I)^{-1}$ is equal to the sparsity K of x_a. If $x_{new}[1]$ is well aligned (have a similar order of magnitudes) with x_a, in the second iteration, the vector Dx_a will be sparser than x_a. As a result, the next solution $x_{new}[2]$ will be closer to and better aligned with x_a since less columns from $(A^T A + D)^{-1}$ will be involved. This will continue in the subsequent iterations until convergence occurs where $Dx_a = 0$ and $x_{new} = x_a$.

From Proposition 3.1 in Needell and Tropp (2009), if A satisfies the restricted isometry condition for sparsity K and parameter δ, the bounds for the error in the first iteration can be written as

$$\frac{\|x_a\|_2}{(2+\delta)} \leq \|(A^T A + I)^{-1} x_a\|_2 \leq \frac{\|x_a\|_2}{(2-\delta)}. \tag{4.15}$$

It is therefore crucial to have a good alignment in the first and subsequent iterations which depends on how close A^TA is to identity and therefore on the restricted isometry constant. Even when $\delta = 0$, the norm of the error in the first iteration is not zero but is perfectly aligned with x_a, therefore in the second iteration actual solution x_a will be recovered exactly. If A satisfies the restricted isometry condition, the reconstruction will be exact yet the value of the RIC parameter δ below which reconstruction is guaranteed needs to be determined.

4.6. Noise and Stable Recovery

The measurements may be corrupted by noise such that $y = Ax + e$, where e represents the noise vector. For a stable recovery, measurement noise e and corresponding signal deviation Δx must be comparable. This can be shown using the main iterated equation in the CR algorithm $x[i] = (A^TA + D)^{-1}A^T(y + e)$. The deviation in x is given by $\Delta x = (A^TA + D)^{-1}A^Te$. Since $(A^TA + D)^{-1}$ is always nonsingular, the norm of Δx will be comparable to the norm of measurement noise e ensuring stability.

4.7. Conclusion

CR performs better than SP for zero-one type sparse signals with nearly equal performance for Gaussian type signals. Despite its apparent high complexity $O(n^3)$, its simulation runtimes were far below than expected and below that of BP-LP perhaps due to much fewer number of iterations and early termination of the Gaussian elimination for sparse signals. In addition, CR does not require prior sparsity knowledge.

CHAPTER 5

SPARSE CHANNEL ESTIMATION

5.1. Introduction

5.1.1. Channel

The medium between the source and the destination through which information signals travel is called channel. Every channel has its own characteristic depending on its structure. Every communication channel is in fact a filter. This is due to the fact that channels are selective on frequencies. Channels can be modeled by magnitude and phase responses in frequency domain and impulse response in time domain. An ideal channel is a channel in which there is only a time delay between the transmitted and received symbols as in a uni-path transmission with constant velocity for all frequencies. However, even in a highly protected single-mode fiber optic channel in which there exists only one path between the source and destination, channel is far from ideal because the velocity of light in the fiber is wavelength dependent which cause dispersion. Also the bandwidth of wavelengths that travel increases with the modulation frequency. Similar considerations also apply to the protected uni-path channels like coaxial and twisted pair cables. The case with the unprotected channels like air and underwater is worse. The are more factors that contribute to the distortion of the signal received by the receiver. Thermal noise, human generated noises, radiation, scattering, reflection are some to count. One of the major causes of distortion in air and underwater transmission is the so called multi-path effect caused by reflections. In this thesis, the focus will be on this type of distortion.

5.1.2. Multi-Path Effect, ISI, Frequency Selectivity, Dispersion

Multi-path effect can be visualized as having one receiver and multiple transmitters clustered at the same position each transmitting the same sequence of symbols with different time delays and attenuation. The samples received at the receiver therefore do not belong to a single symbol instead are the superposition of multiple symbols with different attenuation factors and phases. This prevents direct detection of a symbol from a single sample. This superposition of multiple symbols into one sample is called intersymbol interference (ISI). Channels having ISI are frequency selective. An equivalent term for ISI generally used in fiber optic context is dispersion.

5.1.3. Mobile and Terrestrial Digital Communication Channels

Most digital communication systems involve transmission of highly compressed data in digital packets as in DTV or HDTV broadcasting and cellular mobile communications. High symbol rates coupled with high compression ratios make such channels challenging to estimate because they may span several hundred symbols length. Also the equalization of these channels is equally challenging due to the same reason. Symbol error rate gains more significance as the compression ratio increases as in h264 and MPEG2 video transmissions because decompression of them is not tolerant to symbol errors since they involve a very tight sequential synchronization. If synchronization is lost repeatedly, this may cause fatal frame losses and degraded quality of service.

Robust channel estimation and equalization is required for high symbol rate and high compression ratio transmissions.

5.2. Channel Estimation

It is apparent from the previous section that detection of a symbol from an ISI corrupted single sample is not possible since this sample does not wholly belong to that single symbol. In order to detect a symbol, multiple samples that may bear information about that symbol must be carefully processed together. Intuitively, in order to undo the effect of a channel, channel characteristic must be known by the receiver. In other words, we need to know what the channel is doing to each symbol. Since this information is contained in channel impulse response, we need to estimate it before doing any symbol detection.

Channel impulse response (CIR) estimation or shortly channel estimation is an important inverse problem encountered in digital communication systems. Estimated channel impulse response is eventually used to equalize the frequency response of that channel for efficient symbol detection. Digital wireless and terrestrial communication channels in general exhibit frequency selectivity due to a phenomenon known as multipath effect and the consequent inter-symbol interference (ISI). Estimated CIR is used to equalize the overall frequency response between the transmitter and the symbol detector of the receiver in order to recover efficiently the correct symbols as if they are received through a uni-path constant velocity transmission.

5.3. Channel Estimation Methods

Channel estimation methods can be divided into three main categories depending on the usage of a training sequence.

1. Blind Channel Estimation: Channel is estimated from received signals only. There is no knowledge of what is being transmitted. Blind channel estimation uses the constant modulus property of the transmitted symbols and usually requires long iterations to converge. Although it offers much better channel utilization due to the absence of a training sequence, it is not a robust method in fast changing environments and hence not utilized in practical systems as of now.

2. Training Sequence Based Channel Estimation: Known training sequences are transmitted. Receiver knows the transmitted training sequence and when it is transmitted. Receiver uses the known training sequence and corresponding received signal to estimate the channel. It may require long training sequences for long channels because training sequences must be longer than the channel for a robust estimation. Training sequence based channel estimation provides faster and more accurate estimation due to known training sequence. Channel utilization is penalized due to the transmission of long training sequences.

3. Semi-Blind Channel Estimation: Merges both the blind and the training sequence based channel estimations to be able to use shorter training sequences with less iterations. However, this topic is still in the research stage without an active usage as of now.

All three methods have their advantages and disadvantages. Second and third methods are more likely to be preferred. In the case of high bit-rate transmissions, accuracy gains much significance due to high compression ratios. Training sequence based channel estimation is an inverse problem and can be defined within the context of linear inverse problems formulated by finding unknown input x from known output y and known matrix A such that $y = Ax$. Here y stands for the signal samples received, A is the Toeplitz convolution matrix constructed from the transmitted training sequence and x is the unknown channel.

5.4. Training Sequence Based Channel Estimation and Sparse Channels

Training sequence based channel estimation depends on transmission of a sequence of symbols known by both transmitter and receiver in order to probe the channel characteristic, known as channel impulse response. It is known that the samples received by the receiver while the training sequence is transmitted is the convolution of the channel impulse response with the training sequence. It is straightforward to think that deconvolution of received sample sequence with the known training sequence should yield the channel impulse response.

Training sequences are generally pseudo random sequences with no specific pattern. Training sequences can also be designed in a different way depending on the channel and device characteristics.

Training sequence based channel estimation is an actively used channel estimation method for its robust performance and simplicity. For dynamic channel estimation transmitter sends a known training sequence at certain periodic time instants to receiver. Receiver detects the beginning of the training sequence, stores the received data corresponding to the training sequence and tries to estimate the channel from them.

Sparse Channels And Training Sequence length

The most common way to estimate a channel in practice is by finding the least squares solution to the linear system $y = Ah$ in which y is the received data corresponding to the training sequence, A is the Toeplitz convolution matrix constructed from the known training sequence, and h is the unknown channel impulse response data. A

Toeplitz matrix is the one whose $(n + 1)$th row is the nth row shifted to right by one. For a robust channel estimation, the $n \times n$ Toeplitz matrix A must have only training sequence data which requires a training sequence of length at least $2n$. For low bit-rate and short channels this may not be a concern. However, for high bit-rate long delay spread channels, periodically sent long training sequences with at least two times the length of the channel will cause significant reduction in channel efficiency. Therefore, there is a need for a method using shorter training sequences. With a shorter than $2n$ length training sequence, it is not possible to construct an $n \times n$ square matrix A, instead a smaller $m \times n$ rectangular matrix A with $m < n$ can be constructed. This makes the linear system $y = Ah$ an underdetermined linear system. It has been shown in previous chapters that if the unknown input h to a linear system $y = Ah$ is known to be sparse, then full $n \times n$ matrix A is not necessary to recover h from y, instead an $m \times n$ matrix A with $m < n$ can be sufficient. The value of m depends on the level of the sparsity. High bit-rate DTV channels are almost always sparse allowing training sequences shorter than the length $2n$ to be used for their estimation. Therefore sparse recovery techniques such as CR and SP methods mentioned in previous chapters can be used for this purpose.

There are currently five main training sequence based channel estimation methods:

1. Least Squares And Least Norm Channel Estimation
2. Maximum likelihood Channel Estimation
3. Correlation Based Channel Estimation
4. Blended Least Squares Estimation
5. De-convolution Based Channel Estimation

5.4.1. Least Squares and Least Norm Channel Estimation

Least squares (LS) channel estimation finds the channel that minimizes the L_2 norm of the error between the sampled data and the expected data which is the convolution of the training sequence and the channel impulse response. LS is possible if the training sequence is sufficiently long, that is, the system is overdetermined. In the short training sequence case, when the system is underdetermined, least norm solution finds the channel with the minimum L_2 norm.

As mentioned in Section 5.4, channel estimation problem in the presence of a training sequence is equivalent to solving a linear discrete inverse problem,

$$h = \Phi(y, A) \tag{5.1}$$

for the linear channel model,

$$y = Ah + \eta, \tag{5.2}$$

where y is the $m \times 1$ vector of the received samples, A is the $m \times n$ Toeplitz convolution matrix constructed from the training sequence symbols, η is the $m \times 1$ noise vector and h is the $n \times 1$ channel impulse response vector.

Linear discrete inverse problems have been treated extensively in Chapters 2 and 3. Depending on the length of the training sequence relative to the channel spread, the problem is either posed as an underdetermined linear discrete inverse problem or an overdetermined linear discrete inverse problem as stated in Section 5.4. If the training sequence is sufficiently long hence the system is overdetermined, the solution that minimize the L_2 norm of the error such that

$$h_{LS} = \min_h \|y - Ah\|_2 \tag{5.3}$$

is given by

$$h_{LS} = (A^T A)^{-1} A^T y. \tag{5.4}$$

h_{LS} is called the least squares channel estimation. If the training sequence is not sufficiently long hence the system is underdetermined, the solution with minimum L_2 norm such that

$$h_{LN} = \min_h \|h\|_2 \quad \text{subject to} \quad y = Ah \tag{5.5}$$

is given by

$$h_{LN} = \lim_{\lambda \to 0} (A^T A + \lambda I)^{-1} A^T y = A^T (AA^T)^{-1} y. \tag{5.6}$$

h_{LN} is called the least norm solution.

Although both least squares and least norm solutions have different targets, they have quite similar formulations. Least norm solution in the limiting case when λ goes to zero become equivalent to the least squares formulation. In other words, least norm solution is in fact equivalent to the Tikhonov regularized least squares solution when the regularization parameter λ goes to zero.

Substituting Equation 5.2 into the the least squares solution given by Equation 5.4 produces

$$h_{LS} = (A^T A)^{-1} A^T (Ah + \eta) \tag{5.7}$$

$$h_{LS} = h + (A^T A)^{-1} A^T \eta. \tag{5.8}$$

Substituting Equation 5.2 into the the least norm solution given by Equation 5.6 produces

$$h_{LN} = A^T (AA^T)^{-1} (Ah + \eta) \tag{5.9}$$

$$h_{LN} = A^T (AA^T)^{-1} Ah + A^T (AA^T)^{-1} \eta. \tag{5.10}$$

From matrix inversion lemma,[1]

$$h_{LN} = \lim_{\lambda \to 0} (A^T A + \lambda I)^{-1} A^T Ah + \lim_{\lambda \to 0} (A^T A + \lambda I)^{-1} A^T \eta \tag{5.11}$$

$$h_{LN} = \alpha h + \beta \eta \tag{5.12}$$

The least squares (LS) channel estimate in Equation 5.8 has the actual channel impulse response vector h with the colored noise term $(A^T A)^{-1} A^T \eta$. The meaning of this is that, apart from the noise, there is no interference to the estimated channel. In the absence of noise, the least squares estimate converges to the actual channel. If there is a sufficiently long training sequence at least two times the channel spread, least squares gives quite a robust estimate of the channel but with some penalty to the channel efficiency.

The least norm (LN) channel estimate in Equation 5.11 and 5.12 which uses a shorter training sequence does not perform as good as the least squares solution but offers a better channel usage. The poor performance is indicated in Equation 5.12 such that it does not have an actual channel vector h standing alone and hence it does not converge to the actual channel even in the absence of noise. This is due to the fact that the matrix $\alpha = \lim_{\lambda \to 0} (A^T A + \lambda I)^{-1} A^T A$ that multiply the channel vector h is not an identity matrix because A is underdetermined. This causes an effect called inter-tap interference in the channel estimate. That is, one tap in the channel may affect another tap degrading

[1] $(A^T A + \lambda I)^{-1} A^T = A^T (AA^T + \lambda I)^{-1}$

the estimation performance. Performance degradation increases when the matrix A gets more underdetermined as the training sequence gets shorter.

Least squares and least norm estimations need to be followed by a thresholding stage to be effectively used in practice. There exists different thresholding techniques that can be used. Direct thresholding, constant false alarm rate (CFAR), or protection window based thresholding can be used.

There exists other variants of least squares estimation: One is the thresholded least squares (ThLS) which uses the thresholded output of the least squares as the channel estimate. The other is the sparse least squares (SpLS) which reevaluates the coefficients obtained from the ThLS using another smaller sized least squares.

5.4.2. Maximum Likelihood (ML) Channel Estimation

Maximum likelihood (ML) channel estimation finds the channel impulse response which maximizes the probability of received samples for a given training sequence. ML channel estimation is in fact a generalization of least squares estimation. Apart from LS estimation, ML estimation needs and incorporates the noise information, namely the noise covariance matrix into the solution to obtain a better estimate. If the noise is white Gaussian with zero mean and hence the noise covariance matrix is a scaled identity matrix $\sigma^2 I$, the ML solution is equivalent to the LS solution.

For the overdetermined channel convolution model

$$y = Ah + \eta, \tag{5.13}$$

the ML estimate for the channel is given by

$$h_{ML} = \max_h [p(y|h)] = (A^H C^{-1} A)^{-1} (A^H C^{-1}) y, \tag{5.14}$$

where C is the covariance matrix of the noise η which is computed using the expected value of $\eta\eta^H$ given by

$$C = Cov(\eta) = E(\eta\eta^H). \tag{5.15}$$

It is obvious that if the noise is Gaussian with variance σ^2, the covariance matrix C is given by $C = \sigma^2 I$. Substituting this into the Equation 5.14, the ML estimate for the channel become equivalent to the LS estimate.

If the noise is not white Gaussian, ML performs better than LS provided that there is prior noise information, that is, noise covariance matrix is known. This may not be always possible in which case ML estimate can be iterated to estimate both the channel and noise.

Like LS, ML requires a sufficiently long training sequence, at least two times the channel span, for a robust estimation.

5.4.3. Correlation Based Channel Estimation

Correlation based channel estimation is a low complexity method to estimate a channel. It relies on convolving the training sequence with the received symbols. Since the training sequences are of pseudo-random nature they will have near identity auto-correlation matrices which enables the estimation of the channel. The performance of correlation based channel estimation depends on the length of the training sequence used. There are two cases to consider.

Long Training Sequence: This is the case when the length of the training sequence is at least two times the channel span. In this case, the convolution matrix A consists only of the training sequence symbols such that

$$y = Ah + \eta. \tag{5.16}$$

Multiplying both sides with A^T produces

$$A^T y = A^T Ah + A^T \eta. \tag{5.17}$$

Since

$$A^T A \approx NI, \tag{5.18}$$

correlation estimate is given by

$$h_{corr} = \frac{1}{N} A^T y. \tag{5.19}$$

Substituting Equation 5.16 into 5.19 produces

$$h_{corr} = \frac{1}{N} A^T (Ah + \eta) \tag{5.20}$$

$$h_{corr} = \frac{1}{N} A^T Ah + \frac{1}{N} A^T \eta \tag{5.21}$$

$$h_{corr} = \frac{1}{N} A^T Ah + \epsilon. \tag{5.22}$$

Equation 5.22 indicates that the correlation estimate does not converge to the actual channel in the absence of noise and if $\frac{1}{N}(A^TA)$ is not an identity matrix, which will be the case for most of the time if the training sequence is of finite length. Non-identity matrix $\frac{1}{N}(A^TA)$ that multiplies the actual channel h will cause inter-tap interference. For pseudo-random training sequences, it can be shown that as the length of the training sequence increases, the correlation matrix $\frac{1}{N}(A^TA)$ gets closer to the identity matrix, but this penalizes the channel efficiency.

Short Training Sequence: This is the case when the length of the training sequence is shorter than two times the channel length. Assuming no guard intervals, in this case the convolution matrix A does not wholly consist of the training sequence symbols and includes unknown real data before and after the training sequence. Inclusion of the unknown data degrades the performance of the correlation. The system can be modeled as

$$y = Ah + Dh + \eta. \tag{5.23}$$

The term D is the unknown data before and after the training sequence. Multiplying both sides of the equation with A^T produces

$$A^Ty = A^TAh + A^TDh + A^T\eta \tag{5.24}$$

and assuming

$$A^TA \approx NI, \tag{5.25}$$

the correlation estimate is,

$$h_{corr} = \frac{1}{N}A^Ty \tag{5.26}$$

Substituting Equation 5.23 into 5.26,

$$h_{corr} = \frac{1}{N}A^T(Ah + Dh + \eta) \tag{5.27}$$

$$h_{corr} = \frac{1}{N}A^TAh + \left[\frac{1}{N}A^TDh + \frac{1}{N}A^T\eta\right] \tag{5.28}$$

$$h_{corr} = \frac{1}{N}A^TAh + \epsilon, \tag{5.29}$$

where

$$\epsilon = \underbrace{\left[\frac{1}{N}A^T Dh\right]}_{\text{data interference}} + \underbrace{\left[\frac{1}{N}A^T \eta\right]}_{\text{colored noise}} \tag{5.30}$$

Comparing Equation 5.28 with 5.21, the error ϵ in Equation 5.28 and given in Equation 5.30 includes unknown data interference and the colored noise. Whereas in long training sequence case, the only source of error is colored noise as shown in Equation 5.21.

Another point which may not be obvious from Equation 5.29 is that the matrix $\frac{1}{N}(A^T A)$ is not as close to the identity as in the long training sequence case.

It can be concluded that the long training sequence correlation estimate has a good performance with a penalty to the channel utilization. The performance of the short training sequence correlation estimate gets poorer as the length of the training sequence decreases. And its output is expected to be highly noisy due to the interference from unknown data symbols.

5.4.4. Blended Least Squares Channel Estimation

Blended least squares channel estimation method proposed by Özen, Zoltowski and Fimoff (2002) aims to combine the correlation and least squares based methods to produce a better channel estimate. BLS uses the correlation method and thresholding for an initial estimate of the possible tap positions. CFAR or protection window based thresholding can be applied after correlation. As shown in Figure 5.1, in a real channel estimation context, actual tap positions need not be at the sampling instants. Using this fact, BLS represents each tap as a weighted sum of three half-sample spaced pulse matched filter (PMF) impulse responses to the right and left of the sampling instant. Then using the least squares, BLS estimates the three weights for each channel tap and therefore each tap is represented by a weighted sum of three half-sample spaced PMF impulse responses. Using this three weights for each channel tap, BLS recovers a finer resolution channel estimate including low amplitude pulse tails which otherwise would be buried under noise and lost.

