KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT DER TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
Kardinaal Mercierlaan 94, 3001 Leuven (Heverlee)

STRUCTURED TOTAL LEAST SQUARES: ANALYSIS, ALGORITHMS AND APPLICATIONS

Promotoren:
Prof. dr. ir. S. Van Huffel
Prof. dr. ir. B. De Moor

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de toegepaste wetenschappen

door

Philippe LEMMERLING

May 1999

KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT DER TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
Kardinaal Mercierlaan 94, 3001 Leuven (Heverlee)

STRUCTURED TOTAL LEAST SQUARES: ANALYSIS, ALGORITHMS AND APPLICATIONS

Jury:
Prof. dr. ir. J. Delrue, voorzitter
Prof. dr. ir. S. Van Huffel, promotor
Prof. dr. ir. B. De Moor, promotor
Dr. ir. I. Dologlou (ILSP, Greece)
Prof. dr. ir. M. Moonen
Prof. dr. ir. R. Pintelon (ELEC, VUB)
Prof. dr. ir. J. Vandewalle
Prof. dr. ir. M. Van Barel

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de toegepaste wetenschappen

door

Philippe LEMMERLING

U.D.C. 510.5(043.3)

May 1999

D/1999/7515/27

ISBN 90-5682-197-0

Voorwoord

Bij het beëindigen van dit doctoraat, zou ik een aantal mensen willen bedanken.

Vooreerst gaat mijn dank uit naar mijn promotoren Prof. Sabine Van Huffel en Prof. Bart De Moor voor de mogelijkheid die zij mij boden in hun groep te doctoreren.

Sabine Van Huffel wens ik te bedanken vor haar luisterbereidheid en haar nooit aflatend enthousiasme. De vele discussies alsook de talrijke mensen waarmee ze mij in contact bracht, hebben in belangrijke mate bijgedragen tot dit doctoraat.

Bart De Moor zou ik willen bedanken voor de geestdrift waarmee hij als ongeëvenaard redenaar nieuwe ideeën en concepten aanbracht.

Prof. Marc Moonen zou ik willen bedanken voor de belangrijke suggesties die hij gaf i.v.m.een efficiënte implementering van het nieuwe spraakcompressie schema.

Mijn dank gaat ook uit naar Prof. Joos Vandewalle, voor de zorg waarmee hij dit proefschrift op zeer korte tijd heeft nagelezen.

I would like to express my gratitude to Yannis Dologlou for the help and support he provided in the development of the newly proposed speech compression scheme.

Mijn dank gaat ook uit naar de voorzitter van de jury Prof. J. Delrue, alsook naar de overige leden van de jury Prof. Marc Van Barel en Prof. Rik Pintelon, voor hun bereidwillige medewerking.

Voorts wil ik de talrijke SISTA leden bedanken voor de aangename sfeer. Graag bedank ik ook Ida Tassens voor haar hulpvaardigheid waarop ik kon rekenen bij het oplossen van praktische problemen allerhande.

Ik dank tevens het I.W.T. voor de financiële steun.

Tot slot is het duidelijk dat ik bij het voltooien van dit doctoraat niet enkel vanuit wetenschappelijke hoek steun ondervond. De bijdrage van deze mensen is moeilijk onder woorden te brengen. Er bestaat geen beter dankwoord voor hen, dan het teruggeven van –al was het maar een fractie– de liefde en het vertrouwen die zij in mij stellen. Deze thesis draag ik op aan mijn moeder.

Iedere nacht sluip ik buiten
in de hoop dat jij daar straalt.
Iedere nacht opnieuw
blijf ik wachten
maar de leegte antwoordt niet.
Ik tel de sterren, vraag me af
welke de jouwe zou kunnen zijn,
hopend op een flikkering
om de wanhoop uit te doven....
maar de sterren die ik zie
zijn dezelfde als voorheen.
De nacht slaapt in,
tranen van dauw parelen over mijn wangen.
Onverstoorbaar klopt het heelal verder,
tergend traag.
Bij elke hartslag worden duizenden levens
de wereld ingejaagd,
duizenden worden weggeblazen.
En toch is er dat gevoel
dat je hart en ziel met mij verweven zijn,
als watermerk uit het verleden
dat hoopvol stemt in het licht van de toe-
komst.

Philippe Lemmerling
Leuven, Mei 1999

Abstract

This dissertation focuses on the analysis, algorithms and applications for Structured Total Least Squares problems and includes the following original contributions.

- The different existing approaches for Structured Total Least Squares problems, the Constrained Total Least Squares, Structured Total Least Norm and Riemannian SVD approach are proven to be equivalent despite previous claims of inequality in literature. The Structured Total Least Squares problems are situated in a larger framework of so-called Structured Total Approximation Problems. This extended framework envisages other cost functions than the sum of squares as well as nonlinearly structured matrices instead of only considering affinely structured matrices. The Riemannian SVD framework is extended for nonlinearly structured matrices. The theoretical derivation of this nonlinear Structured Total Least Squares approach is illustrated with a numerical example.

- The Structured Total Least Squares problem is shown to be a nonconvex problem, suffering from many local minima. Using existing theoretical results, a new definition for nongeneric Structured Total Least Squares problems is introduced and it is shown that under certain conditions due to this nongenericity, the Structured Total Least Squares problem fails to have a solution. Theory and definitions are illustrated with new examples.
  Furthermore it is shown by means of examples that besides the local minima problem, the Structured Total Least Squares problem might also have several *optimal* solutions.

- A new projection interpretation of the Structured Total Least Squares problem is derived. Using the latter interpretation, it is possible to view Structured Total Least Squares problems as a kind of structured subspace methods, based on newly defined (structure-dependent) signal and noise subspaces.

- The nature of Structured Total Least Squares problems involving Hankel (Toeplitz) matrices is analyzed in detail, by using a specific parametrization of the data in these Hankel (Toeplitz) matrices. The difficulty of the Structured Total Least Squares problem is shown to depend on the parameters of this new parametrization.

- Existing algorithms and alternative algorithms are discussed for the Constrained Total Least Squares and the Riemannian SVD framework. For the Structured Total Least Norm framework a new algorithm is developed. The main improvement consists in the replacement of the previously used penalty function approach by the numerically robust equality constrained least squares approach. Another important improvement, given the problem of the multiple local minima, is the use of subspace based initialization methods. For Hankel (Toeplitz) matrices this is realized by an ESPRIT-type approach, but also for other structures it is possible to use a subspace based initialization method, based on the projection interpretation of the Structured Total Least Squares problem.

- Using a deconvolution example in a medical application called renography, it is demonstrated how efficient algorithms can be developed for a particular Structured Total Least Squares problem. This is done by exploiting the low displacement rank of the matrices involved in the kernel routine.

- A novel speech compression algorithm is proposed. It is a subband based scheme in which the compression of the low frequency band is essentially a Structured Total Least Squares problem. It is shown that this new scheme is related to the class of sinusoidal coders, the main difference being that damped sinusoids are used instead of undamped sinusoids. This way of encoding allows variable rate speech coding. The latter means that without having to re-encode the data, it is possible to shift to a lower bit rate –depending on the available bandwidth and/or the available computational resources– with a gradual decrease of the audible quality.

- A previously proposed framework that incorporates both the prediction error models and behavioural models is shown to be very similar to the Structured Total Least Squares problem. A novel optimization approach for this framework is derived. The latter approach provides many insight, e.g. that the constraints imposed on the model parameters have a severe impact on the obtained result. It also allows one to actually perform experiments in this new framework, using of-the-shelf optimization routines.

Notation

The current section lists the symbols and acronyms that occur frequently in this thesis. The last column refers to the page on which the symbol or acronym is first used and/or defined.

List of symbols

A^T	transpose of A	
A^H	Hermitean transpose of A	
A^{-1}	inverse of A	
$A^\dagger$	Moore-Penrose pseudo-inverse of A	page 39
$A \otimes B$	Kronecker product of A and B	page 58
$arctan(x)$	inverse tangent of x	page 4
$diag(v)$	diagonal matrix with the entries of vector v as diagonal entries	page 26
E_n	$n \times n$ anti-identity matrix	page 62
i.i.d.	independently identically distributed	page 3
$0_{m\times n}$	$m \times n$ zero matrix	page 104
I_n	$n \times n$ identity matrix	page 28
$vec(A)$	column wise vectorization of a matrix A	page 58

List of acronyms

AR	Auto Regressive	page 115
BTLS	Bootstrapped Total Least Squares	page 84
CB	Code Book	page 116
CELP	code excited linear prediction	page 115
CTLS	Constrained Total Least Squares	page 21
DOA	Direction-of-Arrival	page 8
ECLS	Equality Constrained Least Squares	page 89

EIV	Errors-In-Variables	page 2
EVD	EigenValue Decomposition	page 29
HF	High-Frequency	page 115
IQML	Iterative Quadratic Maximum Likelihood	page 79
ISE	Iterative Signal Enhancement	page 123
LF	Low-Frequency	page 115
LS	Least Squares	page 2
LSI	Latent Semantic Indexing	page 11
LTI	Linear Time-Invariant	page 129
MA	Moving Average	page 139
ML	Maximum Likelihood	page 2
NMR	Nuclear Magnetic Resonance	page 7
PEM	Prediction Error Modeling	page 129
QP	Quadratic Programming	page 88
RiSVD	Riemannian Singular Value Decomposition	page 21
RSVD	Restricted Singular Value Decomposition	page 28
SNR	Signal-to-Noise Ratio	page 95
STLN	Structured Total Least Norm	page 21
TLS	Total Least Squares	page 2
ULA	Uniform Linear Array	page 9
STAP	Structured TotalApproximation Problems	page 21
STLS	Structured Total Least Squares	page 3
VQ	Vector Quantization	page 115

Contents

Gestructureerde totale kleinste kwadraten problemen: analyse, algoritmen en toepassingen

Nederlandse samenvatting

Hoofdstuk1: Inleiding

Lineaire modellen spelen een belangrijke rol in talrijke disciplines: signaal verwerking, systeem theorie, economie, biomedische toepassingen,... Een bijzondere klasse van methoden voor het bepalen van de parameters van dergelijke modellen zijn de Gestructureerde totale kleinste kwadraten methoden (GTKK). GTKK methoden zijn een natuurlijke uitbreiding van de gewone totale kleinste kwadraten (TKK) methode. Zoals de gewone TKK methode is de GTKK methode een manier om lineare modellen te fitten op basis van opgemeten data die in een datamatrix opgeslagen worden. In de praktijk hebben deze datamatrices meestal een bijzondere structuur, waarmee rekening dient gehouden te worden in de methode die gebruikt wordt om een lineair model op te stellen. Dit is precies wat de GTKK methode doet: het in rekening brengen van de structuur van de opgemeten datamatrices zodat lineaire modellen nauwkeuriger bepaald kunnen worden.
Het proefschrift bevat drie luiken: analyse, algoritmen en toepassingen. Het eerste luik beschrijft de aard van het GTKK probleem, alsook interpretaties en eigenschappen van het GTKK probleem. De GTKK methode wordt geïnterpreteerd als een deelruimte gebaseerde methode waarbij gebruik gemaakt wordt van structuur-afhankelijke signaal-en ruisdeelruimten.
Het tweede luik van het proefschrift gaat nader in op mogelijke algoritmen

om het GTKK probleem op te lossen. Na een kort overzicht van bestaande (suboptimale) algoritmen wordt een verbeterd algoritme voor het oplossen van GTKK problemen voorgesteld. Een belangrijke bijdrage hierin is de ontwikkeling van methoden die betere startwaarden leveren alsook de vervanging van de kernroutine door een kleinste kwadraten probleem met gelijkheidsbeperkingen. Tevens worden efficiënte algoritmen voorgesteld voor Hankel/Toeplitz GTKK problemen gebaseerd op de lage verschuivingsrang van de optredende matrices. Het derde luik van het proefschrift beschrijft enkele concrete toepassingen. De efficiëntie van de in het vorige luik ontwikkelde algoritmen voor Hankel/Toeplitz GTKK problemen wordt geïllustreerd aan de hand van een deconvolutie probleem dat optreedt in de medische renografie. Een tweede toepassing die in detail uitgewerkt wordt is een nieuwe spraakcompressie techniek gebaseerd op de GTKK methode. Aangetoond wordt dat deze methode verwant is aan de zogenaamde sinusoïdale coders, met als belangrijk pluspunt dat de nieuw ontwikkelde techniek toelaat aan variabele bitsnelheid compressie te doen. Tot slot wordt er een veralgemeend kader voorgesteld voor de identificatie van lineaire tijdsinvariante dynamische systemen. Het veralgemeend kader verenigt twee belangrijke strekkingen in de moderne identificatie literatuur, namelijk de predictiefoutenmethode en de gedragsgebaseerde benadering. Een nieuwe optimalisatie benadering van dit veralgemeend kader leidt tot interessante interpretaties en een algoritme om de modelselectieprocedure uit te voeren. Dit laatste stelt ons in staat het potentieel van het veralgemeend kader te illustreren aan de hand van enkele voorbeelden. De structuur van de thesis wordt weergegeven in figuur .

Hoofdstuk 2: Inleiding tot GTKK problemen

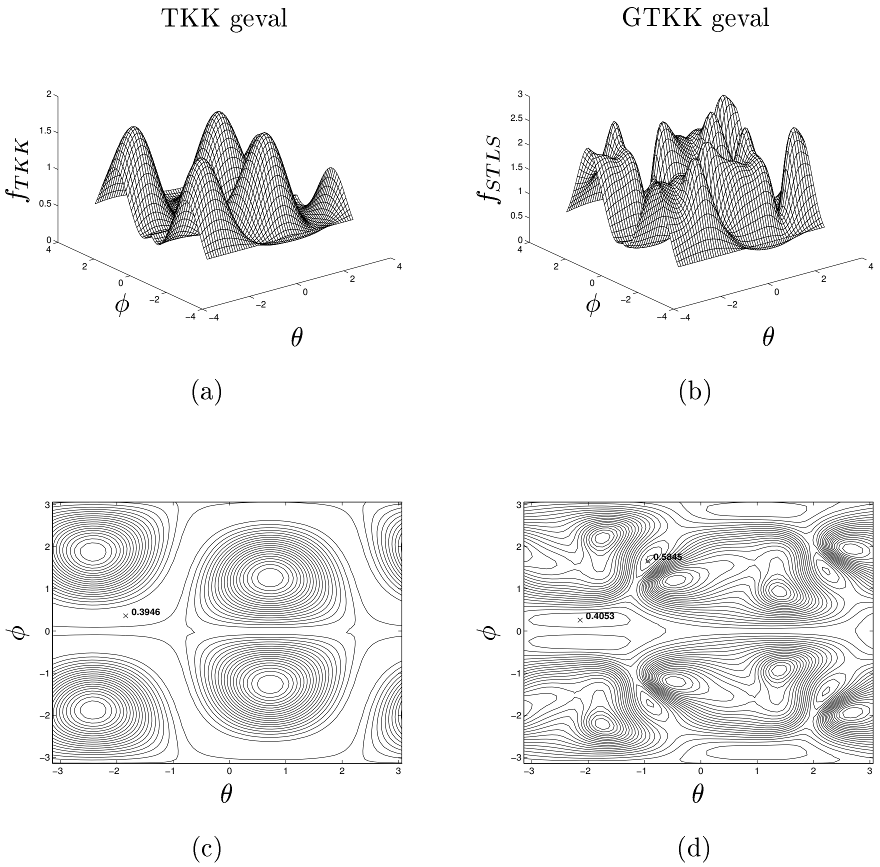
Zoals reeds vermeld en nogmaals aangegeven hieronder, zijn GTKK problemen een natuurlijke uitbreiding van de gewone TKK problemen.

$\min_{\Delta A, \Delta b, x} \|[\Delta A \;\; \Delta b]\|_F^2$

zodat $(A + \Delta A)x = b + \Delta b$ **TKK**

zodat $[A + \Delta A \;\; b + \Delta b]$ en $[A \;\; b]$ eenzelfde affiene structuur hebben **GTKK**

Het is mogelijk om zowel het gewone TKK probleem als het GTKK probleem te formuleren als een optimalisatie probleem zonder beperkingen met de respectievelijke kostfuncties f_{TKK} en f_{GTKK}. In figuur 0.1 wordt van beide functies het functieverloop en de bijhorende contouren weergegeven. Het betreft hier een klein voorbeeldje waarbij $A \in \mathbb{R}^{3\times 2}$ en $b \in \mathbb{R}^{3\times 1}$, waarbij $[A\ b]$ de Hankelstructuur heeft. De figuren aan de linkerzijde geven de functiewaarden (figuur (a)) en bijhorende contourlijnen (figuur (c)) weer van de kostfunctie f_{TKK} die in het TKK geval geminimaliseerd dient te worden. De figuren aan de rechterzijde geven de functiewaarden (figuur (b)) en bijhorende contourlijnen (figuur (d)) weer van de kostfunctie f_{GTKK} die in het GTKK geval geminimaliseerd dient te worden. De gegevens komen uit voorbeeld 2.1.1. Het valt meteen op dat zelfs in dit kleine voorbeeld, de kostfunctie van het GTKK probleem meerdere locale minima heeft in tegenstelling tot het gewone TKK probleem (bemerk dat we om redenen van symmetrie slechts 1 kwadrant hoeven te inspecteren). In dit



Figuur 0.1: *De figuren aan de linkerzijde geven de functiewaarden (figuur (a)) en bijhorende contourlijnen (figuur (c)) weer van de kostfunctie f_{TKK} die in het TKK geval geminimaliseerd dient te worden. De figuren aan de rechterzijde geven de functiewaarden (figuur (b)) en bijhorende contourlijnen (figuur (d)) weer van de kostfunctie f_{GTKK} die in het GTKK geval geminimaliseerd dient te worden. De gegevens komen uit voorbeeld 2.1.1.*

hoofdstuk wordt tevens een overzicht gegeven van de belangrijkste bestaande benaderingen van het GTKK probleem: de Constrained Total Least Squares (CTLS) benadering [2, 3], de Riemanniaanse Singulierewaarden Ontbinding (RiSVD) benadering [66, 67] en de Structured Total Least Norm (STLN) benadering [81, 83, 95]. Ondanks andersluidende beweringen in de literatuur, tonen we in sectie 2.3 aan dat al deze benaderingen onder milde voorwaarden equivalent zijn. Verder wordt het GTKK probleem in het groter kader van de Gestructureerde Totale BenaderingsProblemen (GTBP) geplaatst. Dit kader omvat naast de GTKK problemen uitbreidingen naar niet-lineair gestructureerde matrices, kostfuncties die verschillen van de gewogen som van kwadraten die we in het GTKK probleem terugvinden en uitbreidingen van voorgaande problemen naar hun multivariabele tegenhangers. De uitbreiding naar niet-lineaire structuren wordt in sectie 2.4.2 uitgewerkt voor het RiSVD kader.

Hoofdstuk 2: Eigenschappen van GTKK problemen

In dit hoofdstuk onderzoeken we eerst onder welke omstandigheden GTKK geen oplossing hebben. Het startpunt is de volgende algemene formulering van het GTKK probleem:

$$\begin{aligned} &\min_{\Delta s, x} \Delta s^T W \Delta s \\ &\text{zodat } (A + \Delta A)x = b + \Delta b \\ &\text{en } [A + \Delta A \ b + \Delta b] \text{ dezelfde structuur heeft als } [A \ b], \end{aligned} \tag{0.1}$$

waarbij $\Delta s \in \mathbb{R}^{k\times 1}$ enkel de *verschillende* elementen van de correctiematrix $\Delta A \in \mathbb{R}^{m\times n}$ en de correctievector $\Delta b \in \mathbb{R}^{m\times 1}$ bevat. Op analoge wijze wordt $s \in \mathbb{R}^{k\times 1}$ gedefinieerd als de vector die enkel de *verschillende* elementen van $[A \ b]$ bevat. Het is eenvoudig voorbeelden te vinden waarbij het GTKK probleem (0.1) geen oplossing heeft maar het volgende gelijkaardige probleem wel:

$$\begin{aligned} &\min_{\Delta s, y} \Delta s^T W \Delta s \\ &\text{zodat } [A + \Delta A \ b + \Delta b]y = 0, y^T y = 1 \\ &\text{en } [A + \Delta A \ b + \Delta b] \text{ dezelfde structuur heeft als } [A \ b]. \end{aligned} \tag{0.2}$$

Op basis van voorgaande probleemformuleringen introduceren we niet-generische GTKK problemen.

Definitie 0.0.1 *Een GTKK probleem (0.1) wordt niet-generisch genoemd wanneer alle optimale*[1] *oplossingen van het overeenkomstige probleem (0.2) voldoen*

[1] de optimale oplossing is niet uniek zoals verderdoor vermeld.

aan $y(n+1) = 0$.

In volgende stelling wordt het verband gelegd tussen de definitie van niet-generische problemen en het ontbreken van een oplossing voor (0.1).

Stelling 0.0.2 *Stel dat* $\mathbf{M}(s) = [A\ b]$ *een regulier*[2] *affien gestructureerde matrix is. Indien (0.1) niet-generisch is, dan heeft het ook geen oplossing.*

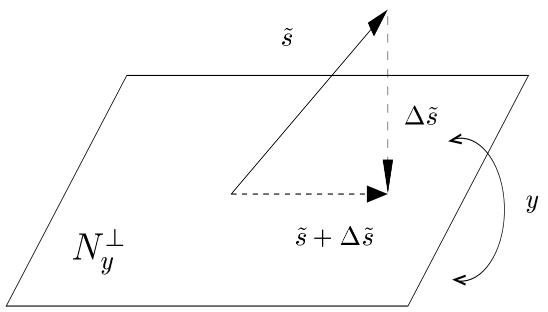
Aan de hand van enkele (nieuwe) voorbeelden tonen we aan dat een GTKK probleem meerdere optimale oplossingen kan hebben.
Een belangrijk resultaat van dit hoofdstuk is de projectie interpretatie van GTKK problemen. De sleutelformule van deze interpretatie is:

$$\tilde{s} + \Delta\tilde{s} = (I - \tilde{\mathbf{H}}_{\mathbf{2}}(y)^T(\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}(y)^T)^{-1}\tilde{\mathbf{H}}_{\mathbf{2}}(y))\tilde{s},$$

waarbij $\tilde{\mathbf{H}}_{\mathbf{2}}(y) \in \mathbb{R}^{m\times k}$ een matrix is die lineair afhangt van y (dit is dezelfde y als in formulering (0.2), aangezien we in het verdere verloop van de thesis enkel generische GTKK problemen beschouwen), $\Delta\tilde{s} = W^{0.5}\Delta s$ en $\tilde{s} = W^{0.5}s$. Laten we een nieuwe ruisdeelruimte definiëren N_y als de ruimte opgespannen door de kolommen van $\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T$. Het orthogonaal complement van de vorige deelruimte is meteen de definitie voor de nieuwe signaaldeelruimte $N_y^{\perp}$. Belangrijk is hierbij op te merken dat de matrix $\tilde{\mathbf{H}}_{\mathbf{2}}(y)$ niet enkel van y afhangt maar vooral ook van de structuur die dient behouden te worden in het GTKK probleem. Bijgevolg zijn de nieuw gedefinieerde signaal- en ruisdeelruimten structuur-afhankelijk. We zien nu dat het oplossen van een GTKK probleem als volgt geformuleerd kan worden (zie figuur 0.2):
vind de parametervector y *zodat de projectie van* $\tilde{s}$ *op de signaaldeelruimte* $N_y^{\perp}$ *zo dicht mogelijk (in* L_2 *norm) bij het originele signaal* $\tilde{s}$ *ligt.*

Door gebruik te maken van de projectie interpretatie is het mogelijk een bovengrens te bepalen voor de waarde van doelfunctie uit (0.2) in de oplossing van formulering (0.2). Dergelijke bovengrens is van praktisch nut in tal van toepassingen aangezien ze een idee geeft van de te verwachten nauwkeurigheid (in de zin “hoe dicht ligt $[A + \Delta A\ \ b + \Delta b]$ in de oplossing van (0.2) bij de originele matrix $[A\ b]$”) van de oplossing van het GTKK probleem. Voor de volledigheid geven we ook de bijhorende ondergrens. De theorie i.v.m. de projectie interpretatie wordt geïllustreerd aan de hand van verschillende voorbeelden. Aansluitend worden ook problemen i.v.m. de inverteerbaarheid van $\mathbf{D}_{\mathbf{2}}(y) \equiv \tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}(y)^T$ onderzocht, aangezien deze matrix een belangrijke rol speelt in de projectie interpretatie alsook op tal van andere plaatsen. De problemen worden in 2 klassen ingedeeld en er wordt nader ingegaan op het geval van een symmetrisch Toeplitz matrix. Hiervoor tonen we aan dat de inverteerbaarheid van $\mathbf{D}_{\mathbf{2}}(y)$ steeds een probleem vormt, maar dat het mogelijk is dit

[2] Voor een definitie zie definitie 3.1.4; het dient gezegd te worden dat de veel voorkomende matrices zoals Hankel- en Toeplitz matrices regulier gestructureerd zijn.



Figuur 0.2: *Deze figuur illustreert de projectie interpretatie van een GTKK probleem. Het GTKK probleem oplossen komt neer op het bepalen van een parametervector y zodat het verschil tussen $\tilde{s}$ en diens projectie op de signaal-deelruimte $N_y^{\perp}$ zo klein mogelijk is in L_2 norm.*

probleem –door de vervanging van $\mathbf{D_2}(y)$ voor symmetrische Toeplitz matrices door $\mathbf{D_2}(y)$ voor gewone Toeplitz matrices– te omzeilen zonder de uiteindelijke oplossing te wijzigen.
Het laatste deel van dit hoofdstuk bevat een gedetailleerde analyse van het GTKK probleem voor het geval de datamatrices een (blok)Hankel of een (blok)Toeplitz structuur hebben. Door over te gaan naar een nieuwe parametrisatie wordt dit speciale GTKK probleem omgezet in een niet-lineair kleinste kwadraten probleem, waarin de originele meetgegevens zo goed mogelijk benaderd dienen te worden door een som van complex gedempte exponentiëlen. We tonen aan dat de waarde van de parameters (dit zijn dus de amplituden, fasen, dempingen en frequenties van de aanwezige componenten) van deze complex gedempte exponentiëlen in de oplossing van het GTKK probleem bepalend zijn voor de moeilijkheidsgraad van het beschouwde GTKK probleem.
Tot slot vermelden we ook nog enkele orthogonaliteits en statistische eigenschappen.

Hoofdstuk 4: Algoritmen voor het oplossen van GTKK problemen

Dit hoofdstuk begint met een overzicht en analyse van 2 vaak gebruikte suboptimale algoritmen voor het oplossen van GTKK problemen: Cadzow's algoritme (algoritme 4.1.1) en het IQML algoritme (algoritme 4.1.2). Voor laatsgenoemd algoritme wordt aangetoond waarom het suboptimaal is voor het oplossen van GTKK problemen en wat de gevolgen hiervan zijn op de statistische nauwkeurigheid van de bekomen resultaten.

Het hoofdstuk vervolgt met een analyse van bestaande algoritmen voor het CTLS en RiSVD kader evenals uitbreidingen hierop. We verkiezen verder te werken met het STLN kader omdat dit het dichtst bij de originele GTKK formulering staat. De andere kaders lossen niet de originele GTKK formulering op maar een hieruit afgeleide formulering. Bijgevolg is het ook duidelijk dat in het STLN kader de structuur van de betrokken matrices beter behouden blijft, aangezien de opgeloste formulering niet afwijkt van de originele GTKK formulering. Dit laatste is van belang wanneer efficiënte algoritmen ontwikkeld dienen te worden: hoe complexer de structuur van de betrokken matrices, hoe moeilijker het zal zijn hun structuur uit te buiten.
Voor het STLN kader ontwikkelen we nieuwe algoritmen, nl. algoritme 4.4.1 en 4.4.2. Het tweede is afgeleid uit het eerste met als voornaamste verbetering een toename van de convergentie snelheid door het gebruik van de exacte Hessiaan (uiteraard tegen een hogere rekenkundige kostprijs). De voornaamste verbetering van algoritme 4.4.1 t.o.v. bestaande STLN algoritmen is

1. het gebruik van een kleinste kwadraten probleem met gelijkheidsbeperkingen als kernroutine (deze aanpak is numeriek robuuster dan de voorheen gebruikte "penalty function" benadering)

2. het gebruik van betere startwaarden.

Zoals zojuist aangegeven ontwikkelen we in dit hoofdstuk tevens nieuwe methoden voor het berekenen van betere startwaarden. Deze algoritmen zijn eveneens gebaseerd op de projectie interpretatie van het GTKK probleem. Hun efficiëntie wordt aangetoond met een aantal voorbeelden.

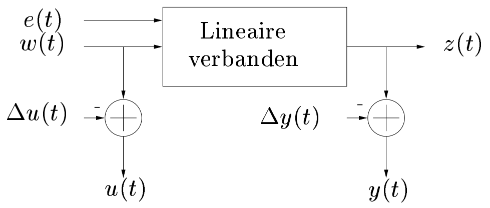
Hoofdstuk 5: Efficiënte implementering van het deconvolutie probleem

Voor het algoritme 4.4.1, dat beschreven wordt in hoofdstuk 4, wordt een efficiënte implementering uitgewerkt voor het bijzondere GTKK probleem dat optreedt in het deconvolutieprobleem. De efficiënte algoritmen maken gebruik van de lage verschuivingsrang van de matrices die optreden in het GTKK probleem. Op deze manier wordt een algoritme bekomen met een complexiteit van $O(mn + n^2)$ flops per iteratie (alle GTKK algoritmen zijn iteratief, een logisch gevolg van de niet-convexiteit van het GTKK probleem), waarbij m en n bepaald worden door de dimensies van de beschouwde datamatrix $[A \;\; b] \in \mathbb{R}^{m\times(n+1)}$. Met een eenvoudig deconvolutie voorbeeld wordt de efficiëntie van deze aanpak met die van een niet geoptimaliseerde verie ($O((m + n)^3)$ flops per iteratie) en een eerder ontwikkelde verbetering [83] ($O(mn^2 + m^2)$ flops per iteratie) vergeleken.