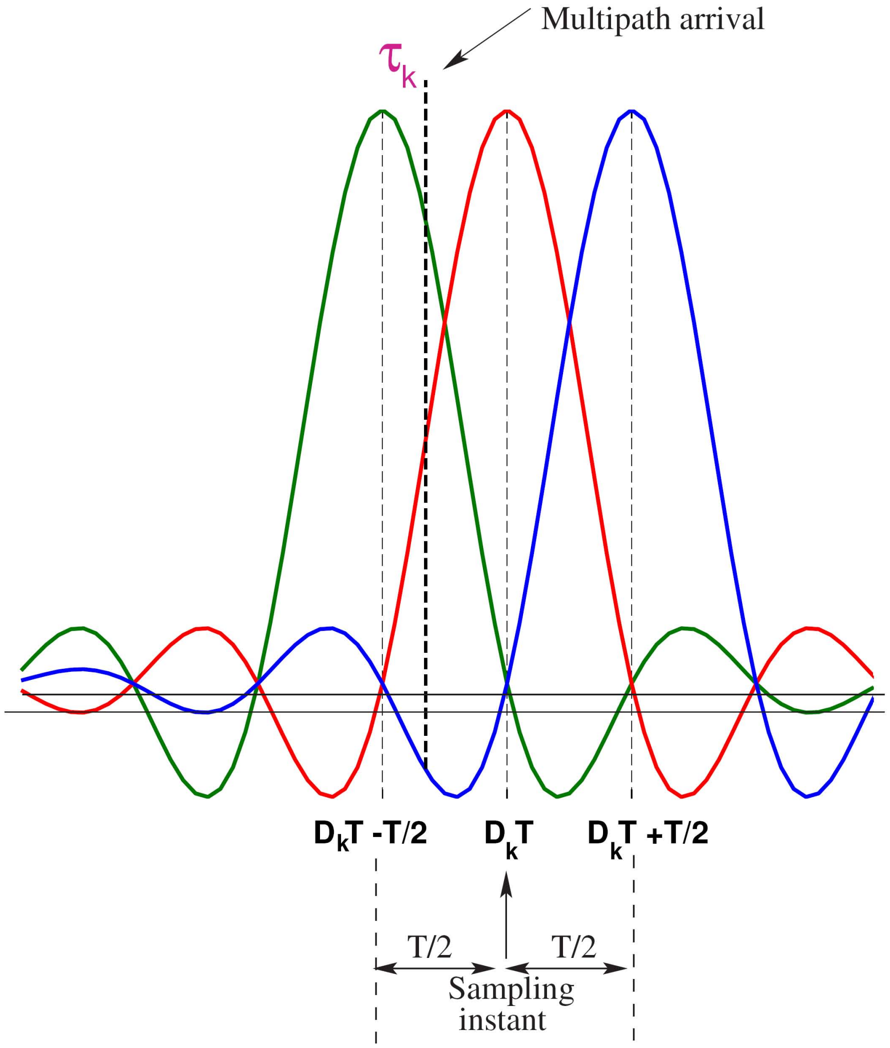


Figure 5.1. BLS algorithm: Each channel tap is represented by the weighted sum of 3 half-sample spaced PMF impulse responses to the right and left of the actual time of arrival. (Source: Özen (2003))

5.4.5. Linear Deconvolution Based Channel Estimation

Direct linear deconvolution, in other words, zero-forcing deconvolution or inverse filtering is generally disregarded as an effective channel estimation method due to its ill-posed nature. Its regularized form, Wiener deconvolution or Wiener filtering alleviates this problem by incorporating the noise variance into the solution.

It is known that the signal received by a receiver is the convolution of the transmitted signal with the channel impulse response, therefore it is straightforward to think that when a training sequence is transmitted, linear deconvolution of the received signal with that training sequence should readily yield the channel impulse response. Assuming no guard intervals around the training sequence, the received signal during training sequence transmission is the convolution of the channel impulse response $h(n)$ with the training sequence $q(n)$ and the surrounding data $d(n)$ plus the random noise $\eta(n)$. This can be described in discrete time domain by the convolution equations

$$y(n) = (q(n) + d(n)) * h(n) + \eta(n) \tag{5.31}$$

$$y(n) = q(n) * h(n) + d(n) * h(n) + \eta(n) \tag{5.32}$$

and in summation form

$$y(n) = \sum_{k=-\infty}^{\infty} (q(k) + d(k))h(n-k) + \eta(n). \tag{5.33}$$

The deconvolution in time domain is not quite obvious from Equation 5.33. However, more conveniently, convolution in time domain is equivalent to multiplication in Fourier domain such that

$$Y(\omega) = H(\omega)(Q(\omega) + D(\omega)) + N(\omega) \tag{5.34}$$

$$Y(\omega) = H(\omega)Q(\omega) + H(\omega)D(\omega) + N(\omega), \tag{5.35}$$

where $Y(\omega)$, $H(\omega)$, $Q(\omega)$, $D(\omega)$ and $N(\omega)$ are the Fourier transforms of the received signal $y(n)$, the channel impulse response $h(n)$, the training sequence $q(n)$, the surrounding data $d(n)$ and the noise term $\eta(n)$ respectively. The deconvolution in Fourier domain for the channel estimate H_{est} is

$$H_{est}(\omega) = \frac{Y(\omega)}{Q(\omega)} \tag{5.36}$$

Substituting Equation 5.35 into Equation 5.36 produces

$$H_{est}(\omega) = \frac{H(\omega)Q(\omega) + H(\omega)D(\omega) + N(\omega)}{Q(\omega)} \tag{5.37}$$

$$H_{est}(\omega) = H(\omega) + \frac{H(\omega)D(\omega) + N(\omega)}{Q(\omega)}, \tag{5.38}$$

where the channel error term for deconvolution is

$$\epsilon_{DEC} = \frac{H(\omega)D(\omega) + N(\omega)}{Q(\omega)}. \tag{5.39}$$

The channel estimate in Equation 5.38 consists of an actual channel term $H(\omega)$, the interference from noise $N(\omega)$ and data symbols $D(\omega)$ coming before and after the training sequence. This means, even in the absence of noise, the channel estimate will not converge to the actual channel due to the unknown data interference sometimes referred to as self-noise.

Random noise $N(\omega)$ is not generally the prime factor in deconvolution and correlation based channel estimations unless the SNR is very low. Instead, the interference from unknown data symbols coming immediately before and after the training sequence has more significance on the estimate. And this interference along with the random noise $N(\omega)$ will be amplified if $Q(\omega)$, the denominator in Equation 5.38, has zeros or near zeros. The solution as in all ill-posed inverse problems is regularization. Wiener deconvolution (filter) is the regularized form of the direct linear deconvolution (inverse filter). Both filters are defined by the equations

$$\text{Inverse filter:} \qquad H_{est}(\omega) = \frac{Y(\omega)}{Q(\omega)} \tag{5.40}$$

$$\text{Wiener filter:} \qquad H_{est}(\omega) = \frac{Q^*(\omega)Y(\omega)}{Q^*(\omega)Q(\omega) + \sigma^2}\ , \tag{5.41}$$

where the regularization parameter is given by $\sigma^2 = \sigma_n^2/\sigma_a^2$. σ_a^2 is the variance of the training sequence signal and σ_n^2 is the variance of the noise plus data interference. It is important to note that even in the absence of random noise, regularization is still necessary since data interference acts as a noise in the deconvolution operation. Data interference is thus sometimes called self-noise.

5.4.6. Permuted Deconvolution (PDEC) Based Channel Estimation

Introduction:

This subsection presents a new deconvolution method called permuted deconvolution (PDEC) which is particularly suited to estimate long delay sparse channels using relatively short training sequences. It is known that data preceding and following training sequences interferes with channel estimation by producing an unwanted component in the estimation output. This can be avoided by using guard intervals which penalize channel utilization. PDEC makes use of the subsequential diversity in training sequences to reduce that data interference component lowering the required threshold level and therefore offering a better estimation in short training sequence and long delay channel scenarios.

Estimation of channel impulse response (CIR) is necessary for receivers to effectively equalize channel and detect correct symbols. Training sequences known both by the receiver and transmitter are periodically sent by the transmitter to identify the CIR. For multi-path channels the samples received by the receiver are the convolution of the CIR with the transmitted symbols. Therefore, deconvolution of the training sequence from the received samples will produce the CIR. There are different techniques for deconvolution depending on channel delay and training sequence length. If the training sequence is sufficiently longer than the channel delay, least squares (LS) or maximum likelihood (ML) solution is possible. If the training sequence length is short and close to the channel delay, LS or ML solutions are not feasible. In that case, a sparse recovery technique can be used provided that training sequence length is above the channel delay. Correlation and Wiener deconvolution are lower complexity alternatives. Diversity techniques such as multi-antenna setup can also be used to increase the estimation performance. Sparse recovery techniques which assume that the channel is very sparse are iterative with high complexities. Correlation is a fast and low complexity estimation technique but suffers from large inter-tap and unknown data interference in the estimation output. Wiener deconvolution uses regularization which prevent excessive interference due to low spectral components in the training sequence but still suffer from significant unwanted interference component in its output which require high threshold values for estimation. PDEC uses Wiener deconvolution together with the subsequential diversity in training sequences to reduce the unwanted interference component in the estimation output reducing the required threshold for a better estimate. PDEC which can be implemented by a linear transversal filter is not iterative and does not assume that the channel is very sparse.

Motivation: Subsequential Diversity

Channel estimation using short training sequence suffer from interference from data immediately preceding and following the training sequence if no guard intervals are used. The motivation behind the PDEC method is that a training sequence have multiple subsequences each of which can be used as a different training sequence to produce subsequential diversity. Estimates from each subsequences can be superimposed and averaged to produce a better estimate. In each estimate, real channel coefficients will be in phase and will add up. However the same is not valid for the unwanted data interferences which are not necessarily in phase for different training sequences. Summing and averaging the interference components will reduce the cumulative interference level in the estimation output allowing to use a lower threshold for channel tap estimation.

Description of PDEC:

PDEC uses Wiener deconvolution. Let $h(n)$ be the CIR, $q(n)$ be the training sequence, $y(n)$ be the received samples and $d(n)$ be the data preceding and following the training sequence and let $H(\omega)$, $Q(\omega)$, $Y(\omega)$ and $D(\omega)$ be their respective Fourier transforms. Since the received samples are the convolution of the channel impulse response with the transmitted symbols, the channel convolution equation in Fourier domain is given by

$$Y(\omega) = H(\omega)\left[Q(\omega) + D(\omega)\right]. \tag{5.42}$$

In order to estimate the channel $H(\omega)$, $Y(\omega)$ need to be deconvolved with $Q(\omega)$. Wiener deconvolution of $Y(\omega)$ with $Q(\omega)$ will produce two components as shown in Equation 5.43 below.

$$\underbrace{\frac{Q(\omega)^{*}Y(\omega)}{Q(\omega)^{*}Q(\omega)+\lambda}}_{deconvolution} = \underbrace{\frac{Q(\omega)^{*}Q(\omega)H(\omega)}{Q(\omega)^{*}Q(\omega)+\lambda}}_{channel\ component} + \underbrace{\frac{Q(\omega)^{*}H(\omega)D(\omega)}{Q(\omega)^{*}Q(\omega)+\lambda}}_{interference\ component}, \tag{5.43}$$

where λ is the positive regularization parameter. As shown in Equation 5.43, Wiener deconvolution has two components. One of them is the channel component with a slight perturbation caused by small λ. The other is the interference component caused by the convolution of the unknown data and the channel.

Let $Q_i(n)$ represent the subsequences of the parent training sequence $Q(n)$. Having the same length with $Q(n)$, $Q_i(n)$ are produced by zeroing a certain fraction of elements in $Q(n)$ permuted for each i. Using Equation 5.43 for each i, summing over i and averaging produces the equation

$$\frac{1}{N}\sum_{i=1}^{N}\underbrace{\frac{Q_i(\omega)^*Y(\omega)}{Q_i(\omega)^*Q_i(\omega)+\lambda}}_{deconvolution} = \frac{1}{N}\sum_{i=1}^{N}\underbrace{\frac{Q_i(\omega)^*Q_i(\omega)H(\omega)}{Q_i(\omega)^*Q_i(\omega)+\lambda}}_{channel\ component} + \frac{1}{N}\sum_{i=1}^{N}\underbrace{\frac{Q_i(\omega)^*H(\omega)D_i(\omega)}{Q_i(\omega)^*Q_i(\omega)+\lambda}}_{interference\ component}. \tag{5.44}$$

D_i contains the unknown data D and the elements of Q not included in Q_i. Since for different i Q_i and D_i will vary, the interference component summation in Equation 5.44 will not be a coherent sum and after division by N is expected to be smaller than the interference component in Equation 5.43. The channel components in Equation 5.44 which involve the same channel except for a slight perturbation caused by the small positive number λ is highly coherent because for small positive λ

$$\frac{Q_i(\omega)^*Q_i(\omega)}{Q_i(\omega)^*Q_i(\omega)+\lambda}\approx 1 \quad \text{and} \quad \frac{Q_i(\omega)^*Q_i(\omega)H(\omega)}{Q_i(\omega)^*Q_i(\omega)+\lambda}\approx H(\omega). \tag{5.45}$$

Therefore, Equation 5.44 becomes

$$\frac{1}{N}\sum_{i=1}^{N}\frac{Q_i(\omega)^*Y(\omega)}{Q_i(\omega)^*Q_i(\omega)+\lambda} = H+\Psi(\omega) \tag{5.46}$$

$$\left[\frac{1}{N}\sum_{i=1}^{N}\frac{Q_i(\omega)^*}{Q_i(\omega)^*Q_i(\omega)+\lambda}\right]Y(\omega) = H+\Psi(\omega) \tag{5.47}$$

$$PD(\omega)Y(\omega) = H+\Psi(\omega). \tag{5.48}$$

The error term $\Psi(\omega)$ contains the interference component in Equation 5.44 plus the regularization error in the channel component due to λ. The term in brackets in Equation 5.47 is the PDEC estimator (PD). PD can be precomputed and stored offline for a given training sequence using a very large N. $pd(n)$, the inverse Fourier transform of $PD(\omega)$, can be used as a transversal channel estimation filter which convolves $pd(n)$ with the received samples $y(n)$ to give an estimate of the channel $h(n)$ with error $\Psi(n)$.

Complexity and Implementation:

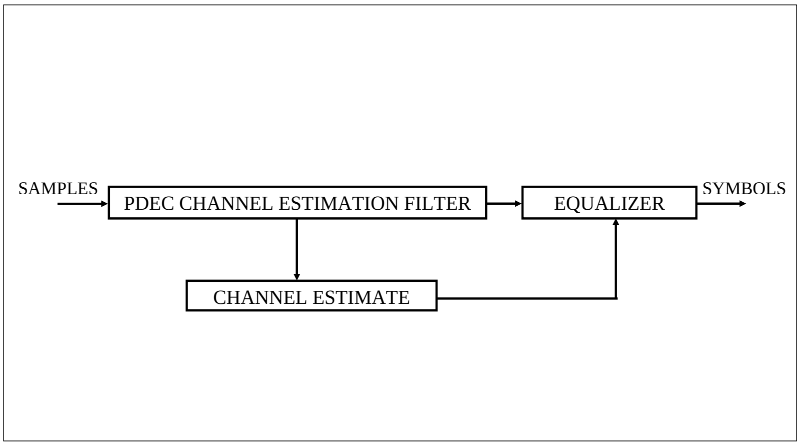


Figure 5.2. PDEC Channel Estimation Filter: PDEC filter is a time domain transversal filter.

The complexity of the PDEC filter is that of a transversal convolutional filter since its coefficients need be computed only once. Therefore it is a low complexity one dimensional symbol by symbol filter. Typical implementation is shown in Figure 5.2.

Noise:

The MMSE criteria sets the regularization parameter λ to be equal to σ_n^2/σ_x^2, where σ_n^2 is the noise variance and σ_x^2 is the signal variance. Empirical tests of PDEC show that a wide range of values for λ produces acceptable results due to the diversity in PDEC. Since all $Q_i(\omega)$s have zeros or near zeros at different positions, PDEC is more immune to noise and interference than the Wiener filter described in Equation 5.43 which use the parent training sequence $q(n)$ only.

5.4.7. Comparison of Sparse Channel Estimation Methods

Comparison of different training sequence based channel estimation methods are shown in Figure 5.3 at SNR = 20 dB, in Figure 5.4 at SNR = 10 dB, in Figure 5.5 at SNR = 5 dB and in Figure 5.6 at SNR = 0 dB. A channel of length 576 and a training sequence of length 704 is used for testing. Four methods, PDEC, Wiener deconvolution, correlation and regularized least norm are tested at 20 dB, 10 dB, 5 dB and 0 dB SNRs.

Each figure consists of 5 plots. Plot (a) is the original channel taps. Plot (b) is the PDEC test. Plot (c) is the Wiener deconvolution test. Plot (d) is the correlation test. Plot (e) is the regularized least norm test.

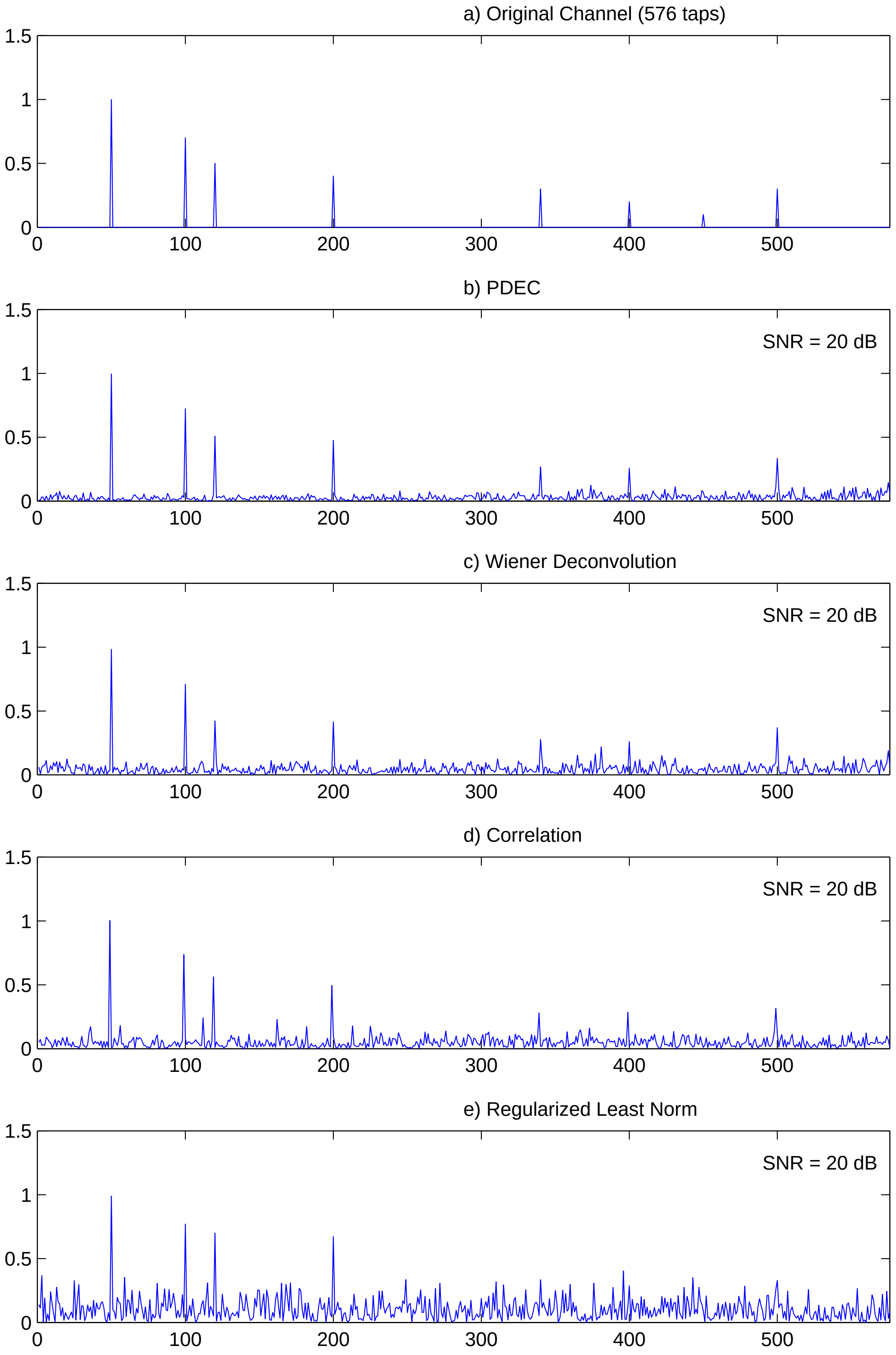


Figure 5.3. Comparison of different training sequence based channel estimation methods at SNR = 20 dB. a) Original channel taps. b) PDEC c) Wiener deconvolution d) Correlation e) Regularized least norm