De efficiënte implementatie van algoritme 4.4.1 wordt nogmaals geïllustreerd aan de hand van het bijzondere deconvolutieprobleem dat optreedt in de medische renografie. Het is hierbij de bedoeling de retentiefunctie –d.i. de impulsresponsie van de nier beschouwd als lineair systeem– te bekomen. Deze retentiefunctie visualiseert de gemiddelde transittijd van een eenheid van een radioactieve tracer in de nier onder beschouwing. Dit laatste laat de geneesheer toe de nierfunctie of disfunctie te evalueren na een transplantatie. Aan de hand van dit voorbeeld wordt eveneens de grotere statistische nauwkeurigheid van de GTKK methode t.o.v. andere schattingsmethoden zoals de gewone TKK methode, bevestigd.

Hoofdstuk 6: Toepassing in spraakcompressie

In dit hoofdsstuk beschrijven we een nieuwe subbandgebaseerde spraakcompressie methode. De methode is werkt bloksgewijs op het inkomend spraaksignaal. De motivering achter de opsplitsing van elk inkomend blok in een hoog-en laagfrequent deel berust op de observatie dat het laagfrequente deel goed te modelleren is m.b.v. een lineair model. Het hoogfrequente deel van een blok vertoont eerder een "niet-lineair karakter". Uit voorgaande blijkt duidelijk dat we de GTKK methode kunnen gebruiken voor het bepalen van het model van het laagfrequente deel. Dit gebeurt dus niet door het bepalen van een klassiek autoregressief (AR) model met bijhorende residus, maar wel door het bepalen van een minimale correctie op het blok zodat het gecorrigeerde signaal exact voldoet aan een AR model. Het voordeel in termen van spraakcompressie is duidelijk: we hoeven enkel de initiële waarden van het blok en de bijhorende modelparameters door te sturen om aan de ontvangstzijde het benaderde signaal perfect te reconstrueren. Het is dus niet nodig bijhorende residus (aangezien die er niet zijn) mee te sturen zoals bij compressietechnieken die gebaseerd zijn op de klassieke AR modelbenadering. Het GTKK probleem dat we beschouwen voor het laagfrequente deel van een inkomend blok bevat een Hankel matrix als de te benaderen matrix. Aangezien we uit hoofdstuk 3 het verband tussen dit GTKK probleem en de parametrisatie van het blok m.b.v. een som van complex gedempte exponentiëlen kennen is het duidelijk dat de methode toegepast op het laagfrequente deel gerelateerd is met de klasse van de zogenaamde sinusoïdale coders. Het belangrijkste verschil is dat dit nieuwe schema een som van *gedempte* sinusoïden gebruikt i.p.v. een som van ongedempte sinusoïden zoals in het geval van de bestaande sinusoïdale coders. Voor het hoogfrequente deel van het nieuwe schema wordt gebruik gemaakt van bestaande vectorquantizatiemethoden.
Het nieuwe schema wordt aan de hand van een spraaksignaal vergeleken met een standaard methode uit de spraakcompressie nl. de Federal Standard 1016 4800 bps CELP methode. Aan de hand van een wiskundige maat (de segmentele signaalruisverhouding) wordt aangetoond dat het nieuwe schema tegen een



Figuur 0.3: *Deze figuur illustreert de nieuwe modelset van lineaire tijdsinvariante dynamische systemen. De waargenomen gegevens bestaan uit $u(t)$ en $y(t)$. Er wordt verondersteld dat deze meetgegevens ruis bevatten zodat $w(t) = u(t) + \Delta u(t)$ en $z(t) = y(t) + \Delta y(t)$ (waarbij $z(t)$ en $w(t)$ de niet-waarneembare gegevens voorstellen). Voorts is er ook nog de niet-gemeten ingang $e(t)$ die ook gebruikt wordt om voor een stuk $z(t)$ te "verklaren".*

bitsnelheid van ongeveer 4800 bits per seconde (nagenoeg gelijk aan die van de CELP methode dus) een vergelijkbare performantie heeft als de CELP methode. Uit de analyse van de FFT spectra van inkomende en gecomprimeerde signalen van beide methoden blijkt tevens hun gelijkaardige kwaliteit.

Hoofdstuk 6: Misfit versus latency

In dit laatste hoofdstuk wordt er een recentelijk geïntroduceerd [69] veralgemeend kader voorgesteld [69] voor de identificatie van lineaire tijdsinvariante dynamische systemen. Het veralgemeende kader verenigt twee belangrijke strekkingen in de moderne identificatie literatuur, namelijk de predictiefoutenmethode en de gedragsgebaseerde benadering. Het basisidee van de nieuwe benadering wordt weergegeven in figuur 0.3. Terwijl de predictiefoutenmethode enkel een "latency" ingang $e(t)$ veronderstelt en geen "misfit" op de ingang ($\Delta u(t)$) noch op de uitgang ($\Delta y(t)$), is dit in de gedragsgebaseerde benadering net omgekeerd. Aangezien de statistische eigenschappen van $e(t)$, $\Delta u(t)$ en $\Delta y(t)$ grondig kunnen verschillen, is het een logische stap om een onderscheid te maken tussen deze grootheden en ze alledrie (met een verschillende weging) te beschouwen zoals in het veralgemeend kader gebeurt. Door gebruik te maken van de techniek van Lagrange vermenigvuldigers wordt aangetoond dat de modelselectieprocedure voor modellen uit dit veralgemeend kader, sterk verwant is aan de GTKK problemen aangezien deze procedure herschreven kan worden als een set van Riemanniaanse SWO vergelijkingen. Er worden tevens een aantal eigenschappen van dit veralgemeend kader bewezen.
Een nieuwe optimalisatiebenadering van dit veralgemeend kader leidt tot interessante interpretaties en een algoritme om de modelselectieprocedure uit te voeren. Op die manier zijn we in staat de modelselectieprocedure uit te testen

op een aantal voorbeelden. Hieruit blijkt het grote potentieel van dit veralgemeend kader: indien de statistische eigenschappen van de “misfit” en “latency” termen gekend zijn, kan het resultaat van klassieke identificatiemethoden sterk verbeterd worden.

Hoofdstuk 8: Besluit

In dit proefschrift werden de analyse, algoritmen en toepassingen van GTKK problemen behandeld. We geven nu kort de belangrijkste besluiten en eigen bijdragen weer.

- Er werd bewezen dat de verschillende benaderingen van GTKK problemen: de Constrained Total Least Squares benadering, de Structured Total Least Norm benadering en de Riemanniaanse SWO benadering equivalent zijn ondanks andersluidende beweringen in de literatuur. De GTKK problemen worden geplaatst in het ruimer kader van Gestructureerde Totale BenaderingsProblemen. De uitbreidingen bevatten andere kostfuncties dan de gewogen som van kwadraten in het GTKK probleem (bv. L_p norm benaderingen met $p \neq 2$), niet-lineair gestructureerde matrices en multivariabele problemen. De uitbreiding naar niet-lineaire structuren werd uitgewerkt en bewezen voor de Riemanniaanse SWO benadering. De theorie werd aan de hand van een Vandermonde systeem geïllustreerd.

- Er werd aangetoond dat het GTKK probleem een niet-convex probleem is, met vele locale minima als gevolg. Gebruik makend van bestaande theoretische resultaten werd de definitie van niet-generische GTKK problemen ingevoerd. Deze definitie liet toe om te bewijzen dat onder bepaalde omstandigheden, niet-generische GTKK problemen geen oplossing hebben. Theorie en definities werden gestaafd aan de hand van nieuwe voorbeelden.
  Eveneens werd er met behulp van enkele voorbeelden aangetoond dat GTKK problemen soms meerdere *optimale* oplossingen hebben.

- Een projectie interpretatie van het GTKK probleem werd afgeleid. Deze interpretatie liet toe het GTKK probleem te interpreteren als een deelruimte gebaseerde methode met structuur-afhankelijke signaal-en ruisdeelruimten. De theorie werd geïllustreerd aan de hand van een Hankel matrix en een ongestructureerde matrix.

- De projectie interpretatie leidde tot een bovengrens voor de kostfunctie (gewogen som van kwadraten) in het GTKK probleem. Deze bovengrens is van belang in talrijke toepassingen. Voor de volledigheid werd ook een bijhorende ondergrens afgeleid. Beide grenzen werden geïllustreerd aan de hand van enkele voorbeelden.

- Problemen i.v.m. de inverteerbaarheid van $\mathbf{D_2}(y)$ werden besproken, aangezien laatstgenoemde matrix een belangrijke rol speelt in de projectie interpretatie alsook in vercheidene andere formuleringen. Aangetoond werd dat de problemen in 2 klassen onderverdeeld kunnen worden. Het geval van symmetrische Toeplitz matrices werd in detail behandeld. Er werd aangetoond dat er bij dit type matrices steeds inverteerbaarheidsproblemen optreden. Vervolgens werd ook aangetoond dat –voor dit speciale geval– problemen vermeden kunnen worden door de matrix $\mathbf{D_2}(y)$ te gebruiken die hoort bij de gewone Toeplitzstructuur, zonder dat hierdoor de oplossing van het GTKK probleem gewijzigd wordt.

- De eigenschappen van GTKK voor Hankel/Toeplitz matrices werden in detail onderzocht, door gebruik te maken van een nieuwe parametrisatie van dit speciale GTKK probleem. Uit deze nieuwe parametrisatie kon afgeleid worden dat de moeilijkheidsgraad van het GTKK probleem afhangt van de waarden van de parameters –van deze nieuwe parametrisatie– in de oplossing van het GTKK probleem.

- Er werd een overzicht gegeven van bestaande algoritmen en uitbreidingen hierop en dit voor de Constrained Total Least Squares benadering en de Riemanniaanse SWO benadering. Voor de Structured Total Least Norm benadering werden nieuwe algoritmen ontwikkeld. De belangrijkste verbetering bestaat erin dat de vroegere "penalty function" benadering vervangen werd door een kleinste kwadraten probleem met gelijkheidsbeperkingen (hiervoor bestaat numeriek robuuste software). Een andere belangrijke verbetering is de ontwikkeling van algoritmen die betere startwaarden leveren. Deze algoritmen zijn gebaseerd op de projectie interpretatie van het GTKK probleem. Het effect van deze verbeteringen werd geïllustreerd met enkele voorbeelden.

- Een deconvolutieprobleem dat optreedt in de medische renografie werd gebruikt om aan te tonen hoe het gebruik van de lage verschuivingsrang van de betrokken matrices kan leiden tot een efficiënte implementering van algoritmen voor het oplossen van GTKK problemen.

- Een nieuw spraakcompressie schema werd voorgesteld. Het gaat om een subbandgebaseerd schema waarbij de compressie in de laagfrequentie band overeenkomt met een GTKK probleem. Het verband met de klasse van sinusoïdale coders werd aangegeven. Een belangrijk verschil met deze laatste klasse is dat het nieuwe schema gebruik maakt van een som van gedempte sinusoïden i.p.v. een som van ongedempte sinusoïden. Een belangrijk voordeel van dit soort schema's, is de mogelijkheid om aan variabele bitsnelheid compressie te doen. Dat laatste betekent dat zonder de gegevens opnieuw te moeten comprimeren, er overgeschakeld kan worden naar lagere bitsnelheden –afhankelijk van de beschikbare bandbreedte en/of de rekenkracht beschikbaar aan de ontvangstzijde– met een geleidelijke verslechtering van de verstaanbaarheid.

- Een recentelijk geïntroduceerd kader [69] dat twee belangrijke strekkingen in de moderne identificatie literatuur – namelijk de predictiefoutenmethode en de gedragsgebaseerde benadering– verenigt, werd voorgesteld. Het verband tussen de modelselectieprocedure van dit nieuwe kader en de GTKK problemen werd aangetoond. Een nieuwe optimalisatie benadering van dit veralgemeend kader werd bewezen en leidde tot interessante interpretaties alsook een algoritme om de modelselectieprocedure uit te voeren. Aan de hand van enkele voorbeelden kon op die manier het potentieel van dit veralgemeend kader aangetoond worden.

GTKK problemen zijn nog relatief jong. Hierna beschrijven we een aantal mogelijkheden voor verder onderzoek.

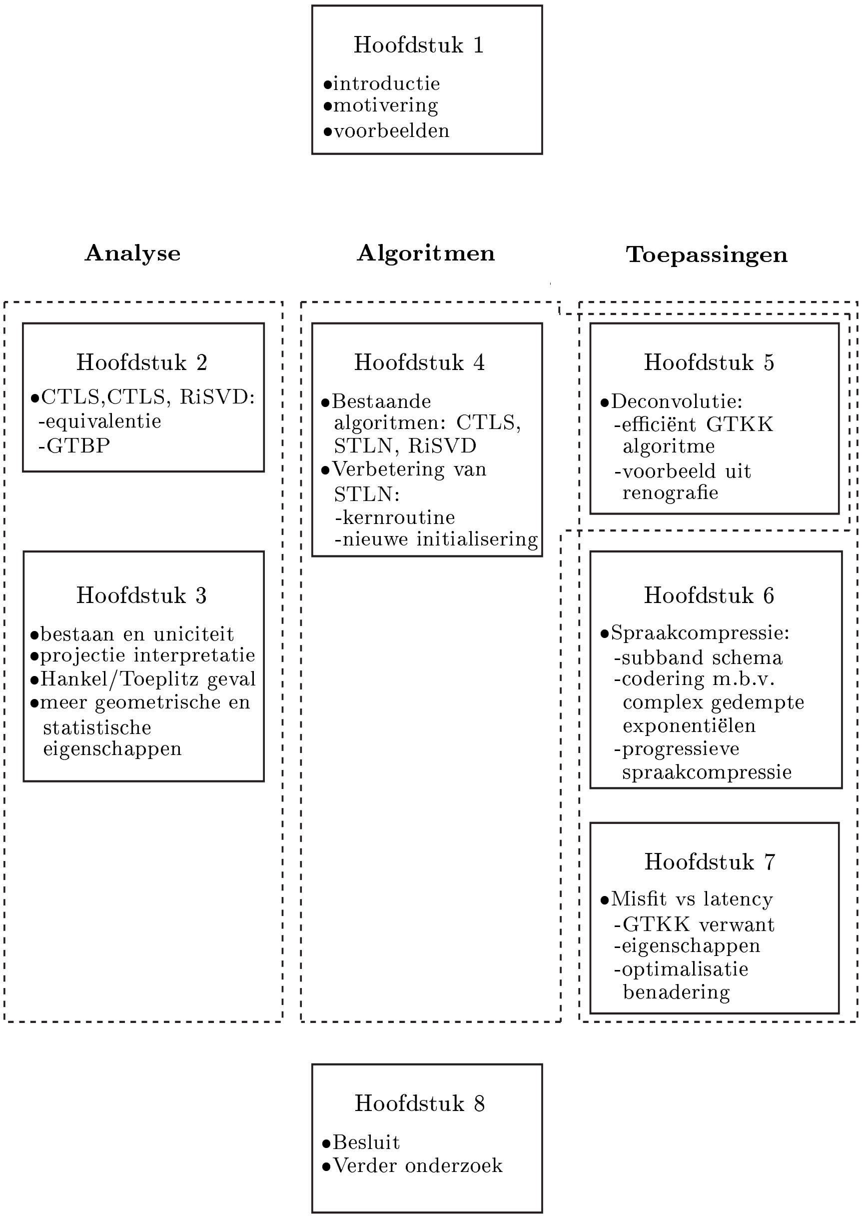
- In dit proefschrift werd in hoofdzaak de L_2 norm formulering (de "kleinste kwadraten" in de term GTKK) behandeld. Andere L_p normen met $p \neq 2$ zijn aangewezen in bepaalde statistische omstandigheden (bv. de aanwezigheid van zogeheten "outliers"). Deze GTBP alsook bijhorende algoritmen zijn grotendeels nog niet onderzocht.
  Een nog algemenere aanpak zou erin bestaan $\|\Delta s\|$ in de GTBP formulering (2.16) te vervangen door f waarbij f de waarschijnlijkheidsdichtheidsfunctie van de elementen van Δs voorstelt.

- Ondanks het feit dat in het proefschrift de theorie voor de niet-lineaire uitbreiding van het GTKK probleem in het RiSVD kader uitgewerkt werd, is het nog onduidelijk onder welke omstandigheden de bekomen statistische schatter optimaal is. Een verband met hogere orde statistiek ligt voor de hand.

- Een interessante uitbreiding van het GTKK probleem is die waarbij een regularisatie term (bv. $\lambda\|x\|$) toegevoegd wordt aan de kostfunctie in (2.16). Interessante resultaten in dit verband werden reeds gepresenteerd in [33, 34, 102] voor het zogeheten "regularized Constrained Total Least Squares" probleem.

- In het hoofdstuk over spraakcompressie kunnen ongetwijfeld betere perceptuele resultaten bekomen worden door de kostfunctie in het GTKK probleem (som van kwadraten) te vervangen door een kostfunctie die beter de perceptuele kwaliteit van het resultaat weergeeft. Een andere optie is het toepassen van voor- of nabewerkingen die de specifieke eigenschappen van het menselijk gehoor uitbuiten.
  Tenslotte leidt het geen twijfel dat een meer geavanceerde quantiseringsmethode (i.p.v. het huidige eenvoudige schema dat een vast aantal bits toekent aan de verschillende parameters) tot een kwalitatief beter resultaat zou leiden voor eenzelfde bitsnelheid of of tot een lagere bitsnelheid voor dezelde kwalitieit.

- In hoofdstuk 7 werd aangetoond dat het veralgemeend "misfit versus latency" kader in staat is de statistische nauwkeurigheid van het geschatte model aanzienlijk te verbeteren. De keuze van de gewichten α, β en γ was eerder intuïtief. Om de mogelijkheden van het veralgemeend kader verder te onderzoeken, is een grondige statistische analyse noodzakelijk (voor resultaten i.v.m. een aantal speciale gevallen van het veralgemeend kader verwijzen we naar [101]).
  Naast deze statistische analyse loont het zeker de moeite volgende punten verder te onderzoeken:
  - Heeft het zin de gewichten α, β en γ te kiezen zodat een stabiel (andere eigenschappen die variëren langsheen de "root locus" komen hier uiteraard ook voor in aanmerking) systeem bekomen wordt?
  - Kan de bepaling van modellen voor verschillende waarden van het drietal (α, β, γ) informatie verschaffen over de (ongekende) "ruisomstandigheden"?



Figuur 0.4: *Structuur van het proefschrift.*

Chapter 1

Introduction

This chapter situates the STLS problem among other linear data fitting techniques such as the Least Squares and Total Least Squares approach. The STLS approach is motivated from a statistical point of view as well as from a practical point of view (section 1.1). The omnipresence of STLS problems in applications is illustrated in section 1.2 with several examples. Finally a chapter by chapter overview is provided in section 1.3 and some notations that will be used throughout the thesis are introduced in section 1.4.

1.1 Introduction

Linear models play an important role in a broad class of disciplines and related applications such as signal processing, control theory, system theory, statistics, economics, biomedical applications etc. In this thesis we analyze a specific method for estimating the parameters that characterize these linear models, we develop algorithms for solving the new approach and show its use in applications.

As stated before, linear models are our starting point. These models can be described in the following way

$$\alpha(1)x(1) + \alpha(2)x(2) + \ldots + \alpha(n)x(n) = \beta \tag{1.1}$$

with $x \in \mathbb{R}^{n\times 1}$ the parameter vector and $\alpha \in \mathbb{R}^{n\times 1}$ the observed variables. In an experiment, typically many measurements are performed, say m, all of which are supposed to satisfy relation (1.1). This leads to an overdetermined set of equations

$$\sum_{j=1}^{n} A(i,j)x(j) \approx b(i), i = 1, \ldots, m \tag{1.2}$$

with $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^{m\times 1}$ and $m > n$ or in matrix form:

$$Ax \approx b. \tag{1.3}$$

The first methods for solving this type of problems date back to the beginning of the 19th century, when Gauss [41] described the well-known Least Squares (LS) method. In the LS approach a solution for the overdetermined system of equations (1.3) is found by solving the following unconstrained optimization problem:

$$\min_x \|Ax - b\|_2. \tag{1.4}$$

Although the Maximum Likelihood (ML) method is usually associated to Fisher [35], it is clear that Gauss already knew that under certain circumstances, the LS method had ML properties, as is shown by the following quotation from [41]:

> "Therefore, that will be the most probable system of values ... if the same degree of accuracy is to be presumed in all the observations."

In the quotation above, the word "observations" refers only to the elements of the vector b in (1.3) and not to the entries of A. Indeed, as can be seen from (1.4), only b is perturbed in order to make the system of equations $Ax \approx b$ consistent. The latter statistical interpretation of the LS problem also reveals its shortcomings: in many applications not only the observed variables $b(i), i = 1, \ldots, n$, but also the observed variables $A(i,j), i = 1, \ldots, m,\ j = 1, \ldots n$ are affected by errors. The latter observation led to an approach known as Total Least Squares (TLS) in the field of numerical analysis , or also as Errors-In-Variables (EIV) regression or orthogonal regression in statistics. The TLS approach allows errors on the entries of b, but also on the entries of A. It can be formulated as the following constrained optimization problem:

$$\min_{\Delta A, \Delta b, x} \|[\Delta A\ \Delta b]\|_F \tag{1.5}$$
$$\text{such that } (A + \Delta A)x = b + \Delta b.$$

A TLS approach for the univariate case ($n = 1$) was already known at the end of the 19th century [6], but it was mainly the discovery of the link between TLS and the Singular Value Decomposition (SVD) [45, 92] that increased the insight in TLS problems and also led to robust SVD-based algorithms, allowing the use of TLS in real-life applications. The algebraic, algorithmic and geometrical properties of the TLS problem as well as extensions towards so-called non-generic TLS problems were extensively studied in [96]. As shown in [96], the TLS approach has ML properties under certain assumptions. In order to specify these assumptions, a statistical framework is now presented. Let A and b contain measurements of the true unobservable values in $A_0 \in \mathbb{R}^{m\times n}$ and $b_0 \in \mathbb{R}^{n\times 1}$. Furthermore let

$$\begin{aligned} A_0 x_0 &= b_0 \\ A &= A_0 + \Delta A_n \\ b &= b_0 + \Delta b_n, \end{aligned}$$

in which $\Delta A_n \in \mathbb{R}^{m\times n}$ and $\Delta b_n \in \mathbb{R}^{n\times 1}$ contain the "measurement" errors on the elements of A_0 and b_0. In [96] it is then shown that x at the solution of (1.5) is a ML estimator for x_0 when the rows of $[\Delta A_n \ \Delta b_n]$ are independently identically distributed (i.i.d.) with common zero mean vector and common covariance matrix that is (an unknown) multiple of the identity matrix I_{n+1}. However, these noise conditions are often violated in real-life situations, when there exist linear relations between the true unobserved variables represented by the entries $[A_0 \ b_0]$ as well as between the observed variables represented by the entries of $[A \ b]$. Mostly the linear relations between the entries of $[A_0 \ b_0]$ are the same as those between the entries of $[A \ b]$. Thus, at least intuitively, it is clear that –in case the *different* entries of $[\Delta A_n \ \Delta b_n]$ are i.i.d. zero mean noise of equal (unknown) variance– a ML estimate of x can only be obtained when the entries of $[\Delta A \ \Delta b]$ in (1.5) obey the same linear relations as the entries of $[A \ b]$ and the entries of $[A + \Delta A \ b + \Delta b]$. An example of such linear relations is a Toeplitz matrix in which the linear relations are a mathematical way of expressing that all the elements on a diagonal are equal. Another way of saying that there exist linear relations between the elements of a matrix $S \equiv [A \ b]$, is by stating that S can be written as a linear combination of given fixed basis matrices:

$$S = t(1)T_1 + \dots t(k)T_k, \tag{1.6}$$

with $T_i \in \mathbb{R}^{m\times(n+1)}, i = 1, \dots, k$ the fixed basis matrices and $t \in \mathbb{R}^{k\times 1}$. Note that an extension of TLS towards a particular affine [1] structure, namely the case of several error free columns, was already discussed in [96]. The Structured Total Least Squares (STLS) approach studied in this thesis is a further generalization of the TLS approach in that direction. The STLS problem can

[1] Note that affine refers to the addition of a constant basis matrix T_0 to the right-hand side in (1.6), without multiplication factor. This extension (from linear to affine) is trivial in the sense that if S is an affine combination of $T_i, i = 0, \dots, k$, then $S - T_0$ is a linear combination of $T_i, i = 1, \dots, k$

be formulated as follows:

$$\begin{aligned}
&\min_{\Delta A, \Delta b, x} \|[\Delta A \ \Delta b]\|_F^2 \qquad (1.7)\\
&\text{such that } (A+\Delta A)x = b + \Delta b\\
&\text{and } [A+\Delta A \ b+\Delta b] \text{ has the same affine structure as } [A \ b].
\end{aligned}$$

Although (1.7) is not the most general formulation of the STLS approach, it clearly indicates the similarities and the differences w.r.t. the TLS approach in (1.5): both approaches are constrained optimization problems, the only difference being that the STLS approach adds an additional constraint on the structure of the matrices involved.

Before continuing, we give a geometric interpretation of the different methods for solving systems of overdetermined equations that we mentioned before: LS, TLS and STLS. Since we want to give a graphical impression of the difference, we consider the following simple (single variable) linear regressions [2] case:

$$A = [6 \ -2 \ -5 \ -8]^T \text{ and } b = [5 \ 6 \ -2 \ -5]^T,$$

where it should be noted that $S \equiv [A \ b]$ is a Toeplitz matrix. In this simple linear regression case, solving the overdetermined system of equations $Ax \approx b$ is geometrically equivalent to fitting a straight line (through the origin) with slope $arctan(x)$ to the points $(A(i), b(i)), i = 1, \ldots, 4$ (indicated by a "+" sign in figure 1.1), such that a specific cost function is minimal and additional constraints –if present– are satisfied. Accordingly, new points $(A(i)+\Delta A(i), b(i)+\Delta b(i)), i = 1, \ldots, 4$ (indicated by a "x" sign in figure 1.1) that lie on the straight line, are obtained.

As indicated in (1.4) the LS problem is solved by determining x such that

$$\sum_{i=1}^{4} (A(i,1)x - b(i))^2$$

is minimal. In figure 1.1(a) we see that in geometrical terms LS determines the slope such that the sum of the squared vertical distances (indicated by dotted lines) is minimized.

When we consider the problem (1.5) in the simple linear regression case, we see that the objective function becomes:

$$\sum_{i=1}^{4} \Delta A(i,1)^2 + \Delta b(i,1)^2 \qquad (1.8)$$

whereas the constraint in (1.5) simply states that the points $(A(i)+\Delta A(i), b(i)+\Delta b(i)), i = 1, \ldots, 4$ have to lie on a straight line. It is possible to reformulate

[2] by simple (or single variable) linear regression, we indiciate that A contains only 1 column

(1.5) as an unconstrained optimization problem, giving in the simple linear regression case:

$$\min_x \sum_{i=1}^{m} \frac{(A(i,1)x - b(i))^2}{1+x^2},$$

which means that we determine the slope of the straight line running through the origin, such that the sum of the squared orthogonal distance (indicated by the dotted lines in figure 1.1(b)) of the points $(A(i,1), b(i)), i = 1, \ldots, 4$ to the straight line is minimal.
The STLS approach is similar to the TLS approach in the sense that we also minimize (1.8) such that the points $(A(i) + \Delta A(i), b(i) + \Delta b(i)), i = 1, \ldots, 4$ lie on a straight line, but in addition we impose the additional constraint that $[\Delta A \ \Delta b]$ has to be Toeplitz. The latter constraint is imposed by the following linear relations between the entries of $[\Delta A \ \Delta b]$

$$\Delta A(i) = \Delta b(i+1), i = 1, \ldots, 3.$$

The latter constraint is clearly satisfied as can be seen in figure 1.1(c) by comparing the consecutive horizontal (i.e. $\Delta A(i)$) dotted lines and vertical (i.e. $\Delta b(i)$) dotted lines. Figure 1.1(d) shows the estimated straight lines for the three different methods: LS(dash-dotted line), TLS (dashed line) and STLS (full line).
In the previous paragraphs, the STLS aproach was mainly motivated by using statistical arguments. Like the LS and the TLS approach, the STLS approach can prove useful outside this statistical framework. As a matter of fact, in the case of LS, this was already noticed at the very beginning by Gauss [41]:

> "In conclusion, the principle that the sum of the squares of the differences between the observed and computed quantities must be minimum may, in the following manner, be considered independently of the calculus of probabilities."