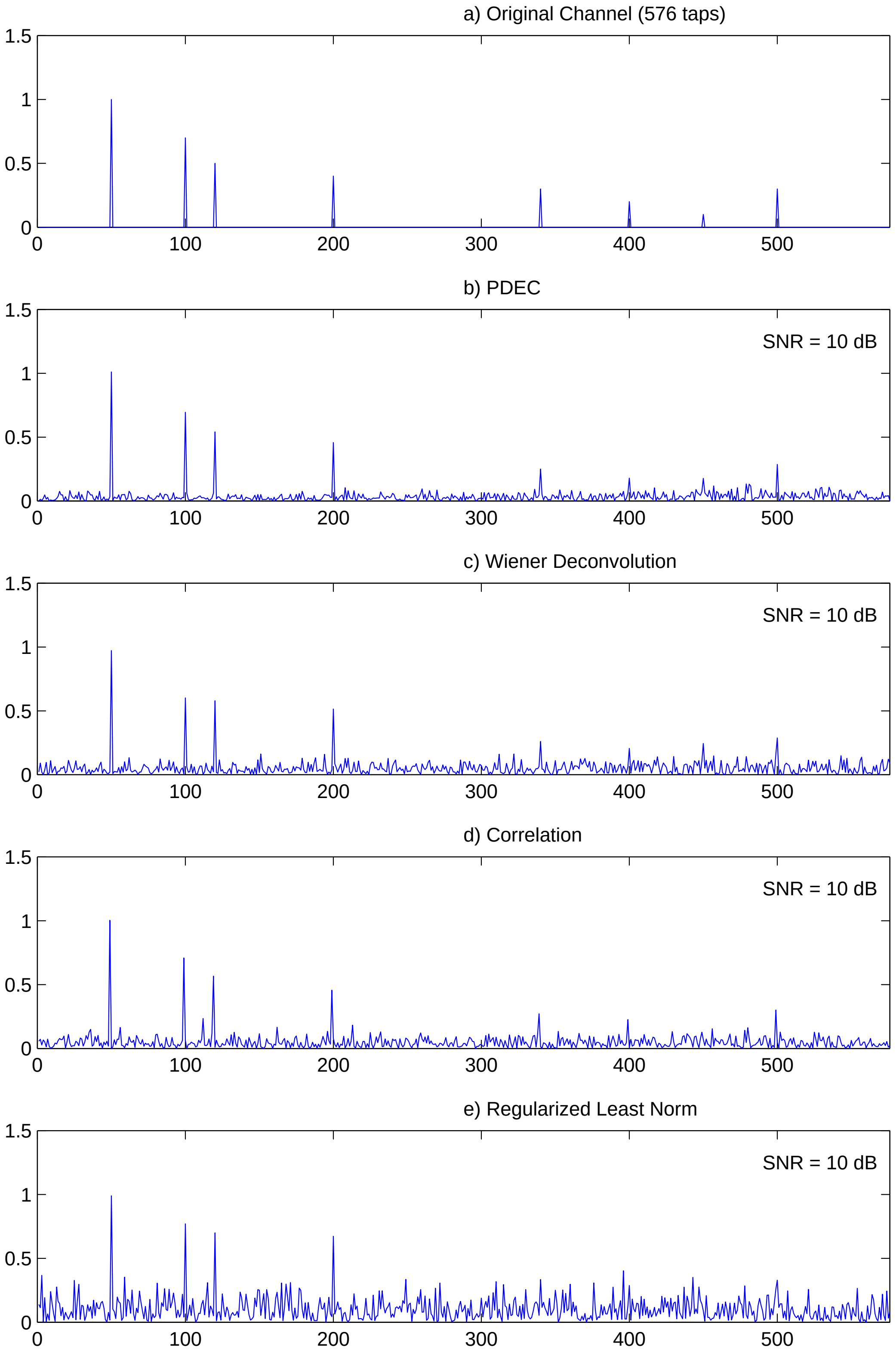


Figure 5.4. Comparison of different training sequence based channel estimation methods at SNR = 10 dB a) Original channel taps. b) PDEC c) Wiener deconvolution d) Correlation e) Regularized least norm

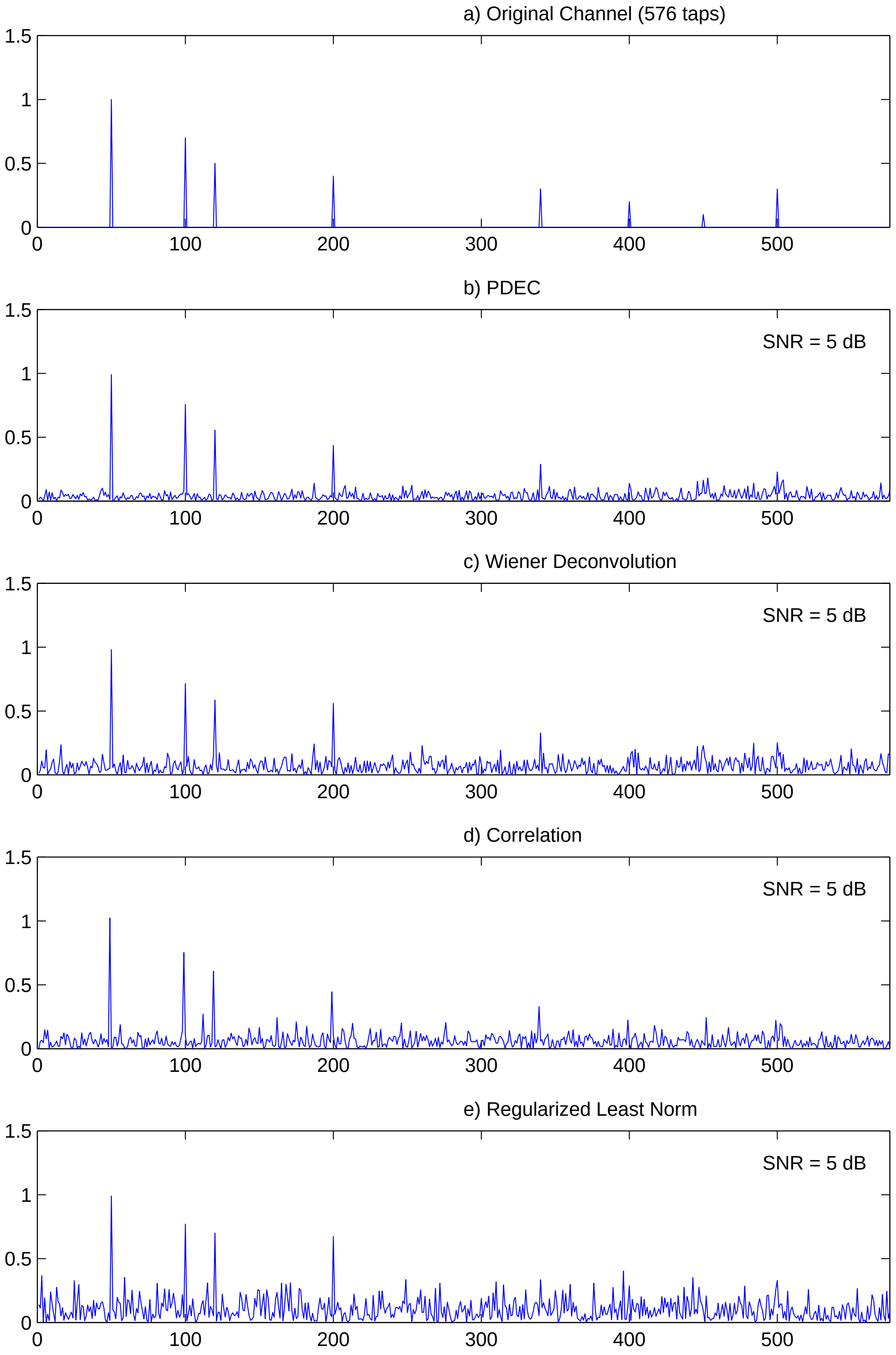


Figure 5.5. Comparison of different training sequence based channel estimation methods at SNR = 5 dB a) Original channel taps. b) PDEC c) Wiener deconvolution d) Correlation e) Regularized least norm

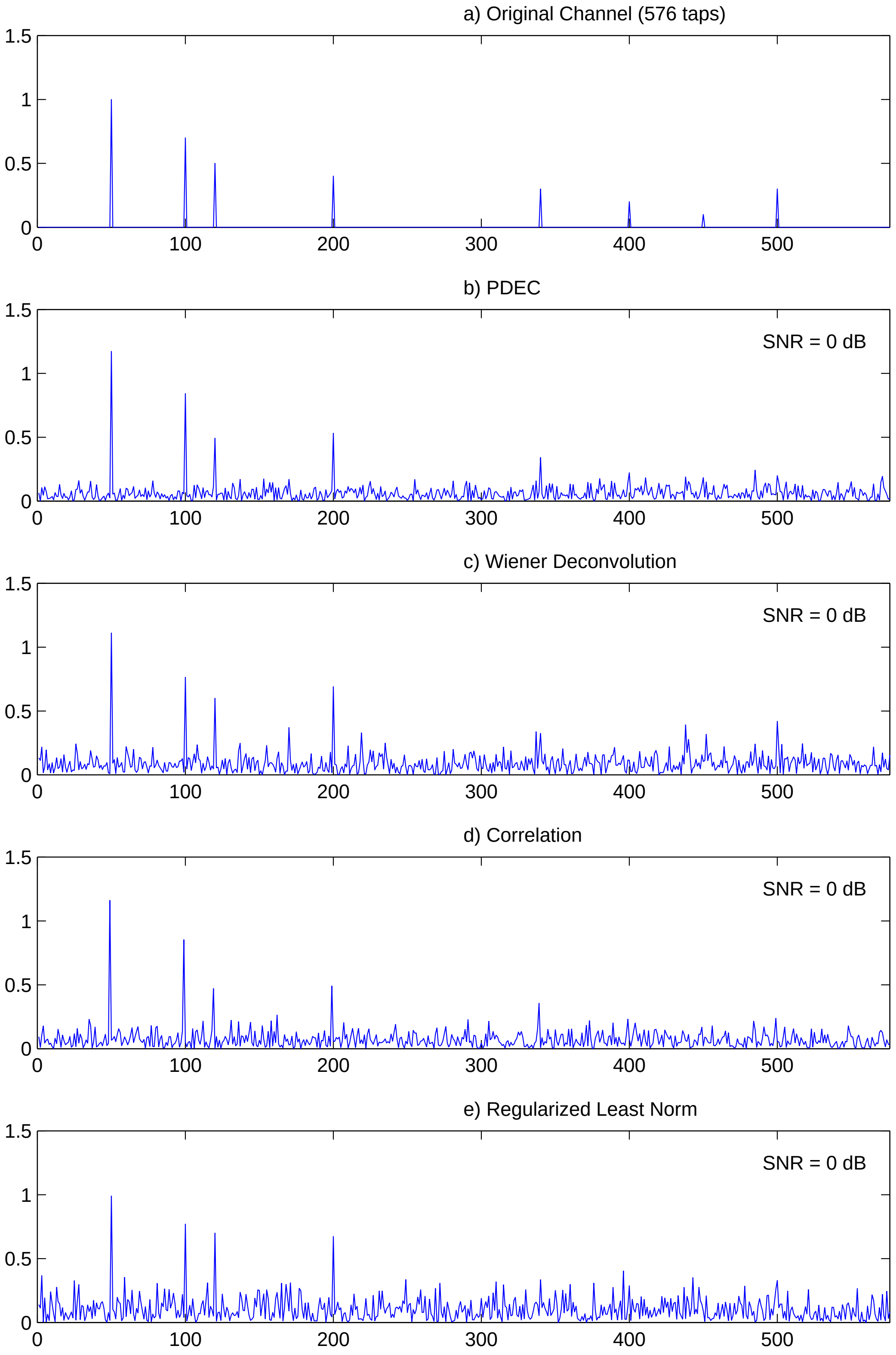


Figure 5.6. Comparison of different training sequence based channel estimation methods at SNR = 0 dB a) Original channel taps. b) PDEC c) Wiener deconvolution d) Correlation e) Regularized least norm

5.4.8. Test Results

PDEC performance is very prominent in comparison to correlation and Wiener deconvolution in Figure 5.3 at 20 dB SNR. The variance of interference in PDEC is around 4 dB less than that of the Wiener deconvolution and around 9 dB less than that of the correlation. Besides, correlation has considerable number of spurious taps.

At 10 dB SNR depicted in Figure 5.4, PDEC has an interference 3.85 dB less than that of Wiener deconvolution and 8.60 dB less than that of correlation. Wiener deconvolution has a lower interference than that of correlation.

At 5 dB SNR depicted in Figure 5.5, PDEC has an interference 3.92 dB less than that of Wiener deconvolution and 7.10 dB less than that of correlation. Wiener deconvolution has a lower interference than that of correlation.

At 0 dB SNR depicted in Figure 5.6, PDEC has an interference 3.69 dB less than that of Wiener deconvolution and 4.19 dB less than that of correlation. Wiener deconvolution has now nearly equal interference to that of correlation.

In all tests PDEC had a stable and superior performance over other methods. Regularized least norm is just given to show that it is not a preferable solution even at high SNR levels. PDEC is prominently superior at high SNR levels because interference overshadows random noise when SNR is high. While Wiener deconvolution is superior to correlation at high SNR, the gap closes as SNR goes to zero and the performance of both methods become very poor.

CHAPTER 6

SPARSE CHANNEL EQUALIZATION

6.1. Equalization

Equalization like channel estimation is an inverse problem. Equalization aims to equalize (flatten) the frequency response of the channel between the transmitter and receiver in order to make the overall channel appear like an ideal uni-path channel in which each individual sample received is the delayed and scaled version of a transmitted symbol. In other words, equalizer aims to remove inter symbol interference from received samples. The eventual purpose of channel estimation is to provide CIR to a channel estimation based equalizer.

The purpose of the overall channel estimation and equalization system is to decrease the symbol or bit error rate of the incoming digital stream down to the point that the remaining errors can be corrected by the error correction modules. Otherwise, the error rate of the incoming stream might be too high to be error corrected, particularly under noisy and weak signal conditions.

6.2. Equalizers

Equalizers are physical devices that perform equalization to remove ISI from received samples. There are different equalizer implementations. Some equalizers work with stored sequences and are called sequence estimators. They are better in terms of performance but are more computationally complex. Besides, they require larger storage. One well-known example is the maximum likelihood sequence estimator (MLSE) which is optimum if the noise is additive white Gaussian noise (AWGN). Other more practical equalizers are filters which perform symbol-by-symbol detection without a large storage and computation. They are called symbol-by-symbol equalizers (SSE). Some SSE equalizers may self-adapt to a channel using the detected symbols and a training sequence without any need for channel estimation. However, they are slow to adapt and therefore are not robust enough to be used in high symbol and compression rate transmissions. Channel estimation based SSE equalizers are faster and more robust since they do not need a self-adaptation period. SSE equalizers can be classified as:

- Linear equalizers
  - Zero forcing linear equalizers (ZF-LE).
  - Minimum mean square error linear equalizers (MMSE-LE).
- Decision feedback equalizers (DFE).

6.3. Linear Equalizers

There exist mainly two types of linear equalizers. They are the zero forcing linear equalizer (ZF-LE) and minimum mean square error linear equalizer (MMSE-LE).

6.3.1. Zero Forcing Linear Equalizer

Zero forcing linear equalizer (ZF-LE) introduced by Lucky (1965) is an inverse filter whose frequency response is the inverse of the channel frequency response. Assuming $H(\omega)$ is the Fourier transform of the channel $h(t)$, the frequency response of the ZF-LE is given by

$$F(\omega) = \frac{1}{H(\omega)}. \tag{6.1}$$

The received samples $y(t)$ are the convolution of the channel impulse response $h(t)$ with the transmitted symbols $s(t)$ plus noise $\eta(t)$, whose Fourier transforms are $Y(\omega)$, $H(\omega)$, $S(\omega)$, and $N(\omega)$ respectively. The convolution of the ZF-LE impulse response $f(t)$ with the received samples will therefore yield the equalizer output $x(t)$ such that in Fourier domain

$$X(\omega) = Y(\omega)F(\omega) \tag{6.2}$$

$$X(\omega) = [S(\omega)H(\omega) + N(\omega)]\,\frac{1}{H(\omega)} \tag{6.3}$$

$$X(\omega) = S(\omega) + \frac{N(\omega)}{H(\omega)}. \tag{6.4}$$

The inverse Fourier transform of $X(\omega)$ given by $x(t) = F^{-1}(X(\omega))$ is the equalizer output which is expected to be close to $s(t)$. It is obvious from Equation 6.4 that if the channel impulse response $h(t)$ has deep spectral nulls, the noise will be excessively amplified and equalization will fail. ZF-LE can only work if the CIR has no deep spectral nulls and the noise is very low. These limitations prohibit its use as a practical equalizer.

In the inverse problems context, inverse filtering and therefore zero forcing equalization is an ill-posed problem and needs to be regularized.

6.3.2. Minimum Mean Square Error Linear Equalizer

The regularized form of ZF-LE is called MMSE-LE (Lucky, Salz and Weldon 1968), which regularize the equalizer output by minimizing the mean square error.
MMSE-LE frequency response is given in frequency domain by

$$F(\omega) = \frac{H(\omega)^*}{H(\omega)^* H(\omega) + \lambda}. \tag{6.5}$$

Therefore, the MMSE-LE filter coefficients are

$$m_f = F^{-1}\left[F(\omega)\right], \tag{6.6}$$

where F^{-1} is the inverse Fourier transform.
The filter coefficients in time domain are

$$m_f = (HH^T + \lambda I)^{-1} H \delta_0, \tag{6.7}$$

where H and δ_0 is the channel convolution matrix and the cursor position respectively as described in Subsection 6.4.1. and Figure 6.4.

The regularization parameter λ prevents the filter frequency response from having deep spectral nulls. Noise amplification is limited and the filter output is regularized. The regularization parameter λ is dependent on the signal to noise ratio and is given by,

$$\lambda = \frac{NoisePower}{SignalPower} = \frac{\sigma_n^2}{\sigma_s^2}, \tag{6.8}$$

where σ_n^2 is the noise variance and σ_s^2 is the signal variance.

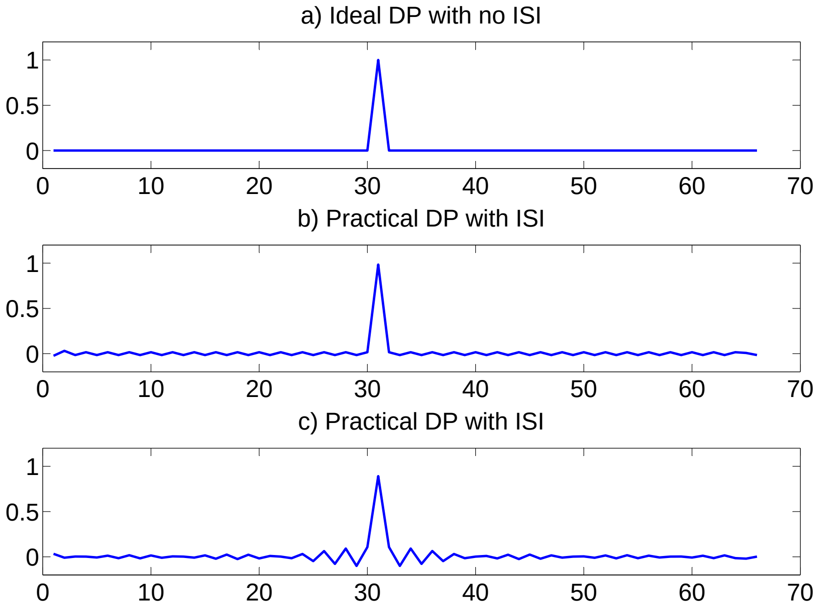


Figure 6.1. Detector pulse (DP) shapes. Tails to the left convolve with future symbols to produce future symbol ISI and tails to the right convolve with past symbols to produce past symbol ISI. a) Ideal unit pulse with zero tail and zero ISI. b) and c) Practical detector pulses with nonzero tails and nonzero ISI.

Detector Pulse: The convolution of the coefficients f of a linear equalizer with the channel impulse response h produces the detector pulse $DP = f * h$ and it should ideally give a unit impulse at the position of the symbol to be detected. In practice, the DP is not an exact unit impulse but a pulse with tails on both sides as shown in Figure 6.1 b and c. This is due to the finiteness of the filter length and regularization of the filter coefficients. Nonzero tails indicate the presence of ISI. The tails to the left of the main pulses cause the ISI from future symbols and tails to the right of the main pulses cause the ISI from the past symbols.

6.4. Decision Feedback Equalizers (DFE)

Decision feedback equalizer is a nonlinear equalizer first introduced by Austin (1967). DFE has a feedback filter which evaluates the interference caused by the previously detected symbols and subtracts it from the feedforward filter output. This way, DFE eliminates the ISI caused by the already detected symbols (assuming they are all correct) leaving only the ISI caused by future symbols. Therefore DFEs perform much better than the linear equalizers. Since MLSE is not practically feasible for long channels and large signal constellations, DFEs are the only plausible option for equalizing such channels. DFEs suffer from error bursts if erroneous symbols are fed back.

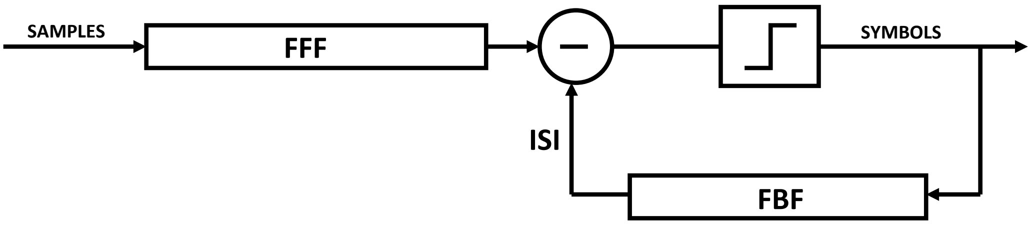


Figure 6.2. DFE Block Diagram. FFF: Feedforward filter. FBF: Feedback filter.