This use of statistically sound methods from a practical point of view that has nothing to do with the statistical framework in which the method is a ML method, occurs in e.g. system identification. Prediction error models [63] are then obtained as a result of a trade-off between sensitivity w.r.t. output measurements on one hand and rapidly decaying effects of erroneous initial conditions on the other hand. So, although prediction error methods do have ML properties in certain noise circumstances, prediction error models can also be derived using practical arguments rather than statistical ones. The same is true for the STLS approach. A novel speech compression method that is based on this particular use of STLS will be developed in chapter 6. Having

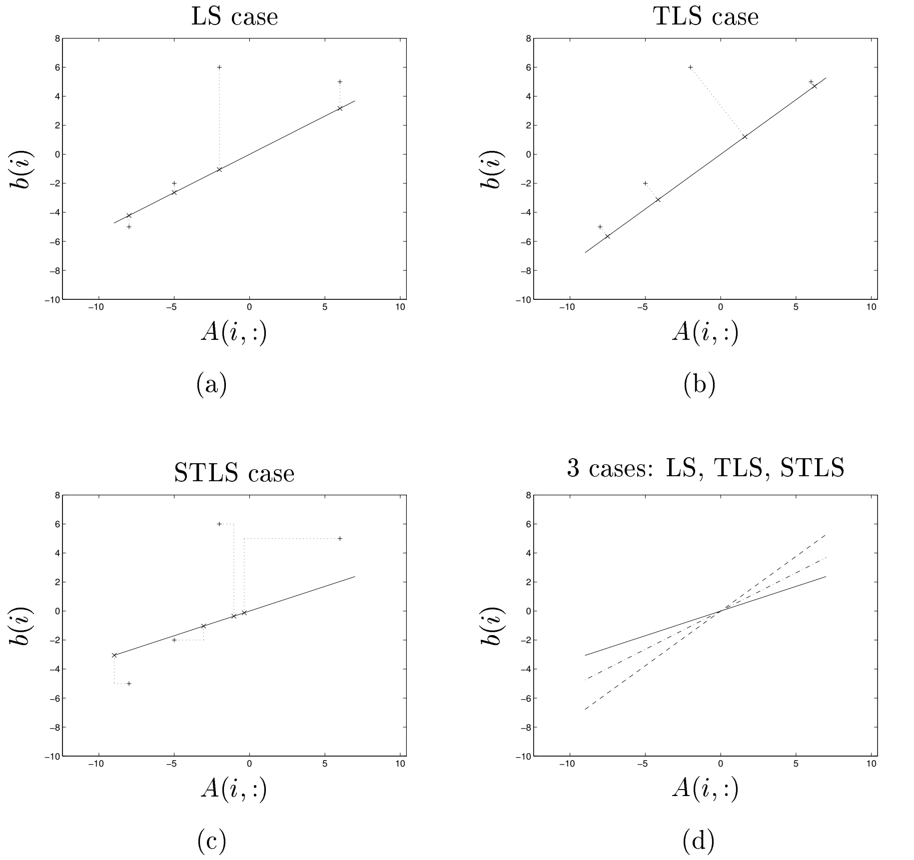


Figure 1.1: *These figures give a geometrical interpretation of the different methods for solving a system of overdetermined equations.*

introduced this new motivation for the use of the STLS approach, we will briefly compare the TLS and STLS approach, but from a purely algebraic point of view. Looking at (1.5), it is clear that the TLS approach represents a way to find the closest (in Frobenius norm) rank deficient matrix $[A + \Delta A \; b + \Delta b]$ to a given data matrix $[A \; b]$. As explained before, the solution of the TLS approach relies on the use of the SVD. In fact, the correction $[\Delta A \; \Delta b]$ is obtained by truncating the SVD of $[A \; b]$. However, this SVD truncation operation is not structure preserving in the sense that the matrix $[A + \Delta A \; b + \Delta b]$ will typically be unstructured, even if $[A \; b]$ is structured. It turns out that in many applications the data matrix $[A \; b]$ is structured either because the matrix has a particular interpretation (e.g. it might be the covariance matrix of a stationary signal and thus Toeplitz) or simply because one constructed the matrix by assigning the different elements of a signal vector to the entries of the matrix $[A\ b]$ (e.g. a Hankel matrix is constructed by storing an impulse response in the first column and last row of $S = [A\ b]$, thereby defining the entire Hankel matrix). In those applications that start from a structured matrix $[A\ b]$, it is often desirable to obtain a rank-deficient matrix $[A + \Delta A\ b + \Delta b]$ with $[\Delta A\ \Delta b]$ as small as possible and of the *same structure* as the original data matrix S. This is exactly what STLS does from an algebraic point of view.

1.2 STLS in applications

To give an idea of the large amount of fields and related applications in which the STLS problem constitutes the kernel problem, three examples are described in some detail in the next three subsections. The last subsection presents more examples together with references from which it can easily be seen that the mentioned applications contain indeed STLS problems as the kernel problem that needs to be solved. The fact that applications figure in this section does not imply that some STLS-type algorithm has been used in the quoted references. It simply means that the kernel problem of the application is truly an STLS problem.

1.2.1 Nuclear Magnetic Resonance spectroscopy

The first example concerns a medical application called Nuclear Magnetic Resonance (NMR) spectroscopy. NMR spectroscopy is a frequently used non-invasive medical diagnosis tool. Somewhat simplified (for a comprehensive treatment of NMR spectroscopy see [40, 32]), the basic principle of NMR spectroscopy can be explained as follows: atomic nuclei in a static magnetic field can be in 2 different spin or energy states. Application of an oscillating magnetic field causes the transition of nuclei to the high-energy state. After removing the oscillating magnetic field, the population of nuclei returns to its equilibrium state, thereby emitting the previously absorbed energy as weak electromagnetic radiation. The latter signal can be modeled (in the time-domain) as a sum of complex damped exponentials perturbed by i.i.d. complex Gaussian noise:

$$s(l) = \sum_{k=1}^{K} a_k e^{jp_k} e^{(j2\pi f_k + d_k) l \Delta t} + v(l), l = 1, \ldots, N, \tag{1.9}$$

with a_k the amplitude, f_k the frequency, p_k the phase and d_k the damping of the kth component, Δt the sampling interval and $v \in \mathbb{C}^{N \times 1}$ a vector of zero-mean i.i.d. Gaussian noise samples. Every spectral component or term in (1.9) corresponds to a different chemical substance or metabolite present in the volume element from which the signal was obtained. The amplitude a_k is the most important parameter since it is directly proportional to the concentration of the corresponding metabolite (i.e. the metabolite associated to the resonance frequency f_k) in the volume element under investigation. Therefore, if signals are obtained from several adjacent volume elements in a cross section of e.g. a patient's head, it is possible to produce an image that visualizes the concentration of a specific metabolite over the cross section. As will be shown in chapter 3, a signal consisting of a sum of K complex damped exponentials, satisfies a linear prediction equation of order K. Writing out the equations for the different time instants we obtain the following system of equations:

$$Ax \approx b \tag{1.10}$$

$$A = \begin{bmatrix} s(1) & s(2) & \dots & s(K) \\ s(2) & s(3) & \dots & s(K+1) \\ \vdots & \vdots & & \vdots \\ s(N-K) & s(N-K+1) & \dots & s(N-1) \end{bmatrix}, b = \begin{bmatrix} s(K+1) \\ s(K+2) \\ \vdots \\ s(N) \end{bmatrix},$$

which is not consistent because the signal s is a sum of complex damped exponentials perturbed by i.i.d. additive complex Gaussian noise. In order to remove the noise and make (1.10) consistent, it is intuitively obvious (for a more detailed explanation see [27]) that the correction $[\Delta A \;\; \Delta b]$ applied to $[A \;\; b]$ should have the same Hankel structure. If not, it could mean e.g. that the entries on the fourth anti-diagonal of $[\Delta A \;\; \Delta b]$ are different although they all represent the "perturbation" to one and the same element $s(4)$. Due to the i.i.d. complex Gaussian noise, a ML estimate of x (and therefore also ML estimates of the parameters $a_k, p_k, f_k, d_k, k = 1, \dots, K$, since they can be determined from x and s in a straightforward way, see e.g. [93]) is obtained by solving the following problem:

$$\begin{aligned} &\min_{\Delta s} \Delta s^H \Delta s \\ &\text{such that } (A + \Delta A)x = b + \Delta b \\ &\text{and } \;\; [A + \Delta A \;\; b + \Delta b] \text{ has the same Hankel structure as } [A \;\; b]. \end{aligned}$$

where $\Delta s(l), l = 1, \dots N$ represents the correction to $s(l), l = 1, \dots, N$. Clearly, the latter kernel problem of the NMR spectroscopy application is an STLS problem.

1.2.2 Direction-of-Arrival problem

In a sense this problem is much related to the previous NMR example. By that we mean that whereas the previous NMR application is a problem in the time-domain, the Direction-of-Arrival (DOA) problem is its spatial counterpart. The DOA problem is omnipresent in applications such as radar, sonar, seismic reflection prospection, wireless communication, ... For a comprehensive treatment of the DOA problem we refer to [50, 21]. The problem setting is depicted in figure 1.2: m narrow-band wavefronts impinge on an array of n sensors. First consider a noise free environment and let $y_i(j)$ be the signal received by the jth sensor at time i. For the moment suppose also that there is only one source in the far field. Due to the narrow-band characteristic of the wavefront, the signal received by sensor 2 is a phase-shifted version of the signal received by sensor 1, the phase shift being

$$\phi_1 = 2\pi \frac{d}{\lambda} sin(\theta_1),$$

where θ_1 is the angle of incidence (w.r.t. the normal to the array), λ is the wavelength of the propagating disturbance and d is the distance between sensor 1 and 2. We will consider a Uniform Linear Array (ULA) , meaning that the array consists of equidistant sensors, linearly aligned and having the same response characteristics. The latter phase shift relation then holds for all consecutive sensors. Returning to the scenario of m impinging wavefronts and assuming that the sensors are perturbed by additive i.i.d. complex Gaussian noise, we obtain the following model equations for the output of the different sensors:

$$y_i = As_i + v_i, i = 1, \dots, q, \tag{1.11}$$

where

$$A = \begin{bmatrix} 1 & 1 & \dots & 1 \\ e^{j\phi_1} & e^{j\phi_2} & & e^{j\phi_m} \\ \vdots & \vdots & & \vdots \\ e^{j(n-1)\phi_1} & e^{j(n-1)\phi_2} & \dots & e^{j(n-1)\phi_m} \end{bmatrix}$$

is the so-called array phase steering matrix, $s_i \in \mathbb{C}^{m \times 1}$ contains the complex amplitudes of the m wavefronts impinging on the array at time instant i, $y_i \in \mathbb{C}^{m \times 1}$ contains the outputs of the m sensors at time i, $v_i \in \mathbb{C}^{n \times 1}$ contains i.i.d. complex Gaussian noise and q is the number of snapshots. From the equations (1.11), it is clear that this problem is similar to the previous example, only now the framework is a spatial one, instead of a temporal one. As a matter of fact the output of the sensors at 1 time instant are sums of complex undamped exponentials and instead of consecutive time samples which satisfy a prediction equation, we now have consecutive spatial samples (the outputs of the consecutive sensors at one time instant) which satisfy a prediction equation. As indicated in [3] this gives rise to the following set of forward-backward linear prediction equations:

$$Cp \approx 0, C = [C_1^T \ \dots C_q^T]^T \in \mathbb{C}^{2(n-m)q \times (m+1)}$$

and

$$C_k = \begin{bmatrix} y_k(1) & y_k(2) & \dots & y_k(m+1) \\ y_k(2) & y_k(3) & \dots & y_k(m+2) \\ \vdots & \vdots & & \vdots \\ y_k(n-m) & y_k(n-m+1) & \dots & y_k(n) \\ \bar{y}_k(m+1) & \bar{y}_k(m) & \dots & \bar{y}_k(1) \\ \bar{y}_k(m+2) & \bar{y}_k(m+1) & \dots & \bar{y}_k(2) \\ \vdots & \vdots & & \vdots \\ \bar{y}_k(n) & \bar{y}_k(n-1) & \dots & \bar{y}_k(n-m) \end{bmatrix} \in \mathbb{C}^{2(n-m)\times(m+1)},$$

where $\bar{y}$ denotes the complex conjugate of y. Intuitively it is clear that if all the sensors are perturbed by additive i.i.d. complex Gaussian noise, a ML estimate of p (and thus also of the angles $\phi_i, i = 1, \dots, m$, since the quantities $e^{j\phi_i}, i = 1, \dots, m$ are the roots of the following polynomial in $\zeta : p(1) + p(2)\zeta + \dots + p(m+1)\zeta^m = 0$) is obtained by solving the following problem:

$$\begin{aligned} &\min_{\Delta C, p} \Delta c^H \Delta c \\ &\text{such that } (C + \Delta C)p = 0 \\ &\text{and } \; C + \Delta C \text{ has the same structure as } C \end{aligned} \tag{1.12}$$

where $\Delta c \in \mathbb{R}^{2(n-m)(m+1)\times 1}$ contains the different entries of ΔC (for more mathematical arguments we refer again to [3]).

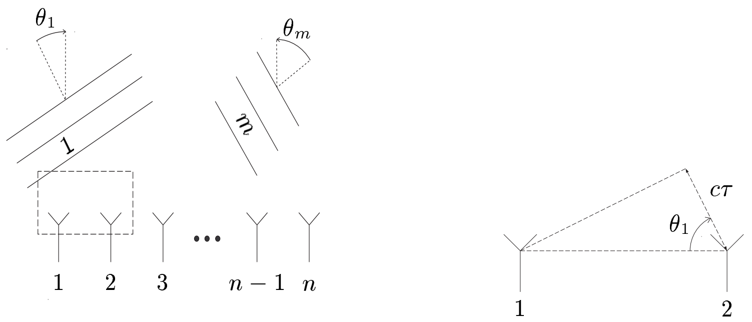


Figure 1.2: *This figure shows the setting for the DOA problem: m narrow-band wavefronts impinge on an array of n sensors. The figure on the right is an enlargement of the rectangular area indicated in the figure on the left.*

1.2.3 Information retrieval

The third example we consider is an application in information retrieval. The exponentially growing amount of electronically available documents creates the need for efficient and reliable techniques to retrieve information from large databases. Basic search engines retrieve information by literally matching terms in a user's query with terms extracted from the documents in the database. This approach has two major defaults. First of all, in most cases there exist different words to describe one and the same subject. Unexperienced users tend to forget these synonyms, thereby missing a great deal of relevant information. On the other hand, some words might have different meanings depending on the context in which they appear. This is called polysemy and results in the retrieval of documents that are of no interest to the user but nevertheless lexically match the query. Latent Semantic Indexing (LSI) [28, 13, 15] attempts to circumvent these problems by determining a model that captures the latent semantic structure in the associations of terms with documents. Somewhat simplified, the latter means that the LSI approach groups the documents rather by concept than by looking for single words that appear in them. Typically we start from a term-document matrix $A \in \mathbb{R}^{m\times n}$ in which $A(i,j)$ is the frequency that term (or word) i occurs in document j. As shown in e.g. [28], the latent semantic structure is obtained via a truncated SVD of A:

$$A_k \equiv U(:,1:k)\Sigma(1:k,1:k)V(:,1:k)^T,$$

where $A = U\Sigma V^T, U \in \mathbb{R}^{m\times m}, \Sigma \in \mathbb{R}^{m\times n}, V \in \mathbb{R}^{n\times n}$ represents the SVD of A. From a geometrical point of view the rows of $V(1:k,:)$ can be seen as a representation of the n documents in a k dimensional space, whereas the rows of $U(1:k,:)$ can be seen as a representations of the n terms in a k dimensional space. $\Sigma(1:k,1:k)$ then represents the latent semantic structure by mapping points of the "reduced" document space onto the "reduced" term space. Let $q \in \mathbb{R}^{m\times 1}$ stand for a query, with $q(i)$ the frequency that the ith term occurs in the query. To find the closest match within the latent semantic model, an inverse transformation (from the "reduced" term space to the "reduced" document space) is performed on q (note that a query is similar to an element in term space):

$$d = \Sigma(1:k,1:k)^{-1}U(:,1:k)^T q.$$

Note that the premultiplication of q with $U(1:k,:)^T$ first maps the query onto the "reduced" term space and next a point in the "reduced" document space is obtained via the transformation matrix $\Sigma(1:k,1:k)$. The actual search consists of comparing the angles between d and the rows of $V(1:k,:)$. The row with the smallest angle w.r.t. d is ranked highest in the output of the search result.

As shown in [51, 52] it is of course still possible that due to e.g. polysemy the LSI approach yields undesired documents. In [51, 52] it is demonstrated that the efficiency of the LSI approach can further be increased by perturbing the reduced term-document matrix A_k in a particular way. The problem solved is the following:

$$\begin{aligned} & \min_{y,B_k} \|A_k - B_k\|_F \\ & \text{such that } B_k y = 0 \\ & \text{and perturbations are only allowed on row } i \text{ and in certain columns} \end{aligned} \tag{1.13}$$

Referring to the previously mentioned problem of polysemy, row i corresponds to a term in the user's query having different meanings in different contexts (e.g. the term "depression" can appear with different meanings in the context of bussiness recession, tectonics, mental disorders, ...). The columns of row i that are allowed to change in (1.13) are those that relate to documents belonging to contexts in which the user is not interested. Experiments have shown that the STLS problem (1.13) considerably improves the efficiency of LSI.

1.2.4 Further examples

Further examples of applications in which STLS problems appear are briefly mentioned below. The fact that they figure in this list does not mean implicitly that the problem has been solved as an STLS problem.

- Blind identification and equalization of channels based on second-order statistics [90, 62].
- Blind estimation of non-minimum phase ARMA systems using fourth order statistics [16].
- Restoration of degraded images (spatial degradation caused by blurring) [65].
- Gradient-based optical flow estimation [72].
- Target classification of elastic targets by the analysis of their resonant response [1].
- Estimation of the DOA of wideband cyclostationary signals in which estimates of the carrier frequency and symbol rate are available [5].
- Estimation of 2-dimensional frequencies of superimposed sinusoids in Gaussian noise [4].
- Improvement of ESPRIT-type (Estimation of Signal Parameters via Rotational Invariance Techniques) high-resolution techniques using STLS [48, 49].

- Total Least Squares with elementwise weights (see [66] and the references therein)
- The noisy realization problem [67].
- H_2 model reduction [70], frequency weighted Hilbert-Schmidt-Hankel model reduction [85, 84].
- The calculation of the essential matrix, which plays a central role in the structure and motion from motion problem in computer vision [26].

1.3 Chapter by chapter overview

The structure of the thesis is presented in figure 1.3. Chapter 1 introduces the STLS problem and compares it with other linear data fitting techniques such as the Least Squares and Total Least Squares approach. The STLS approach is motivated from a statistical point of view as well as from a practical point of view. The omnipresence of STLS problems in applications is illustrated with several examples. Finally a chapter by chapter overview is provided and some notations that will be used throughout the thesis are introduced.

Chapters 2 to 7 are situated on three levels: analysis (chapter 2 and 3), algorithms (chapter 4 and 5) and applications (chapter 5, 6 and 7) of STLS problems.

Chapter 2 starts by showing that the STLS approach is more than just a simple extension of the TLS problem. An overview of existing approaches for the STLS problem is given: the CTLS approach, the STLN approach and the RiSVD approach. It is proven that, despite misleading claims in literature, the three approaches are equivalent under mild conditions. Further extensions of the STLS problem towards multivariate STLS problems, nonlinearly structured matrices,... are brought together in the Structured Total Approximation Problem framework. The extension of STLS towards nonlinearly structured matrices is derived for the RiSVD framework and confirmed by a numerical example.

Chapter 3 concerns properties and interpretations of the univariate STLS problem. The existence of a solution to the STLS problem is considered. By introducing the definition of nongeneric STLS problems, it is shown that under mild conditions, these nongeneric STLS problems fail to have a solution. The theory is illustrated by some examples. Furthermore, the non-uniqueness of the solution is illustrated by means of several examples.
A projection interpretation of the STLS problem is developed. This interpretation leads to a new definition of structure-dependent signal and noise

subspaces, whereby the STLS approach can be seen as a subspace approach involving structure-dependent subspaces. Using the geometrical insight provided by the projection interpretation an upperbound to the STLS objective function (weighted sum of squares) is derived. For reasons of completeness a lower bound to the same function is added as well. The structured-dependent subspace interpretation is illustrated on a Hankel matrix example and on an unstructured matrix example. The usefulness of the derived bounds is demonstrated using several examples. In addition to this projection interpretation, the invertibility of the matrix $\mathbf{D_2}(y)$, appearing in the projection interpretation as well as in many other places (e.g. RiSVD framework), is addressed. The theory is illustrated by several examples. For one of them (symmetric Toeplitz matrix), it is proven in detail where the invertibility problem comes from and how it can be circumvented in STLS problems involving this particular structure.
The second last section of chapter 3 analyzes the STLS problem involving Hankel matrices in detail. The analysis proceeds by using a new parametrization of the set of rank deficient Hankel matrices. It is shown how the difficulty of solving this particular STLS problem is related to the parameters of the new parametrization at the solution of the STLS problem. The analysis can straightforwardly be extended to Toeplitz matrices, block Hankel matrices and block Toeplitz matrices.
Finally some orthogonality properties of the STLS solution as wel as a ML interpretation of the STLS problem are given.

Chapter 4 describes algorithms for solving the STLS problem. First, two well-known suboptimal algorithms, namely Cadzow's algorithm and IQML, are discussed and shown to be suboptimal. Next, an overview of existing algorithms and possible alternatives in the CTLS and RiSVD framework is presented. An improved algorithm for the STLN framework is proposed. The preference for this framework is inspired by the fact that the latter framework stands closest in formulation to the original STLS problem formulation. Intuitively it should be clear that this makes it easier to exploit the structure of the involved matrices and thus develop efficient algorithms. One of the major improvements of the newly proposed algorithm for the STLN framework, is the introduction of new initialization methods (first for Hankel (Toeplitz) matrices and further on for an arbitrarily affine structure). The improvement consists of the fact that the final result obtained with an STLS algorithm when using this new initialization is of a better quality and reached in less iterations. The other major improvement of the new algorithm is the replacement of the penalty function approach used previously by an equality constrained least squares approach.

In chapter 5 a fast algorithm for the basic deconvolution problem is developed in order to show how an efficient STLS algorithm can be obtained once a particular problem or application has been specified. A simulation experiment based on a medical application (whose kernel problem is the deconvolution problem) in renography is described. This example is used to demonstrate the improved efficiency of the new algorithm compared to other algorithms for solving this type of STLS problem. By means of a Monte-Carlo simulation, using several noise levels, we show the improved statistical accuracy of the deconvolution results obtained with the STLS estimator as compared to other estimators such as the TLS estimator that do not impose a structure on the correction matrix $[\Delta A \;\; \Delta b]$.

Chapter 6 describes a novel speech compression scheme. It is a subband based coding scheme, in which the kernel problem of the low frequency band is an STLS problem. The coder is shown to be related to sinusoidal coding based coders, the main difference being that sinusoidal coders use a sum of *undamped* sinusoids to fit a frame of the signal, whereas our scheme uses *exponentially damped* sinusoids. The proposed scheme is demonstrated to have a quality similar to existing coders, when compared at the same bit rate (4800 kbits/sec). The major advantage however of the coding based on complex damped exponentials is the possibility to do progressive speech compression with a variable degree of analysis/synthesis at the emitter/receiver depending on the availability of channel capacity and/or the requirements of the specific application, without having to encode the signal each time the bit rate changes.

Mathematical models are the heart of many applications in areas such as control system design, prediction, simulation, fault detection,... In chapter 7 a new framework for modeling dynamic linear time-invariant systems is developed. It combines the ideas of the prediction error modeling approach [63, 86] and the behavioural modeling approach [100, 79, 80, 71]. A general framework that integrates both approaches is proposed and shown to be related to the STLS problem. Several properties of the new framework are proven. A new optimization approach is developed that provides more insight into the new framework. This new approach makes it also possible to identify the parameters of the models of this new framework, by simply using of-the-shelf nonlinear optimization routines.

Finally, chapter 8 summarizes the conclusions and outlines suggestions for further research.

1.4 Notations

In this section some notation that will be used throughout the thesis will be introduced. First of all, matrices will always be denoted by capital letters and vectors by lowercase letters. To refer to specific entries or parts of a matrix or a vector, we adopt a Matlab-like [64] notation:

- A(i,j): the entry in the jth column of the ith row of A.
- A(i,:): the ith row of A.
- A(:,j): the jth column of A.
- A(p:q,r:s): the $(q-p+1) \times (s-r+1)$ submatrix of A containing the entries that belong to rows p till q **and** to columns r till s.
- b(i): the entry on the ith row of column vector b.
- b(p:q): the $(q-p+1) \times 1$ subvector of b containing the entries of row p till row q.

In general, the structured matrices in this thesis come from an overdetermined system of equations

$$Ax \approx b, \tag{1.14}$$

with $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^{m \times 1}$. The matrix $S \equiv [A \; b] \in \mathbb{R}^{m \times (n+1)}$ is the so-called extended data matrix. The overdetermined set of equations (1.14) can be made consistent by adding a correction matrix $\Delta A \in \mathbb{R}^{m \times n}$ to A and a correction vector $\Delta b \in \mathbb{R}^{m \times 1}$ to b. This leads to the definition of the extended correction data matrix $\Delta S = [\Delta A \; \Delta b] \in \mathbb{R}^{m \times (n+1)}$ and eventually the corrected extended data matrix $T \equiv S + \Delta S$.
If the matrix S is structured, it is obvious that it can be characterized with less than $m(n+1)$ numbers. Therefore we introduce a vector s that contains the k *different* entries of S. In this sense s is a minimal representation of S: knowledge of s and the structure of S completely characterizes the matrix S. Furthermore $s_1 \in \mathbb{R}^{k_1 \times 1}$ contains the entries of s that appear in A and $s_2 \in \mathbb{R}^{k_2 \times 1}$ contains the entries of s that appear in b and are not yet present in s_1 (thus $k = k_1 + k_2$). By convention we will always make sure that $s = [s_1^T \; s_2^T]^T$.
In most cases it is desired to obtain a corrected data matrix T having the same structure as S. It is clear that –in a similar way as was done for S– T can then be characterized by a vector $t \in \mathbb{R}^{k \times 1}$, that contains the different entries of T. By choosing t appropriately (i.e. by ensuring that $t(i), i = 1, \ldots, k$ is the corrected version of $s(i), i = 1, \ldots, k$) we can define a correction vector $\Delta s \in \mathbb{R}^{k \times 1}$ such that $s + \Delta s = t$. Sometimes a matrix function will be used to define a particular structure:

$$\mathbf{M}(m) : \mathbb{R}^{k \times 1} \to \mathbb{R}^{m \times (n+1)} : m \to \mathbf{M}(m).$$

For the sake of clarity, matrix functions are represented by boldface capital letters, immediately followed –between brackets– by the vector(s) on which they operate. In case S and T have the same structure, it is obvious that a matrix function exists such that: $S = \mathbf{M}(s)$ and $T = \mathbf{M}(t) = \mathbf{M}(s + \Delta s)$. If in addition

the structure is a linear one, it is allowed to write $\mathbf{M}(s+\Delta s) = \mathbf{M}(s)+\mathbf{M}(\Delta s)$ and $\mathbf{M}(\Delta s) = \Delta S$. We illustrate the previous on 2 small examples.
The first example concerns an extended data matrix with Hankel structure. Let $A = [1\ 2\ 3]^T$, $A + \Delta A = [1+\alpha \quad 2+\beta \quad 3+\gamma]^T$, $b = [2\ 3\ 4]^T$ and $b + \Delta b = [2+\beta \ \ 3+\gamma \ \ 4+\delta]^T$. The previously defined matrices, vectors and matrix functions then become:

$$S = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 4 \end{bmatrix},\ T = \begin{bmatrix} 1+\alpha & 2+\beta \\ 2+\beta & 3+\gamma \\ 3+\gamma & 4+\delta \end{bmatrix},\ \Delta S = T - S$$

$$s = [1\ 2\ 3\ 4]^T,\ \Delta s = [\alpha\ \beta\ \gamma\ \delta]^T,\ t = s + \Delta s,\ k = 4$$

$$s_1 = [1\ 2\ 3]^T,\ s_2 = 4,\ k_1 = 3,\ k_2 = 1$$

$$\mathbf{M}(m) : \mathbb{R}^{4\times 1} \rightarrow \mathbb{R}^{3\times 2} : m \rightarrow \begin{bmatrix} m(1) & m(2) \\ m(2) & m(3) \\ m(3) & m(4) \end{bmatrix}.$$

Note that in this example we can say that $\mathbf{M}(s + \Delta s) = \mathbf{M}(s) + \mathbf{M}(\Delta s)$ and $\Delta S = \mathbf{M}(\Delta s)$, because the Hankel structure is a linear one.
The next example consists of a nonlinearly structured extended data matrix. The A matrix is a Vandermonde matrix and the vector b is unstructured. Let

$$A = \begin{bmatrix} 1 & 1 \\ 2 & 3 \\ 4 & 9 \end{bmatrix},\ A + \Delta A = \begin{bmatrix} 1 & 1 \\ 2+\alpha & 3+\beta \\ (2+\alpha)^2 & (3+\beta)^2 \end{bmatrix},$$

$$b + \Delta b = \begin{bmatrix} 7 \\ -3 \\ 2 \end{bmatrix} \text{ and } b + \Delta b = \begin{bmatrix} 7+\gamma \\ -3+\delta \\ 2+\epsilon \end{bmatrix}.$$

The previously defined matrices, vectors and matrix functions then become:

$$S = \begin{bmatrix} 1 & 1 & 7 \\ 2 & 3 & -3 \\ 4 & 9 & 2 \end{bmatrix},\ T = \begin{bmatrix} 1 & 1 & 7+\gamma \\ 2+\alpha & 3+\beta & -3+\delta \\ (2+\alpha)^2 & (3+\beta)^2 & 2+\epsilon \end{bmatrix},\ \Delta S = T - S$$

$$s = [2\ 3\ 7\ \ -3\ 2]^T,\ \Delta s = [\alpha\ \beta\ \gamma\ \delta\ \epsilon]^T,\ t = s + \Delta s,\ k = 5$$

$$s_1 = [2\ 3]^T, s_2 = [7\ \ -3\ 2]^T,\ k_1 = 2,\ k_2 = 3$$

$$\mathbf{M}(m) : \mathbb{R}^{5\times 1} \rightarrow \mathbb{R}^{3\times 3} : m \rightarrow \begin{bmatrix} 1 & 1 & m(3) \\ m(1) & m(2) & m(4) \\ m(1)^2 & m(2)^2 & m(5) \end{bmatrix}.$$

Note that for this nonlinear structure it is no longer allowed to write $\mathbf{M}(s + \Delta s) = \mathbf{M}(s) + \mathbf{M}(\Delta s)$ or $\Delta S = \mathbf{M}(\Delta s)$. The following however still holds: $S = \mathbf{M}(s)$ and $T = \mathbf{M}(t) = \mathbf{M}(s + \Delta s)$.

1.5 Conclusion

In this chapter we have situated the STLS problem among other linear data fitting techniques such as the LS and TLS approach. The STLS approach has been motivated from a statistical point of view as well as from a practical point of view. The omnipresence of STLS problems in applications has been illustrated with several examples. Finally a chapter by chapter overview has been provided and some notations that will be used throughout the thesis have been introduced.

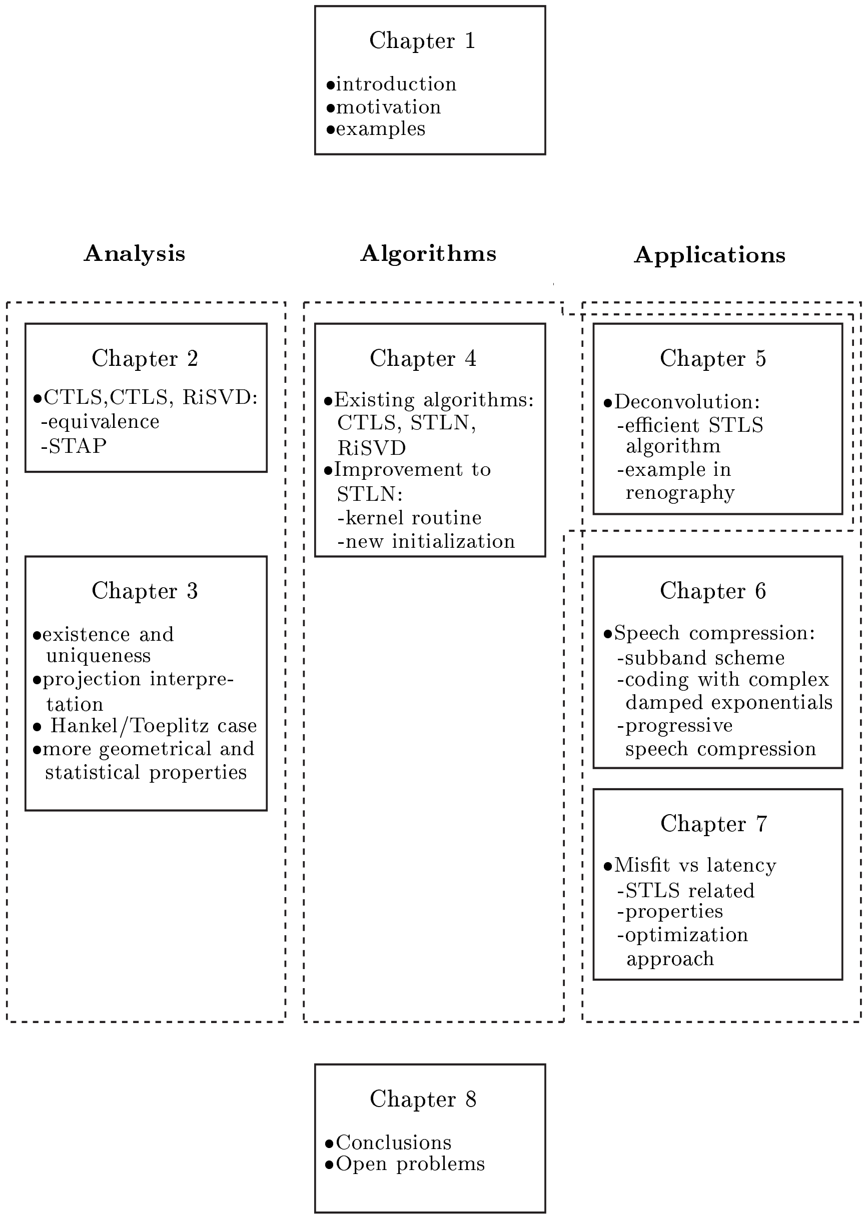


Figure 1.3: *Organization of the thesis.*

Chapter 2

Introduction to Structured Total Least Squares

This chapter introduces the Structured Total Least Squares problem. By using simple examples, we show that the STLS problem is more than a simple extension of the TLS problem (section 2.1). Next, we present in section 2.2 an overview of the 3 *most important approaches found in literature: the Constrained Total Least Squares (CTLS) approach, the Riemannian Singular Value Decomposition (RiSVD) approach and the Structured Total Least Norm (STLN) approach. Under mild conditions we can prove the equivalence of the three approaches when used for solving the STLS problem (section 2.3). Further extensions towards multivariate STLS problems, nonlinearly structured matrices,... are brought together in the Structured Total Approximation Problems (STAP) framework (section 2.4). The theory for the extension of the RiSVD framework towards nonlinear structures is derived and illustrated with a Vandermonde system example. Finally, we demonstrate in section 2.5 how an STLS problem involving complex data matrices can be transformed into an STLS problem involving only real data matrices.*

2.1 Problem formulation

As already mentioned in the introductory chapter, the STLS problem is an extension of the well-known TLS problem:

$$\boxed{\begin{array}{l}\boxed{\begin{array}{l}\min_{\Delta A,\Delta b,x} \|[\Delta A \;\; \Delta b]\|_F^2 \\ \text{such that } (A+\Delta A)x = b+\Delta b\end{array}} \;\; \textbf{TLS} \\ \text{such that } [A+\Delta A \;\; b+\Delta b] \text{ has same affine structure as } [A \;\; b]\end{array}} \;\; \textbf{STLS} \qquad (2.1)$$

where $A, \Delta A \in \mathbb{R}^{m\times n}, b, \Delta b \in \mathbb{R}^{m\times 1}$ and $x \in \mathbb{R}^{n\times 1}$. On first sight, the addition of an extra constraint on the structure of $[A+\Delta A \;\; b+\Delta b]$ seems a simple extension of the TLS approach. To take the edge of this argument, we consider a small example that will give an idea of the increased difficulty when going from TLS to STLS.

Example 2.1.1 Let $S \equiv [A \; b]$ be the following Toeplitz matrix

$$\begin{bmatrix} 0.2190 & 0.0470 & 0.6789 \\ 0.9347 & 0.2190 & 0.0470 \\ 0.3835 & 0.9347 & 0.2190 \end{bmatrix}.$$

At the solution of the STLS problem, the matrix $[\Delta A \;\; \Delta b]$ will have to be Toeplitz too. □

If we set $y = [x^T \;\; -1]^T$, it is well known that the TLS problem can be rewritten as the following unconstrained optimization problem

$$\min_y f_{TLS}(y), \text{ with } f_{TLS}(y) = \frac{y^T S^T S y}{y^T y}. \qquad (2.2)$$

Note that the change from $[x^T \;\; -1]^T$ to y does not alter the solution of the TLS problem, unless $y(n+1) = 0$ at the global minimum of f_{TLS}. The latter situation is referred to as *nongeneric* TLS (see [96, Chapter 3] for an extensive discussion on nongeneric TLS problems). For the STLS problem it is also possible to derive an equivalent unconstrained optimization problem

$$\min_y f_{STLS}(y), \text{ with } f_{STLS}(y) = y^T S^T \mathbf{D_2}(y)^{-1} S y, \qquad (2.3)$$

where $\mathbf{D_2}(y) \in \mathbb{R}^{(n+1)\times(n+1)}$ is a symmetric positive definite matrix, the entries of which are quadratic functions of y. An exact definition of $\mathbf{D_2}(y)$ and a derivation of this unconstrained optimization formulation follow in section 2.2 and 2.3. Also in the STLS case, the change to y could lead to a different solution for (2.3) compared to the STLS problem stated in the beginning of this chapter when $y(n+1) = 0$ at the global solution of (2.3). In analogy to TLS, such problems are referred to as nongeneric STLS problems. They will be discussed in chapter 3. Example 2.1.1 is a generic problem in both the TLS and STLS framework. For f_{TLS} it is clear and for f_{STLS} it will be shown in section 2.3, that both functions are homogeneous of zeroth order. The latter means that $f_{TLS}(y) = f_{TLS}(\alpha y)$ and $f_{STLS}(y) = f_{STLS}(\alpha y)$ for $\alpha \neq 0$. Therefore, if we want to visualize both functions, we can constrain the domain to the unit sphere $y^T y = 1$. The natural way to represent points on the unit sphere is by using spherical coordinates (θ, ϕ) instead of the euclidean coordinates $(y(1), y(2), y(3))$. The transformations formulas are

$$\begin{aligned} y(1) &= |sin(\phi)| cos(\theta) \\ y(2) &= |sin(\phi)| sin(\theta) \\ y(3) &= cos(\phi) \end{aligned}$$

as can be seen from figure 2.1. It is clear that in spherical coordinates both functions are periodic in θ and ϕ with period 2π, and thus we only have to visualize the functions for $-\pi < \phi \leq \pi$, $-\pi < \theta \leq \pi$. The left hand side of figure 2.2 shows the surface plot (figure (a)) and the corresponding contour plot (figure (c)) of f_{TLS} whereas the right-hand side shows the surface plot (figure (b)) and the corresponding contour plot (figure (d)) of f_{STLS} for example 2.1.1. From the surface plots we can already see that the transition from TLS to STLS leads to a substantially more difficult function. Further information is obtained by comparing both contour plots. First of all there is an obvious symmetry around the axis $\phi = 0$. This is because (θ, ϕ) and $(\theta, -\phi)$ represent the same point on the unit sphere. There is however a second symmetry in the contour plots that stems from the fact that $f_{(S)TLS}(y) = f_{(S)TLS}(-y)$[1] (remember that both functions are zeroth order homogeneous functions), yielding in spherical coordinates the equality: $f_{(S)TLS}(\theta, \phi) = f_{(S)TLS}(\theta - \pi, \pi - \phi)$. So for symmetry reasons, we only have to inspect e.g. the upper left quadrant ($-\pi < \theta \leq 0$, $0 < \phi \leq \pi$) of the contour plots, when looking for (local) minima of $f_{(S)TLS}$. The crosses is the contour plots indicate the location of local minima that were found in the upper left quadrant by function evaluation over a grid of points. The accompanying numbers indicate the function value at that point. The main conclusion from the contour plots is that even in this small example the STLS problem has 2 local minima, whereas in the TLS case, there is only one stationary point that is a minimum and thus also the global minimum. The other stationary points of the TLS problem are saddle points, except for one that is a maximum.

2.2 Overview of existing approaches

In this section we give an overview of the most important existing methods for solving STLS problems: the Constrained Total Least Squares (CTLS) approach

[1] $f_{(S)TLS}$ is a symbolic notation that refers to f_{TLS} as well as to f_{STLS}

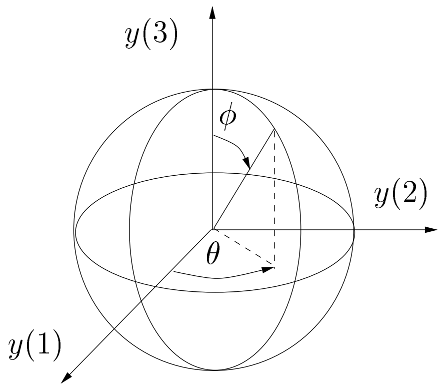


Figure 2.1: *This figure shows the relation between the euclidean coordinates* $(y(1), y(2), y(3))$ *and the spherical coordinates* (θ, ϕ) *of a point on the unit sphere.*

[2, 3], the Riemannian SVD (RiSVD) approach [66, 67] and the Structured Total Least Norm (STLN) approach [81, 83, 95]. For each of them we state the initial problem formulation and the problem that is ultimately solved (most of the approaches start from an initial problem formulation –that is very similar for the three methods– but solve a different problem that is derived from the initial one).
For the sake of clarity, we sometimes use a nomenclature different from the one used in the key references. However, the one-to-one correspondence between our notation and the notations in the key references should be clear. To give an idea of the structure and size of the different matrices and vectors that appear in these approaches, we will apply them on the following example.

Example 2.2.1 Let S be the following Hankel matrix

$$S = \begin{bmatrix} 5 & 6 & 7 \\ 6 & 7 & 8 \end{bmatrix}^T$$
$$\Delta S = \begin{bmatrix} \tilde{\alpha} & \tilde{\beta} & \tilde{\gamma} \\ \tilde{\beta} & \tilde{\gamma} & \tilde{\delta} \end{bmatrix}^T.$$

The STLS problem to solve is (2.1) where the structure to be preserved is the Hankel structure of the original Hankel matrix S. □

As will become clear in the next three subsections, the different approaches are conceptually equivalent in the sense that they use a minimal representation for ΔS (or $S + \Delta S$), i.e. only the *different* elements of ΔS (or $S + \Delta S$) are represented. The relation between the elements of this minimal representation

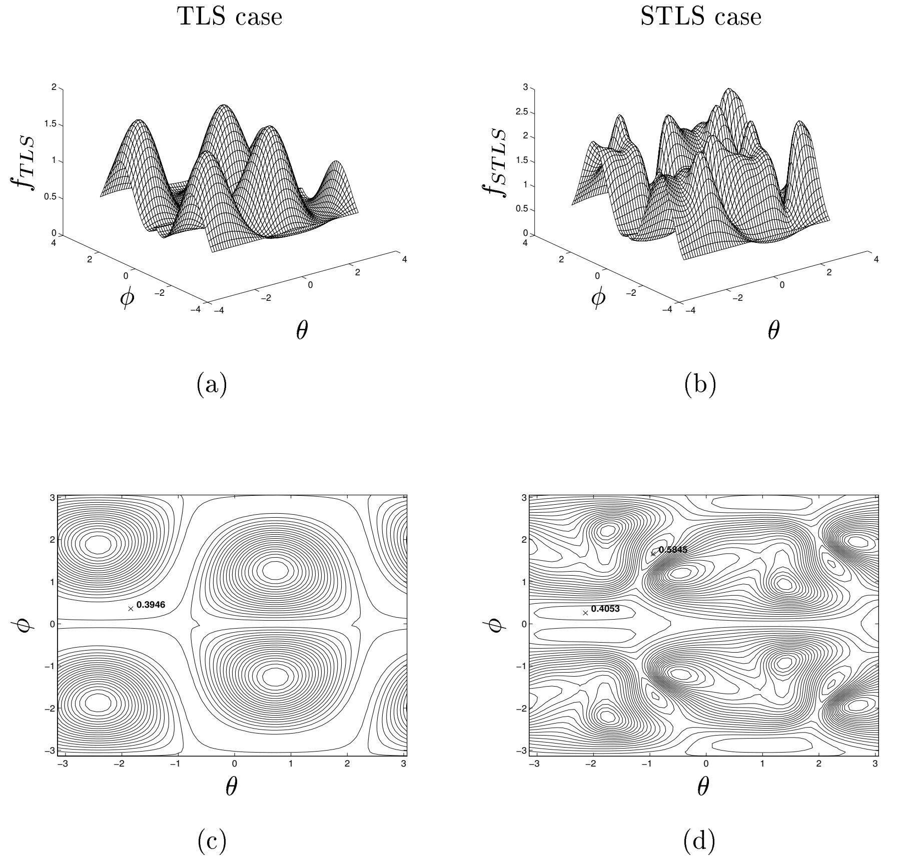


Figure 2.2: *The figures on the left show the surface plot (figure (a)) and the contour plot (figure (c)) of the cost function f_{TLS} that has to be minimized in order to solve the TLS problem applied to Example 2.1.1. The figures on the right show the surface plot (figure (b)) and the contour plot (figure (d)) of the cost function f_{STLS} that has to be minimized in order to solve the STLS problem applied to Example 2.1.1.*

and their location inside ΔS (or $S + \Delta S$) is invariably given by some structure-dependent matrices. Finally, subsection 2.2.4 will reveal an interesting analogy between the described approaches and the possible ways for solving the TLS problem.

2.2.1 CTLS

In the CTLS approach the different entries of ΔS are represented by the elements of a so-called noise vector $\Delta s \in \mathbb{R}^{k\times 1}$ of minimal dimensionality. The structure-dependent matrices $F_i \in \mathbb{R}^{m\times k}, i = 1, \ldots, n+1$ give the relation between the entries of Δs and their location inside ΔS. They are defined as follows

$$\Delta S(:,i) \equiv F_i \Delta s, i = 1, \ldots, n+1.$$

The CTLS approach then starts from the following formulation

$$\begin{aligned} &\min_{\Delta s, x} \Delta s^T W \Delta s \\ &\text{such that } (A + [F_1 \Delta s \ldots F_n \Delta s])x = b + F_{n+1}\Delta s, \end{aligned} \tag{2.4}$$

where $W \in \mathbb{R}^{k\times k}$ is a weighting matrix. In [3, Theorem 1] a new equivalent problem formulation is derived from (2.4)

$$\min_x \begin{bmatrix} x \\ -1 \end{bmatrix}^T S^T (\mathbf{H}(x) W^{-1} \mathbf{H}(x)^T)^{-1} S \begin{bmatrix} x \\ -1 \end{bmatrix}, \tag{2.5}$$

with $\mathbf{H}(x) = \sum_{i=1}^n x(i) F_i - F_{n+1}$. For the standard STLS problem as defined in (2.1) , applied to example 2.2.1, we get the following scalars, vectors and matrices

$$\begin{aligned} &k = 4, \ \Delta s = [\tilde{\alpha} \ \tilde{\beta} \ \tilde{\gamma} \ \tilde{\delta}]^T \\ &F_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \ F_2 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &\mathbf{H}(x) = \begin{bmatrix} x(1) & -1 & 0 & 0 \\ 0 & x(1) & -1 & 0 \\ 0 & 0 & x(1) & -1 \end{bmatrix}, \ W = diag([1\ 2\ 2\ 1]), \end{aligned}$$

where $diag(v)$ is a diagonal matrix such that it has $v(i)$ as ith diagonal entry. Note that we have to assume that $(\mathbf{H}(x)W^{-1}\mathbf{H}(x)^T)$ is invertible. This is not always the case. E.g. when $k < m$, $(\mathbf{H}(x)W^{-1}\mathbf{H}(x)^T)$ is rank deficient. Since the same problem occurs when interpreting the STLS as a projection operation, we will discuss this in the corresponding section of chapter 3.

2.2.2 RiSVD

The RiSVD approach is somewhat different from the two other approaches in the sense that it deals with matrices S and $T \equiv S + \Delta S$ instead of using the matrices S and ΔS. In other words, the RiSVD is formulated in terms of the given matrix S and the perturbed version T instead of being formulated in terms of the given matrix S and its perturbation ΔS. The k *different* elements of T are arranged in a vector $t \in \mathbb{R}^{k\times 1}$. The link between the elements of t and their location inside T is given by the following equality, involving the structure-dependent constant basis matrices $T_i \in \mathbb{R}^{m\times(n+1)}, i = 1, \ldots, k$

$$T = \sum_{i=1}^{k} T_i t(i).$$

If we stack the different elements of S in a vector $s \in \mathbb{R}^{k\times 1}$ in the same way [2] as we created t (meaning that $t(i)$ is an entry of T that represents a perturbed version of the corresponding entry $s(i)$ of S), S can be written as

$$S = \sum_{i=1}^{k} T_i s(i).$$

The initial formulation of the RiSVD approach then is

$$\begin{aligned} &\min_{t,y} \sum_{i=1}^{k} W(i,i)(t(i) - s(i))^2 \\ &\text{such that } \; Ty = 0 \; \text{ and } \; y^T y = 1, \end{aligned} \tag{2.6}$$

Using the technique of Lagrange multipliers, (2.6) is transformed [67, Theorem 1] into the following non-linear generalized SVD.

$$\begin{aligned} &\text{Find the triplet } (u,\tau,v) \text{ correponding to the smallest } \tau \text{ such that} \\ &Sv = \mathbf{D_2}(v)u\tau \quad u^T\mathbf{D_2}(v)u = 1 \\ &S^T u = \mathbf{D_1}(u)v\tau \quad v^T\mathbf{D_1}(u)v = 1, v^T v = 1, \end{aligned} \tag{2.7}$$

with $\mathbf{D_2}(v) = \sum_{i=1}^{k} \frac{1}{W(i,i)}(T_i v)(T_i v)^T$, $\mathbf{D_1}(u) = \sum_{i=1}^{k} \frac{1}{W(i,i)}(T_i^T u)(T_i^T u)^T$ and $W \in \mathbb{R}^{k\times k}$ a weighting matrix. Thus $\mathbf{D_1}(u)$ and $\mathbf{D_2}(v)$ are symmetric non-negative definite matrices, the entries of which are quadratic functions of u, respectively v. Furthermore the corrected data matrix can be obtained using the following formula:

$$t(i) = s(i) - \frac{1}{W(i,i)} u^T T_i v \tau, \tag{2.8}$$

and due to constraint $v^T v = 1$ we have

$$y = v. \tag{2.9}$$

[2] We assume here that S will be approximated by a matrix T having the same structure. In almost all practical applications, this is the only choice that makes sense.

Remark 2.2.2 We added an extra constraint $v^Tv = 1$ to the original formulation as described in [67]. Without this constraint u and v cannot be determined uniquely. The reason for this is that the constraint $v^T\mathbf{D_1}(u)v = 1$ is automatically satisfied once the constraint $u^T\mathbf{D_2}(v)u = 1$ is satisfied. This can immediately be verified by writing

$$u^T\mathbf{D_2}(v)u = \sum_{i=1}^{k} \frac{1}{W(i,i)}(u^TT_iv)(u^TT_iv)^T =$$

$$\sum_{i=1}^{k} \frac{1}{W(i,i)}(v^TT_i^Tu)(vT_i^Tu)^T = v^T\mathbf{D_1}(u)v.$$

So without the extra constraint, once a solution (u, τ, v) is known every triplet $(\frac{u}{\xi}, \tau, v\xi)$ with $\xi \neq 0$ also satisfies the equations (2.7) without the constraint $v^Tv = 1$. ◇

It is clear that if $\mathbf{D_1}(u)$ were equal to I_m and $\mathbf{D_2}(v)$ equal to I_{n+1}, formulation (2.7) would be the SVD formulation of the TLS approach. Furthermore, note that if $\mathbf{D_1}(u)$ and $\mathbf{D_2}(v)$ could be assumed constant, then (2.7) would be equivalent to finding the triplet (u, τ, v) corresponding to the smallest restricted singular value of the Restricted Singular Value Decomposition (RSVD) [68] of the triplet $(A, \mathbf{D_2}(v)^{1/2}, \mathbf{D_1}(u)^{1/2})$.
Applied to example 2.2.1, we obtain the following scalars, vectors and matrices

$$k = 4,\ s = [5\ \ 6\ \ 7\ \ 8]^T,\ s = [5 + \tilde{\alpha}\ \ 6 + \tilde{\beta}\ \ 7 + \tilde{\gamma}\ \ 8 + \tilde{\delta}]^T,$$

$$T_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}^T,\ T_2 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}^T,\ T_3 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}^T,$$

$$T_4 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}^T,$$

$$\mathbf{D_1}(u) = \begin{bmatrix} u(1)^2 + \frac{u(2)^2}{2} + \frac{u(3)^2}{2} & \frac{u(1)u(2)}{2} + \frac{u(2)u(3)}{2} \\ \frac{u(1)u(2)}{2} + \frac{u(2)u(3)}{2} & \frac{u(1)^2}{2} + \frac{u(2)^2}{2} + u(3)^2 \end{bmatrix},$$

$$\mathbf{D_2}(v) = \begin{bmatrix} v(1)^2 + \frac{v(2)^2}{2} & \frac{v(1)v(2)}{2} & 0 \\ \frac{v(1)v(2)}{2} & \frac{v(1)^2+v(2)^2}{2} & \frac{v(1)v(2)}{2} \\ 0 & \frac{v(1)v(2)}{2} & \frac{v(1)^2}{2} + v(2)^2 \end{bmatrix},$$

$$W = diag([1\ 2\ 2\ 1])$$

2.2.3 STLN

The STLN approach (we first consider the STLN approach for L_2 norms, extensions to other norms are given in section 2.4) is the most straightforward approach. The *different* entries of ΔA are stacked in a vector $\alpha \in \mathbb{R}^{q\times 1}$ while the different entries of Δb (which are not already accounted for in α) are stacked in a vector $\beta \in \mathbb{R}^{d\times 1}$. The structure-dependent matrices relating the elements of α and β to their location in ΔA and Δb are: $X \in \mathbb{R}^{m\times q}$, $P_2 \in \mathbb{R}^{m\times d}$, $P_1 \in \mathbb{R}^{m\times q}$. They are defined by the following equalities:

$$\begin{aligned}\Delta A x &= X\alpha\\ \Delta b &= P_2\beta + P_1\alpha.\end{aligned}$$

Using this nomenclature, the STLS problem is formulated as follows:

$$\begin{aligned}&\min_{\alpha,\beta,x} \alpha^T D_\alpha^2 \alpha + \beta^T D_\beta^2 \beta\\ &\text{such that } \; Ax + X\alpha = b + P_2\beta + P_1\alpha,\end{aligned} \tag{2.10}$$

where $D_\alpha \in \mathbb{R}^{q\times q}$ and $D_\beta \in \mathbb{R}^{d\times d}$ represent weighting matrices. Unlike the previous approaches, no new formulation is derived from this one. Applied to example 2.2.1, the above mentioned scalars, vectors and matrices become:

$$\begin{aligned}&q = 3,\ d = 1,\ \alpha = [\tilde{\alpha}\ \ \tilde{\beta}\ \ \tilde{\gamma}]^T,\ \beta = \tilde{\delta},\\ &P_2 = [0\ \ 0\ \ 1]^T,\ P_1 = \begin{bmatrix} 0 & 1 & 0\\ 0 & 0 & 1\\ 0 & 0 & 0\end{bmatrix},\ X = \begin{bmatrix} x(1) & 0 & 0\\ 0 & x(1) & 0\\ 0 & 0 & x(1)\end{bmatrix}\\ &D_\alpha^2 = \begin{bmatrix} 1 & 0 & 0\\ 0 & 2 & 0\\ 0 & 0 & 2\end{bmatrix},\ D_\beta^2 = 1\end{aligned}$$

2.2.4 Interpretation of the three approaches

A closer look at the three final formulations (2.5), (2.7) and (2.10), reveals an interesting analogy between on one hand the CTLS, RiSVD and STLN approach for solving the STLS problem and on the other hand the possible ways for solving the ordinary TLS problem. From (2.1) it is clear that, just like the STLS problem, the TLS problem is basically a constrained optimization problem. It can therefore be solved in that framework, using some advanced nonlinear constrained optimization package. For the STLS problem this corresponds to the STLN approach (for L_2 norm). Exploiting the algebraic properties of the TLS problem, namely the fact that it corresponds to an eigenvalue problem in which we are looking for the eigenvector corresponding to the smallest eigenvalue of the matrix S^TS, allows for other solution methods: an eigenvalue decomposition (EVD) approach applied to S^TS or an SVD approach applied to S. The counterpart for the SVD method in the STLS framework is the RiSVD approach. As a matter of fact, by writing down the RiSVD approach for an unstructured matrix S, we find that $\mathbf{D_2}(v) = \|v\|_2 I_{n+1}$ and $\mathbf{D_1}(u) = \|u\|_2 I_m$.

The equations in (2.7) then simply become the ordinary SVD equations.
The EVD method for solving the TLS problem becomes the CTLS approach in the STLS case. Indeed, the objective function in (2.5) can be viewed as a new kind of Rayleigh quotient, that needs to be minimized in order to find the smallest eigenvalue and the corresponding eigenvector of the non-linear EVD that can be associated with the Riemannian SVD (2.7). The equivalences are summarized once more in figure 2.3.

Remark 2.2.3 The EVD equations that correspond to the RiSVD equations (2.7) can easily be deduced from the latter:

$$\begin{aligned} S^T \mathbf{D_2}(v)^{-1} S v &= \mathbf{D_1}(u) v \tau^2 \\ S \mathbf{D_1}(u)^{-1} S^T u &= \mathbf{D_2}(v) u \tau^2. \end{aligned}$$

As can be seen, we obtain a set of "coupled" EVD equations, the Rayleigh quotient of which is the objective function in (2.5). $\diamond$

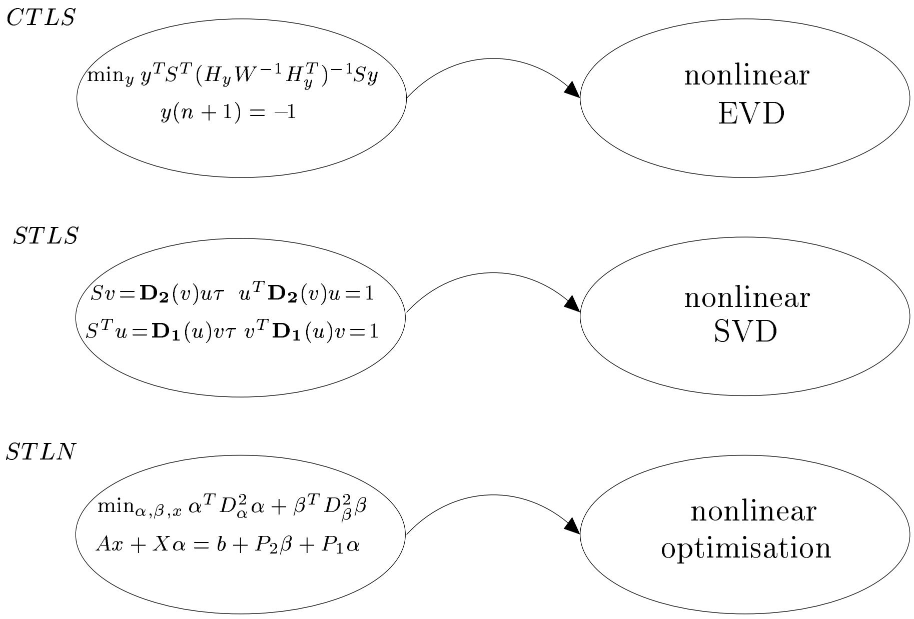


Figure 2.3: *This figure shows the interesting analogy that exists between the different approaches of the STLS problem and the possible methods for solving the TLS problem.*

2.3 Equivalence of existing approaches

We have now described three approaches for the STLS problem. Although the initial formulations (2.4), (2.6) and (2.10) are very similar the actually used formulations (obtained after some derivation) (2.5), (2.7) and (2.10) have caused a great deal of confusion on whether some of these methods are suboptimal or not. In [94] it is claimed that CTLS is suboptimal and thus yields a solution different from the one obtained with e.g. the RiSVD approach. The main argument used to support this claim [94, 67] is that in the CTLS method a linear constraint is used, whereas the constraint that appears in the RiSVD approach is a nonlinear one. As in [61], we will prove the equivalence of the three approaches CTLS, RiSVD and STLN under mild conditions. We do so by deriving for each of these approaches an equivalent cost function that only depends on the so-called extended parameter vector $y \equiv -y(n+1)[x^T \;\; -1]^T$. We show that these cost functions are equal. The only difference then left is that in the CTLS and STLN approach, the constraint on the extended parameter vector is linear, whereas the constraint on the extended parameter vector in the RiSVD case is quadratic or nonlinear. Using Lemma 2.3.4 we will be able to prove that this difference does not change the solution for the parameter vector x, that can be derived from the extended parameter vector as follows: $x = -y(1:n)/y(n+1)$.
To derive a cost function for the CTLS approach is very easy, since the final CTLS formulation (2.5) derived from (2.4) is exactly what we are looking for. Thus from (2.5) we find for the CTLS case:

$$\begin{aligned} &\min_y y^T S^T (\mathbf{H_2}(y) W^{-1} \mathbf{H_2}(y)^T)^{-1} S y, \\ &\text{such that } y(n+1) = -1, \end{aligned} \tag{2.11}$$

where $\mathbf{H_2}(y) = \sum_{i=1}^{n+1} y(i) F_i$. To obtain a cost function equivalent to the STLN formulation (2.10), we apply the method of Lagrange multipliers to (2.10). In appendix A.1 we prove the following Lemma:

Lemma 2.3.1 *The STLN formulation (2.10) can be reformulated as the following optimization problem:*

$$\begin{aligned} &\min_y y^T S^T (\tilde{P}_2 D_\beta^{-2} \tilde{P}_2^T + (\tilde{X} - \tilde{P}_1) D_\alpha^{-2} (\tilde{X} - \tilde{P}_1)^T)^{-1} S y \\ &\text{such that } y(n+1) = -1, \end{aligned} \tag{2.12}$$

where $\tilde{X} \equiv -y(n+1)X$, $\tilde{P}_1 \equiv -y(n+1)P_1$ *and* $\tilde{P}_2 \equiv -y(n+1)P_2$.