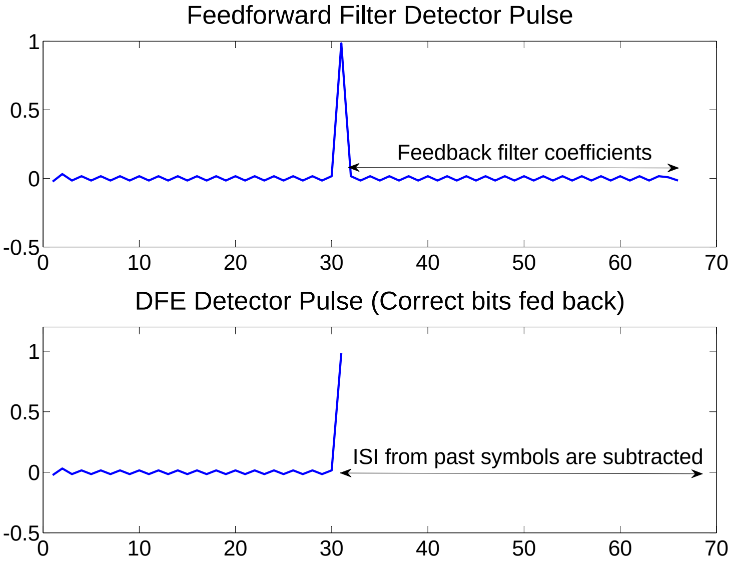


Figure 6.3. Detector pulse (DP) shapes of the DFE feedforward filter (top) and Overall DFE (bottom).

DFE consists of two filters as shown in Figure 6.2. Feedforward filter (FFF) is a regularized linear filter. Feedback filter (FBF) is also a linear filter. The coefficients of both filters are jointly optimized for a given criterion usually for MMSE. If the feedback filter is removed, FFF still works as a linear equalizer with a much lower performance.

One important consideration in DFE performance is the length of filters which depend on a couple of factors given as,

- Channel Length
- Hardware Constraint
- Target Performance

Using too long or too short filters may degrade the performance. Filter lengths should be comparable to the channel length.

Another important factor in DFE performance is the cursor position. Cursor position is the delay that FFF detects a symbol. Cursor position affects both the FBF length and detection performance. Cursor can be anywhere through the length of FFF but it should not be very close to the edges. Performance jumps may be observed if cursor position is even slightly changed particularly for short filter designs due to hardware limitations (Al-Dhahir 1996).

6.4.1. Inverse Problem Derivation of Minimum Mean Square Error DFE Filter Coefficients

Figure 6.4 describes the convolution of the transmitted symbols x with the channel impulse response $h = [h_0, h_1, h_2, ..., h_L]$ using a convolution matrix H. Figure 6.4 emphasizes that the transmitted symbol vector x can be partitioned into a past symbol vector x_p, and a future symbol vector x_f. This partitioning will help understand the operational mechanism of DFEs. DFEs use past decisions x_p to subtract the part of the ISI caused by them. Therefore, it will prove to be useful to partition the symbol vector into past and future symbol vectors in order to understand the DFE operation. The convolution equation described in Figure 6.4 is

$$y = H_f x_f + H_p x_p, \tag{6.9}$$

where

$$H = [H_f | H_p] \quad \text{and} \quad x = \left[\frac{x_f}{x_p}\right].$$

If the contribution $H_p x_p$ due to the symbols already detected is subtracted from y, the new

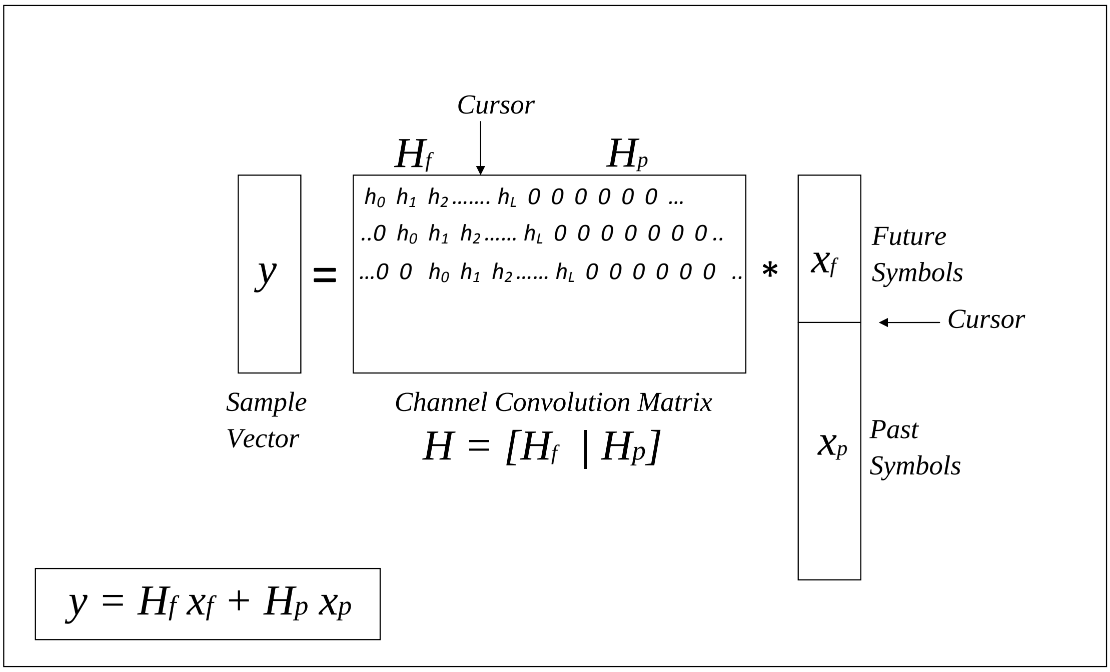

Cursor
Hf
Hp
y
=
h0 h1 h2……. hL 0 0 0 0 0 0 …
..0 h0 h1 h2…… hL 0 0 0 0 0 0 0 ..
…0 0 h0 h1 h2…… hL 0 0 0 0 0 0 ..
*
xf
Future Symbols
Cursor
xp
Past Symbols
Sample Vector
Channel Convolution Matrix
H = [Hf | Hp]
y = Hf xf + Hp xp

Figure 6.4. Convolution of transmitted symbols x with channel impulse response h using a channel convolution matrix H (noiseless).

equation is

$$y - H_p x_p = H_f x_f. \tag{6.10}$$

Using the pseudo inverse of the matrix of H_f, x_f is given by

$$x_f = H_f^T (H_f H_f^T)^{-1} (y - H_p x_p). \tag{6.11}$$

Since $H_f H_f^T$ may not be well posed, a regularization term λI is normally added. The regularized equation is

$$x_f = H_f^T (H_f H_f^T + \lambda I)^{-1} (y - H_p x_p). \tag{6.12}$$

Rewriting the above equation produces

$$x_f = H_f^T (H_f H_f^T + \lambda I)^{-1} y - H_f^T (H_f H_f^T + \lambda I)^{-1} H_p x_p. \tag{6.13}$$

The symbol x_0 to be detected is at the cursor position and is given by $x_0 = x_f^T \delta_0$, where $\delta_0 = [0, 0, .., 0, 1, 0, .., 0]^T$. $x_0 = x_f^T \delta_0$ is given by

$$x_0 = x_f^T \delta_0 = \underbrace{y^T (H_f H_f^T + \lambda I)^{-1} H_f \delta_0}_{Feedforward} - \underbrace{x_p^T H_p^T (H_f H_f^T + \lambda I)^{-1} H_f \delta_0}_{Feedback}. \tag{6.14}$$

Equation 6.14 reveals the feedforward and feedback parts of DFE.

Feedforward filter coefficients are:

$$m_f = (H_f H_f^T + \lambda I)^{-1} H_f \delta_0, \tag{6.15}$$

Feedback filter coefficients are:

$$\begin{aligned} m_b &= H_p^T (H_f H_f^T + \lambda I)^{-1} H_f \delta_0 & (6.16) \\ m_b &= H_p^T m_f & (6.17) \end{aligned}$$

It needs to be noted that from whatever process the feedforward coefficient m_f are obtained, they are used also to produce the feedback coefficients by convolving m_f with the channel impulse response to produce the detector pulse DP. The feedback filter coefficients are the values of the DP to the right of the main pulse as shown in Figure 6.3. The regularization parameter λ that minimizes the mean square error is,

$$\lambda = \frac{Noise\ Power}{Signal\ Power} = \frac{\sigma_n^2}{\sigma_x^2} \tag{6.18}$$

This standard derivation of DFE filter coefficients isolates the past and future symbols x_p and x_f and partitions the convolution matrix into two parts H_f and H_p and calculates the FFF coefficients using only H_f submatrix. This has the advantage that H_f is more deterministic compared to the underdetermined matrix H and therefore interference from future symbols will be greatly reduced. The main disadvantage of this derivation is that the FFF coefficients are derived on the assumption that all past decisions are correct. In this derivation, past symbols x_p and corresponding part of H, H_p is neglected. This has the consequence of larger FBF coefficients in turn for a lower ISI from future symbols which may increase the probability and the duration of burst errors at high noise levels. In low or no noise cases, DFE can achieve very low error rates which is not possible using linear equalizers (Proakis 1995).

6.5. Frequency Compensated Linear Equalizer (FC-LE)

6.5.1. Introduction

In this section, a new equalizer architecture called frequency compensated linear equalizer (FC-LE) is introduced. It is well known that linear equalizers (LE) suffer from noise amplification when the channel has deep fades namely very low spectral components. Figure 6.5 shows a typical fading channel (Proakis 1995) and its magnitude response. Linear equalizers can not achieve very low error rates at low noise levels or even in the absence of noise. Therefore linear equalizers are not a choice for high performance equalization. DFEs suffer from frequent burst errors particularly at high noise levels but they perform well in the low to no-noise region. FC-LE combines the simplicity of linear equalizer with the probability based decision capability of optimal detector MLSE.

6.5.2. Architecture and Operation

As shown in Figure 6.6, frequency compensated linear equalizer consists of three consecutive blocks, prefilter, linear equalizer, and a decision device. Prefilter filters received symbols before applying a linear equalization. If channel impulse response has a low pass characteristic, prefilter is a low pass moving sum filter or if the channel impulse response has a high pass characteristic, prefilter is a high pass moving difference filter. Linear equalizer receives either the sum or difference of two samples. This way, frequency components of the received and filtered symbols will have been lowered at positions the channel have fades and hence noise amplification will be somewhat reduced. Depending on the prefilter type, the output of the linear equalizer will now be groups of two symbols added or subtracted instead of a single symbol. A decision device at the output of the linear equalizer will determine the most likely symbol.

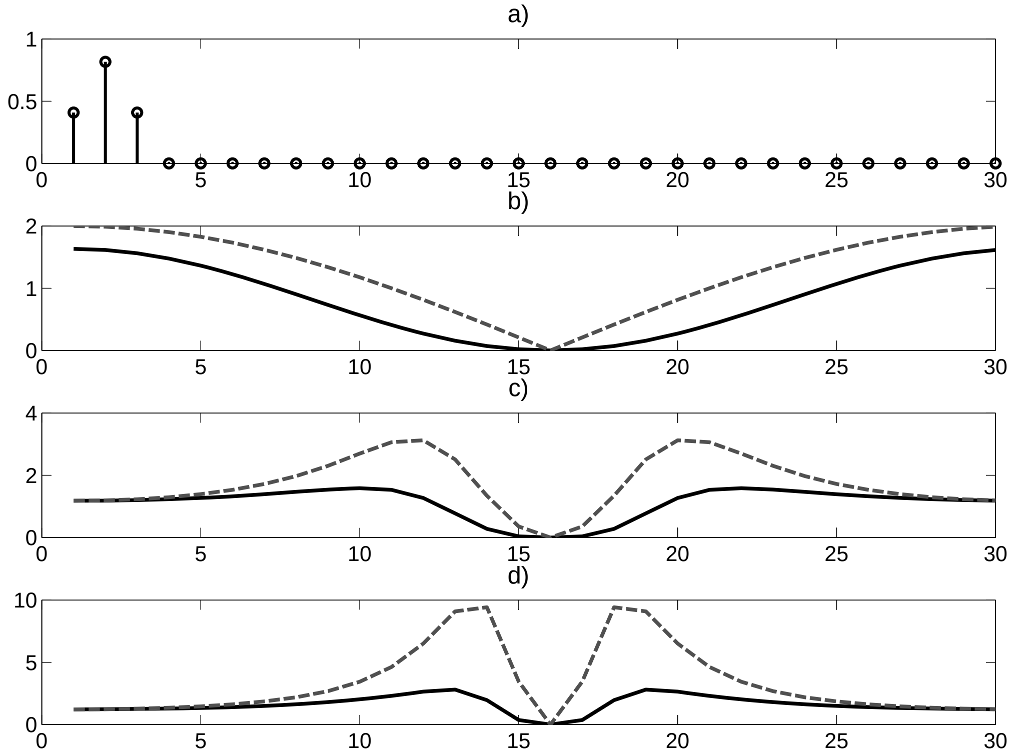


Figure 6.5. a) Coefficients of the channel in Figure 10.2.5(b) in Proakis (1995). b) magnitude responses of the channel (black) and the low-pass prefilter (gray-dashed). c) Combined MMSE-LE and prefilter magnitude response (black) and MMSE-LE magnitude response (gray-dahed) for SNR = 10 dB. d) Combined MMSE-LE and prefilter magnitude response (black) and MMSE-LE magnitude response (gray-dahed) for SNR = 20 dB.

6.5.3. Description of the FC-LE Blocks

Low Pass Prefilter: It is a simple moving sum filter. It convolves the input samples with the vector [1,1]. It sums two consecutive samples.

High Pass Prefilter: It is a simple moving difference filter. It convolves the input samples with the vector [-1,1]. It subtracts two consecutive samples.

Linear Equalizer: Minimum mean square error linear equalizer.

Low Pass Decision Device (LPDD): LPDD slices the output of the linear equalizer into three regions. Let $u[n]$ represent the linear equalizer output and $x[n]$ represent the symbols outputted by the decision device. Then the rules that produces the symbols are given as

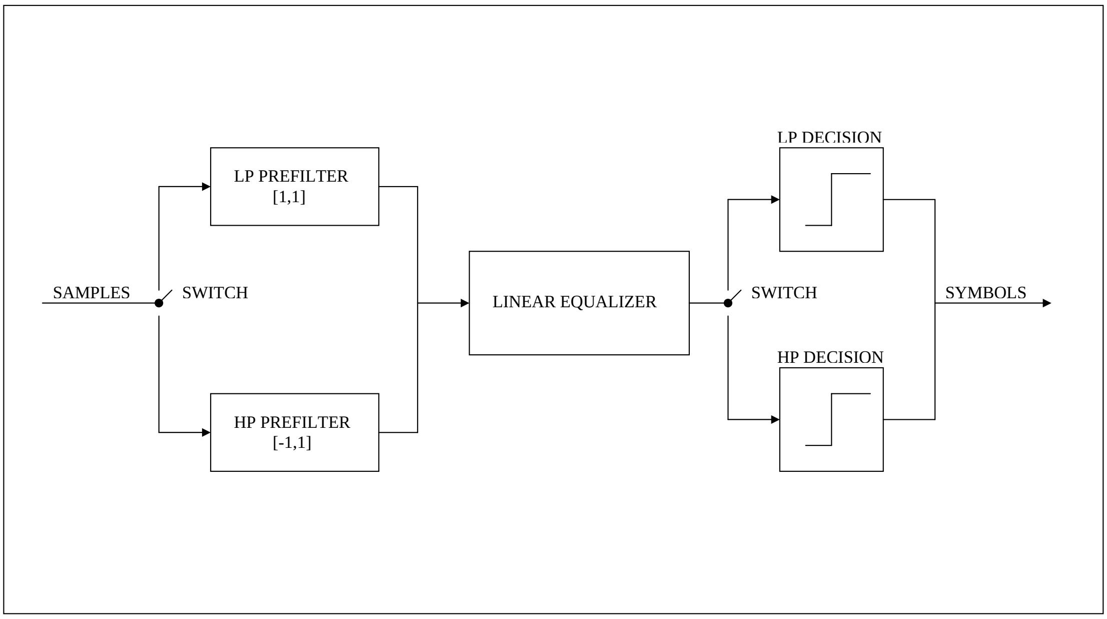


Figure 6.6. Blocks of a frequency compensated linear equalizer (FC-LE)

follows,

$$\begin{aligned} -1 \le u[n] \le 1 &\qquad x[n] = -x[n-1] \\ u[n] \ge 1 &\qquad x[n] = 1 \\ u[n] \le -1 &\qquad x[n] = -1 \end{aligned}$$

High Pass Decision Device (HPDD): Like LPDD, HPDD slices the output of the linear equalizer into three regions. Then the rules that produces the symbols are given as follows,

$$\begin{aligned} -1 \le u[n] \le 1 &\qquad x[n] = x[n-1] \\ u[n] \ge 1 &\qquad x[n] = 1 \\ u[n] \le -1 &\qquad x[n] = -1 \end{aligned}$$

The only difference between LPDD and HPDD is that when the equalizer output $u[n]$ is between minus one and one LPDD inverts the previous symbol whereas HPDD outputs the previous symbol as the current symbol.

6.5.4. Embedded Prefilter FC-LE Architecture

As shown in Figure 6.7, an alternative architecture for FC-LE is the one which combines each prefilter with linear equalizer to produce two linear equalizers each for low and high pass channels. Decision devices are as described in Subsection 6.5.3. The advantage of the embedded prefilter architecture is that the number of switches are reduced to one and prefilter blocks are eliminated.

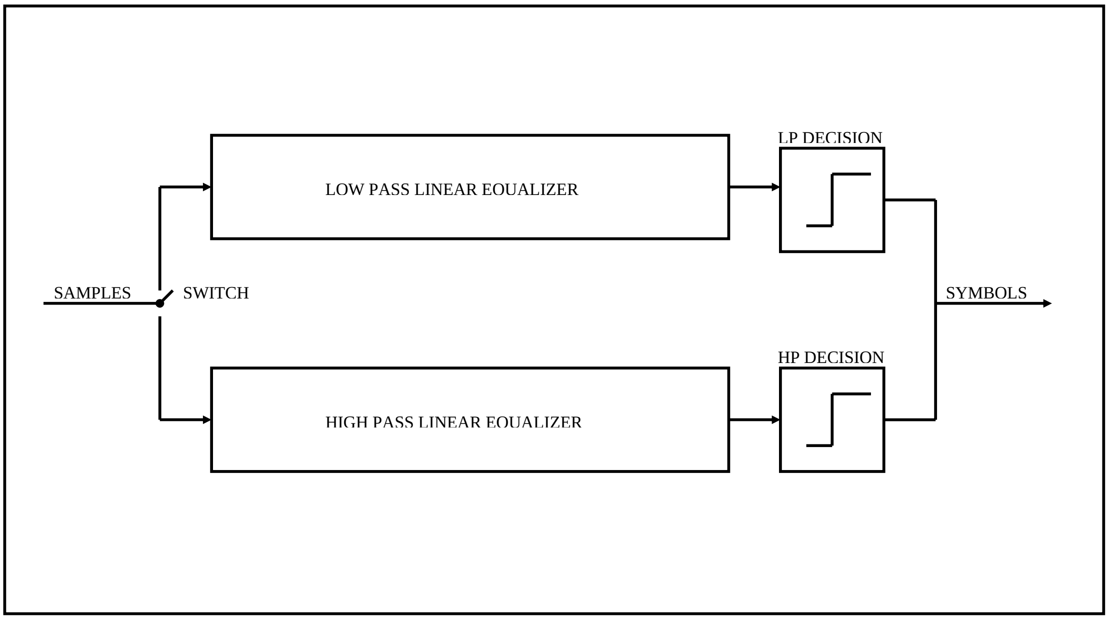


Figure 6.7. Embedded prefilter architecture for the frequency compensated linear equalizer (FC-LE).

6.5.5. FC-LE Linear Equalizer Coefficients and Detector Pulses

The equalizer coefficients in the non embedded FC-LE are derived according to the MMSE criteria as described in Subsection 6.3.2. In the embedded prefilter architecture, MMSE-LE coefficients are convolved with the vector [1,1] for the low pass linear equalizer and with the vector [-1,1] for the high pass linear equalizer.

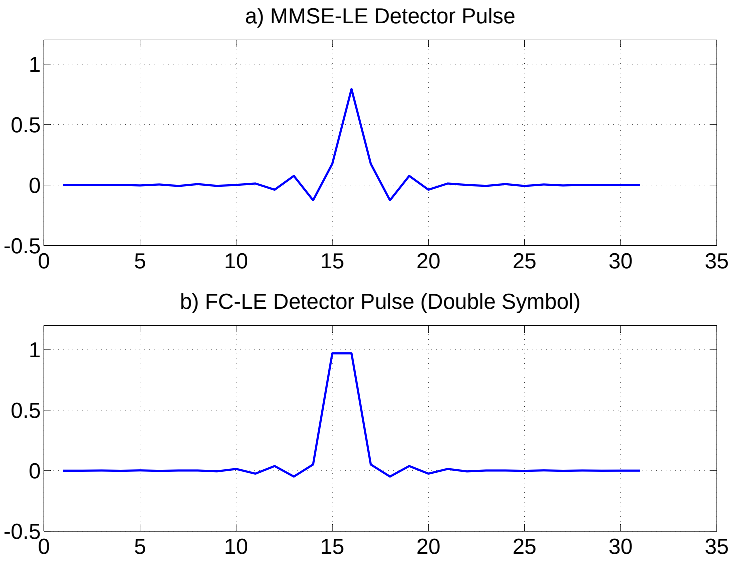


Figure 6.8. Detector pulse (DP) shapes of the MMSE-LE filter (top) and FC-LE filter (bottom) for the channel of Figure 10.2.5(b) in Proakis (1995). The linear equalizers of both filters are identical. Note that the pulse magnitude of the MMSE-LE is lower than that of the FC-LE and the tails of the MMSE-LE have larger coefficients than those of the FC-LE. This indicates larger noise and ISI amplification.