Also the RiSVD formulation can easily be reformulated as an optimization problem that only depends on the extended parameter vector.

Lemma 2.3.2 *The RiSVD equations (2.7) can be reformulated as the following optimization problem:*

$$\begin{aligned} &\min_y y^T S^T \mathbf{D_2}(y)^{-1} S y \\ &y^T y = 1. \end{aligned} \tag{2.13}$$

The proof of this lemma can be found in appendix A.2. In the next Lemma (see appendix A.3 for the proof), we show that the three objective functions in (2.11), (2.12) and (2.13) are equal.

Lemma 2.3.3

$$y^T S^T (\mathbf{H_2}(y) W^{-1} \mathbf{H_2}(y)^T)^{-1} S y = y^T S (\tilde{P}_2 D_\beta^{-2} \tilde{P}_2^T + (X - \tilde{P}_1) D_\alpha^{-2} (X - \tilde{P}_1)^T)^{-1} S y \tag{2.14}$$

and

$$y^T S^T (\mathbf{H_2}(y) W^{-1} \mathbf{H_2}(y)^T)^{-1} S y = y^T S^T \mathbf{D_2}(y)^{-1} S y. \tag{2.15}$$

From the previous lemmas it is clear that the CTLS and STLN approach are equivalent. To prove the equivalence of the RiSVD with the other 2 approaches, we have to show that that quadratic constraint in (2.13) can be replaced with the linear constraint $y(n+1) = -1$, without affecting the parameter vector $x = -y(1:n)/y(n+1)$. Therefore, we first prove the following Lemma.

Lemma 2.3.4 *Let $M \in \mathbb{R}^{g \times h}, y \in \mathbb{R}^{h \times 1}, \mathbf{Q}(y) \in \mathbb{R}^{g \times g}$, where the entries of $\mathbf{Q}(y)$ are quadratic functions of the entries of y. Furthermore, let f be the following multivariable function of y: $f(y) = y^T M^T \mathbf{Q}(y)^{-1} M y$. Then $f(\gamma y) = f(y), \forall \gamma \neq 0$.*

Proof: Applying the definition of f we find that $f(\gamma y) = (\gamma y)^T M^T \mathbf{Q}(\gamma y)^{-1} M (\gamma y)$. Since the entries of $\mathbf{Q}(\gamma y)$ are quadratic functions of the elements of γy, it is obvious that $\mathbf{Q}(\gamma y) = \gamma^2 \mathbf{Q}(y)$. Therefore $f(\gamma y) = (\gamma y)^T M^T \gamma^{-2} \mathbf{Q}(y)^{-1} M (\gamma y) = f(y), \forall \gamma \neq 0$ □

Since lemma 2.3.4 applies to the cost function in (2.13), it is clear that the RiSVD cost function is scaling invariant. Thus the only effect of the constraint $y^T y = 1$ is to fix 1 solution y out of the infinite number of solutions $\gamma y, \forall \gamma \neq 0$. However it does not put any constraints on the direction of y. The constraint $y(n+1) = -1$ is also a way to select one out of an infinite number of solutions. This time however, the directions for which $y(n+1) = 0$ are excluded. As mentioned in the introduction, problems that have $v(n+1) = 0$ at their solution are referred to as nongeneric STLS problems. Summarizing we can say that for generic STLS problems, the parameter vector x does not change when in formulation (2.13) the constraint is changed into the linear constraint $y(n+1) = -1$, thereby proving that under the condition that the STLS problem at hand is generic, all three approaches CTLS, RiSVD and STLN are equivalent in the sense that they all yield the same parameter vector x.

2.4 Extensions of the STLS problem

In this section we consider further extensions of the STLS. Although the thesis will mainly focus on STLS problems, it is interesting to situate the STLS problem in a larger framework that we will denote by Structured Total Approximation Problems (STAP). We define the STAP as follows:

$$\begin{aligned} &\min_{\Delta s, X} \|\Delta s\| \\ &\text{such that } (A + \Delta A)X = B + \Delta B \\ &\text{and } [A + \Delta A \ B + \Delta B] \text{ has the same structure as } [A \ B], \end{aligned} \tag{2.16}$$

with $A, \Delta A \in \mathbb{R}^{m \times n}$, $B, \Delta B \in \mathbb{R}^{m \times d}$, $X \in \mathbb{R}^{n \times d}$, Δs contains the *different* entries of $[\Delta A \ \Delta B]$ and $\|.\|$ is a vector norm. Table 2.1 gives an overview of the STAP framework. Each cell –except for the TLS, LS and the STLS cell– contain references that address that particular problem. Note that each

	$[\Delta A \ \Delta B]$ unstructured	affinely structured	nonlinearly structured
(weighted) L_1	[9, 10, 99]	[81, 83]	[82]
(weighted) $L_p, 1 < p < 2$	[73, 98, 99]		
(weighted) L_2	LS, TLS	STLS	[82]
(weighted) $L_p, 2 < p < \infty$	[99]		
(weighted) L_∞	[99]	[81, 83]	

Table 2.1: *This table gives an overview of the STAP framework. Each cell can be further subdivided in a univariate ($d = 1$) and a multivariate $d > 1$ variant.*

cell of the table can further be subdivided into a univariate ($d = 1$) STAP and a multivariate $d > 1$ STAP. From the table it is clear that the STLS problem is a special case of the STAP framework. Although an unstructured matrix is a special case of an affinely structured matrix, we have deliberately created a different column, since the unstructured problems are easier to solve and well-understood. The two dimensions in the table represent extensions towards nonlinear structures (horizontal direction) and towards different norms

(vertical direction). The extension towards other norms is obviously inspired by statistical motives. For example, the use of an L_2 norm is recommended when the entries of Δs obey a Gaussian distribution, whereas the L_∞ provides so-called worst case solutions. Furthermore, the L_1 norm is known to be insensitive to a number of outliers present in Δs. From the previous explanation it should be clear that –under certain statistical conditions– someone might select an L_p norm with $1 < p < 2$ or $2 < p < \infty$ as the appropriate norm. Like in the unstructured case, L_1 and L_∞ problems (at least for the affinely structured matrices) can be treated as linear programming problems. For $1 < p < 2$, matters become more complicated since the objective function of the STAP is only once differentiable.
In the next subsection we briefly cover the multivariate STLS problem. We show how the multivariate STLS problem can be transformed into a larger univariate STLS problem. In the last subsection we present our extension of the STLS towards nonlinear structures, called Nonlinear STLS (NSTLS) and illustrate it by means of a Vandermonde system. In the remainder of the thesis the focus will be mainly on the STLS problem for affinely structured matrices. However, if a statement can be generalized towards a broader class of STAP we will do so.

2.4.1 The multivariate STLS problem

As shown in [95], the multivariate STLS problem can easily be transformed into a univariate STLS problem. Therefore we start from (2.16) with $d > 1$ and $\|.\|$ the L_2 norm. It is clear that $AX = B, A \in \mathbb{R}^{m\times n}, X \in \mathbb{R}^{n\times d}, B \in \mathbb{R}^{m\times d}$ can be rewritten as follows: $(I_d \otimes A)vec(X) = vec(B)$, where $vec(M)$ denotes the vector obtained by stacking the different columns of M into one long vector and $\otimes$ is the Kronecker product. Thus the multivariate STLS problem

$$\begin{aligned}
&\min_{\Delta s, X} \Delta s^T W \Delta s \qquad (2.17)\\
&\text{such that } (A + \Delta A)X = B + \Delta B\\
&\text{and } [A + \Delta A \; B + \Delta B] \text{ has the same structure as } [A \; B],
\end{aligned}$$

where $\Delta s \in \mathbb{R}^{k\times 1}$ contains the *different* elements of $[\Delta A \; \Delta B] \in \mathbb{R}^{m\times(n+d)}$ and $W \in \mathbb{R}^{k\times k}$ a weighting matrix, can be reformulated as

$$\begin{aligned}
&\min_{\Delta s, \tilde{x}} \Delta s^T W \Delta s \qquad (2.18)\\
&\text{such that } (\tilde{A} + \Delta\tilde{A})\tilde{x} = \tilde{B} + \Delta\tilde{B}\\
&\text{and } [\tilde{A} + \Delta\tilde{A} \; \tilde{B} + \Delta\tilde{B}] \text{ has the same structure as } [\tilde{A} \; \tilde{B}],
\end{aligned}$$

with $\tilde{A} = I_d \otimes A \in \mathbb{R}^{md\times nd}$, $\Delta\tilde{A} = I_d \otimes \Delta A \in \mathbb{R}^{md\times nd}$, $\tilde{B} = vec(B) \in \mathbb{R}^{md\times 1}$, $\Delta\tilde{B} = vec(\Delta B) \in \mathbb{R}^{md\times 1}$, $\tilde{x} = vec(X) \in \mathbb{R}^{nd\times 1}$.

Remark 2.4.1 This approach (i.e. going from (2.17)) to (2.18) can lead to an unnecessarily large number of variables. Let us consider the case in which $[A\ B]$ is a Hankel matrix. The equality constraints in (2.17) can be restated as follows:

$$[A + \Delta A\ B + \Delta B] \begin{bmatrix} X \\ -I_d \end{bmatrix} = 0.$$

So it is clear that the rows of $[X^T\ -I_d^T]$ form a basis for the nullspace of $[A + \Delta A\ B + \Delta B]$. In (2.17) and (2.18), we see that this basis is parametrized by nd parameters (namely the entries of X or $\tilde{x}$). However, it is well known that this basis can be parametrized with n parameters, by replacing $[X^T\ -I_d^T]$ with

$$H^T \equiv \begin{bmatrix} h(1) & h(2) & \dots & h(n) & 0 & \dots & \dots & 0 \\ 0 & h(1) & \dots & h(n-1) & h(n) & 0 & \dots & 0 \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \\ 0 & \dots & 0 & 0 & h(1) & \dots & h(n-1) & h(n) \end{bmatrix}$$

and thus replacing $\tilde{x} \in \mathbb{R}^{nd\times 1}$ in (2.18) by $h \in \mathbb{R}^{n\times 1}$, thereby reducing the number of parameters with $n(d-1)$. ◇

2.4.2 Extension of STLS for nonlinear structures

The extension of the STLS approach towards nonlinear structures was first described in [82] for the STLN framework. Independently we developed a nonlinear extension for the RiSVD framework. In a first subsection we give the derivation that leads to the nonlinear extension in the RiSVD framework. We show that in analogy to the RiSVD framework for affinely structured matrices, the problem is equivalent to solving a set of Riemannian SVD equations for the smallest Riemannian singular value. Therefore we propose a heuristic inverse iteration algorithm for solving this nonlinear extension. The second subsection illustrates the theory and algorithm on a small 4×3 example of a Vandermonde system. In conclusion we briefly consider the ML properties of this STLS estimator for nonlinearly structured matrices.

Derivation

In the RiSVD framework, the extension of the STLS approach towards nonlinear structures can be formulated as follows:

$$\min_t \sum_{i=1}^{k} (t(i) - s(i))^2 \tag{2.19}$$
$$\text{such that } Ty = 0 \text{ and } y^T y = 1,$$

where $T \equiv \mathbf{M}(t) \in \mathbb{R}^{m\times(n+1)}$ is a matrix that depends nonlinearly on the elements of the vector $t \in \mathbb{R}^{k\times 1}$. An example could be

$$\mathbf{M}(t) = \begin{bmatrix} 1 & \dots & 1 & t(n+1) \\ t(1) & \dots & t(n) & t(n+2) \\ t(1)^2 & \dots & t(n)^2 & t(n+3) \\ \vdots & & & \vdots \\ t(1)^{(m-1)} & \dots & t(n)^{(m-1)} & t(n+m) \end{bmatrix},$$

with $\mathbf{M}(\mathbf{t}) : \mathbb{R}^{k\times 1} \to \mathbb{R}^{m\times(n+1)}$, $k = m+n$. If we define $S \equiv \mathbf{M}(s)$, then (2.19) corresponds –in algebraic terms– to finding a rank deficient approximation $T = \mathbf{M}(t)$ of S such that the objective function in (2.19) is minimized and S has the same nonlinear structure as T.
We will now derive a set of Riemnannian SVD equations, by applying the method of Lagrange multipliers to (2.19). The Lagrangian for the latter problem is:

$$L(t,y,l,\lambda) = \sum_{i=1}^{k}(t(i)-s(i))^2 + l^T\mathbf{M}(t)y + \lambda(1-y^Ty). \tag{2.20}$$

At first sight, the nonlinear dependence of T on t makes a derivation similar to the one for affinely structured matrices T impossible. The reason is that at a certain point in the derivation we need to eliminate t out of our equations. However, we can get around this problem in the following way. Make a Taylor series expansion of each entry of $T = \mathbf{M}(t)$ around its corresponding entry in $S = \mathbf{M}(s)$. Symbolically this can be represented as follows:

$$\mathbf{M}(t) = \mathbf{M}(s) + M^{(1)} + M^{(2)} + \dots ,$$

in which the entries of $M^{(j)} \in \mathbb{R}^{m\times(n+1)}$ contain the jth order terms of the Taylor series expansion mentioned before. Let us now define $\hat{H}_2 \in \mathbb{R}^{m\times k}$ as follows

$$(T-S)y \equiv \hat{H}_2(t-s). \tag{2.21}$$

From the definition it should be clear that the entries of $\hat{H}_2$ are functions of the elements of y, s and $t-s$. If necessary we will indicate this dependence by using the matrix function notation: $\hat{H}_2 = \hat{\mathbf{H}}_\mathbf{2}(y,s,t-s)$. Note that if we only retain the first order term of $T = \mathbf{M}(t)$ (i.e. replacing T by $S + M^{(1)}$), the entries of $\hat{H}_2$ are no longer functions of the elements $t-s$, but only of the elements y and s. The Lagrangian (2.20) can now be rewritten as

$$L(t,y,l,\lambda) = \sum_{i=1}^{k}(t(i)-s(i))^2 + l^T(\hat{H}_2(t-s)+Sy) + \lambda(1-y^Ty). \tag{2.22}$$

Taking the derivatives of L w.r.t. t, y, l and λ and setting these equal to 0 we obtain the following equations:

$$2(t-s)+\hat{H}_2^T l = 0 \tag{2.23}$$
$$T^T l = \lambda y \tag{2.24}$$
$$\hat{H}_2(t-s)+Sy = 0 \text{ or } Ty = 0 \tag{2.25}$$
$$y^T y = 1. \tag{2.26}$$

Apparently, the introduction of $\hat{H}_2$ and the corresponding reformulation of the Lagrangian (2.20) into (2.22), allows us to avoid the difficulty related to the nonlinear dependence of T on t. It should however be noted that (2.23) contains an approximation. The exact Lagrange multiplier equation (2.23) should be

$$2(t-s)+\hat{H}_2^T l + \xi = 0$$

where $\xi \in \mathbb{R}^{k\times 1}$ and

$$\xi(i) = l^T \frac{\delta \hat{H}_2}{\delta t(i)}(t-s).$$

Thus by omitting this vector from (2.23), we assume that the entries of $\hat{H}_2$ do not depend on the elements of $t-s$ (since then $\xi = 0$). As mentioned before this corresponds to an approximation of $T = \mathbf{M}(t)$ by its first order approximation $S + M^{(1)}$. The latter means that the simplification performed in (2.23) is valid as long as $t-s$ is small enough to justify the replacement of T by its first order approximation.
We now continue our derivation using the 4 Lagrange multiplier equations (2.23)-(2.26). Multiplying the left and right hand side of (2.24) with y^T and combining this with (2.25) and (2.26) we find that $\lambda = 0$. Substituting (2.23) into (2.25) we obtain

$$0.5\hat{D}_2 l = Sy, \tag{2.27}$$

where $\hat{D}_2 \in \mathbb{R}^{m\times m}$ is defined by $\hat{D}_2 \equiv \hat{H}_2\hat{H}_2^T$. As a consequence the entries of $\hat{D}_2$ are functions of the elements of s, t and $t-s$. If necessary we will indicate this dependence by using the matrix function notation: $\hat{D}_2 = \hat{\mathbf{D}}_{\mathbf{2}}(y, s, t-s)$. Since the entries of $\hat{H}_2$ are linear in the elements of y (see (2.21)) it is always possible to define the matrix $\hat{H}_1 \in \mathbb{R}^{(n+1)\times k}$ such that

$$\hat{H}_1^T y \equiv \hat{H}_2^T l, \tag{2.28}$$

with $\hat{H}_1 \in \mathbb{R}^{(n+1)\times k}$ a matrix which entries depend on the elements of l, s and $t-s$. If necessary we will indicate this dependence by using the matrix function notation: $\hat{H}_1 = \hat{\mathbf{H}}_{\mathbf{1}}(l, s, t-s)$. Multiplying both sides of the equation in (2.21) with l^T to the left and combining this with (2.28) we obtain:

$$y^T (T-S)^T l = y^T \hat{H}_1 (t-s).$$

Since the latter equation is valid for all y, we have:

$$(T-S)^T l = \hat{H}_1(t-s). \tag{2.29}$$

Combining (2.29) and (2.24) we obtain:

$$-S^T l = \hat{H}_1(t-s). \tag{2.30}$$

Substituting (2.23) in (2.30) yields

$$S^T l = 0.5\hat{H}_1\hat{H}_2^T l. \tag{2.31}$$

Substituting (2.28) into (2.31) we obtain

$$S^T l = 0.5\hat{D}_1 y \tag{2.32}$$

where $\hat{D}_1 \in \mathbb{R}^{(n+1)\times(n+1)}$ is defined by $\hat{D}_1 \equiv \hat{H}_1\hat{H}_1^T$. As a consequence the entries of $\hat{D}_1$ are functions of the elements of l, t and $t-s$. Again if necessary this dependence can be shown explicitly by using the matrix function notation: $\hat{D}_1 = \hat{\mathbf{D}}_{\mathbf{1}}(l,s,t-s)$. Absorbing irrelevant constants in l we have derived (see (2.27), (2.32) and (2.26)) the following set of Riemannian equations:

$$\begin{aligned} Sy &= \hat{D}_2 l \\ S^T l &= \hat{D}_1 y \\ y^T y &= 1. \end{aligned} \tag{2.33}$$

Let $l = \tau u$ with $u^T u = 1$. We then have

$$\hat{\mathbf{D}}_{\mathbf{1}}(l,s,t-s) = \tau^2 \hat{\mathbf{D}}_{\mathbf{1}}(u,s,t-s),$$

because the entries of $\hat{D}_1$ are quadratic functions of the elements in l. To avoid notational clutter, we agree that from this point on we assume the symbol $\hat{D}_1$ represents $\hat{\mathbf{D}}_{\mathbf{1}}(u,s,t-s)$. Furthermore from (2.19) it can be seen that the only purpose of the constraint $y^T y = 1$ is to exclude the trivial solution $y = 0$, without excluding any possible direction of y. This is also the reason why the Lagrange multiplier λ associated with the constraint $y^T y = 1$ turned out to be 0. Therefore it is allowed to change the constraint $y^T y = 1$ into $y^T \Xi y = 1$ where $\Xi \in \mathbb{R}^{(n+1)\times(n+1)}$ is a positive semidefinite matrix. Since the matrix $\hat{D}_1$ is positive semidefinite by construction (see the definition of $\hat{D}_1$ in (2.32)) the set of equations (2.33) is equivalent to

$$Sy = \hat{D}_2 u\tau \tag{2.34}$$

$$S^T u = \hat{D}_1 y\tau \tag{2.35}$$

$$u^T u = 1 \tag{2.36}$$

$$y^T \hat{D}_1 y = 1. \tag{2.37}$$

Multiplying the left and right-hand side of (2.34) with $S^T \hat{D}_2^{-1}$ and using (2.35) we obtain

$$S^T \hat{D}_2^{-1} S y = \hat{D}_1 y \tau^2.$$

In combination with (2.37) this becomes

$$y^T S^T \hat{D}_2^{-1} S y = \tau^2. \tag{2.38}$$

In order to give an interpretation to τ we will derive a new formula for the objective function in (2.19) which will also be necessary in the heuristic algorithm described further on. Combining (2.23) with (2.25) we find that

$$l = 2(\hat{H}_2 \hat{H}_2^T)^{-1} S y.$$

Substituting the latter in (2.23) yields

$$t - s = -\hat{H}_2^T (\hat{H}_2 \hat{H}_2^T)^{-1} S y. \tag{2.39}$$

Thus using the definition of $\hat{D}_2$ we find:

$$(t - s)^T (t - s) = y^T S^T \hat{D}_2^{-1} S y. \tag{2.40}$$

Comparing (2.40) with (2.38), we see that the nonlinear extension of STLS can be formulated as follows:

> Find the triplet (u, τ, y) corresponding to the smallest τ, that solves the Riemannian SVD equations (2.34)-(2.37).

Note that to arrive at (2.39) and the interpretation for τ we assumed $\hat{D}_2$ to be invertible. It is however still possible to give the same interpretation for τ when $\hat{D}_2$ is rank deficient. We could e.g. normalize l in the following way: $u = \frac{l}{\tau} = \frac{l}{(l^T \hat{D}_2 l)^{0.5}}$ and then the interpretation follows right away at the expense of a more difficult constraint on u, namely $u^T \hat{D}_2 u = 1$. Formula (2.39) can be changed into

$$t - s = -\hat{H}_2^T (\hat{H}_2 \hat{H}_2^T)^{\dagger} S y, \tag{2.41}$$

where $M^{\dagger}$ denotes the pseudo-inverse of M . The fact that $\hat{D}_2$ is not invertible, means that the system of equations

$$0.5 \hat{D}_2 l = S y$$

is possibly insolvable. Since the latter equation corresponds to one of the Riemannian SVD equations we derived, it is obvious that in the algorithm based on these Riemannian SVD equations the system has to be solvable at convergence. In that case the pseudo-inverse is merely a way to select the smallest

norm solution. As a matter of fact, the example we describe in the next subsection contains a rank deficient $\hat{D}_2$.
Following the interpretation of τ, we propose an inverse iteration algorithm for solving the Riemannian SVD equations (2.34)-(2.37). For notational convenience we use $\Delta s \equiv t - s$. Furthermore, the superscript $[k]$ denotes an entity at the kth iteration.

Nonlinear STLS Algorithm

Input: extended data matrix $S \equiv \mathbf{M}(s)$ (note the structure that has to be preserved is contained in the definition of the matrix function)
Output: y, t

Step 1: Initialization: $k = 0$, $\Delta s^{[0]} = 0$, $[Q\ R] = qr(S)$,
$Q_1 = Q(:, 1 : n+1)$, $Q_2 = Q(:, n+2 : m)$, $R_1 = R(1 : n+1, :)$,
$[U\ \Sigma\ V] = svd(S)$, $x^{[0]} = U(:, n+1)$, $y^{[0]} = V(:, n+1)$,
$\tau^{[0]} = \Sigma(n+1, n+1)$
Step 2: $z^{[k+1]} = R_1^{-T} \hat{D}_1^{[k]}$
Step 3: $w^{[k+1]} = -(Q_2^T \hat{D}_2^{[k]} Q_2)^{-1} Q_2^T \hat{D}_2^{[k]} Q_1 z$
Step 4: $u^{[k+1]} = Q_1 z^{[k+1]} + Q_2 w^{[k+1]}$
Step 5: $y^{[k+1]} = R_1^{-1} Q_1^T \hat{D}_2^{[k+1]} u^{[k+1]}$
Step 6: $u^{[k+1]} = u^{[k+1]} / \|u^{[k+1]}\|_2$
Step 7: $\Delta s^{[k+1]} = -\hat{H}_2^{[k+1]} (\hat{D}_2^{[k+1]})^{\dagger} S y^{[k+1]}$
Step 8: $\gamma^{[k+1]} = (y^{[k+1]})^T \hat{D}_1^{[k+1]} y^{[k+1]}$
Step 9: $y^{[k+1]} = y^{[k+1]} / (\gamma^{[k+1]})^{0.5}$
Step 10: $\tau^{[k+1]} = (y^{[k+1]})^T S^T u^{[k+1]}$
Step 11: $t^{[k+1]} = s + \Delta s^{[k+1]}$
Step 12: if $rank(M(t^{[k+1]})) < n+1$ then stop
else goto Step 2

Note that $[Q\ R] = qr(S)$ is a shorthand notation indicating that $Q \in \mathbb{R}^{m \times m}$ and $R \in \mathbb{R}^{m \times (n+1)}$ form the QR decomposition of $S = QR$, $[U\ \Sigma\ V] = svd(S)$ is a shorthand notation indicating that the columns of $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{(n+1) \times (n+1)}$ contain the left and right singular vectors corresponding to the singular values on the diagonal of $\Sigma \in \mathbb{R}^{m \times (n+1)}$. Furthermore, entities that are calculated using other entities always use the most recently calculated entities. E.g. $\hat{D}_2^{[k+1]}$ in Step 7 is calculated using $y^{[k+1]}$ and $\Delta s^{[k]}$.

Example

In this subsection we describe an example consisting of a nonlinearly structured overdetermined system of equations and determine a solution using the nonlinear extension of the RiSVD framework. We start from a Vandermonde system of equations:

$$\begin{bmatrix} 1 & 1 \\ s(1) & s(2) \\ s(1)^2 & s(2)^2 \\ s(1)^3 & s(2)^3 \end{bmatrix} \begin{bmatrix} x(1) \\ x(2) \end{bmatrix} \approx \begin{bmatrix} b(1) \\ b(2) \\ b(3) \\ b(4) \end{bmatrix}. \tag{2.42}$$

Note that the relation between the vector y in formulation (2.19) and the vector x in the previously mentioned example is $x = -y(1:n)/y(n+1)$, with $n = 2$ in this case. We assume that the right-hand side $b \in \mathbb{R}^{4\times 1}$ is noise free and that the vector $s \in \mathbb{R}^{2\times 1}$ contains noisy measurements. In order to obtain a solution $x \in \mathbb{R}^{2\times 1}$ we have to solve (2.19) with

$$T = \mathbf{M}(t) = \begin{bmatrix} 1 & 1 & b(1) \\ t(1) & t(2) & b(2) \\ t(1)^2 & t(2)^2 & b(3) \\ t(1)^3 & t(2)^3 & b(4) \end{bmatrix}. \tag{2.43}$$

To give an idea on how the involved matrices depend on the different variables of the problem, we give an overview of the most important matrices used in the derivation and in the algorithm:

$$\hat{H}_1 = \begin{bmatrix} \begin{array}{c} u(2)+u(3)(2s(1)+\Delta s(1))+ \\ u(4)(3s(1)^2+3s(1)\Delta s(1)+\Delta s(1)^2) \end{array} & 0 \\ 0 & \begin{array}{c} u(2)+u(3)(2s(2)+\Delta s(2))+ \\ u(4)(3s(2)^2+3s(2)\Delta s(2)+\Delta s(2)^2) \end{array} \\ 0 & 0 \end{bmatrix}$$

$$\hat{H}_2 = \begin{bmatrix} 0 & 0 \\ y(1) & y(2) \\ (2s(1)+\Delta s(1))y(1) & (2s(2)+\Delta s(2))y(2) \\ (3s(1)^2+3s(1)\Delta s(1)+\Delta s(1)^2)y(1) & (3s(2)^2+3s(2)\Delta s(2)+\Delta s(2)^2)y(2) \end{bmatrix}.$$

To test the algorithm, we first generate a consistent Vandermonde system, i.e. we start from a $t_{ex} \in \mathbb{R}^{2\times 1}$ and an $y_{ex} \in \mathbb{R}^{3\times 1}$ such that $T(t_{ex})y_{ex} = 0$ with $\mathbf{M}(t)$ as in (2.43), $t_{ex} = [1.5\ 2.52]^T$, $x_{ex} = [1\ 2]^T$ and $y_{ex} = [x_{ex}^T\ -1]^T/\|[x_{ex}^T\ -1]^T\|_2$. The consistency is obtained by setting $b = \mathbf{M}(t_{ex})(:,1:2)x_{ex}$. Let $w \in \mathbb{R}^{2\times 1}$ be a noise vector, the elements of which are i.i.d. Gaussian noise of standard deviation 0.1. We will now do a Monte-Carlo simulation using 500 noise realizations. In each run we determine the solution $x_{NSTLS} = -(y(1:2))/y(3)$ of problem (2.19) with $s = t_{ex}+w$ and $\mathbf{M}(t)$ as in (2.43). For reasons of comparison we also calculate for each run the LS (x_{LS}) and the TLS solution of the overdetermined Vandermonde system of equations. For the three methods, the relative accuracy of the solution $acc_{meth} \equiv \frac{\|x_{meth}-x_{ex}\|_2}{\|x_{ex}\|_2}$, $meth$ =LS, TLS, NSTLS is averaged over the 500 runs. We obtain the following results:

$$acc_{LS} = 1.934e-2, acc_{TLS} = 1.935e-2, acc_{NSTLS} = 1.233e-13.$$

It is thus obvious that using the nonlinear STLS algorithm we obtain far better results. In the result presented here it seems that the algorithm not only finds a rank deficient approximation $\mathbf{M}(t)$ of $\mathbf{M}(s)$ having the same nonlinear structure, but under the given noise circumstances, it turns out that the nonlinear STLS algorithm also yields a far better statistical accuracy than the LS or TLS estimate. As will be shown in chapter 3, the STLS approach can be shown to be a ML estimator in the case of affinely structured matrices in which the different entries are perturbed by i.i.d. Gaussian noise. As a matter of fact, the cost function (2.5) that is finally used to solve the CTLS formulation, can be seen as a LS minimization of the equation error $h \equiv Ax - b = S[x^T \ -1]^T$. This can be explained as follows. The elements of h obey a Gaussian distribution since they are obtained as linear combinations of Gaussian random variables. Because of the structure in the matrix $[A \ b]$ the elements of h will not be independent and therefore we have to introduce the inverse of the covariance matrix of h as weighting into the LS minimization. The latter covariance matrix is exactly $\mathbf{H}(x)W^{-1}\mathbf{H}(x)^T$ that is used as weigthing in (2.5). Summarizing we can say that because of the Gaussianity of h, we are able to express the maximization of the Likelihood function as a LS problem. It should be clear that when the elements s in (2.42) are perturbed by i.i.d. Gaussian noise, the equation error $Ax - b$ will no longer obey a Gaussian distribution. This is due to the nonlinear structure of the system: when $t(i)$ is an i.i.d. Gaussian random variable $t(i)^2$ is no longer Gaussian. Without going into details, it turns out that for small perturbations –compared to the size of the elements to which they are applied– the distributions of $t(i)^p, p = 2, 3, \ldots$ are still very close to a Gaussian distribution (but with increasing standard deviation of course), meaning that it still might be possible to use only second order statistics to find a statistical optimal solution. However if e.g. the right-hand side in (2.42) is perturbed with i.i.d. Gaussian noise too, the nonlinear STLS looses its statistical optimality (see e.g. a similar example in [82]). A possible explanation could be that now the elements of the equation error h are random variables that are obtained as linear combinations of non-Gaussian and Gaussian distributed random variables, making it very unlikely that a method that uses only the second order statistical information –by solving a LS problem involving the equation error covariance matrix– will also be an estimator with good statistical accuracy.

2.5 STLS problems involving complex data matrices

Until now we have mainly considered STLS problems involving real data matrices. In the remainder of the thesis we will continue to do so. However, since many applications deal with complex data matrices (see the NMR and DOA applications in section 1.2), we describe here how these STLS problems can easily be transformed into STLS problems that only contain real data matrices.

Our starting point is the multivariate STLS formulation (2.17):

$$\begin{aligned} &\min_{\Delta s, X} \Delta s^H W \Delta s \\ &\text{such that } (A+\Delta A)X = B + \Delta B \\ &\text{and } [A+\Delta A \; B+\Delta B] \text{ has the same structure as } [A \; B], \end{aligned} \tag{2.44}$$

with $A, \Delta A \in \mathbb{C}^{m\times n}$, $B, \Delta B \in \mathbb{C}^{m\times d}$, $X \in \mathbb{C}^{n\times d}$ and thus also $\Delta s \in \mathbb{C}^{k\times 1}$. Furthermore $W \in \mathbb{R}^{k\times k}$ is supposed to be symmetric ($W = W^T$). Now decompose A in its real and imaginary part by writing

$$A = A_r + jA_i,$$

where $A_r, A_i \in \mathbb{R}^{m\times n}$ contain respectively the real and imaginary part of the entries of A. If the same is done for ΔA, B, ΔB, Δs and X, it is straightforward to show that (2.44) can be transformed into:

$$\begin{aligned} &\min_{\Delta \check{s}, \check{X}} \Delta \check{s}^T \check{W} \Delta \check{s} \\ &\text{such that } (\check{A}+\Delta \check{A})\check{X} = \check{B} + \Delta \check{B} \\ &\text{and } [\check{A}+\Delta \check{A} \; \check{B}+\Delta \check{B}] \text{ has the same structure as } [\check{A} \; \check{B}], \end{aligned} \tag{2.45}$$

where

$$\begin{aligned} &\Delta\check{s} \equiv [\Delta s_r^T \; \Delta s_i^T]^T \in \mathbb{R}^{2k\times 1}, \check{X} \equiv [X_r^T \; X_i^T]^T \in \mathbb{R}^{2n\times d}, \\ &\check{W} \equiv I_2 \otimes W \in \mathbb{R}^{2k\times 2k}, \\ &\check{A} \equiv \begin{bmatrix} A_r & -A_i \\ A_i & A_r \end{bmatrix} \in \mathbb{R}^{2m\times 2n}, \Delta\check{A} \equiv \begin{bmatrix} \Delta A_r & -\Delta A_i \\ \Delta A_i & \Delta A_r \end{bmatrix} \in \mathbb{R}^{2m\times 2n} \\ &\check{B} = [B_r^T \; B_i^T]^T \in \mathbb{R}^{2m\times d}, \Delta\check{B} = [\Delta B_r^T \; \Delta B_i^T]^T \in \mathbb{R}^{2m\times d}. \end{aligned}$$

Note that if e.g. $[A \; B]$ has a Hankel structure, then $[\check{A} \; \check{B}]$ has a block Hankel structure, thereby implying that an algorithm developed for real Hankel matrices needs to be modified if complex Hankel matrices are used.

2.6 Conclusion

In this chapter we have introduced the Structured Total Least Squares problem. By using simple examples, we have shown that the STLS problem is more than a simple extension of the TLS problem. Furthermore, we have presented an overview of the 3 most important approaches found in literature: the CTLS approach, the RiSVD approach and the STLN approach. Under mild conditions we have proven the equivalence of the three approaches when used for solving the STLS problem. Further extensions towards multivariate STLS problems, nonlinearly structured matrices,... have been brought together in the STAP framework. The theory for the extension of the RiSVD framework towards nonlinear structures has been derived and illustrated with a Vandermonde system example. Finally it has been demonstrated how an STLS problem involving complex data matrices can be transformed into an STLS problem involving only real data matrices.

Chapter 3

Properties of the STLS problem

This chapter concerns properties and interpretations of the univariate STLS problem. In the first section we consider the existence of a solution to the STLS problem. By introducing the definition of nongeneric STLS problems, it is shown that under mild conditions, these nongeneric STLS problems fail to have a solution. The theory is backed by some examples. Furthermore, the non-uniqueness of the solution is illustrated by means of several examples.
A projection interpretation of the STLS problem is developed in the second section. The latter interpretation leads to a new definition of structure-dependent signal and noise subspaces, whereby the STLS approach can be seen as a subspace approach involving structure-dependent subspaces. Using the geometrical insight provided by the projection interpretation, it is straightforward to derive an upperbound to the STLS objective function (weighted sum of squares). For reasons of completeness a lower bound to the same function is added as well. The structured-dependent subspace interpretation is illustrated on a Hankel matrix example and on an unstructured matrix example. The usefulness of the derived bounds is demonstrated using 4 *examples.*
Furthermore, the invertibility of the matrix $\mathbf{D_2}(y)$*, appearing in the projection interpretation as well as in many other places (e.g. RiSVD framework), is addressed. The theory is backed by several examples. For one of them (symmetric Toeplitz matrix), it is proven in detail where the invertibility problem comes from and how it can be circumvented in STLS problems involving this particular structure.*
Section 3.3 treats the STLS problem involving Hankel matrices. The analysis proceeds by using a new parametrization of the set of rank deficient Hankel matrices. It is then shown how the difficulty of this particular STLS prob-

lem is related to the parameters of the new parametrization. The analysis can straightforwardly be extended to Toeplitz matrices, block Hankel matrices and block Toeplitz matrices.
The last section contains some orthogonality properties of the STLS solution as wel as a ML interpretation of the STLS problem.

3.1 Existence and uniqueness

In this section the existence and uniqueness of a solution of the STLS problem are discussed. We consider the univariate STLS problem:

$$\begin{aligned} &\min_{\Delta s, x} \Delta s^T W \Delta s \\ &\text{such that } (A + \Delta A)x = b + \Delta b \\ &\text{and } [A + \Delta A \; b + \Delta b] \text{ has the same structure as } [A \; b], \end{aligned} \tag{3.1}$$

where $\Delta s \in \mathbb{R}^{k \times 1}$ contains the different elements of the correction matrices $\Delta A \in \mathbb{R}^{m \times n}$ and $\Delta b \in \mathbb{R}^{m \times 1}$. In the following example a univariate STLS problem is stated in which the Hankel structure of the extended data matrix $[A \; b]$ needs to be preserved.

Example 3.1.1 Let $A = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$ and $b = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$. The problem to be solved is the STLS problem (3.1) with $W = diag([1 \; 2 \; 3 \; 2 \; 1])$ and preservation of the Hankel structure of $[A \; b]$. □

It will now be shown that the particular STLS problem described in example 3.1.1 has no solution. If the following choices are made

$$[\Delta A \; \Delta b] = \begin{bmatrix} -1 & \epsilon & 0 \\ \epsilon & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \text{ or } \Delta s = [-1 \; \epsilon \; 0 \; 0 \; 0]^T \tag{3.2}$$

$$\text{and } x = [\frac{1}{\epsilon} \; 0]^T,$$

it is clear that the constraints in (3.1) are satisfied. Furthermore, as will be shown in section 3.2 1 is a lower bound on the objective function $\Delta s^T W \Delta s$ in this particular example. Using the ΔA and Δb proposed in (3.2) it is easy to see that the objective function equals $1 + 2\epsilon^2$. The latter means that for $\epsilon \to 0$ the objective function approaches its lower bound 1. However, since the corresponding $x = [1/\epsilon \; 0]^T$, the lower bound is never achieved and thus there

is no solution to the STLS problem described in Example 3.1.1.
Consider the following problem

$$\begin{aligned}
&\min_{\Delta s, y} \Delta s^T W \Delta s \\
&\text{such that } [A+\Delta A \ b+\Delta b]y = 0, y^T y = 1 \\
&\text{and } [A+\Delta A \ b+\Delta b] \text{ has the same structure as } [A \ b],
\end{aligned} \tag{3.3}$$

where $\Delta s \in \mathbb{R}^{k\times 1}$ contains the different elements of the correction matrices $\Delta A \in \mathbb{R}^{m\times n}$ and $\Delta b \in \mathbb{R}^{m\times 1}$, $y \in \mathbb{R}^{(n+1)\times 1}$. Note that the only difference between (3.1) and (3.3) resides in the constraints: it is easy to turn (3.3) into (3.1) by replacing the constraint $y^T y = 1$ with the constraint $y(n+1) = -1$. It is clear that a solution of problem (3.3) with the matrices ΔA, Δb and W as described in example 3.1.1 is given by:

$$[\Delta A \ \Delta b] = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, y = [1 \ 0 \ 0]^T. \tag{3.4}$$

The reason why a solution can be found for (3.3) but not for (3.1), follows from the different constraint on y. The constraint $y(n+1) = -1$ prevents $y(n+1)$ from becoming 0 as happens in (3.4). As indicated [11] the problem with example 3.1.1 is caused by the fact that the set of feasible points of (3.1) (i.e. the set of points that satisfy the constraints in (3.1)), is not closed. Therefore, the set of feasible points does not contain all its limit points. More specifically if we take the sequence of matrices $[\Delta A \ \Delta b]$ described in (3.2) for $\epsilon \to 0$, we see that the objective function value approaches its lower bound but since the limit point of the sequence (which is the matrix $[\Delta A \ \Delta b]$ from the solution of (3.3)) is not in the set of feasible points of problem (3.1)), (3.1) has no solution. As proved by the example 3.1.1, problem (3.1) sometimes fails to have a solution. Using a proof similar to the one presented in [11], it is possible to show –under mild conditions– that (3.3) always has a solution.

Theorem 3.1.2 *Let $f_{\Delta s} : \Delta s \to [A+\Delta A \ b+\Delta b] = \mathbf{M}(s+\Delta s)$ be a continuous function (where Δs contains the different elements of $[\Delta A \ \Delta b]$, s contains the different elements of S and M is the matrix function defining the structure). Furthermore assume W in (3.3) to be positive definite. If there exist a $\Delta\tilde{s}$ and corresponding $\Delta\tilde{A}$ and $\Delta\tilde{b}$ such that the constraints in (3.3) are satisfied, then (3.3) itself has a solution.*

Proof: First of all note that the set of rank deficient matrices is closed. Furthermore the set of structured matrices $\mathbf{M}(s+\Delta s)$ is closed. Thus the set of matrices that satisfies the constraints of (3.3), being the intersection of two closed sets, is a closed set itself. Moreover, since $f_{\Delta s}$ is a continuous function, the set $\{\Delta s | rank(\mathbf{M}(s+\Delta s)) < n+1\}$ is closed too. Due to the existence of a feasible point $\Delta\tilde{s}$ for the constraints in (3.3) (i.e. that $rank(\mathbf{M}(s + \Delta\tilde{s})) < n + 1$), the domain of Δs over which the problem needs to be solved can be restricted to the intersection of the following sets: $\{\Delta s | rank(\mathbf{M}(s + \Delta s)) < n+1\}$ and $\{\Delta s | \Delta s^T W \Delta s \leq \Delta\tilde{s}^T W \Delta\tilde{s}\}$. The first set is closed and the second set is compact (i.e. closed and bounded) because of the positive definiteness of W and therefore their intersection is compact too. Summarizing, we have to find the minimum of the objective function in (3.3) over a compact nonempty set. Thus, according to Weierstrass' theorem there will always exist a solution to (3.3). □

The fact that it is always possible under mild conditions to find a solution to (3.3), allows us to define the following class of STLS problems (3.1).

Definition 3.1.3 *An STLS problem (3.1) is called nongeneric when all*[1] *the optimal solutions of the corresponding problem (3.3) satisfy* $y(n + 1) = 0$.

To link this property of nongenericity to the existence or non-existence of a solution to (3.1), the following definition, introduced in [11], concerning affinely structured matrices is stated.

Definition 3.1.4 *Consider an affinely structured matrix* $\mathbf{M}(s) = [A\ b] \in \mathbb{R}^{m\times(n+1)}$, $s \in \mathbb{R}^{k\times 1}$. *Using the matrix function notation for A and b we can say* $\mathbf{M}(s) = [\mathbf{M_1}(s)\ \mathbf{m_2}(s)]$ *with* $A = \mathbf{M_1}(s)$, $b = \mathbf{m_2}(s)$. *A matrix structure* $\mathbf{M}(s)$ *is called regularly structured when for all* $t \in \mathbb{R}^{k\times 1}, h \in \mathbb{R}^{n\times 1}$ *for which* $\mathbf{M_1}(t)h = 0$, *the following holds:* $rank([\mathbf{M_1}(t)\ \mathbf{H}(h)]) = m$ *with* $\mathbf{H}(h) \in \mathbb{R}^{m\times k}$ *defined by* $(\mathbf{M_1}(t) - \mathbf{M_1}(s))h = \mathbf{H}(h)(t - s)$.

It can easily be checked that e.g. Toeplitz and Hankel matrices are regularly structured. The following lemma is needed in the proof that relates the nongenericity property to the existence of a solution of (3.1).

Lemma 3.1.5 *Consider the following system of equations:* $Gh = g$, *with* $G \in \mathbb{R}^{p\times q}$, $g \in \mathbb{R}^{p\times 1}$, $h \in \mathbb{R}^{q\times 1}$ *and* $rank(G) = p$. *Let* h_0 *be a solution of* $Gh = g$. *Furthermore, let* $F : \mathbb{R}^q \times \mathbb{R} \to \mathbb{R}^p$ *be a continuous function satisfying* $\lim_{\epsilon\to 0} F(h, \epsilon) = 0, \forall h \in \mathbb{R}^{q\times 1}$. *Then for all* $\delta > 0$ *there exists an* $\epsilon_0 > 0$ *such that the system of equations* $Gh = g + F(h, \epsilon)$ *has a solution denoted by* h_ϵ *for all* $0 < \epsilon \leq \epsilon_0$ *and such that* $\|h_\epsilon - h_0\|_2 \leq \delta$.

Proof: See [12]. □

The latter allows us to prove the following theorem, which is an adaptation of Theorem 2.3 in [12].

[1] the optimal solution might not be unique as will be shown further on

Theorem 3.1.6 *Let* $\mathbf{M}(s) = [A\ b]$ *be a regularly and affinely structured matrix. If problem (3.1) is nongeneric, it has no solution.*

Proof: The fact that (3.1) is nongeneric means that all optimal solutions of the corresponding problem (3.3) satisfy $y(n+1) = 0$. Let ΔA^*, Δb^* and y^* be an optimal solution of (3.3), with $y^*(n+1) = 0$. To prove the theorem, we will show that $[A + \Delta A^*\ \ b + \Delta b^*]$ is a limit point of a sequence of matrices $[A + \Delta A_\epsilon\ b + \Delta b_\epsilon]$ that together with an x_ϵ satisfy the constraints in (3.1). Due to the nongenericity the limit point can never be achieved, since that would contradict the assumptions of the theorem.
In order to simplify notations we will use the matrix function representation of A and b: $\mathbf{M}(s) = [A\ b]$. Let $\mathbf{M_1}(s) = \mathbf{M}(s)(:, 1:n)$ and $\mathbf{m_2}(s) = \mathbf{M}(s)(:, n+1)$. Thus if $[A + \Delta A^*\ b + \Delta b^*] = \mathbf{M}(t^*)$, we have $A + \Delta A^* = \mathbf{M_1}(t^*)$ and $b + \Delta b^* = \mathbf{m_2}(t^*)$. Furthermore let $x^* = y^*(1:n)$. It is easy to show that the following system of equations

$$\mathbf{M_1}(t^* + \epsilon\bar{t})(x^*/\epsilon + \bar{x}) = \mathbf{m_2}(t^* + \epsilon\bar{t}) \tag{3.5}$$

is equivalent to

$$[\mathbf{M_1}(t^*)\ \mathbf{H}(x^*)] \begin{bmatrix} \bar{x} \\ \bar{t} \end{bmatrix} = \mathbf{m_2}(t^*) + F(\epsilon, \bar{t}, \bar{x}), \tag{3.6}$$

with $\mathbf{H}(x^*) \in \mathbb{R}^{m \times k}$ similarly defined as in definition 3.1.4, namely $\mathbf{H}(x^*)\epsilon\bar{t} = (\mathbf{M_1}(t^* + \epsilon\bar{t}) - \mathbf{M_1}(t^*))x^*$ and $F(\epsilon, \bar{t}, \bar{x}) = \mathbf{m_2}(t^* + \epsilon\bar{t}) - \mathbf{m_2}(t^*) + (\mathbf{M_1}(t^* + \epsilon\bar{t}) - \mathbf{M_1}(t^*))\bar{x}$. Since $\mathbf{M}(s)$ is regularly structured, $rank([\mathbf{M_1}(t^*)\ \mathbf{H}(x^*)]) = m$. Since furthermore the map F is continuous and $\lim_{\epsilon \to 0} F(\epsilon, \bar{t}, \bar{x}) = 0$, we can apply lemma 3.1.5. The latter implies that it is always possible to find an ϵ_0 such that $\forall \epsilon : 0 < \epsilon \leq \epsilon_0$ a $\bar{t}$ and $\bar{x}$ can be found such that (3.5) is satisfied. Furthermore, according to lemma 3.1.5, $\bar{t}$ is bounded. Thus for all ϵ, $0 < \epsilon \leq \epsilon_0$ an x_ϵ exists such that $\mathbf{M_1}(t^* + \epsilon\bar{t})x_\epsilon = \mathbf{m_2}(t^* + \epsilon\bar{t})$. In other words we have found a sequence of solutions of (3.1) and the limit point of the sequence of matrices $[\mathbf{M_1}(t^* + \epsilon\bar{t})\ \mathbf{m_2}(t^* + \epsilon\bar{t})]$ is the set of matrices that appears in the solution of (3.3), namely $[\mathbf{M_1}(t^*)\ \mathbf{m_2}(t^*)]$. Thus the cost function of (3.1) at the limit point of the sequence of solutions of (3.1), equals the value of the cost function of (3.3) at its solution. However, if the above mentioned sequence would achieve its limit point, we would have found a solution to (3.1) with $y(n+1) \neq 0$. Since every solution of (3.1) is obviously a solution of (3.3), this contradicts the assumption of nongenericity of (3.1). $\square$

The following example illustrates the importance of the regularity assumption of the structure for relating the nongenericity property to the non-existence of a solution of (3.1).

Example 3.1.7 Consider the following matrix function $\mathbf{M}(m) : \mathbb{R}^{2\times 1} \rightarrow \mathbb{R}^{3\times 3} : m \rightarrow \begin{bmatrix} 1 & 1 & 2 \\ 2 & 2 & m(2) \\ 1 & m(1) & 2 \end{bmatrix}$. Let $s = [0\ 6]^T$ and $[A\ b] = \mathbf{M}(s)$. □

The solution of (3.3) with the matrices A and b as defined in example 3.1.7, is obviously $[A + \Delta A\ \ b + \Delta b] = \mathbf{M}(t)$ with $t = [1\ 0]^T$ and $y = [1\ \ -1\ 0]$ or $y = [-1\ 1\ 0]$. Both optimal solutions satisfy $y(n+1) = 0$. However (3.1) still has an optimal solution, namely $[A + \Delta A\ \ b + \Delta b] = \mathbf{M}(t)$ with $t = [0\ -2]^T$ and $y = [1\ 0\ 1]$ or $y = [-1\ 0\ \ -1]$. The reason is of course that the matrix structure defined by the matrix function $\mathbf{M}(m)$ in example 3.1.7 is not regular. This can be easily seen by checking the rank of the matrix $[\mathbf{M_1}(t)\ \mathbf{H}(h)] = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 2 & 2 & 0 & 0 \\ 1 & t(1) & h(2) & 0 \end{bmatrix}$ for $t = [1\ 0]^T$ and $h = [1\ \ -1]^T$ (i.e. we take h equal to $y(1:n)$ of the first solution y of (3.3)). We then obtain:

$$[\mathbf{M_1}(t)\ \mathbf{H}(h)] = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 2 & 2 & 0 & 0 \\ 1 & 1 & -1 & 0 \end{bmatrix}$$

which has rank 2 and thus it can be concluded that the matrix $\mathbf{M}(s)$ is not regularly structured.
At this point it is interesting to note that the different formulations reviewed in section 2.3 correspond to either the formulation in (3.1) or in (3.3). The STLN and CTLS approach correspond to (3.1), whereas the RiSVD approach is equivalent to (3.3). However, most STLS problems are generic and in that case (3.1) and (3.3) are fully equivalent as was proven in section 2.3 (where the equivalence of the STLS, CTLS and STLN approach was proven under the genericity assumption and thus also the equivalence of (3.1) and (3.3) under the same assumption).
Finally we indicate with 2 small examples that the solution of the STLS problem is not necessarily unique.

Example 3.1.8 Let $S = \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$, the structure to be preserved in (3.1) is the Toeplitz structure of S. □

In this paragraph it is assumed that the STLS problems are generic. The latter will be confirmed by the fact that –for the examples considered– there are always optimal solutions of problem (3.3) with $y(n+1) \neq 0$. Therefore, in order to investigate the number of optimal solutions of the STLS problem we can evaluate the cost function described in lemma 2.3.2 over the unit circle. For convenience the cost function is repeated here:

$$f_{STLS} = y^T S^T \mathbf{D_2}(y)^{-1} S y,$$

where the entries of $\mathbf{D_2}(y) \in \mathbb{R}^{k \times k}$ are quadratic functions of the elements of y. Moreover, since the cost function is evaluated over the unit circle, the cost function f_{STLS} in figure 3.1 is expressed in polar coordinates:

$$y(1) = cos(\theta), y(2) = sin(\theta).$$

Note that the cost function is only plotted over the region $0 \leq \theta < \pi$. This is because the y vectors in the region $\pi \leq \theta < 2\pi$ correspond to opposite y vectors of the region depicted. Since the cost function was shown to be scaling invariant in section 2.3, the cost function over the region $\pi \leq \theta < 2\pi$ is a copy of the cost function over the depicted region. From figure 3.1 it is obvious that there are 2 optimal solutions, namely one at $\theta = 0.3927$ and one at $\theta = 1.9635$. In both θ values the cost function is 2. Using the theory presented in the next

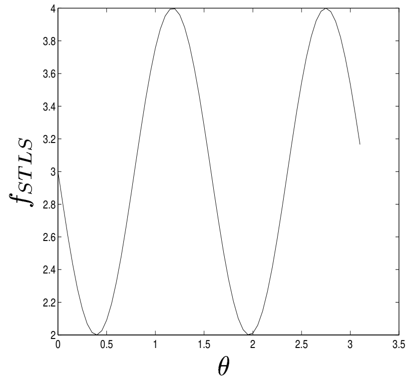


Figure 3.1: *This figure shows the cost function $f_{STLS} = y^T S^T \mathbf{D_2}(y)^{-1} S y$ as a function of y expressed in the polar coordinate θ, for example 3.1.8.*

section it is possible to compute the corresponding matrix ΔS once the vector y is known. This yields: $\Delta S = \begin{bmatrix} -0.5 & 1.2071 \\ 0.2071 & -0.5000 \end{bmatrix}$ for the first θ value, and $\Delta S = \begin{bmatrix} -0.5 & -0.2071 \\ -1.2071 & -0.5000 \end{bmatrix}$ for the second θ value.
The next example even has an infinite number of optimal solutions.

Example 3.1.9 Let $S = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$. The structure to be preserved in the STLS problem is the Toeplitz structure of S. □

Instead of showing the graph of f_{STLS} applied to example 3.1.9, it is shown that the problem can be solved analytically. Straightforward calculations show that for the simple 2×2 Toeplitz structure (as in the previous example, we only have to evaluate the cost function over the unit circle, using polar coordinates),

$$\mathbf{D_2}(y)^{-1} = \begin{bmatrix} \frac{2+cos(\theta)^2 sin(\theta)^2}{1+sin(\theta)^2} & -cos(\theta)sin(\theta) \\ -cos(\theta)sin(\theta) & 1+sin(\theta)^2 \end{bmatrix}.$$

It then follows that for the matrix S described in example 3.1.9, the following holds:

$$[cos(\theta)\ sin(\theta)]S^T\mathbf{D_2}(y)^{-1}S\begin{bmatrix} cos(\theta) \\ sin(\theta) \end{bmatrix} = 1.$$

The latter means that every vector y on the unit circle is an optimal STLS solution. Again using the theory of the next section, we find the corresponding matrices ΔS:

$$\Delta S = \begin{bmatrix} -sin(\theta)cos(\theta) & cos(\theta)^2 \\ sin(\theta)^2 & -sin(\theta)cos(\theta) \end{bmatrix}.$$

3.2 Projection interpretation of the STLS problem

In this section the STLS problem for affinely structured data matrices S is interpreted in terms of a projection on a signal subspace. The latter differs from the classical SVD based definition of a signal subspace, since it depends on the structure of the matrix S. It is assumed that the STLS problems are generic, so that formulation (3.3) can be used as STLS formulation.
In subsection 3.2.1 the projection interpretation is derived using the method of Lagrange multipliers. Furthermore, new definitions for signal and noise subspace are introduced. The next subsection shows how the projection interpretation can be used to derive an upper bound on the objective function $\Delta s^T W \Delta s$ at the solution of the STLS problem. Knowledge of such an upper bound is very useful since due to the non-convexity of the STLS problem (3.3) (see example 2.1.1 and also section 3.3), any algorithm for solving it risks to get stuck in a local minimum. As a matter of fact, all the existing algorithms –whether they are derived from the CTLS, STLN or STLS formulation– suffer from this local minima problem, making the choice of good starting values an important issue (see section 4.5). It is clear that an upper bound on the objective function in (3.3) at the solution of (3.3) provides a way to detect bad starting values (i.e. starting values that lead to a solution for which the objective function is larger than the upper bound). For reasons of completeness, we also add the derivation of a lower bound on the same objective function. Subsection 3.2.3 illustrates subsection 3.2.1 on two special cases: a Hankel matrix and an unstructured matrix. The latter example is used to show that in that case our definition of signal subspace coincides with the classical one based on the SVD of the matrix S. Finally, in subsection 3.2.4 we illustrate the bounds with some numerical examples of Hankel matrices.