As shown in Figure 6.8, FC-LE makes double symbol detection instead of the single symbol detection of LEs. Detector pulses in Figure 6.8 clearly indicate that frequency compensation in FC-LE greatly reduces the inter-symbol interference and noise amplification by reducing the pulse tail coefficients and increasing the pulse magnitude.

6.5.6. Performance

Performance comparison between FC-LE and DFE for the channel of Figure 10.2.5(b) in Proakis (1995) yields the data in Table 6.1. As shown in Table 6.1, in the low SNR range DFE performs better than FC-LE by around 2.8 dB. In the high SNR range above 15 dB, the difference decreases towards zero. FC-LE has a non-feedback configuration and its performance is close to DFE in medium to high SNR region.

Table 6.1. Error probability versus SNR for FC-LE and DFE.

Error Probability	SNR FC-LE	SNR DFE	SNR Difference
10^{-6}	20.0 dB	18.5 dB	1.5 dB
10^{-5}	19.0 dB	16.7 dB	2.3 dB
10^{-4}	17.6 dB	15.3 dB	2.3 dB
10^{-3}	15.7 dB	13.0 dB	2.7 dB
10^{-2}	12.8 dB	10.0 dB	2.8 dB

CHAPTER 7

DTV CHANNEL ESTIMATION

7.1. DTV Channel

DTV Data Transmission Model

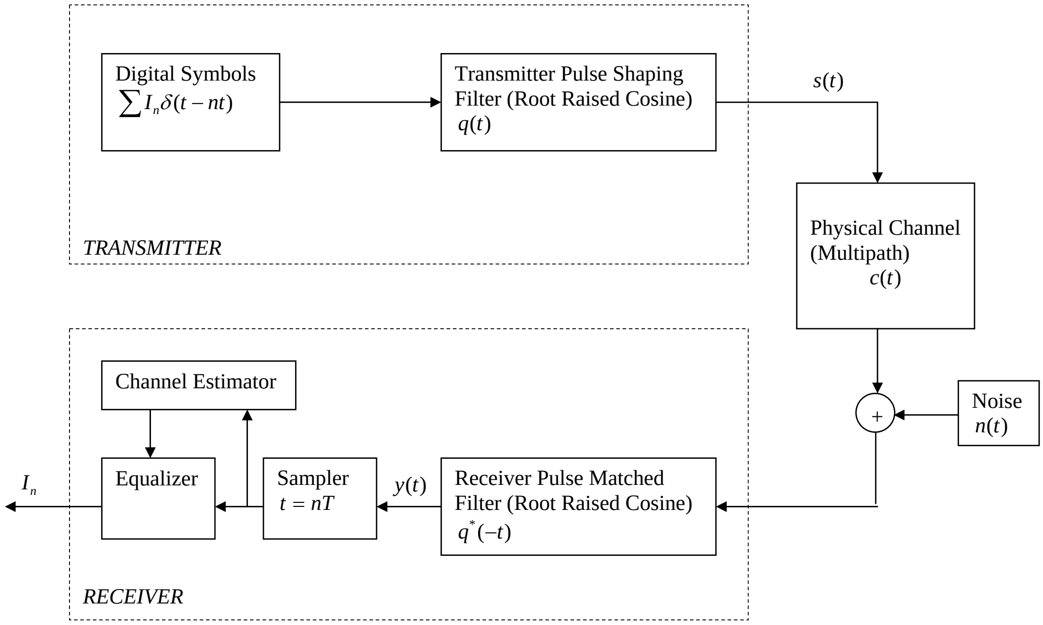


Figure 7.1. General DTV data transmission model block diagram.

The DTV transmission model depicted in Figure 7.1 consists of three main parts. The first part is the transmitter where the digital data is passed through a root-raised cosine pulse shaping filter for better ISI immunity. The second part is the receiver where received samples are passed through a pulse matched filter and equalized for the channel. The third part is the physical channel which may have multi-path characteristics. There exists a channel estimation module in the receiver for channel equalization purposes.

7.2. Properties of DTV CIR Data

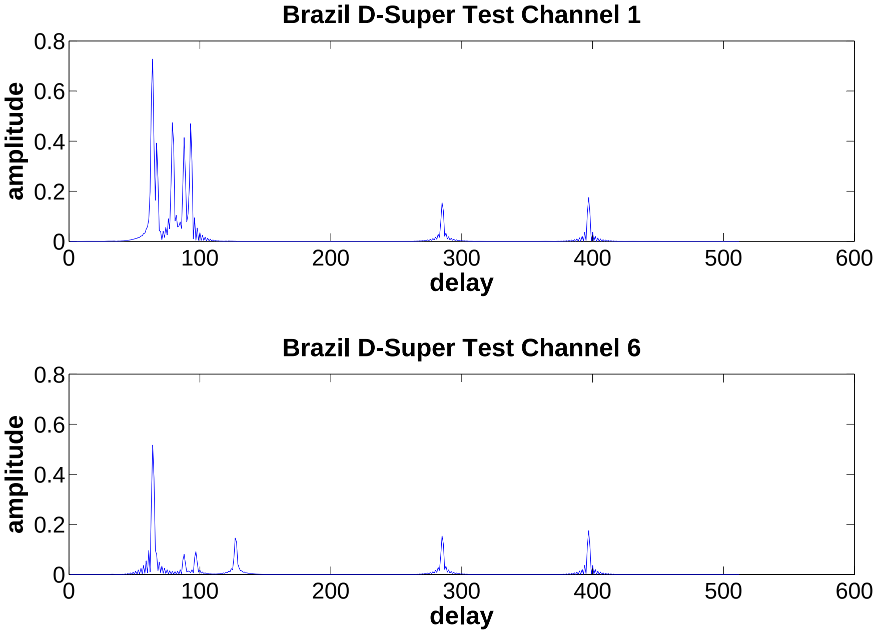


Figure 7.2. Two Brazil D-Super test channels are shown. These channels are designed to measure the performance of DTV receivers against different real-life channel characteristics. Note the sparsity of the channel with only a few major taps and their tails as compared to the channel lengths of 512.

In digital communications, channel impulse response (CIR) data is represented by a vector of complex numbers. Each vector element represents a time delayed and scaled reception of the transmitted stream due to a reflection. The index of each vector element is the amount of delay in symbol periods and the value of each element is the strength of the reception at this time delay. For example, if there are 12 multi-path receptions perhaps due to 11 reflecting objects and 1 direct reception, there will be 12 significant values on the CIR vector all others being close to zero. These significant values are called taps and will be used in the calculation of equalizer filter coefficients. As transmission bit-rate goes high, the length of the CIR vector becomes longer because time intervals in terms of bit periods is proportional to the bit-rate as indicated in the equation $\delta = (L/c)BR$, where L is the path difference, c is the speed of light, δ is the time difference in bit periods, BR is the transmission bit-rate. For example, for a 12 Mbit/s DTV transmission, 1 km path difference between two nearest receptions is roughly equal to $(1000/c)12.10^6 = 40$ bits. This means that there are 39 zero taps between the two receptions. It is not unusual to have 10 km difference between the closest and farthest receptions making the CIR length at least 400 bits for HDTV broadcasts. Number of objects causing significant multi-path reflections are generally limited and are much less than the channel length. Therefore CIR for DTV broadcasts are sparse having usually less than 10% of their taps nonzero. High bit-rate channels can be said to have 3 main properties. 1) They are long. 2) They are sparse. 3) In general, the magnitude of significant taps decreases as their index increase indicative of the fact that larger index taps represent farther receptions and hence weaker signals. The sparsity of the CIR is an important advantage in its estimation as will be indicated in the following sections. Figure 7.2 depicts two examples of DTV channels as seen from the PMF output in the receiver.

7.3. Terrestrial DTV Frame Structure and Channel Estimation Challenge

The 8-VSB Digital Broadcast Standard has been standardized in ATSC (1995) for the terrestrial video broadcasting in the United States. As shown in Figure 7.3, this standard has a frame structure consisting of 313 segments. Each segment has 832 symbols. First four symbols of each segment are allocated for segment synchronization. First segment of each frame has 700 frame synchronization symbols following the first 4 symbols for segment synchronization. The subsequent 128 symbols in the first segment are reserved. Each of the remaining 312 segments have 828 data symbols following the first 4 segment synchronization symbols. Therefore each 8-VSB DTV frame consists of $313 \times (4 + 828) = 260416$ symbols. Of which $312 \times 828 = 258336$ symbols are for data, $313 \times 4 = 1252$ symbols are for segment synchronization and 700 symbols for frame synchronization. It should be noted that all synchronization symbols are known by both the transmitter and receiver. Although the 700 frame synchronization symbols are not truly intended for channel estimation, it is possible to utilize them for this objective. The challenge here is 700 symbols plus the first four symbols making 704 total are not sufficient to get a least squares estimate of a channel more than 352 symbols long. And indeed, DTV channels are almost always longer than 352 symbols ranging between 500 and 600 symbols. That long a channel makes the system heavily underdetermined and hence unsolvable unless some other constraints are available. Luckily, besides being very long, DTV channels are very sparse. Otherwise estimating a 576 symbols long channel using only a 704 symbols long sequence would not be possible. The challenge here is to estimate a long delay spread sparse channel using a short training sequence. The problem can be treated as a ULDIP type inverse problem and sparse recovery techniques such as SP, CoSamp, SCR and CR which have been introduced in the previous chapters can be applied.

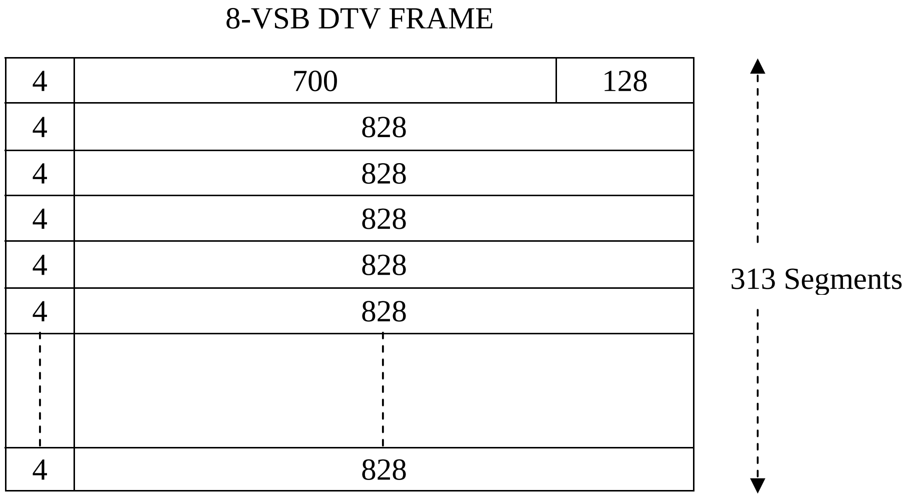


Figure 7.3. The 8-VSB DTV Frame Structure is shown (ATSC 1995). Each frame consists of 313 segments. Each segment begins with 4 symbols allocated for segment synchronization. The remaining 828 symbols in each segment is the payload carrying video and audio data and associated overhead. The first 704 symbols in the first segment of every frame are for frame synchronization, the remaining 128 symbols are reserved. As a secondary purpose, these 704 bits can be used for channel estimation.

7.4. Estimation of Long Delay Spread Sparse DTV Channels through Sparse Recovery Methods Using Short Training Sequences

7.4.1. Problem Statement

As stated in the previous section, in 8-VSB standard there are only 704 training symbols available to estimate a channel of length more than 500 samples. In order to evaluate a least squares estimate of the channel, a linear discrete forward problem must be formulated and the corresponding inverse problem must be solved. The system can be described by the well known formula $y = Ah + n$ where y is the received samples, A is the Toeplitz matrix constructed from the training sequence, h is the channel impulse response and n is the noise. It is known that the received samples are the convolution of the transmitted symbols with the channel impulse response (CIR). This convolution is formulated in discrete time by using a Toeplitz matrix A whose each row is a one sample shifted version of the row above itself. This can best be explained by an example. Assume that a training sequence of length 10 is given by

$$T = [a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9]$$

To estimate a channel of length 7 using the received data corresponding only to the known symbols, the following 4 by 7 Toeplitz matrix must be constructed.

$$A = \begin{bmatrix} a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 \\ a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 \\ a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 \\ a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 \end{bmatrix}$$

Then the forward system of convolution $y = Ah + n$ is

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} a_3 & a_4 & a_5 & a_6 & a_7 & a_8 & a_9 \\ a_2 & a_3 & a_4 & a_5 & a_6 & a_7 & a_8 \\ a_1 & a_2 & a_3 & a_4 & a_5 & a_6 & a_7 \\ a_0 & a_1 & a_2 & a_3 & a_4 & a_5 & a_6 \end{bmatrix} \cdot \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \end{bmatrix} + \begin{bmatrix} n_0 \\ n_1 \\ n_2 \\ n_3 \end{bmatrix}$$

This is an underdetermined system since the matrix A has more columns than rows. From this example, it can be seen that if a training sequence of length L is used to estimate a CIR of length N using received samples corresponding only to known training symbols, a Toeplitz convolution matrix of dimension $M \times N$ is needed where $M = L - N + 1$.

For the case of 8-VSB DTV channel case, there is a training sequence of length $L = 704$ and a CIR more than 500 samples long. For the time being, a channel length of $N = 576$ samples will be assumed. The convolution Toeplitz matrix will then be a 129×576 matrix. This indeed is a heavily underdetermined matrix whose number of rows are less than one forth of number of columns and hence the stated problem is a ULDIP type problem requiring sparse recovery algorithms to solve.

7.4.2. Solution

The solution to the inverse problem stated in above subsection is to use one of the sparse recovery methods developed for ULDIP type problems whose solution is expected to be highly sparse. Among such methods as mentioned in Chapter 2 are SCR, CR, SP, CoSamp or IHT. The best way to evaluate their performances on this particular problem is to make test simulations. In the following subsection, simulations are carried out and their results are displayed for comparison.

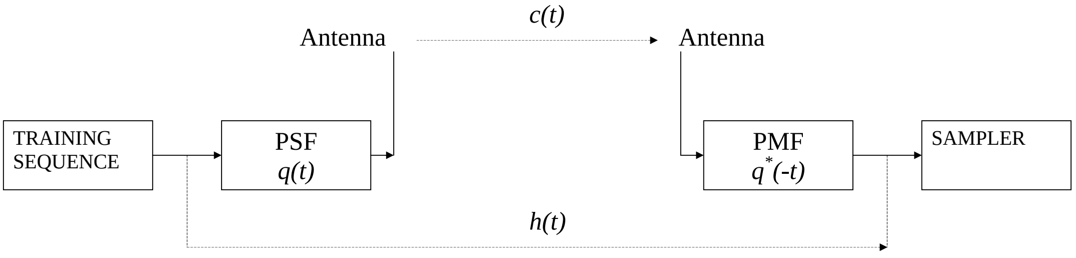
7.4.3. Testing CR and SP for Long Delay Spread DTV Channel Estimation

Test Environment:

DTV channels 5 through 10 are used for testing. Their delay and attenuation values $c(t)$ are given in Appendix A. Referring to the DTV transmission model block diagram in Figure 7.1, the physical CIR $c(t)$ represents the physical multi-path channel as seen between the transmitter and receiver antennas. Overall channel CIR $h(t)$ represents the channel between the input of the pulse shaping filter in the transmitter and the output of the pulse matched filter in the receiver. In the following tests, only the channel impulse response $h(t)$ will be recovered. The CIR $h(t)$ of the channel observed from the pulse matched filter output in the receiver is the triple convolution of the physical channel $c(t)$ with the pulse shaping filter impulse response $q(t)$ in the transmitter and pulse matched filter impulse response $q^*(-t)$ in the receiver and is given by the convolution equations

$$h(t) = c(t) * p(t), \quad \text{where} \quad p(t) = q(t) * q^*(-t).$$

DTV Physical Channel Impulse Response *c(t)* Versus Overall Channel Impulse Response *h(t)*



PSF	: Pulse Shaping Filter
PMF	: Pulse Matched Filter
c(t)	: Physical channel between the antennas
h(t)	: Overall channel between the input of PSF and the output of PMF $h(t) = c(t) * q(t) * q^*(-t)$

Figure 7.4. $c(t)$ versus $h(t)$. Impulse response $c(t)$ between the transmitter and receiver antennas versus impulse response $h(t)$ between the input of the pulse shaping filter in the transmitter and the output of the pulse matched filter in the receiver. Therefore $h(t)$ is the triple convolution of the physical channel impulse response $c(t)$ and the transmitter pulse shaping filter impulse response $q(t)$ and the receiver pulse matched filter $q^*(-t)$ as given by the equation $h(t) = c(t) * p(t)$, where $p(t) = q(t) * q^*(-t)$.

For 8-VSB DTV channels, pulse shaping and pulse matched filters are identical $q(t) = q^*(-t)$.

There will be 6 tests using the CR and SP algorithms for no noise and 20 dB SNR cases. 704 symbols long 8-VSB training sequence (ATSC 1995) will be used as the training sequence. The tests will be carried out for no noise case and for 20 dB SNR with Gaussian noise. A 129 by 576 Toeplitz matrix will be formed for sparse recovery.

Each figure for all 6 tests will have vertically positioned 5 plots (a), (b), (c), (d) and (e). The top most plot (a) will be the original CIR $h(t)$ of the corresponding channel. Plots (b) and (c) will depict the CR output for noiseless and 20 dB SNR cases respectively. Plots (d) and (e) will depict the SP output for noiseless and 20 dB SNR cases respectively.

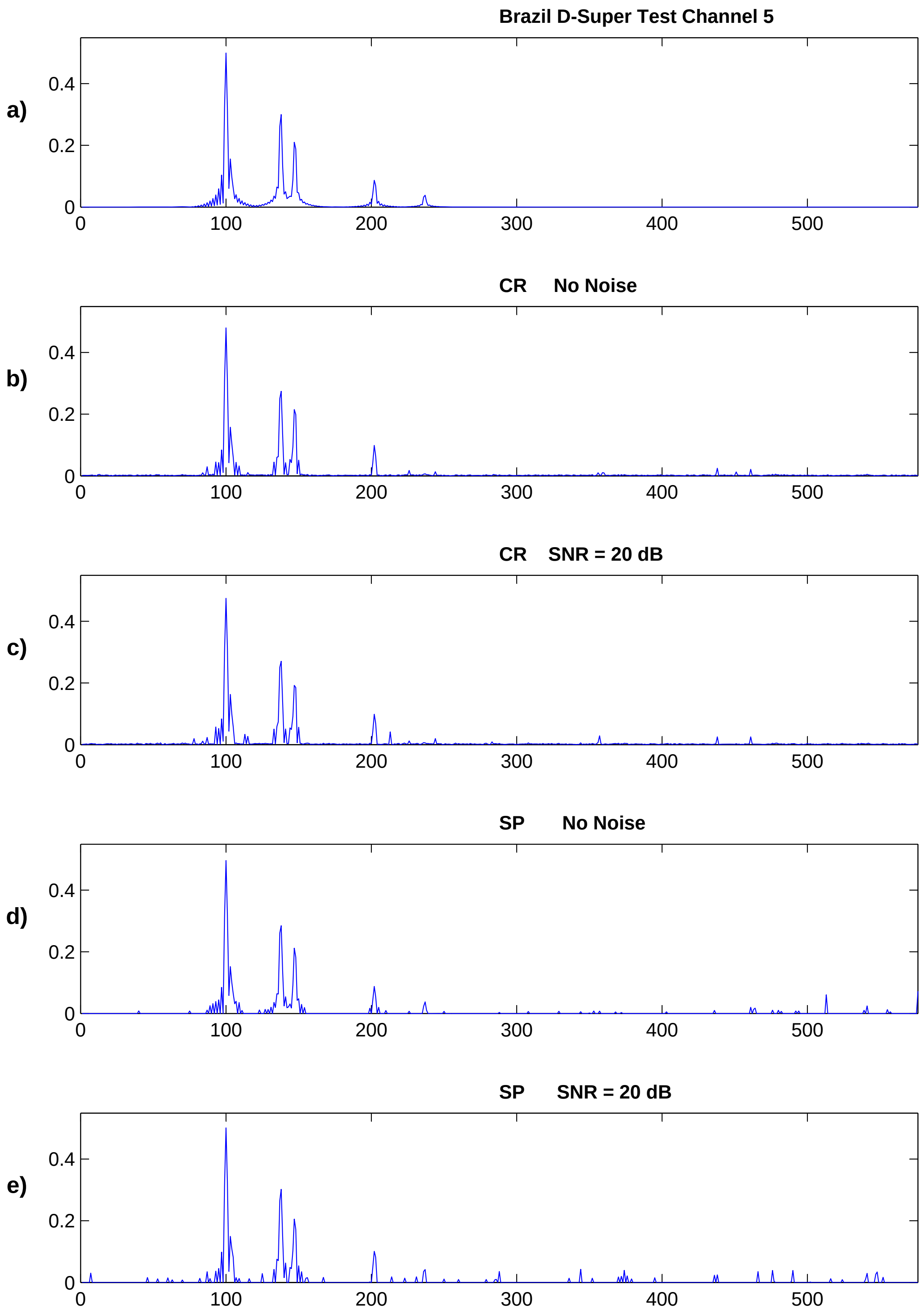


Figure 7.5. a) Channel 5 b) CR estimate without noise c) CR estimate at SNR = 20 dB d) SP estimate without noise e) SP estimate at SNR = 20 dB.