3.2.1 Derivation of projection interpretation of STLS

Before applying the method of Lagrange multipliers to (3.3) we repeat some important matrices and matrix functions that were defined in previous chapters and that will frequently be used throughout this derivation. First of all we have the structure dependent matrices $F_i \in \mathbb{R}^{m\times k}$ that were defined in subsection 2.2.1 in the following way:

$$\Delta S(:,i) \equiv F_i \Delta s, i = 1,\dots,n+1.$$

We then have that

$$\begin{aligned} \Delta S y &= [F_1\Delta s \;\dots\; F_{n+1}\Delta s]y \\ &= \sum_{i=1}^{n+1} F_i \Delta s y(i) \\ &= (\sum_{i=1}^{n+1} F_i y(i))\Delta s \\ &= \mathbf{H_2}(y)\Delta s, \end{aligned} \tag{3.7}$$

where the matrix function $\mathbf{H_2}(y)$ is defined by the last equality, i.e. $\mathbf{H_2}(y) : \mathbb{R}^{(n+1)} \to \mathbb{R}^{m\times k} : y \to \sum_{i=1}^{n+1} F_i y(i)$. Since the weight matrix W would only increase the computational complexity, the following new vectors are introduced:

$$\tilde{s} \equiv W^{0.5}s \text{ and } \Delta\tilde{s} \equiv W^{0.5}\Delta s.$$

Defining a matrix $\tilde{\mathbf{H}}_\mathbf{2}(y) \equiv \mathbf{H_2}(y)W^{-0.5}$ the last equation of (3.7) becomes

$$\Delta S y = \tilde{\mathbf{H}}_\mathbf{2}(y)\Delta\tilde{s}. \tag{3.8}$$

It is assumed that ΔS has the same structure as S and therefore, using a derivation similar to (3.7) and (3.8), we find also that

$$Sy = \tilde{\mathbf{H}}_\mathbf{2}(y)\tilde{s}. \tag{3.9}$$

With the newly introduced variables, (3.3) can be written as:

$$\begin{aligned} &\min_{\Delta\tilde{s},y} \Delta\tilde{s}^T\Delta\tilde{s} \\ &\text{such that } [A+\Delta A\; b+\Delta b]y = 0, y^Ty = 1 \\ &\text{and } [A+\Delta A\; b+\Delta b] \text{ has the same structure as } [A\; b], \end{aligned} \tag{3.10}$$

Using the theory of Lagrange multipliers we can derive a set of equations that characterizes the stationary points of problem (3.10). The Lagrangian for problem (3.10) is:

$$L(\Delta\tilde{s}, y, l, \lambda) = \Delta\tilde{s}^T\Delta\tilde{s} + l^T(S+\Delta S)y + \lambda(y^Ty-1), \tag{3.11}$$

with $l \in \mathbb{R}^{m\times 1}$ and $\lambda \in \mathbb{R}$ the Lagrange multipliers. Taking the derivatives of the Lagrangian L with respect to $\Delta\tilde{s}$, y, l and λ and putting them equal to 0 we get the following equations:

$$2\Delta\tilde{s} + \tilde{\mathbf{H}}_{\mathbf{2}}(y)^T l = 0 \tag{3.12}$$
$$(S+\Delta S)^T l + 2\lambda y = 0 \tag{3.13}$$
$$(S+\Delta S)y = 0 \tag{3.14}$$
$$y^T y = 1. \tag{3.15}$$

Multiplying (3.13) with y^T to the left and using (3.14) and (3.15), we find that $\lambda = 0$. This is just a mathematical way for saying that the constraint $y^T y = 1$ is mainly used to avoid the trivial solution $y = 0$, but could easily be replaced by $y^T D y = 1$ with D a square positive definite matrix. From (3.12) we obtain that

$$\Delta\tilde{s} = -0.5\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T l. \tag{3.16}$$

Using (3.8) and substituting (3.16) into (3.14) yields

$$l = 2(\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T)^{-1} S y. \tag{3.17}$$

Substituting (3.17) into (3.16) we see that

$$\Delta\tilde{s} = -\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T(\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T)^{-1} S y. \tag{3.18}$$

Applying property (3.9) to (3.18) gives

$$\Delta\tilde{s} = -\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T(\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T)^{-1}\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{s}. \tag{3.19}$$

Looking at (3.19) we see that this represents a projection operation. In fact $\Delta\tilde{s}$ is the result of a projection of $-\tilde{s}$ on the column space of $\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T$. Let us now define the noise subspace N_y as the space generated by the columns of $\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T$ and its orthogonal complement $N_y^{\perp}$ is then called the signal subspace. Note that with this definition of noise and signal subspace, both spaces depend on the weighting matrix W, the parameter vector y and the structure of the matrix S. To clarify the latter dependence we write down the formula for $\tilde{\mathbf{H}}_{\mathbf{2}}(y)$:

$$\tilde{\mathbf{H}}_{\mathbf{2}}(y) = (\sum_{i=1}^{n+1} F_i y(i)) W^{-0.5}.$$

The dependence on the structure of S stems from the fact that the matrices $F_i, i = 1, \ldots, n+1$ are different when another structure for S is considered. Using these definitions of signal and noise subspace we can say –using (3.19)– that $\tilde{s} + \Delta\tilde{s}$ being

$$(I - \tilde{\mathbf{H}}_{\mathbf{2}}(y)^T(\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T)^{-1}\tilde{\mathbf{H}}_{\mathbf{2}}(y))\tilde{s},$$

is the projection of $\tilde{s}$ on the signal subspace. This is an interesting observation if we compare this to the original problem formulation in (3.10). In words, the latter could be expressed as follows: find the rank-deficient matrix $S+\Delta S$ that is closest (in some metric defined by W) to the original matrix S and has the same structure. Arriving at (3.19) we have changed from a matrix formulation to a signal formulation. Since (3.19) combined with the objective function in (3.10) and the constraint $y^T y = 1$, is equivalent to problem (3.10) itself, the STLS approach can be reformulated as follows: find a parameter vector y such that the projection of $\tilde{s}$ on the signal subspace $N_y^{\perp}$ is as close (in L_2 norm) as possible to the original signal $\tilde{s}$. This is illustrated in figure 3.2. In section 3.2.3 the new notions of signal and noise subspace will be applied to Hankel structured matrices and unstructured matrices.

3.2.2 Derivation of an upper bound

In signal processing the knowledge of an upper bound of the objective function in (3.10) at the global minimum of problem (3.10), can be interesting for several reasons. As already mentioned in the introduction, it provides a means for detecting bad starting values. However also from the application point of view an upper bound could provide interesting information. E.g. in the speech compression example described in [59, 60], it could be used to determine a priori the number of complex damped exponentials that are needed to fit a frame of a speech signal in a satisfactory way (i.e. the value of the objective function in (3.10) at the solution, should be small enough). We will now explain how such an upper bound can be deduced from the projection interpretation derived in section 3.2.1.
The importance of the projection interpretation for deriving an upper bound, lies in formula (3.19). This formula allows us to construct a rank-deficient structured matrix $S + \Delta S$ starting from an arbitrary y. Basically this means that (3.10) can equivalently be formulated as:

$$\min_y \mathbf{m}(y)^T \mathbf{m}(y)$$
$$\text{such that } y^T y = 1,$$

where $\mathbf{m}(\mathbf{y}) = \Delta\tilde{\mathbf{s}}$ as given by formula (3.19), i.e.

$$\mathbf{m} : \mathbb{R}^{(n+1)\times 1} \rightarrow \mathbb{R}^{k\times 1} : y \rightarrow -\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T (\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T)^{-1}\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{s}.$$

If we denote the parameter vector at the solution of (3.10) by y_*, we get for an arbitrary y

$$\mathbf{m}(y)^T \mathbf{m}(y) \geq \mathbf{m}(y_*)^T \mathbf{m}(y_*). \tag{3.20}$$

However, picking an arbitrary y would lead to a conservative upper bound. Therefore we would like to choose a y such that in case S is already rank-deficient, the equality in (3.20) holds. Obviously this can be achieved by setting

y equal to the right singular vector of S corresponding to the smallest singular value. This gives the following upper bound:

$$\tilde{s}^T \tilde{\mathbf{H}}_2(y_{**})^T (\tilde{\mathbf{H}}_2(y_{**}) \tilde{\mathbf{H}}_2(y_{**})^T)^{-1} \tilde{\mathbf{H}}_2(y_{**}) \tilde{s}, \tag{3.21}$$

with $y_{**} = V(:, n+1)$ and $U \Sigma V^T = S, U \in \mathbb{R}^{m \times m}, \Sigma \in \mathbb{R}^{m \times (n+1)}, V \in \mathbb{R}^{(n+1) \times (n+1)}$ the singular value decomposition of S.
For reasons of completeness, we also give a derivation for a lower bound on the value of the objective function in (3.10) at its solution, for the case W is a diagonal matrix with positive weights. Construct a vector $w(i) \equiv W(i,i), i = 1, \ldots, k$. First define a vector $a \in \mathbb{R}^{k \times 1}$ such that $a(i)$ represents the number of times the error $\Delta s(i)$ appears in the matrix ΔS. The following then is obvious:

$$\begin{aligned}
& w(1)(\Delta s(1))^2 + \ldots + w(k)(\Delta s(k))^2 = \\
& \underbrace{\frac{w(1)}{a(1)}(\Delta s(1))^2 + \ldots + \frac{w(1)}{a(1)}(\Delta s(1))^2}_{a(1)} + \ldots + \underbrace{\frac{w(k)}{a(k)}(\Delta s(k))^2 + \ldots + \frac{w(k)}{a(k)}(\Delta s(k))^2}_{a(k)} \geq \\
& \min_i \left(\frac{w(i)}{a(i)}\right)(a(1)(\Delta s(1))^2 + \ldots + a(k)(\Delta s(k))^2) = \\
& \min_i \left(\frac{w(i)}{a(i)}\right) \|\Delta S\|_F^2 \geq \min_i \left(\frac{w(i)}{a(i)}\right) \sigma_{n+1}^2,
\end{aligned} \tag{3.22}$$

where σ_{n+1} is the smallest singular value of the matrix S. The last inequality follows from the fact that a rank-deficient structured approximation can never be closer (in Frobenius norm) to the original matrix than the closest unstructured rank-deficient approximation.

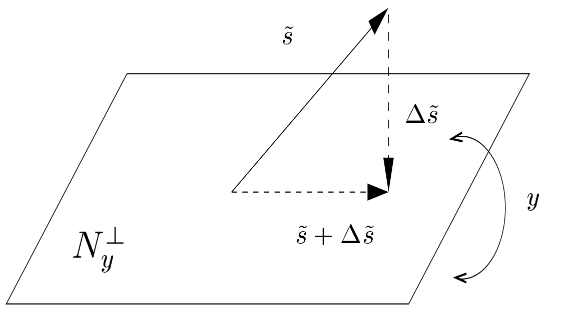


Figure 3.2: *This figure gives a graphical interpretation of problem (3.10). Solving the problem consists in finding that parameter vector y, such that the difference between the signal vector $\tilde{s}$ and its projection on the signal subspace $N_y^{\perp}$ is as small as possible in L_2 norm.*

3.2.3 Two examples

In this section we illustrate the analysis of subsection 3.2.1 on 2 examples: a Hankel matrix and an unstructured matrix. We start with the natural decomposition of a (rank deficient) Hankel matrix, i.e. the Vandermonde decomposition. We show that writing this decomposition in vector notation leads directly to the definitions of signal and noise subspace that we defined in subsection 3.2.1 in a rather abstract manner.
Since every[2] rank-deficient $m \times (n+1)$ Hankel matrix $S + \Delta S$ of rank n can be written as the following Vandermonde decomposition, we have

$$S + \Delta S = \begin{bmatrix} s(1)+\Delta s(1) & s(2)+\Delta s(2) & \dots & s(n+1)+\Delta s(n+1) \\ s(2)+\Delta s(2) & s(3)+\Delta s(3) & \dots & s(n+2)+\Delta s(n+2) \\ \vdots & \dots & \dots & \vdots \\ s(m)+\Delta s(m) & s(m+1)+\Delta s(m+1) & \dots & s(m+n)+\Delta s(m+n) \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & \dots & 1 \\ z_1 & z_2 & \dots & z_n \\ z_1^2 & z_2^2 & \dots & z_n^2 \\ \vdots & \vdots & \vdots & \vdots \\ z_1^{m-1} & z_2^{m-1} & \dots & z_n^{m-1} \end{bmatrix} \begin{bmatrix} r_1 & 0 & \dots & 0 \\ 0 & r_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & r_n \end{bmatrix} \begin{bmatrix} 1 & z_1 & \dots & z_1^n \\ 1 & z_2 & \dots & z_2^n \\ \vdots & \vdots & \vdots & \vdots \\ 1 & z_n & \dots & z_n^n \end{bmatrix} \tag{3.23}$$

with $z_i, i = 1, \dots, n$ the so-called complex signal poles and $r_i, i = 1, \dots, n$ the so-called complex amplitudes. Furthermore, we take $W = I$, such that $s = \tilde{s}$ and $\Delta s = \Delta\tilde{s}$. From (3.23) it is obvious that the vector $s + \Delta s$ can be written as the following linear combination

$$s + \Delta s = r_1 b_1 + r_2 b_2 + \dots + r_n b_n, \tag{3.24}$$

with $b_i = [1 \; z_i \; z_i^2 \; \dots \; z_i^{m+n-1}]^T, i = 1, \dots, n$. Let us call the space spanned by $b_1, \dots, b_n$ the signal subspace. Now we construct a vector y such that the following polynomial in γ

$$y(1) + \gamma y(2) + \gamma^2 y(3) + \dots + \gamma^n y(n+1) \tag{3.25}$$

has roots $z_i, i = 1, \dots, n$.

From (3.23) and the definition of y, it is obvious that y is a null vector of $S + \Delta S$. Moreover, from (3.24) and the definition of y it is clear that the rows of the following $m \times (m+n)$ matrix

$$\begin{bmatrix} y(1) & y(2) & \dots & y(n) & y(n+1) & 0 & \dots & \dots & 0 \\ 0 & y(1) & y(2) & \dots & y(n) & y(n+1) & 0 & \dots & 0 \\ \vdots & & & & & & & & \vdots \\ 0 & \dots & \dots & \dots & \dots & 0 & y(1) & \dots & y(n+1) \end{bmatrix} \tag{3.26}$$

form a basis for the orthogonal complement of the space spanned by the vectors $b_i, i = 1, \dots, n$. Now we can easily derive the relation between the Vandermonde decomposition of $S + \Delta S$ and what we have called in section 3.2.1 the

[2]This is only true for the generic cases in which $z_i \neq z_j$ for $i \neq j$.

signal and noise subspace. Straightforward calculations show that $\mathbf{H_2}(y)$ as defined in (3.7) is equal to the matrix in (3.26), so that its rows span what we defined in subsection 3.2.1 to be the noise subspace N_y. This implies also that the vectors $b_i, i = 1, \ldots, n$ form a basis for what we called in section 3.2.1 the signal subspace $N_y^\perp$. In other words, the latter shows that the rather abstract definition of signal subspace in section 3.2.1 coincides –in the Hankel case– with the space spanned by the vectors that follow in a natural way from the Vandermonde decomposition.
Next we consider the case of an unstructured matrix S and its unstructured rank deficient approximation $S + \Delta S$. The natural decomposition of an unstructured matrix being the SVD we find the following decomposition for the rank deficient matrix $S + \Delta S$ of rank n:

$$S + \Delta S = U(:, 1:n)\Sigma(1:n, 1:n)V(:, 1:n)^T \tag{3.27}$$

where $S = U\Sigma V^T, U \in \mathbb{R}^{m\times m}, \Sigma \in \mathbb{R}^{m\times(n+1)}, V \in \mathbb{R}^{(n+1)\times(n+1)}$ is the SVD of S. In the unstructured case the vector $s \equiv vec(S)$ and $\Delta s \equiv vec(\Delta S)$, where $vec(.)$ is an operator that stacks the consecutive columns of a matrix into a column vector (again we take $W = I$). Transforming (3.27) into vector form, we can write the vector $s + \Delta s$ as the following linear combination:

$$\begin{aligned} s + \Delta s &= \Sigma(1,1)V(:,1) \otimes U(:,1) + \ldots + \Sigma(n,n)V(:,n) \otimes U(:,n) \qquad (3.28) \\ &= \sum_{i=1}^{m}\sum_{j=1}^{n} \Sigma(j,j)U(i,j)V(:,j) \otimes e_i, \end{aligned}$$

where $\otimes$ is the Kronecker product . The last equality of (3.28) shows that the signal $s + \Delta s$ can be written as a linear combination of the mn vectors $b_{ij} \equiv V(:,j) \otimes e_i, j = 1, \ldots n,\ i = 1, \ldots, m$. Furthermore it is clear[3] that the columns of the matrix

$$[V(:, n+1) \otimes e_1 \ldots V(:, n+1) \otimes e_n] \tag{3.29}$$

form a basis for the orthogonal complement of the space spanned by the vectors $b_{ij}, j = 1, \ldots n,\ i = 1, \ldots, m$. Straightforward calculations show that –in the unstructured case– $\mathbf{H_2}(y)^T$ as defined in (3.7) is equal to the matrix in (3.29), so that its columns span what we defined in subsection 3.2.1 to be the noise subspace N_y. This implies also that the vectors $b_{ij}, j = 1, \ldots n,\ i = 1, \ldots, m$ form a basis for what we called in subsection 3.2.1 the signal subspace $N_y^\perp$. In other words, the latter shows once more that the rather abstract definition of signal subspace in subsection 3.2.1 coincides –in the unstructured case– with the space spanned by the vectors that follow in a natural way from the SVD, i.e. the classical definition of signal subspace.

[3] make use of the properties that $(A \otimes B)^T = (A^T \otimes B^T)$ and $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$.

3.2.4 Numerical examples

In this section we illustrate the upper and lower bound derived in section 3.2.2 on 2 examples of a 5×3 Hankel matrix. The examples are constructed from small frames taken from the speech signal described in [60].

$$S_1 = \begin{bmatrix} -4 & -22 & -28 \\ -22 & -28 & -24 \\ -28 & -24 & -15 \\ -24 & -15 & -7 \\ -15 & -7 & -7 \end{bmatrix} \quad S_2 = \begin{bmatrix} -3 & -11 & -9 \\ -11 & -9 & 19 \\ -9 & 19 & 26 \\ 19 & 26 & 23 \\ 26 & 23 & 0 \end{bmatrix}$$

We consider 2 different weighting matrices W in (3.10). The first weighting matrix W_1 we consider is the identity matrix I and the second one is $W_2 = diag([1\ 2\ \ldots\ n \underbrace{n+1\ \ldots\ n+1}_{m-n}\ n\ \ldots\ 2\ 1]), m = 5, n = 2$. With this W_2, it is clear that $\Delta s^T W_2 \Delta s = \|\Delta S\|_F^2$. Using the formulas (3.21) and (3.22) we obtain for the upper and the lower bound on the objective function in (3.10) at its global solution, the values mentioned in table 3.1. The solution is obtained via the STLN1 algorithm described in chapter 4.

Table 3.1: This table shows the values for the objective function in (3.10) at its global solution, as well as the corresponding lower and upper bounds, for the 2 Hankel matrices and the 2 different weighting matrices described in section 3.2.4.

case		lower bound	solution	upper bound
S_1	W_1	4.282605	17.470175	22.408128
	W_2	12.847815	28.028628	40.438179
S_2	W_1	41.379714	119.176226	151.761821
	W_2	124.139141	262.303179	282.674756

3.2.5 Invertibility issues

In the derivation of the projection interpretation, it is assumed in formula (3.17) and further on, that $\tilde{\mathbf{H}}_\mathbf{2}(y)\tilde{\mathbf{H}}_\mathbf{2}(y)^T$ is invertible. As already shown in the proof of lemma 2.3.3,

$$\tilde{\mathbf{H}}_\mathbf{2}(y)\tilde{\mathbf{H}}_\mathbf{2}(y)^T = \mathbf{H}_\mathbf{2}(y)W^{-1}\mathbf{H}_\mathbf{2}(y)^T = \mathbf{D}_\mathbf{2}(y),$$

where $\mathbf{D}_\mathbf{2}(y) \in \mathbb{R}^{m \times m}$ is the matrix function that appears in the RiSVD and the CTLS approach. Therefore the invertibility of $\tilde{\mathbf{H}}_\mathbf{2}(y)\tilde{\mathbf{H}}_\mathbf{2}(y)^T$ might (as

will be indicated in chapter 4) cause problems in algorithms for the RiSVD approach or the CTLS approach.
Basically, there are 2 situations in which $\mathbf{D_2}(y)$ becomes rank deficient. The first situation occurs when $k < m$, yielding a $\mathbf{D_2}(y) \in \mathbb{R}^{m\times m}$ of maximum rank k. Loosely speaking we can say that this concerns structured matrices in which very few elements are allowed to change. The second situation concerns matrices where the rank deficiency of $\mathbf{D_2}(y)$ is not caused by the fact that $k < m$, but by the rank deficiency of $\mathbf{H}_2$ itself.
Similar to the explanation given in [3], we will now show how (3.10) can be reformulated as a problem in y, in case $\mathbf{D_2}(y)$ is rank deficient. Whether $\mathbf{D_2}(y)$ is rank deficient or not, problem (3.10) will only be solvable, if there exists at least one feasible point $(\Delta\tilde{s}, y)$ that satisfies the constraints

$$(S + \Delta S)y = 0, y^T y = 1$$

or equivalently

$$Sy = \tilde{\mathbf{H}}_\mathbf{2}(y)\Delta\tilde{s}, y^T y = 1.$$

If for a given feasible point $(\Delta\tilde{s}, y)$ there would be more than one possible $\Delta\tilde{s}$ for that particular y, it is obvious from the objective function in (3.10), that we have to choose the minimum norm solution

$$\Delta\tilde{s} = -\tilde{\mathbf{H}}_\mathbf{2}(y)^\dagger Sy.$$

Summarizing, in the case where $\mathbf{D_2}(y)$ is rank deficient, problem (3.10) can be restated as follows:

$$\min_y y^T S^T \tilde{\mathbf{H}}_\mathbf{2}(y)^{\dagger T} \tilde{\mathbf{H}}_\mathbf{2}(y)^\dagger Sy \tag{3.30}$$

$$\text{such that} (I_m - \tilde{\mathbf{H}}_\mathbf{2}(y)\tilde{\mathbf{H}}_\mathbf{2}(y)^\dagger)Sy = 0 \text{ and } y^T y = 1. \tag{3.31}$$

Note that in case $\mathbf{D_2}(y)$ is of full rank, $(I_m - \tilde{\mathbf{H}}_\mathbf{2}(y)\tilde{\mathbf{H}}_\mathbf{2}(y)^\dagger) = 0$ and (3.30) simply becomes the STLS problem as described e.g. in (2.13).
We will now describe some small examples in which $\mathbf{D_2}(y)$ is rank deficient.

Example 3.2.1 Consider the matrix function $\mathbf{M}(m) : \mathbb{R}^{3\times 1} \to \mathbb{R}^{4\times 2} : m \to \begin{bmatrix} m(1) & m(3) \\ m(2) & 1 \\ 2 & m(3) \\ m(1) & 5 \end{bmatrix}$. Let $s = [2\ 7\ -1]^T$ and $[A\ b] = \mathbf{M}(s)$. The problem to solve is problem (3.10) with $W = I$ and $S + \Delta S$ having the same structure as S, i.e. $S + \Delta S = \mathbf{M}(s + \Delta s)$. □

Straightforward calculations show that

$$\tilde{\mathbf{H}}_{\mathbf{2}}(y) = \mathbf{H}_{\mathbf{2}}(y) = \begin{bmatrix} y(1) & 0 & y(2) \\ 0 & y(1) & 0 \\ 0 & 0 & y(2) \\ y(1) & 0 & 0 \end{bmatrix}$$

It is clear that example 3.2.1 belongs to the class of problems in which $k < m$ and thus $\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T$ will always –i.e. for all y vectors– be rank deficient. In the next example $k = m$.

Example 3.2.2 Consider the matrix function $\mathbf{M}(m) : \mathbb{R}^{4\times 1} \to \mathbb{R}^{4\times 2} : m \to \begin{bmatrix} m(1) & m(3) \\ m(1) & 2m(3) \\ 2m(1) & 3m(3) \\ 2m(2) & m(4) \end{bmatrix}$. Let $s = [1\ 7\ \ -1\ 5]^T$ and $[A\ b] = \mathbf{M}(s)$. The problem to solve is problem (3.10) with $W = I$ and $S + \Delta S$ having the same structure as S, i.e. $S + \Delta S = \mathbf{M}(s + \Delta s)$. □

Straightforward calculations show that

$$\tilde{\mathbf{H}}_{\mathbf{2}}(y) = \mathbf{H}_{\mathbf{2}}(y) = \begin{bmatrix} y(1) & 0 & y(2) & 0 \\ y(1) & 0 & 2y(2) & 0 \\ 2y(1) & 0 & 3y(2) & 0 \\ 0 & 2y(1) & 0 & y(2) \end{bmatrix}$$

It is clear that example 3.2.2 belongs to the second class of problems in which $\tilde{\mathbf{H}}_{\mathbf{2}}(y)$ and thus also $\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T$ is rank deficient. Again $\mathbf{D}_{\mathbf{2}}(y)$ is rank deficient for all possible y vectors. As a last example we consider the example of symmetric Toeplitz matrices described in [66]. We consider a simple 3×3 example.

Example 3.2.3 Consider the matrix function $\mathbf{M}(m) : \mathbb{R}^{3\times 1} \to \mathbb{R}^{3\times 3} : m \to \begin{bmatrix} m(1) & m(2) & m(3) \\ m(2) & m(1) & m(2) \\ m(3) & m(2) & m(1) \end{bmatrix}$. Let $s = [1\ 7\ \ 5]^T$ and $[A\ b] = \mathbf{M}(s)$. The problem to solve is problem (3.10) with $W = I$ and $S + \Delta S$ having the same structure as S, i.e. $S + \Delta S = \mathbf{M}(s + \Delta s)$. □

Straightforward calculations show that

$$\tilde{\mathbf{H}}_{\mathbf{2}}(y) = \mathbf{H}_{\mathbf{2}}(y) = \begin{bmatrix} y(1) & y(2) & y(3) \\ y(2) & y(1) + y(3) & 0 \\ y(3) & y(2) & y(1) \end{bmatrix}.$$

It can easily be shown that $\mathbf{H_2}(y)$ becomes rank deficient in the following cases

$$y(3) = y(1) \text{ or } y(3) + y(1) = \pm y(2).$$

This means that in the course of an algorithm for solving the STLS problem posed in example 3.2.3, $\mathbf{D_2}(y)$ might become rank deficient. In the following paragraph it is shown that this happens exactly at the solution of the STLS problem. The latter implies that algorithms that expect $\mathbf{D_2}(y)$ to be of full rank, will run into problems near the solution. However, we also present a way to avoid rank deficiency problems in this particular problem. It is shown that $\mathbf{D_2}(y)$ can be replaced by a different matrix function, which is always of full rank for $y \neq 0$, without affecting the solution of the particular STLS problem.

Definition 3.2.4 *A square matrix $A \in \mathbb{R}^{q\times q}$ is called centrosymmetric, if $A = E_qAE_q$ where $E_q \in \mathbb{R}^{q\times q}$ is the anti-identity matrix . A vector $v \in \mathbb{R}^{q\times 1}$ is called centrosymmetric if $v = E_qv$.*

Definition 3.2.5 *A square matrix $A \in \mathbb{R}^{q\times q}$ is called skew centrosymmetric, if $A = -E_qAE_q$ where $E_q \in \mathbb{R}^{q\times q}$ is the anti-identity matrix. A vector $v \in \mathbb{R}^{q\times 1}$ is called skew centrosymmetric if $v = -E_qv$.*

Theorem 3.2.6 *Let $A \in \mathbb{R}^{q\times q}$ and $B \in \mathbb{R}^{q\times q}$, be both centrosymmetric matrices. Consider the generalized eigenvalue problem $Av = \lambda Bv$. If q is even, the generalized eigenvectors can be divided into the following 2 classes: $q/2$ vectors of the form*
$[v_1 \;\; v_2 \dots v_{q/2} \;\; v_{q/2} \dots v_2 \;\; v_1]^T$ and $q/2$ vectors of the form
$[v_1 \;\; v_2 \dots v_{q/2} \;\; -v_{q/2} \dots -v_2 \;\; -v_1]^T$.
If q is odd, the generalized eigenvectors can be divided into the following 2 classes: $(q-1)/2$ vectors of the form
$[v_1 \;\; v_2 \dots v_{(q-1)/2} \;\; 0 \;\; -v_{(q-1)/2} \dots -v_2 \;\; -v_1]^T$ and $(q+1)/2$ eigenvectors of the form
$[v_1 \;\; v_2 \dots v_{(q-1)/2} \;\; v_{(q-1)/2+1} \;\; v_{(q-1)/2} \dots v_2 \;\; v_1]^T$.