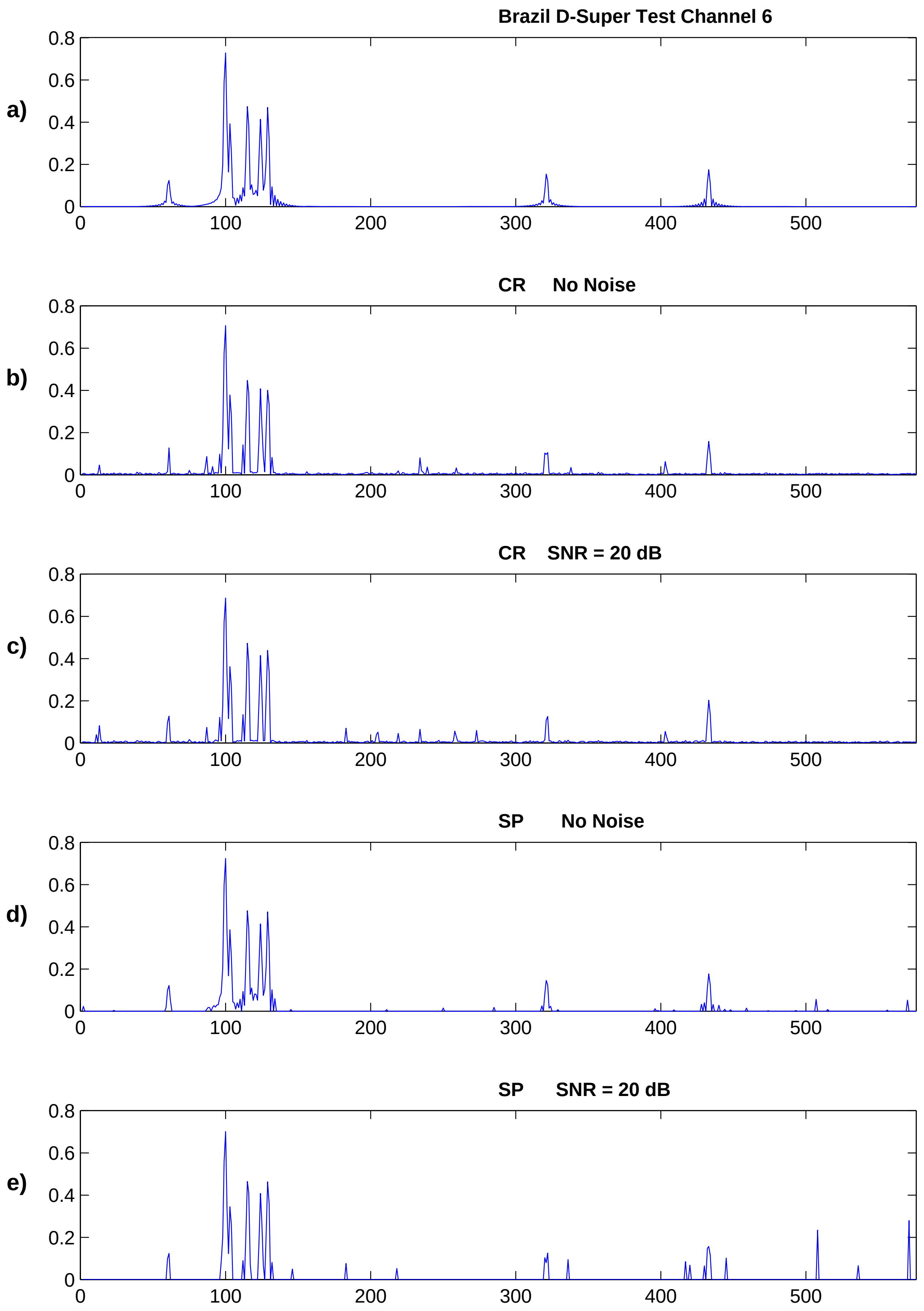


Figure 7.6. a) Channel 6 b) CR estimate without noise c) CR estimate at SNR = 20 dB d) SP estimate without noise e) SP estimate at SNR = 20 dB.

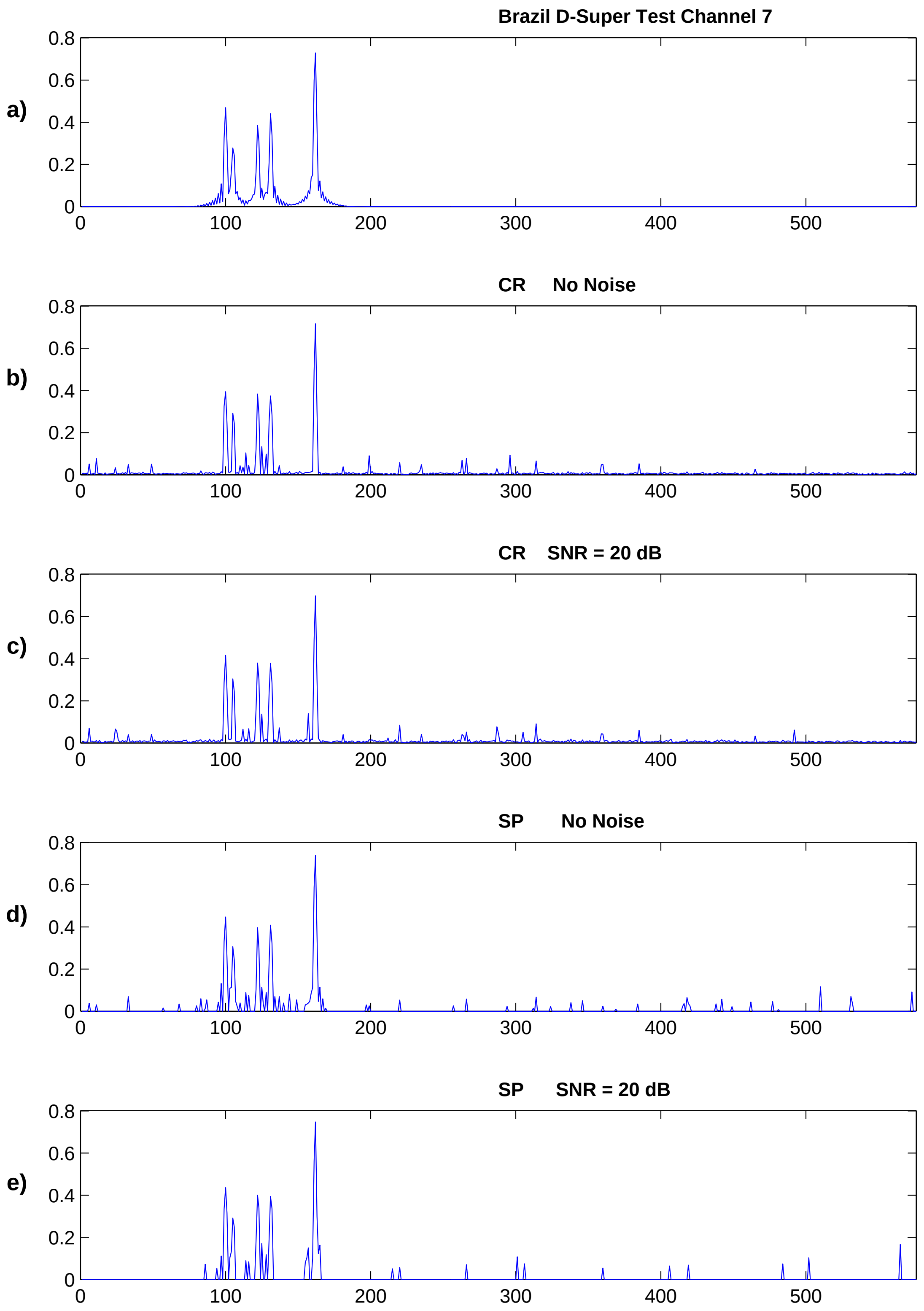


Figure 7.7. a) Channel 7 b) CR estimate without noise c) CR estimate at SNR = 20 dB d) SP estimate without noise e) SP estimate at SNR = 20 dB.

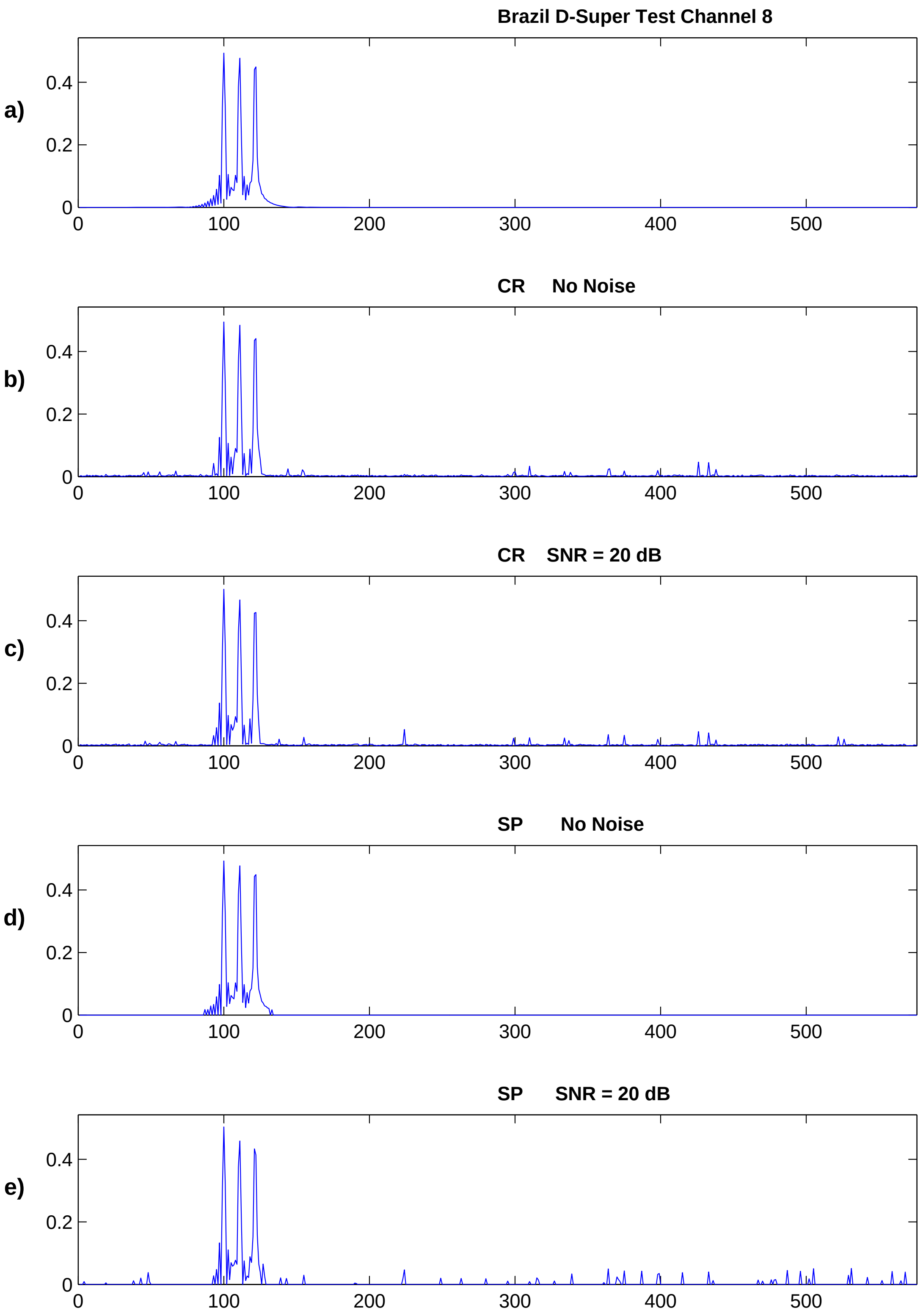


Figure 7.8. a) Channel 8 b) CR estimate without noise c) CR estimate at SNR = 20 dB d) SP estimate without noise e) SP estimate at SNR = 20 dB.

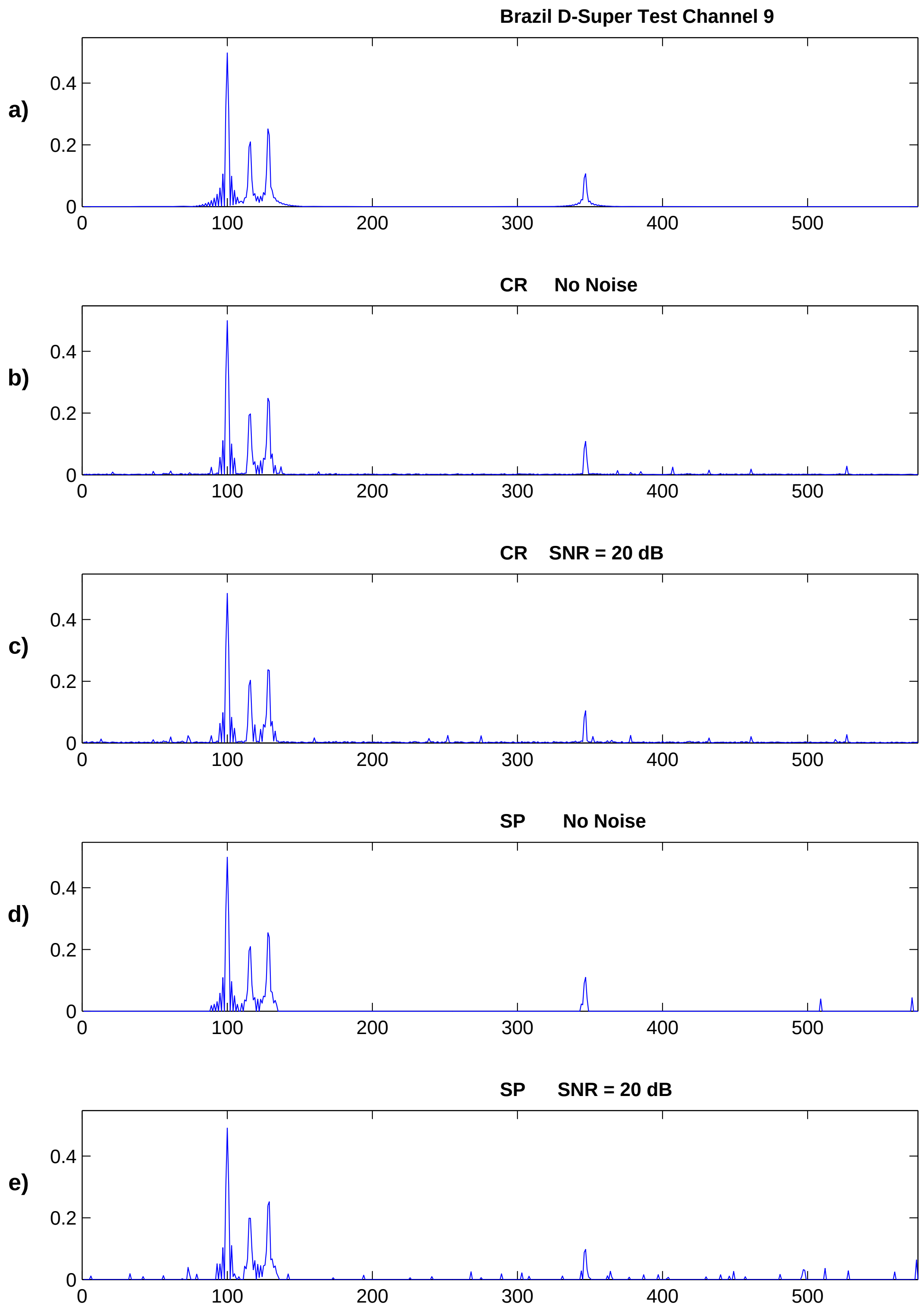


Figure 7.9. a) Channel 9 b) CR estimate without noise c) CR estimate at SNR = 20 dB d) SP estimate without noise e) SP estimate at SNR = 20 dB.

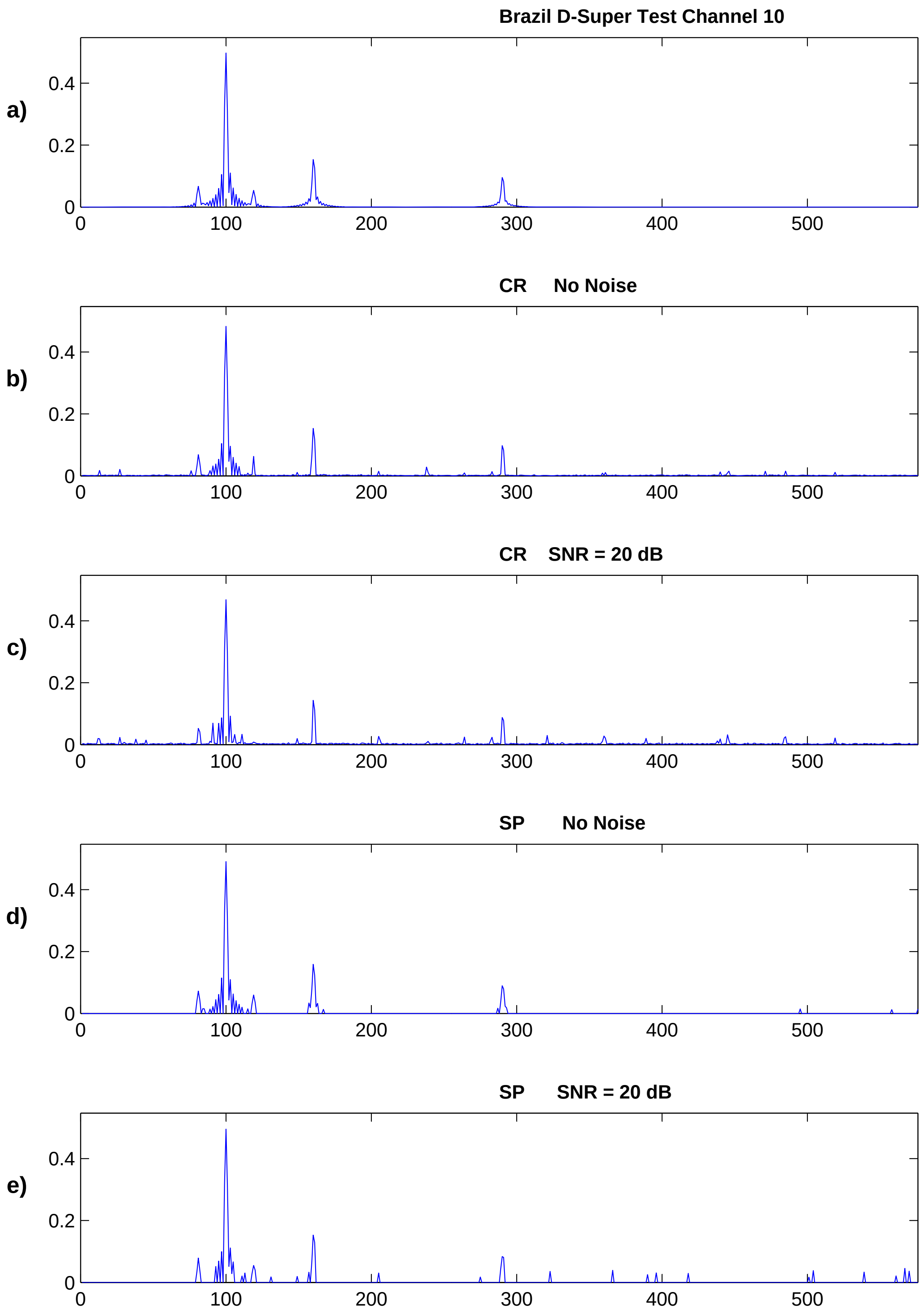


Figure 7.10. a) Channel 10 b) CR estimate without noise c) CR estimate at SNR = 20 dB d) SP estimate without noise e) SP estimate at SNR = 20 dB.

7.4.4. Test Results

Tests with channel 5 in Figure 7.5 show that both CR and SP have estimated all the taps but SP seems to introduce some small spurious taps at 20 dB SNR. Tests with channel 6 in Figure 7.6 show that CR has small spurious taps in the no noise case. But SP has more spurious taps at 20 dB SNR. Both algorithms exhibit similar performances in the tests with channel 7 as shown in Figure 7.7. SP performs good in the test with channel 8 for the no noise case but has more spurious taps at 20 dB SNR as shown in Figure 7.8. Both algorithms exhibit nearly equal performances for channel 9 and channel 10 as shown in Figures 7.9 and 7.10.

It is of primary importance to estimate correctly the channel tap positions rather tan the channel tap values since once the positions of the major taps are determined, method of least squares of a small dimension can be used to accurately determine the tap values. In all tests, CR seems to loose low amplitude tails slightly more than SP. However at 20 dB SNR, SP introduces spurious taps slightly more than CR. This is more prominent in Figures 7.6 and 7.7. All 6 tests demonstrate that both SP and CR algorithms can recover sparse DTV channels with nearly equal performances.

7.4.5. Direct Estimation of the Physical Channel Taps *c(t)*

Problem Statement: As can be seen from the tests in the previous subsection and from the sample channel in Figure 7.11, the overall channel impulse response $h(t)$ to be recovered may not be perfectly sparse although the underlying physical channel $c(t)$ is perfectly sparse having only a few nonzero taps as shown in the table in Appendix A. Due to the convolution with PSF and PMF impulse responses, the taps in $h(t)$ has long and sometimes overlapping tails which are non zero as shown in Figure 7.11. Since the sparse recovery methods that estimate the CIR assumes that the channel $h(t)$ is perfectly sparse, their performance may suffer by missing some low amplitude tails particularly in the presence of noise as shown in the tests in the previous subsection.

The Solution: The solution to this problem is to estimate the physical CIR $c(t)$ directly instead of the overall CIR $h(t)$. Since $c(t)$ is much sparser than $h(t)$ as shown in Figure 7.11, much better performance is expected from sparse recovery algorithms

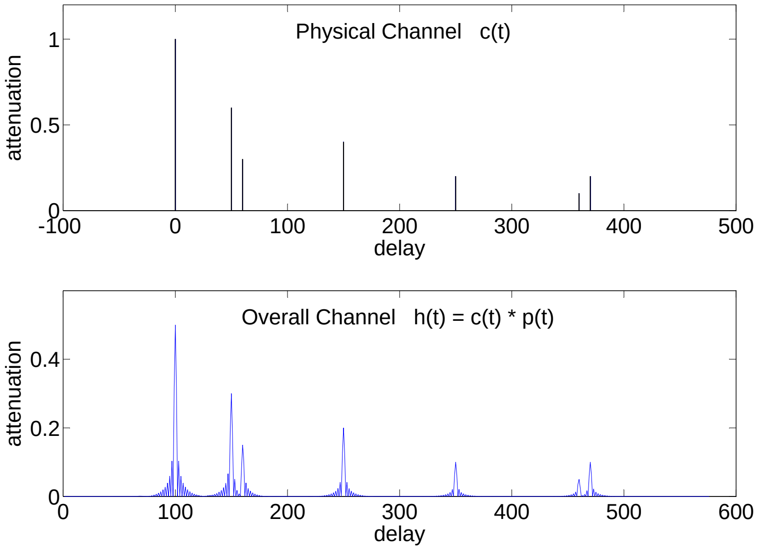


Figure 7.11. A sample physical channel impulse response $c(t)$ (top) versus overall channel impulse response $h(t)$ (bottom). Note that $c(t)$ is much sparser than $h(t)$.

for the estimation of $c(t)$. Once $c(t)$ is estimated, it is trivial to obtain $h(t)$ through the convolution operation $h(t) = c(t) * p(t)$. Instead of using the 8-VSB training sequence (ts) of 704 symbols directly, it will be convolved with the PSF and PMF filter impulse responses before being used in the sparse recovery of the channel. The new training sequence ts_{new} will be calculated by the convolution $ts_{new} = ts * p(t)$ where $p(t) = q(t) * q^*(-t)$ is the combined impulse response of the PSF and PMF. The system matrix A which has previously been used to estimate $h(t)$ will now be the Toeplitz of ts_{new} instead of ts, that is, $A = Toeplitz(ts_{new})$. Using the new convolved training sequence ts_{new} instead of ts and solving the ULDIP type problem $y(t) = A \times c(t)$, the physical channel taps $c(t)$ can be estimated.

7.4.6. Tests for Estimating the Physical CIR *c(t)*

DTV channels 5 through 7 are used for testing. Their delay and attenuation values $c(t)$ are given in Appendix A. Referring to the DTV transmission model block diagram in Figure 7.1, the physical CIR $c(t)$ represents the physical multi-path channel as seen between the transmitter and receiver antennas. There will be 6 tests using the CR and

SP algorithms for no noise and 20 dB SNR cases. 704 symbols long 8-VSB frame synchronization symbols (ATSC 1995) will be used as the training sequence. The tests will be carried out for no noise case and for 20 dB SNR with Gaussian noise. A 129 by 576 Toeplitz matrix will be formed for sparse recovery.

Each figure for all 6 tests will have vertically positioned 5 plots (a), (b), (c), (d) and (e). The top most plot (a) will be the original CIR $h(t)$ of the corresponding channel. Plots (b) and (c) will depict the CR output for noiseless and 20 dB SNR cases respectively. Plots (d) and (e) will depict the SP output for noiseless and 20 dB SNR cases respectively.

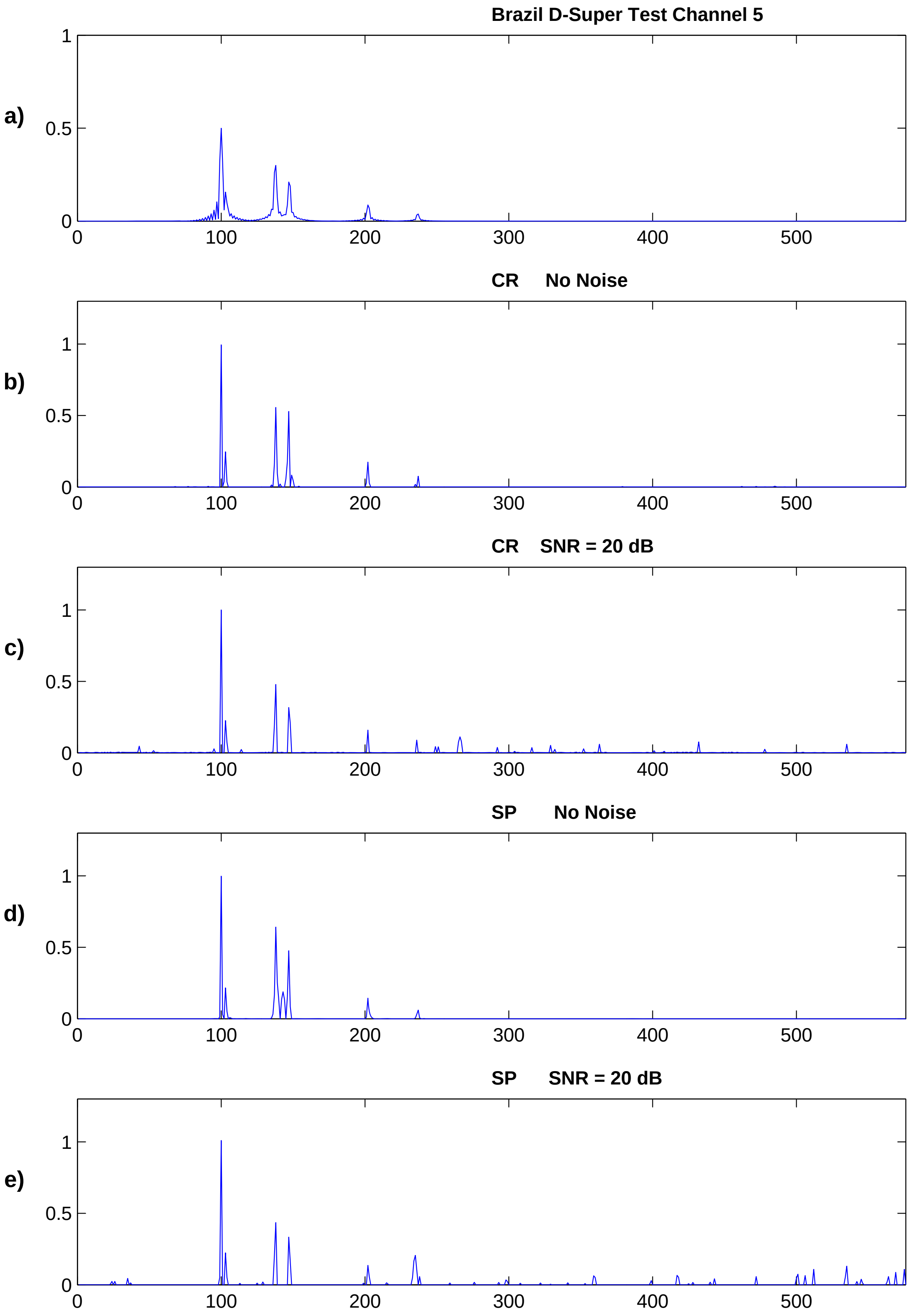


Figure 7.12. a) Channel 5 b) CR estimate without noise c) CR estimate at SNR = 20 dB d) SP estimate without noise e) SP estimate at SNR = 20 dB.

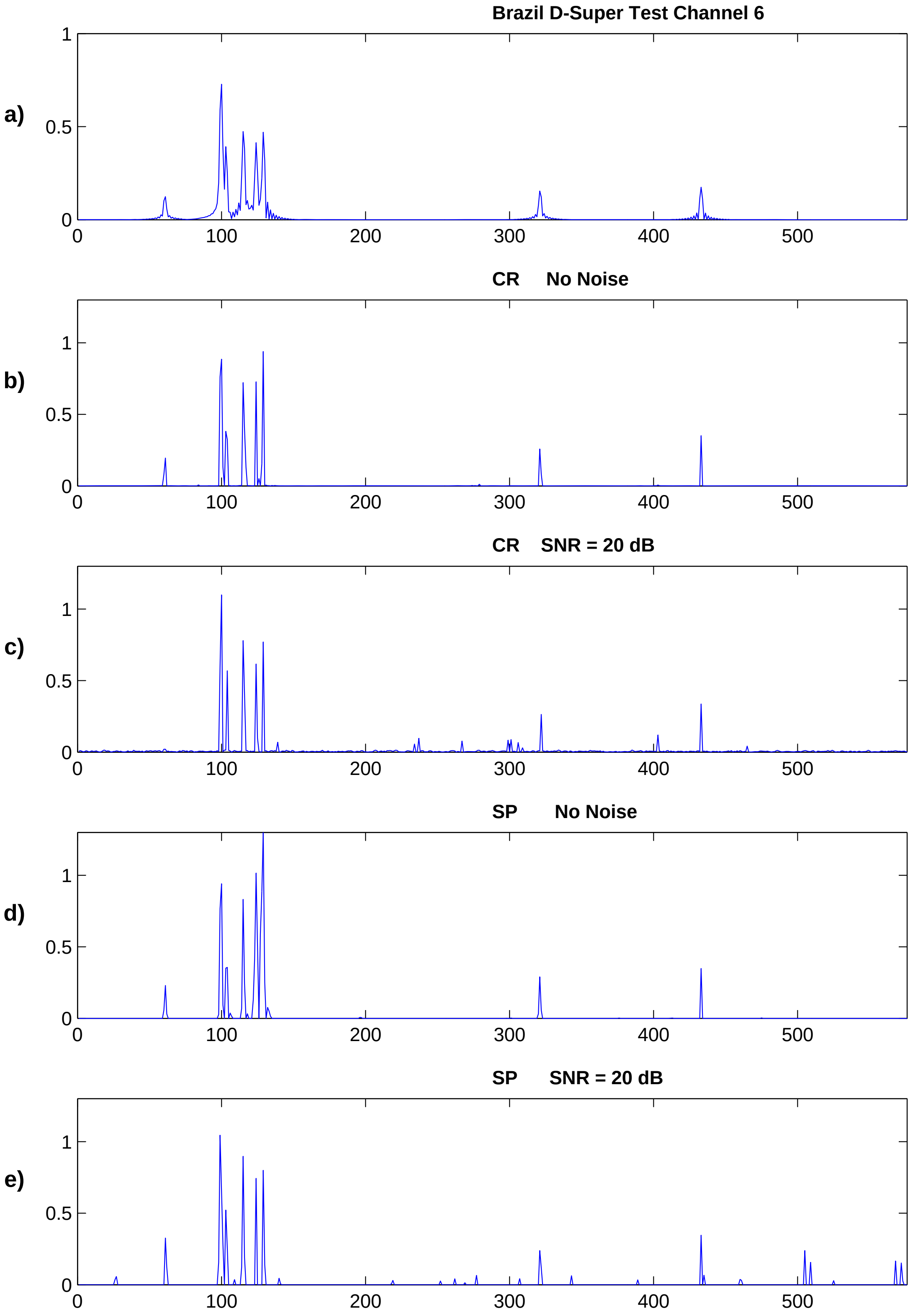


Figure 7.13. a) Channel 6 b) CR estimate without noise c) CR estimate at SNR = 20 dB d) SP estimate without noise e) SP estimate at SNR = 20 dB.

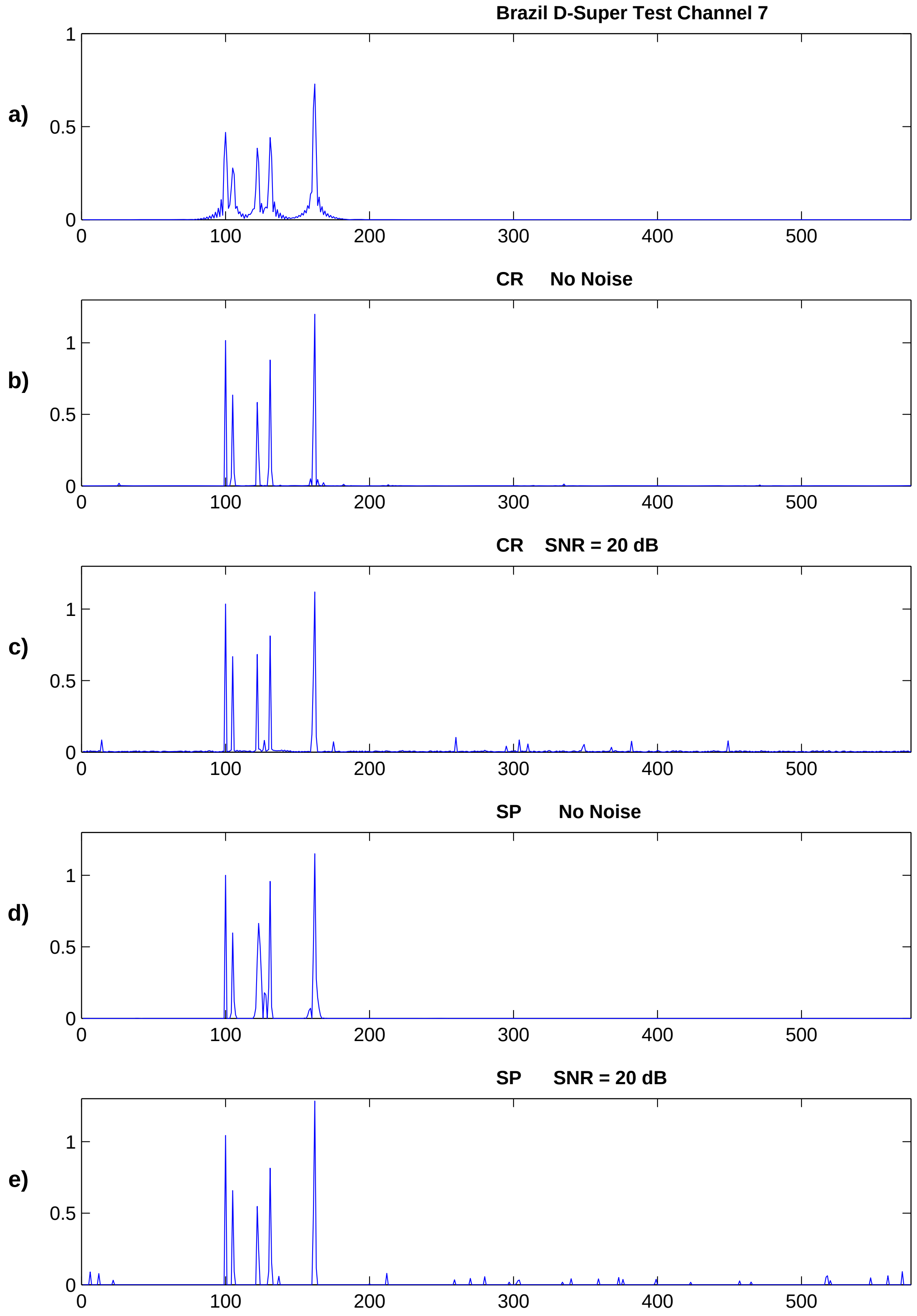


Figure 7.14. a) Channel 7 b) CR estimate without noise c) CR estimate at SNR = 20 dB d) SP estimate without noise e) SP estimate at SNR = 20 dB.

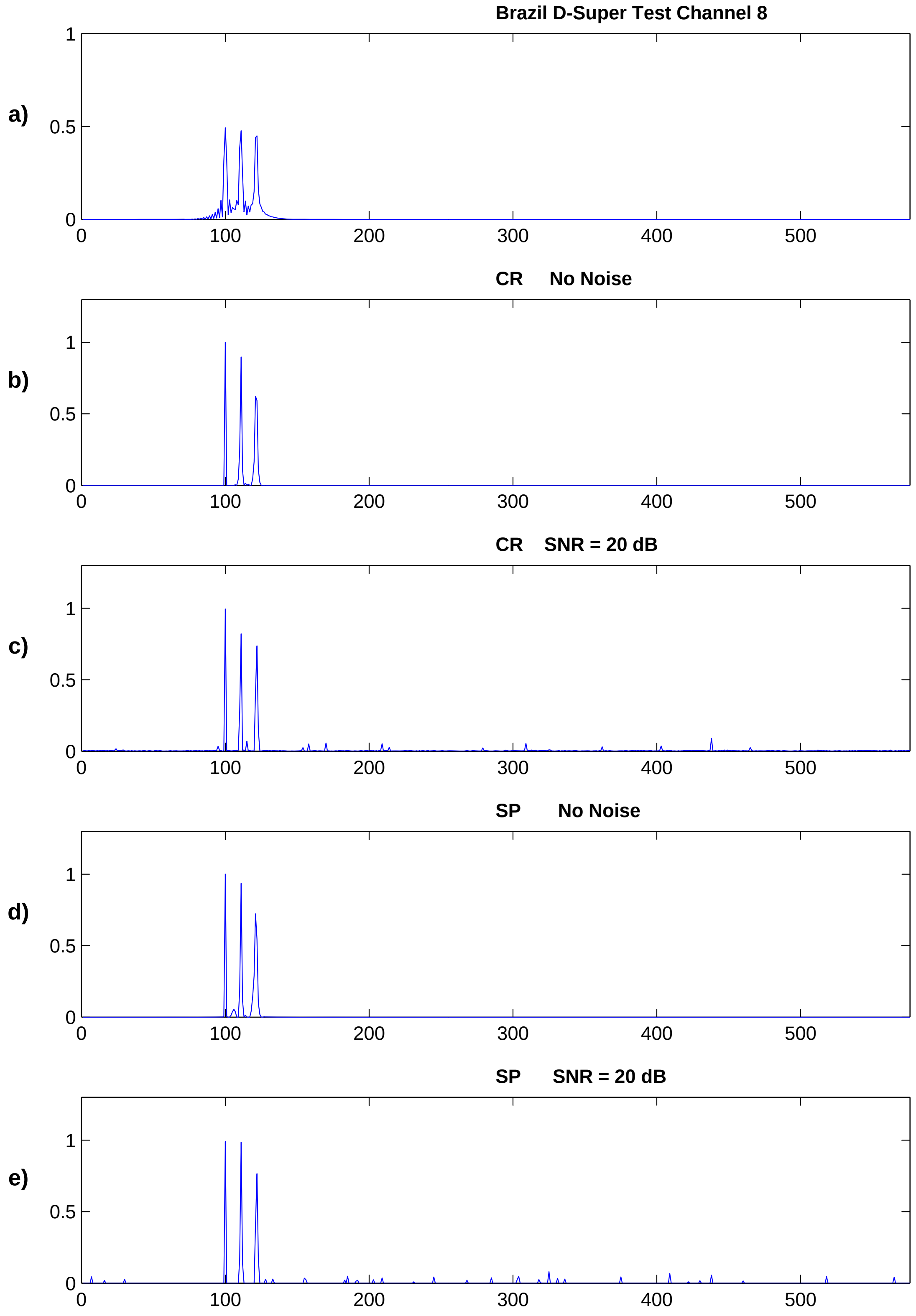


Figure 7.15. a) Channel 8 b) CR estimate without noise c) CR estimate at SNR = 20 dB d) SP estimate without noise e) SP estimate at SNR = 20 dB.

7.4.7. Test Results

Tests with channel 5 in Figure 7.12 show that both CR and SP have recovered the physical taps correctly for the no noise cases. Both SP and CR seems to introduce some small spurious taps at 20 dB SNR. Tests with channel 6 in Figure 7.13 show that both CR and SP have recovered the physical taps correctly for the no noise cases. But SP has several spurious taps and CR misses one tap at 20 dB SNR. Both algorithms exhibit similar performances in the tests with channel 7 shown in Figure 7.14. Figure 7.15 demonstrates that both SP and CR perform well in the tests with channel 8 in the no noise cases but both have several small spurious taps at 20 dB SNR.

All four tests show that it is possible to undo the effect of PSF and PMF filters and recover the physical taps directly using sparse recovery algorithms.

CHAPTER 8

CONCLUSION

In this thesis, a review of inverse problems and regularization has been presented, some important signal processing problems particularly sparse channel estimation have been studied from an inverse problem perspective. New solution methods for these problems are proposed and used for signal processing applications.

Inverse problems are generally harder to solve than direct problems. This is generally because of the scarcity of information available to solve the inverse problem. In many practical inverse problems, despite large number of sensory measurements taken, information is still very scarce due to physical constraints and indirect sampling. Large number of near identical sensory measurements makes the inverse problem ill-posed and its solution very sensitive to measurement errors and noise. Direct solution of inverse problems generally produce unrealistic, irregular solutions. Additional constraints are needed to make the solutions regular i.e. realistic. This enforcement of additional constraints is called regularization. Regularization is usually done by imposing norm minimization on the solution. Mostly L_2 norm (energy) of the solution is minimized. Tikhonov regularization which enforce parametric L_2 norm minimization both on the solution and the error of the inverse problem has a convenient closed form. L_0 norm minimization is not feasible and is not necessary unless the solution has to be sparse. L_1 norm minimization is more robust against outliers than L_2 norm minimization but has no convenient closed form. In addition, L_1 norm minimization induces sparsity which may not be necessary in some practical cases. With a very convenient closed form, Tikhonov regularization stands out as the most practical regularization method. Optimum regularization parameter selection for regularization is being actively researched and some heuristic methods are proposed. Manual selection by visual evaluation is still a preferred choice. If the energy of the solution and noise variance are known a priori, optimum regularization parameter for Tikhonov regularization is the variance of the noise divided by the variance of the solution. Apart from the random noise, if measurement errors exist (measurement matrix is perturbed) this error variance also need to be included in the regularization parameter.

Most indirect (noninvasive) imaging problems such as biomedical and seismic imaging fall into the class of overdetermined linear discrete inverse problems (OLDIP). Practical OLDIP are ill-posed due to large number of proximate measurements required for an image of useful resolution.

Some important inverse problems encountered in signal processing and communication are of underdetermined linear discrete inverse problem type (ULDIP). All ULDIP are ill-posed by nature. They need to be regularized in a similar fashion to OLDIP. Among infinitely many possible solutions to ULDIP, the one with the minimum norm is selected. In contrast to ULDIP, practical OLDIP does not have sparse solutions and generally the energy of the solution and the error is minimized. Some important practical ULDIP in signal processing and communication have sparse or near sparse solutions. Therefore L_0 norm minimization need to be used. Since L_0 norm minimization is not practically feasible and not suitable for near sparse solutions, L_1 norm minimization is preferred as a substitute for L_0 minimization provided that the measurement matrix satisfies the restricted isometry property. Yet, the complexity of L_1 norm minimization is not low enough for most practical problems.

Alternative sparse recovery methods SP and CoSaMP based on the matching pursuit (MP) algorithm are proposed for ULDIP. The initial MP algorithm and its successor OMP have very low complexities but they perform well only if the solution is guaranteed to be very sparse. MP based SP and CoSaMP algorithms perform satisfactorily for less sparse solutions with some complexity penalty. The constraint removal (CR) algorithm proposed in this thesis is another sparse recovery algorithm which is not based on matching pursuit. The CR algorithm has better performance than all other sparse recovery algorithms for zero-one type signals and a similar performance for Gaussian type signals. Another additional advantage of CR is that it does not require prior sparsity knowledge in contrast to SP and CoSaMP.

Sparse recovery algorithms SP and CR are demonstrated to be successful in long delay sparse channel estimation with short training sequences. Yet, their complexities are not low enough for practical real time channel estimation since all of them use least squares evaluation at all iterations. They need to be computationally optimized before they can be used for practical inverse problems. If the channel can be guaranteed to be very sparse, matching pursuit or its derivative orthogonal matching pursuit can still be a fast and feasible solution (Cotter and Rao 2002).

Being able to estimate long delay spread channels with short training sequences is highly desirable for an increased channel efficiency. In addition, broadcast standards (e.g. ATSC 1995) may not allow any training sequences other than what is available in the standard. High data rate wireless transmissions such as DTV and HDTV broadcasts involve channel impulse responses with several hundred samples length. Conventional methods like ML, LS and MAP methods need accordingly long training sequences which penalize the channel efficiency. Correlation and linear deconvolution are two possible alternatives. Their performance is low if the channel is not very sparse and if the noise variance is high. Diversity can be used to improve the estimation performance. Permuted deconvolution (PDEC) which use the subsequential diversity within the training sequence itself is proposed to improve the performance of linear deconvolution. PDEC can be implemented by a low complexity 1-D transversal filter. It can either be used as a stand alone channel estimator or as an initial step for other estimators.

From an inverse problem perspective, equalization is a ULDIP type inverse problem. For long delay sparse channels and for limited filter lengths, equalization becomes a heavily underdetermined and therefore a severely ill-posed problem. Regularization of this problem using SNR results in linear equalization. When the measurement system is severely ill-posed, linear equalization performance becomes unsatisfactory despite regularization. One remedy to this problem is decision feedback. In inverse problem terms, decision feedback is equivalent to transforming a heavily underdetermined inverse problem into a mildly underdetermined or overdetermined one by reducing the horizontal dimension of the measurement (convolution) matrix. This model assumes that the previous decisions are correct and part of the convolution matrix corresponding to the previous decisions can be truncated. In this thesis, a new linear equalizer called frequency compansated linear equalizer (FC-LE) has been proposed. FC-LE is based on double symbol detection to overcome intersymbol interference without a feedback path. FC-LE can achieve very low error rates at high SNR levels in contrast to linear equalizers which have an error floor even in the absence of noise. The ultimate goal of FC-LE idea is to achieve blind equalization through frequency compensation and multiple symbol detection.

REFERENCES

Al-Dhahir N. (1996) Efficient computation of the delay-optimized Finite length MMSE-DFE, *IEEE Transactions on Signal Processing*, vol. 44 no. 5, pp. 1288-1292.

ATSC. (1995) Digital television standard. A/53, September 1995.

Austin M.E. (1967) *Decision-feedback equalization for digital communication over dispersive channels*, M.I.T. Res. Lab. Electron., Tech. Rep. 461, Aug.

Bertero M., Mol C.D. and Pike E.R. (1985) Linear inverse problems with discrete data: I. General formulation and singular system analysis, *Inverse Problems.*, vol. 1, no. 4, pp. 301-330.

Bertero M., Mol C.D. and Pike E.R. (1988) Linear inverse problems with discrete data: II. Stability and regularization, *Inverse Problems.*, vol. 4, no. 3, pp. 573-594.

Bertero M. and Boccacci P. (1998) *Introduction to Inverse Problems in Imaging*. Institute of Physics, Bristol.

Bertero M. and Piana M. (2006) *Inverse problems in biomedical imaging: modeling and methods of solution*, in Complex Systems in Biomedicine, eds. Quarteroni A., Formaggia L., and Veneziani A., 1-33 (Springer, Berlin)

Biondi B. (2006) *3d Seismic Imaging: Three Dimensional Seismic Imaging*. Society of Exploration Geophysicists. ISBN 0-07-011117-0.

Boyd S. and Vandenberghe L. (2004) *Convex Optimization*. Cambridge Univ. Press.

Brown A.R. (2004) *Interpretation of three-dimensional seismic data (sixth ed. ed.)*. Society of Exploration Geophysicists and American Association of Petroleum Geologists. ISBN 0891813640.

Blumensath T. and Davies M.E., (2009) A simple, efficient and near optimal algorithm for compressed sensing, *in Proceedings of the Int. Conf. on Acoustics, Speech and Signal Processing*.

Blumensath T. and Davies M.E. (2010) Normalised Iterative Hard Thresholding; guaranteed stability and performance, *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 298-309.

Candès E.J. and Tao. T. (2005) Decoding by linear programming. *IEEE Trans. Inform. Theory*, vol. 51, no. 12, pp. 4203-4215.

Candès E.J., Romberg J., and Tao T. (2005) Stable signal recovery from incomplete and inaccurate measurements, *Commun. Pure Appl. Math.*, vol. 59, no. 8, pp. 1207-1223.

Candès E.J. (2006) Compressive sampling, *in Proceedings of International Congress of Mathematics*, Madrid, Spain. vol. 3, pp. 1433-1452.

Candès E.J. and Tao T. (2006) Near optimal signal recovery from random projections: universal encoding strategies? *IEEE Trans. Inform. Theory*, vol. 52, no. 12, pp. 5406-5425.

Candès E.J., Romberg J., and Tao T. (2006) Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information, *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489-509.

Candès E.J and Tao T. (2007) The Dantzig selector: Statistical estimation when p is much larger than n, *Annals of Statistics*, vol. 35, no. 6, pp. 2313-2351.

Candès E.J. (2008) The restricted isometry property and its implications for compressed sensing. *C. R. Math. Acad. Sci. Paris, Serie I*, vol. 346, no. 9-10, pp. 589-592.

Chartrand R. (2007) Exact reconstruction of sparse signals via nonconvex minimization, *IEEE Signal Processing Letters*, vol. 14, no. 10, pp. 707-710.

Chartrand R. and Yin W. (2008) Iteratively reweighted algorithms for compressive sensing. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).*, pp. 3869-3872.

Chen S.S., Donoho D.L., and Saunders M.A. (1998) Atomic decomposition by basis pursuit, *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 33-61.

Cotter S.F. and Rao B. D. (2002) Sparse channel estimation via matching pursuit with application to equalization, *IEEE Trans. Commun.*, vol. 50, no. 3, pp. 374-377.

Dai W. and Milenkovic O. (2008) Subspace pursuit for compressive sensing: Closing the gap between performance and complexity. CoRR, abs/0803.0811.

Dantzig G.B. (1951) Maximization of a linear function subject to linear inequalities, in T. C. Koopmans (ed.), Activity Analysis of Production and Allocation, John Wiley & Sons, New York, pp. 339-347.

Dantzig G.B. (1963) *Linear Programming and Extensions*, Princeton University Press, Boston, MA.

Dantzig G.B. and Thapa M.N. (1997) *Linear Programming*. Springer, New York, NY.

Daubechies I., DeVore R., Fornasier M., and Güntürk C. (2010) Iteratively re-weighted least squares minimization for sparse recovery. *Comm. Pure Appl. Math.*, vol. 63, no. 1, pp. 1-38.

Desoer C. A. (1970) *Notes for a Second Course on Linear Systems, Van Nostrand Reinhold Notes on System Sciences*, New York: Van Nostrand Reinhold Company.

Donoho D.L. (2006a) Compressed sensing, *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1289-1306.

Donoho D.L. (2006b) For most large underdetermined systems of linear equations, the minimal l_1 solution is also the sparsest solution. *Comm. Pure Appl. Math.*, vol. 59, no. 6, pp. 797-829.

Donoho D., Tsaig Y., Drori I., and Starck J. (2006) Sparse solution of underdetermined linear equations by stagewise orthogonal matching pursuit, Tech. Report 2006-02, Standford, Department of Statistics.

Enderle J. (2005) *Introduction to Biomedical Engineering*, Second Edition. Academic Press, London.

Engl H.W., Hanke M., and Neubauer A. (1996) *Regularization of Inverse Problems*. Kluwer, Dordrecht.

Foucart S. (2010) A note on guaranteed sparse recovery via ℓ_1-minimization. *Appl. Comput. Harmon. Anal.*, vol. 29, no. 1, pp. 97-103.

Gill P.E., Murray W. and Wright M.H. (1991) *Numerical Linear Algebra and Optimization*, Addison Wesley, Redwood City, CA.

Gorodnitsky I.F. and Rao B.D. (1997) Sparse signal reconstruction from limited data using FOCUSS: A re-weighted minimum norm algorithm. *IEEE Trans. Signal Process.*, vol. 45, no. 3, pp. 600-616.

Groetsch C.W. (2007) Integral equations of the first kind, inverse problems and regularization: a crash course. *Journal of Physics: Conference Series: Inverse Problems in Applied Sciences, towards breakthrough*, vol. 73, pp. 1-32.

Hadamard, J. (1902) Sur les problmes aux drives partielles et leur signification physique, pp. 49-52.

Hadamard J. (1923) *Lectures on Cauchy's problem in linear partial differential equations*, Dover Phoenix editions, Dover Publications, New York, 2003.

Hansen P.C. (1987) The truncated SVD as a method for regularization. *BIT*, vol. 27, no. 4, pp. 534-553.

Hansen P.C. (2007) Regularization Tools version 4.0 for Matlab 7.3, *Numerical Algorithms*, vol. 46, no. 2, pp. 189-194.

Hansen P.C. (2010) *Discrete Inverse Problems: Insight and Algorithms*, SIAM, Philadelphia.

Hansen P.C. and O'Leary D.P. (1993) The use of the L-curve in the regularization of discrete ill-posed problems, *SIAM J. Sci. Comput.*, vol. 14, no. 6, pp. 1487-1503.

Hounsfield G. N. (1973) Computerized Transverse Axial Scanning (Tomography). *Br. J. Radiol.*, vol. 46, no. 552, pp. 1016-1022.

Keller J. (1976) Inverse problems. *Amer. Math. Monthly*, vol. 83, no. 2, pp. 107-118

Lauterbur P.C. (1973) Image Formation by Induced Local Interactions: Examples of Employing Nuclear Magnetic Resonance. *Nature*, vol. 242, no. 5394, pp. 190-191.

Lucky R.W. (1965) Automatic equalization for digital communications, *Bell System Technical Journal*, vol. 44, no. 4, pp. 547-588.

Lucky R.W., Salz J. and Weldon E. (1968) *Principles of Data Communication.* New York, NY: McGraw Hill.

Mallat S. and Zhang Z. (1993) Matching pursuits with time-frequency dictionaries. *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397-3415.

Mallows C.L. (1973) Some comments on Cp, *Technometrics*, vol. 15, no. 4, pp. 661-675.

Mendelson S., Pajor A. and Tomczak-Jaegermann N. (2008) Uniform uncertainty principle for Bernoulli and subgaussian ensembles, *Constr. Approx.*, vol. 28, no. 3, pp. 277-289.

Menke W. (1989) *Geophysical Data Analysis: Discrete Inverse Theory*, Revised Edition (textbook), Academic Press Inc., New York.

Moore E.H. (1920) On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society*, vol. 26, pp. 394-395.

Munk, W.H., Worcester P. and Wunsch C. (1995). *Ocean Acoustic Tomography*. Cambridge: Cambridge University Press. ISBN 0-521-47095-1.

Natarajan B.K. (1995) Sparse approximate solutions to linear systems, *SIAM J. Comput.*, vol. 25, no. 2, pp. 227-234.

Needell D. and Tropp J.A. (2008) CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *ACM Technical Report* 2008-01, California Institute of Technology, Pasadena, July.

Needell D. and Tropp J.A. (2009) Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comp. Harm. Anal.*, vol.26, no.3, pp. 301-321.

Needell D. and Vershynin R. (2009) Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. *Foundations of Comp. Math.*, vol. 9, no. 3, pp. 317-334.

Nesterov Y.E. and Nemirovski A.S. (1994) *Interior Point Polynomial Algorithms in Convex Programming*. SIAM, Philadelphia.

Özen S. (2003) *Topics on channel estimation and equalization for sparse channels with applications to digital TV systems.* PhD thesis, Purdue University.

Özen S., Zoltowski M.D., Fimoff M. (2002) A Novel Channel Estimation Method: Blending Correlation and Least-Squares Based Approaches, *Proceedings of ICASSP 2002*, vol. 3, pp. 2281-2284.

Penrose R. (1955) A generalized inverse for matrices. *Proceedings of the Cambridge Philosophical Society*, vol. 51, no. 3, pp. 406-413.

Proakis J.G. (1995) *Digital Communications*, 3th Edition, McGraw-Hill.

Richardson M.R. and Zandt G. (2009) *Inverse Problems in Geophysics: A Set of Lecture Notes*, Revised and updated Fall 2009, University of Arizona.

Saab R. and Yilmaz O. (2010) Sparse recovery by nonconvex optimization instance optimality, *Appl. Comput. Harmonic Anal.*, vol. 29, no. 1, pp. 30-48.

Şahin A. and Özen S. (2012) Constraint removal for sparse signal recovery, *Elsevier Signal Processing*, vol. 92, no. 4, pp. 1172-1175.

Tibshirani R. (1996) Regression shrinkage and selection via the lasso, *Journal of the Royal Statistical Society. Series B*, vol. 58, no. 1, pp. 267-288.

Tikhonov A.N. and Arsenin V.Y. (1977) *Solutions of Ill-Posed Problems*. New York: Winston. ISBN 0470991240.

Tropp J.A. and Gilbert A.C. (2007) Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Trans. Info. Theory*, vol. 53, no. 12, pp. 4655-4666.

Vogel C.R. (2002) *Computational Methods for Inverse Problems*, The Society for Industrial and Applied Mathematics.

Wahba G. (1977) Practical approximate solutions to linear operator equations when the data are noisy, *SIAM J. Numer. Anal.*, vol. 14, no. 4, pp. 651-667.

Wang M., Polydorides N., and Bertsekas D.P. (2009) *Approximate simulation-based solution of large-scale least squares problems*. LIDS Report, MIT.

APPENDIX A

TEST CHANNELS

The following Tables A.1 and A.2 from Özen (2003) list the DTV test channel delays (time of arrivals) $\{\tau_k\}$ in symbol periods (T), and the relative gains $\{|c_k|\}$.

Table A.1 Simulated channel delays in symbol periods, relative gains for channels 1-6.

Taps	Channel 1		Channel 2		Channel 3	
k	$\{\tau_k\}$	$\{\|c_k\|\}$	$\{\tau_k\}$	$\{\|c_k\|\}$	$\{\tau_k\}$	$\{\|c_k\|\}$
–2			-23.89216836	0.9848		
–1			-1.6143357	0.6457		
0	**0**	**1**	**0**	**1**	**0**	**1**
1	1.6143357	0.6457	32.8248259	0.7456	1.6143357	0.12
2	23.89216836	0.9848	63.06671468	0.8616	23.89216836	0.12
3	32.8248259	0.7456	63.82007134	0.315	32.8248259	0.12
4	63.06671468	0.8616			63.06671468	0.12
5	63.82007134	0.315			63.82007134	0.12

Taps	Channel 4		Channel 5		Channel 6	
k	$\{\tau_k\}$	$\{\|c_k\|\}$	$\{\tau_k\}$	$\{\|c_k\|\}$	$\{\tau_k\}$	$\{\|c_k\|\}$
–2	-23.89216836	0.12				
–1	-1.6143357	0.12			-0.957839182	0.7263
0	**0**	**1**	**0**	**1**	**0**	**1**
1	32.8248259	0.12	3.2286714	0.2512	3.55153854	0.6457
2	63.06671468	0.12	37.667833	0.631	15.25009125	0.9848
3	63.82007134	0.12	47.3538472	0.4467	24.03207745	0.7456
4			102.241261	0.1778	29.16566498	0.8616
5			136.6804226	0.0794		

Table A.2 Simulated channel delays in symbol periods, relative gains for channels 7-12.

Taps	Channel 7		Channel 8		Channel 9	
k	$\{\tau_k\}$	$\{\lvert c_k\rvert\}$	$\{\tau_k\}$	$\{\lvert c_k\rvert\}$	$\{\tau_k\}$	$\{\lvert c_k\rvert\}$
–2						
–1						
0	**0**	**1**	**0**	**1**	**0**	**1**
1	5.16587424	0.65575	10.762238	1	15.541	0.46388
2	22.27783266	0.75697	21.524476	1	28.39	0.54405
3	31.2104902	0.87482			246.66	0.22324
4	61.45237898	1.01565				
5	62.20573564	0.7379				

Taps	Channel 10		Channel 11		Channel 12	
k	$\{\tau_k\}$	$\{\lvert c_k\rvert\}$	$\{\tau_k\}$	$\{\lvert c_k\rvert\}$	$\{\tau_k\}$	$\{\lvert c_k\rvert\}$
–2						
–1	−19.0332	0.12589				
0	**0**	**1**	**0**	**1**	**0**	**1**
1	1.5861	0.1	1.6143357	0.2045	32.8248259	0.1
2	19.0332	0.1	23.89216836	0.15		
3	60.2718	0.31623	32.8248259	0.1		
4	190.332	0.19953	63.06671468	0.2078		
5			63.82007134	0.1509		

VITA

Ahmet Şahin was born on April 4, 1970 in Muğla, Turkey. He received his B.S. degree in Electrical and Electronics engineering from Boğaziçi University, İstanbul in June 1991 and received his M.S. degree in Microelectronics from the University of Southern California, Los Angeles in December 1993. He then worked for the İzmir Institute of Technology as a specialist during its establishment phase until January 1997 and then worked for the University of Ege as a Lecturer until April 1999. He later joined OZ Optics Canada inc., Ottawa, Canada where he worked as a production and quality engineer until July 2000. He continued his career as a research and development engineer in Vestelkom inc., İzmir until January 2006. He then joined Vestek inc., İstanbul as a senior research and development engineer until February 2008. He is currently a Lecturer in İzmir University. His area of interests includes inverse problems in signal processing and communication, video compression and computer vision.