Proof: Let us first consider case 1, namely when q is even. If we can find a vector $g_1 = [v_1 \;\; v_2 \;\; \dots v_{q/2} \;\; v_{q/2} \dots v_2 \;\; v_1]^T$ and λ_1 such that

$$A(1:q/2,:)g_1 = \lambda_1 B(1:q/2,:)g_1, \tag{3.32}$$

then it can easily be checked that –due to the fact that A and B are centrosymmetric – also $A(q/2+1:q,:)g_1 = \lambda_1 B(q/2+1:q,:)g_1$. As a result g_1 would be a generalized eigenvector of (A, B) corresponding to the eigenvalue λ_1. Given the structure of g_1, it is clear that (3.32) can be rewritten as

$$\tilde{A}\tilde{g}_1 = \lambda_1 \tilde{B}\tilde{g}_1, \tag{3.33}$$

with $\tilde{g}_1 = g_1(1:q/2,1) \in \mathbb{R}^{q/2\times 1}$, $\tilde{A} = A(1:q/2,1:q/2) + A(1:q/2,q:-1:q/2+1) \in \mathbb{R}^{q/2\times q/2}$, $\tilde{B} = B(1:q/2,1:q/2) + B(1:q/2,q:-1:q/2+1) \in \mathbb{R}^{q/2\times q/2}$. Since (3.33) is again a generalized eigenvalue problem, we find $q/2$ different generalized eigenvectors $\tilde{g}_1$, from which we easily derive $q/2$ different vectors $g_1 = [\tilde{g}_1^T \ \tilde{g}_1(q/2:-1:1,1)^T]^T$. Since the latter vectors g_1 solve (3.32) they are generalized eigenvectors of (A,B). Similarly, if we can find a vector $g_2 = [v_1 \ \ v_2 \ \ \ldots v_{q/2} \ \ -v_{q/2}\ldots -v_2 \ \ -v_1]^T$ and λ_2 such that

$$A(1:q/2,:)g_2 = \lambda_2 B(1:q/2,:)g_2, \tag{3.34}$$

then it can easily be checked that –due to the fact that A and B are centro-symmetric – also $A(q/2+1:q,:)g_2 = \lambda_2 B(q/2+1:q,:)g_2$. As a result g_2 would be a generalized eigenvector of (A,B) corresponding to the eigenvalue λ_2. Given the structure of g_2, it is clear that (3.34) can be rewritten as

$$\tilde{A}\tilde{g}_2 = \lambda_2 \tilde{B}\tilde{g}_2, \tag{3.35}$$

with $\tilde{g}_2 = g_2(1:q/2,1) \in \mathbb{R}^{q/2\times 1}$, $\tilde{A} = A(1:q/2,1:q/2) - A(1:q/2,q:-1:q/2+1) \in \mathbb{R}^{q/2\times q/2}$, $\tilde{B} = B(1:q/2,1:q/2) - B(1:q/2,q:-1:q/2+1) \in \mathbb{R}^{q/2\times q/2}$. Since (3.35) is again a generalized eigenvalue problem, we find $q/2$ different generalized eigenvectors $\tilde{g}_2$, from which we easily derive $q/2$ different vectors $g_2 = [\tilde{g}_2^T \ \ -\tilde{g}_2(q/2:-1:1,1)^T]^T$. Since the latter vectors g_2 solve (3.34) they are generalized eigenvectors of (A,B). Summarizing we can say that for q even, we have found q different generalized eigenvectors of (A,B). Half of them has the structure represented by g_1 and the other half has the structure represented by g_2.
We now consider the case when q is odd. If we can find a vector
$g_3 = [v_1 \ \ v_2 \ldots v_{(q-1)/2} \ \ v_{(q+1)/2} \ \ v_{(q-1)/2}\ldots v_2 \ \ v_1]^T$ such that

$$A(1:(q+1)/2,:)g_3 = \lambda_3 B(1:(q+1)/2,:)g_3, \tag{3.36}$$

then it can easily be checked that –due to the fact that A and B are centro-symmetric – also $A((q+3)/2:q,:)g_3 = \lambda_3 B((q+3)/2:q,:)g_3$. As a result g_3 would be a generalized eigenvector of (A,B) corresponding to the eigenvalue λ_3. Given the structure of g_3, it is clear that (3.36) can be rewritten as

$$\tilde{A}\tilde{g}_3 = \lambda_3 \tilde{B}\tilde{g}_3, \tag{3.37}$$

with $\tilde{g}_3 = g_3(1:(q+1)/2,1) \in \mathbb{R}^{(q+1)/2\times 1}$, $\tilde{A} = [A(1:(q+1)/2,1:(q-1)/2) + A(1:(q+1)/2,q:-1:(q+3)/2) \ \ A(1:(q+1)/2,(q+1)/2)] \in \mathbb{R}^{(q+1)/2\times(q+1)/2}$, $\tilde{B} = [B(1:(q+1)/2,1:(q-1)/2) + B(1:(q+1)/2,q:-1:(q+3)/2) \ \ B(1:(q+1)/2,(q+1)/2)] \in \mathbb{R}^{(q+1)/2\times(q+1)/2}$. Since (3.37) is again a generalized eigenvalue problem, we find $(q+1)/2$ different generalized eigenvectors $\tilde{g}_3$, from which we easily derive $(q+1)/2$ different vectors $g_3 = [\tilde{g}_3^T \ \tilde{g}_3((q-1)/2:-1:1,1)^T]^T$. Since the latter vectors g_3 solve (3.36) they are generalized eigenvectors of (A,B). Similarly, if we can find a vector $g_4 = [v_1 \ \ v_2\ldots v_{(q-1)/2} \ \ 0 \ \ -v_{(q-1)/2}\ldots -v_2 \ \ -v_1]^T$ such that

$$A(1:(q-1)/2,:)g_4 = \lambda_4 B(1:(q-1)/2,:)g_4, \tag{3.38}$$

then it can easily be checked that also –due to the fact that A and B are centrosymmetric – also $A((q+3)/2:q,:)g_4 = \lambda_4 B((q+3)/2:q,:)g_4$. Furthermore, note that, due to the particular structure of A, B and g_4, the following is always true

$$A((q+1):2,:)g_4 = \lambda_4 B(q+1):2,:)g_4.$$

As a result g_4 would be a generalized eigenvector of (A,B) corresponding to the eigenvalue λ_4. Given the structure of g_4, it is clear that (3.38) can be rewritten as

$$\tilde{A}\tilde{g}_4 = \lambda_4 \tilde{B}\tilde{g}_4, \tag{3.39}$$

with $\tilde{g}_4 = g_4(1:(q-1)/2,1) \in \mathbb{R}^{(q-1)/2\times 1}$, $\tilde{A} = A(1:(q-1)/2,1:(q-1)/2) - A(1:(q-1)/2,q:-1:(q+3)/2) \in \mathbb{R}^{(q-1)/2\times(q-1)/2}$, $\tilde{B} = B(1:(q-1)/2,1:(q-1)/2) - B(1:(q-1)/2,q:-1:(q+3)/2) \in \mathbb{R}^{(q-1)/2\times(q-1)/2}$. Since (3.39) is again a generalized eigenvalue problem, we find $(q-1)/2$ different generalized eigenvectors $\tilde{g}_4$, from which we easily derive $(q-1)/2$ different vectors $g_4 = [\tilde{g}_4^T \; 0 \; -\tilde{g}_4((q-1)/2:-1:1,1)^T]^T$. Since the latter vectors g_4 solve (3.38) they are generalized eigenvectors of (A,B). Summarizing we can say that for q odd, we have found q different generalized eigenvectors of (A,B). $(q+1)/2$ of them have the structure represented by g_3 and $(q-1)/2$ of them have the structure represented by g_4. □

Lemma 3.2.7 *Consider the STLS problem (3.10) with $W = I$, in which the symmetric Toeplitz structure of S needs to be preserved. The Riemannian SVD equations (2.7) reduce in this particular case to*

$$Sy = \mathbf{D_2}(y)y\tau, y^Ty = 1,$$

where at the solution of the STLS problem y is a (skew) centrosymmetric vector and $\mathbf{D_2}(y)$ is a centrosymmetric matrix.

Proof: At the solution of this particular STLS problem the following must be true

$$(S+\Delta S)y = 0,$$

where $S+\Delta S$ is a symmetric Toeplitz matrix. The latter implies that y is an eigenvector of a symmetric Toeplitz matrix with corresponding eigenvalue 0. From theorem 3.2.6 it then follows that y is (skew) centrosymmetric. As indicated in (2.7) $\mathbf{D_2}(y)$ can also be expressed in this case ($W = I$) as follows:

$$\mathbf{D_2}(y) = \sum_{i=1}^{k}(T_iy)(T_iy)^T.$$

Then it is straightforward to show that $\mathbf{D_2}(y)$ is centrosymmetric:

$$\begin{aligned} E_m \mathbf{D_2}(y) E_m &= \sum_{i=1}^{k} (E_m T_i y)(E_m T_i y)^T \\ &= \sum_{i=1}^{k} (E_m T_i E_m E_m y)(E_m T_i E_m E_m y)^T \\ &= \sum_{i=1}^{k} (T_i E_m y)(T_i E_m y)^T \\ &= \sum_{i=1}^{k} (T_i y)(T_i y)^T = \mathbf{D_2}(y), \end{aligned}$$

where the second equation follows from the fact that $E_m E_m = I_m$, the third equation results from the observation that the basis matrices $T_i, i = 1, \ldots k$ are centrosymmetric themselves and the fourth equality is obtained by noting that y is centrosymmetric in this particular case. □

Lemma 3.2.8 *Consider the STLS problem (3.10) with $W = I$, in which the symmetric Toeplitz structure of S needs to be preserved. At the solution of the problem the rank of the matrix $\mathbf{H_2}(y)$ is:*

- *$m/2$ when m is even and thus the solution y has the structure of g_1 or g_2 in theorem 3.2.6*
- *$(m+1)/2$ when m is odd and the solution y is centro symmetric (has the structure of g_3 in theorem 3.2.6)*
- *$(m-1)/2$ when m is odd and the solution y is skew centrosymmetric (has the structure of g_4 in theorem 3.2.6)*

Proof: Let us first consider the case when m is even. Form the definition of $\mathbf{H_2}(y)$, namely $\mathbf{H_2}(y)\Delta s = \Delta S y$ and the fact that $\Delta s \in \mathbb{R}^{k\times 1}$ is centrosymmetric (because ΔS is a symmetric Toeplitz matrix), it is clear that if the entry $(i,j), 1 \leq i \leq m/2, 1 \leq j \leq k$ of $\mathbf{H_2}(y)$ contains the element $y(l), 1 \leq l \leq k$, then the entry $(m-i+1, j)$ of $\mathbf{H_2}(y)$ contains the element $y(k-l+1)$. Therefore , since y is (skew) centrosymmetric, $\mathbf{H_2}(y)(i,:) = \pm\mathbf{H_2}(y)(m-i+1,:), 1 \leq i \leq m/2$, resulting in a matrix $\mathbf{H_2}(y)$ of rank $m/2$ at the solution of the STLS problem.
When m is odd, the previous reasoning still holds for the rows $\mathbf{H_2}(y)(1 : (m-1)/2,:)$ compared to the rows $\mathbf{H_2}(y)((m+3)/2 : m,:)$. By that we mean that by following the same reasoning it can be seen that at the solution of the STLS problem $\mathbf{H_2}(y)(i,:) = \pm\mathbf{H_2}(y)(m-i+1,:), 1 \leq i \leq (m-1)/2$. The only

thing that is left to investigate is $\mathbf{H_2}(y)((m+1)/2,:)$. Since ΔS is a symmetric Toeplitz matrix, we have that in the case y is skew centrosymmetric, that $\mathbf{H_2}(y)((m+1)/2,:) = 0$ and thus the rank of $\mathbf{H_2}(y)$ will be $(m-1)/2$. If y is centrosymmetric, the rank of $\mathbf{H_2}(y)$ will be $(m+1)/2$. □

Lemma 3.2.9 *Consider the STLS problem (3.10) with $W = I$, in which S is a symmetric Toeplitz matrix but the structure that will be preserved is the Toeplitz structure. The solution of this STLS problem is a* symmetric *Toeplitz matrix (although only a Toeplitz structure is imposed by the formulation).*

Proof: First of all, using the equations in (2.7) it can easily be shown that the v and u in these equations can easily be found from the following two minimization problems:

$$\min_{v, v^T v = 1} v^T S^T \mathbf{D_2}(v) S v \tag{3.40}$$

and

$$\min_{u, u^T u = 1} u^T S^T \mathbf{D_1}(u) S u. \tag{3.41}$$

Since the structure that will be preserved in this particular STLS problem is the Toeplitz structure (no symmetry constraints are imposed!), it is possible to choose the basis matrices $T_i, i = 1, \ldots, k$ from the RiSVD formulation, in the following way:

$$T_i = T_{k-i+1}^T, i = 1, \ldots, k. \tag{3.42}$$

Taking (3.42) into account it is easy to see that the matrix functions $\mathbf{D_1}$ and $\mathbf{D_2}$ are equal. It then follows form (3.40) and (3.41) that $u = v$ for this particular STLS problem. Furthermore from (2.8) we find that

$$\begin{aligned} t(i) &= s(i) - u^T T_i v \tau \\ &= s(k-i+1) - u^T T_i u \tau \\ &= s(k-i+1) - u^T T_i u \tau \\ &= s(k-i+1) - u^T T_{k-i+1}^T u \tau \\ &= s(k-i+1) - u^T T_{k-i+1} u \tau \\ &= t(k-i+1), \end{aligned}$$

meaning that $S + \Delta S$ will be a *symmetric* Toeplitz matrix. □

As can easily be checked, the matrix $\mathbf{H_2}(y)$ is always of full rank for the Toeplitz structure, thereby leading to an invertible matrix $\mathbf{D_2}(y)$.

Remark 3.2.10 In [66] it is mentioned that $\mathbf{D_2}(y)$ of the STLS problem described in example 3.2.3 is a symmetric Toeplitz matrix. This is not true. It is symmetric but not Toeplitz, even not at the solution of the particular STLS problem, as can easily be checked with a small example. The only thing that can be said on $\mathbf{D_2}(y)$ is that it is centrosymmetric and this only at the solution of the particular STLS problem. Since $\mathbf{D_2}(y)$ is also symmetric we can specify $\mathbf{D_2}(y)$ even further: it is symmetric and persymmetric (note: a matrix $A \in \mathbb{R}^{q\times q}$ is persymmetric if $E_q A E_q = A^T$). $\diamond$

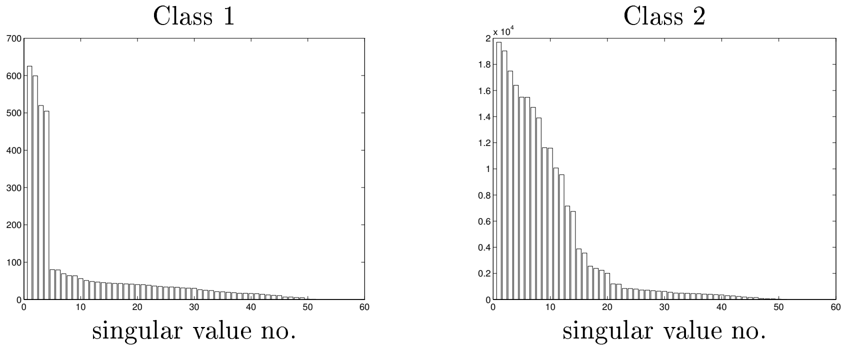
3.3 Analysis of the STLS problem for Hankel/ Toeplitz matrices

In this section we analyze the STLS problem for Hankel matrices (the theory can be extended in a straightforward way to Toeplitz matrices, block Hankel and block Toeplitz matrices). Using a particular parametrization of rank deficient Hankel matrices, we show that the STLS problem suffers from multiple local minima, the properties of which depend on the parameters of the new parametrisation. The latter observation makes initial estimates an important issue in STLS problems. In chapter 4 we will therefore propose new initialization methods.

Concerning the initial estimates, it is important to draw attention to the fact that the STLS problems can be divided into 2 classes. The first class contains STLS problems in which the data matrix S is almost rank deficient. Most of the time these examples are simulated examples, in which a rank deficient matrix is corrupted by a certain amount of noise. It is clear that especially in the tests where S is obtained by adding a small amount of noise to a rank deficient matrix, the STLS problem is not that difficult to solve and basically a standard algorithm with simple starting values will do. In the second class of STLS problems there is no nearby rank deficient matrix to S. Note in this respect that the first class mainly corresponds to applications in which STLS is used for its statistical benefits (ML estimator). However, as already mentioned in chapter 1, STLS is often used outside this statistical framework. The latter applications typically belong to the second class. To illustrate the previous we show an example of each class in figure 3.3. The left hand side represents an example of the first class. The plotted SVD spectrum belongs to a 51×51 Hankel matrix that is obtained in the following way. First a signal s is obtained by adding noise to a sum of 4 complex damped exponentials having the following parameters:

$$f_1 = 200, d_1 = -0.02, a_1 = 12, p_1 = 0.4$$
$$f_2 = -200, d_2 = -0.02, a_2 = 12, p_2 = -0.4$$
$$f_3 = 300, d_3 = -0.03, a_3 = 10.5, p_3 = 0.2$$
$$f_4 = 300, d_4 = -0.03, a_4 = 10.5, p_4 = -0.2.$$

Let $s(l) = \sum_{i=1}^{4} a_i e^{jp_i} e^{(d_i + j2\pi f_i)(l-1)\Delta t} + n(l), l = 1, \ldots, 101$, where $\Delta t = 1/1000$ and n is an i.i.d. Gaussian noise vector with standard deviation 5. Finally, S is a 51×51 Hankel matrix constructed in such a way that we can find s in the first column and last row of S, starting in the upper left corner and ending in the lower right corner. The example on the right of figure 3.3 is the SVD spectrum of a Hankel matrix obtained in the following way. The vector s contains 101 consecutive samples of the speech signal described in chapter 6. Next, a 51×51 Hankel matrix S is constructed using s, in the same way as described above.



(a) (b)

Figure 3.3: *These figures show the SVD spectra of two* 51×51 *Hankel matrices belonging to two different classes of STLS problems. Figure (a) is a simulated example in which the matrix S came about by adding a certain amount of noise to a rank deficient* 51×51 *Hankel matrix (of rank* 4*, since it is constructed from a sum of* 4 *complex damped exponentials). Figure (b) shows the SVD spectrum of a Hankel matrix constructed from a real life speech signal.*

The section is structured as follows. In the next subsection we analyze the cost function associated with the STLS problem in the case we are dealing with Hankel matrices (Toeplitz matrices are included in this category since an STLS problem with a Toeplitz matrix can be rearranged into an equivalent STLS problem with a Hankel matrix, simply by doing a permutation of the rows of the extended data matrix $S \equiv [A \;\; b]$; block Hankel and block Toeplitz matrices are also included since the parametrisation we will propose can be extended to those cases [22]). Although this limits the applicability of our analysis, a large number of applications fits into this category. First we propose sums of complex damped exponentials for the parametrisation of rank deficient Hankel matrices. This parametrisation method is related to the problem of harmonic retrieval and Linear Prediction (LP) theory. Using this parametrisation we are able to show that the STLS problem suffers from local minima located around a narrow global minimum.

3.3.1 Properties of the STLS cost function in the Hankel/Toeplitz case

As mentioned in the introduction of this section, we consider here STLS (3.1) problems in which the matrix S has a Hankel structure. Although this choice limits the applicability of our analysis, Hankel matrices (and their straightforward extensions towards Toeplitz, block Hankel and block Toeplitz matrices) are amongst the most frequently encountered structured matrices. Moreover, rank-deficient matrices having these structures allow a particular parametrisation, that will make the analysis of the STLS cost function tractable. Since we focus on the class of Hankel matrices the STLS problem (3.1) can be formulated as follows:

$$min_{\Delta s(l),l=0,\ldots,N-1,\; x(l),l=1,\ldots,n} \sum_{t=0}^{N-1} (\Delta s(l))^2 \tag{3.43}$$

$$\begin{aligned}&\text{such that } (A+\Delta A)x = b + \Delta b\\ &\text{and } [A+\Delta A\ b+\Delta b] \text{ has the same structure as } [A\ b]\end{aligned} \tag{3.44}$$

with

$$\begin{aligned}
&A \in \mathbb{R}^{m\times n} \text{ with } m = N-n,\\
&A = \begin{bmatrix} s(1) & s(2) & \ldots & s(n)\\ s(2) & s(3) & \ldots & s(n+1)\\ \vdots & & & \vdots\\ s(N-n) & s(N-n+1) & \ldots & s(N-1)\end{bmatrix},\\
&\Delta A = \begin{bmatrix} \Delta s(1) & \Delta s(2) & \ldots & \Delta s(n)\\ \Delta s(2) & \Delta s(3) & \ldots & \Delta s(n+1)\\ \vdots & & & \vdots\\ \Delta s(N-n) & \Delta s(N-n+1) & \ldots & \Delta s(N-1)\end{bmatrix},\\
&b = [s(n+1)\ s(n+2)\ldots s(N)]^T,\\
&\Delta b = [\Delta s(n+1)\ \Delta s(n+2)\ldots \Delta s(N)]^T.
\end{aligned}$$

It is obvious that we only consider the Hankel case, since the Toeplitz case can be transformed into a Hankel case by simple permutation of the rows. The corresponding solution of the Toeplitz case can be derived by rearranging the solution ΔS, Δb and x of the Hankel case. When we look at (3.43), we see that there is an obvious relation with the so-called "harmonic retrieval" problem. This will be explained in the next 2 paragraphs. In the first paragraph, it is shown that if $S + \Delta S \in \mathbb{R}^{m\times(n+1)}$ is a rank deficient Hankel matrix, $s + \Delta s$ can be written as a sum of n complex damped exponentials. In the second paragraph it is proven that when $s + \Delta s$ is a sum of n complex damped exponentials, then $S + \Delta S$ is a rank deficient Hankel matrix. It then follows that the parameters of these n complex damped exponentials can be used to parametrize the set of rank deficient Hankel matrices.
If $S + \Delta S$ is a rank deficient Hankel matrix, then it is obvious that there must exist a vector $y \in \mathbb{R}^{(n+1)\times 1}$ such that $(S + \Delta S)y = 0$. Given the structure of $S+\Delta S$, the latter implies that $s+\Delta s$ satisfies the following difference equation in u:

$$y(n+1)u(l) + y(n)u(l-1) + \ldots + y(2)u(l-n+1) + y(1)u(l-n) = 0.$$

The solutions of the latter equation are given by

$$u(l) = \sum_{i=1}^{n} c_i z_i^l,$$

where $z_i, i = 1, \ldots, n$ are the roots of the following polynomial in γ:

$$y(n+1)\gamma^n + y(n)\gamma^{n-1} + \ldots + y(2)\gamma + y(1) = 0.$$

Call z_i the complex signal poles and c_i the complex amplitudes . Being complex numbers it is obvious that they can be parametrized in the following way:

$$z_i = e^{j2\pi f_i} e^{d_i} \text{ and } c_i = a_i e^{jp_i}.$$

Thus it is proven that $s+\Delta s$ can indeed be written as a sum of complex damped exponentials.
Secondly, if $s+\Delta s$ can be written as a sum of n complex damped exponentials, i.e.

$$s(l) + \Delta s(l) = \sum_{i=1}^{n} c_i z_i^l, l = 1, \ldots, N,$$

with c_i and z_i the previously defined complex amplitudes and complex signal poles, it is obvious that $S+\Delta S$ can be decomposed as in (3.23). Since in (3.23) the middle diagonal matrix is an $n\times n$ matrix, $S+\Delta S$ has to be rank deficient. We have thus shown that the set of rank deficient $m \times (n+1)$ matrices can be parametrized by the frequencies $f_i, i = 1, \ldots, n$, the dampings $d_i, i = 1, \ldots, n$,

the amplitudes $a_i, i = 1, \ldots, n$ and the phases $p_i, i = 1, \ldots, n$ of the n underlying complex damped exponential components.
Since we only consider real data, the signal poles

$$e^{d_i + j2\pi f_i}, i = 1, \ldots, n$$

appear in complex conjugate pairs and the same holds for the complex amplitudes

$$a_i e^{jp_i}, i = 1, \ldots, n.$$

The latter means that when n is odd, there will at least be 1 component k such that $f_k = 0$ and $p_k = 0$ (we say "at least" since there is an even number of real exponentials when n is even and an odd number of real exponentials when n is odd). From now on we will assume, without loss of generality, that n is even and that there are no real exponentials.
Using this parametrisation of the elements $s(l) + \Delta s(l)$ of the rank deficient Hankel matrix, it is possible to recast (3.43) as the minimization of the following cost function F:

$$F(\hat{\theta}) = \sum_{l=0}^{N-1} \epsilon^2(l, \hat{\theta}), \tag{3.45}$$

with

$$\epsilon(l, \hat{\theta}) = s(l) - \sum_{i=1}^{n/2} \hat{a}_i e^{\hat{d}_i l} sin(2\pi \hat{f}_i l + \hat{p}_i) \text{ and}$$
$$\hat{\theta} = [\hat{a}_1 \ldots \hat{a}_{n/2} \;\; \hat{p}_1 \ldots \hat{p}_{n/2} \;\; \hat{d}_1 \ldots \hat{d}_{n/2} \;\; \hat{f}_1 \ldots \hat{f}_{n/2}].$$

Symbols with a caret indicate estimates obtained during consecutive iterations. The corresponding symbols without caret are the values upon termination, i.e. at a local solution of the problem. Note that the constraints still present in (3.43) have been eliminated by the particular choice of parametrisation of the manifold of rank-deficient Hankel matrices. Moreover, since the latter reparametrization transforms the STLS problem (3.43) into the separable Non-Linear Least Squares (NLLS) problem (3.45), the analysis in the next subsection is also valid for that particular type of separable NLLS problems (see e.g. [47]). In [89] an analysis was made for the less general problem in which $\hat{d}_i = 0, i = 1, \ldots, n/2$. Whenever $\hat{d}_i < 0$ and/or $\hat{d}_i > 0$ an asymptotic analysis is of no use, since in the first case the component dies out whereas in the second case it explodes. It is however still possible to show some properties of the cost function (3.45) (and thus of the STLS problem).

3.3.2 Analysis of the cost function

In this section we will decompose the cost function $F(\hat{\theta})$ into different components. This will allow us to analyze the properties of the cost function. Let

$$x(l,\theta) = \sum_{i=1}^{n/2} a_i e^{d_i l} sin(2\pi f_i l + p_i).$$

Then

$$\begin{aligned}\epsilon(l,\hat{\theta}) &= s(l) - x(l,\hat{\theta}) \\ &= s(l) - x(l,\theta) + x(l,\theta) - x(l,\hat{\theta}) \\ &= \epsilon(l,\theta) + x(l,\theta) - x(l,\hat{\theta}). \end{aligned} \tag{3.46}$$

Substitution of (3.46) in (3.45) yields the following decomposition of the cost function:

$$F(\hat{\theta}) = F_1(\theta) + F_2(\theta,\hat{\theta}) + F_3(\theta,\hat{\theta}), \tag{3.47}$$

with

$$F_1(\theta) \equiv \sum_{l=0}^{N-1} \epsilon^2(l,\theta) \tag{3.48}$$

$$F_2(\theta,\hat{\theta}) \equiv \sum_{l=0}^{N-1} (x(l,\theta) - x(l,\hat{\theta}))^2 \tag{3.49}$$

$$F_3(\theta,\hat{\theta}) \equiv \sum_{l=0}^{N-1} 2\epsilon(l,\theta)(x(l,\theta) - x(l,\hat{\theta})) \tag{3.50}$$

A closer look at (3.47) reveals the following. $F(\theta) = \sum_{l=0}^{N-1} \epsilon^2(l,\theta) \equiv F_1(\theta)$, i.e. $F(\theta)$ is a modeling error, measuring the quality of fit between the final sum of complex damped exponentials $x(t,\theta)$ and the original signal $s(t)$. This measure is a constant term that does not depend upon $\hat{\theta}$ and as a consequence is of no importance in the solution of (3.43). During the first iterations, F_2 will dominate F_3, mainly because (3.49) is a sum of squares, whereas (3.50) is a sum of terms that might have (and will have in practice) alternating signs, leading to cancellations and a lower value for F_3. Only when $\hat{\theta}$ reaches θ (i.e. upon convergence), F_3 will become important as well. The difficulty in solving (3.43) (and thus in solving the STLS problem) lies in avoiding the many local minima near the global minimum. Since it is mainly the term F_2 which determines the shape of the global minimum and the position of the many local minima in its neighbourhood, it is especially this term we are interested in. To illustrate this, we use an example of a speech compression application [59, 60]. Each frame

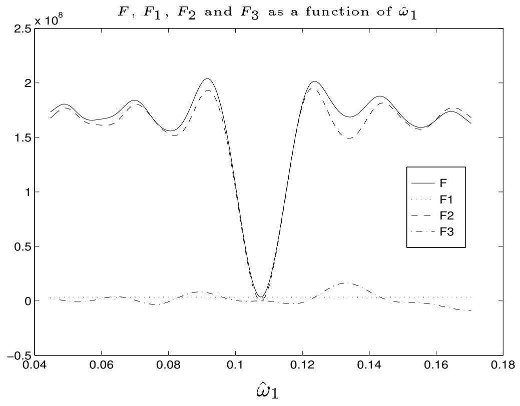


Figure 3.4: *This figure shows the different components of F as a function of $\hat{\omega}_1$, applied to example (3.51).*

of the speech signal is compressed by determining and storing the parameters of the n complex damped exponentials, whose sum lies closest (in L_2 norm) to that particular frame of the original speech signal s. As indicated earlier this boils down to an STLS problem. The frame we consider ($N = 301$) leads to the following set of parameters ($n = 6$, but each time we only give one of the two complex conjugated signal poles and one of the two complex conjugated amplitudes):

$$\begin{aligned} &\omega_1 = 0.1076, \omega_2 = 0.2132, \omega_3 = 0.2224 \\ &a_1 = 412.61, a_2 = 262.11, a_3 = 410.96 \\ &d_1 = 6.666e-04, d_2 = 8.699e-04, d_3 = 2.425e-02 \\ &p_1 = 273.27°, p_2 = 151.52°, p_3 = 241.18°, \end{aligned} \tag{3.51}$$

with $\omega = 2\pi f$. Figure 3.4 shows F and its different components F_1, F_2 and F_3 as a function of $\hat{\omega}_1$. This figure confirms the previous discussion.
Applying the definition of F_2, yields

$$F_2(\theta, \hat{\theta}) = \sum_{l=0}^{N-1} (\sum_{i=1}^{n/2} C_i)^2 \tag{3.52}$$

$$= \sum_{l=0}^{N-1} (\sum_{i=1}^{n/2} C_i^2 + 2 \sum_{i=1}^{n/2-1} \sum_{j=i+1}^{n/2} C_i C_j) \tag{3.53}$$

with

$$C_i = a_i e^{d_i l} sin(\omega_i l + p_i) - \hat{a}_i e^{\hat{d}_i l} sin(\hat{\omega}_i l + \hat{p}_i), \;\; \omega_i = 2\pi f_i, \hat{\omega}_i = 2\pi \hat{f}_i.$$

In appendix B.2 we derive a ratio that allows us to predict whether the sum of cross terms $\sum_{l=0}^{N-1} \sum_{i=1}^{n/2} \sum_{j=i+1}^{n/2} 2C_iC_j$ will be negligible with respect to the sum of quadratic terms $\sum_{l=0}^{N-1}\sum_{i=1}^{n/2} C_i^2$. In that appendix it is shown that the sum of the cross terms between two components becomes important with respect to the sum of the corresponding quadratic terms, whenever the angular frequencies ω_i and ω_j approach each other. The lower the value of the associated dampings d_j and d_i, the closer the angular frequencies may be before the sum of cross terms starts dominating the corresponding sums of quadratic terms.

The reason why we go through all these efforts to derive conditions under which the sum of cross terms is negligible with respect to the corresponding sums of quadratic terms, is because in that case the analysis of the F_2 component becomes a lot more tractable. Therefore we will suppose from now on, that the latter condition is satisfied. Then we can approximate $F_2(\theta, \hat{\theta})$ with

$$\tilde{F}_2(\theta, \hat{\theta}) \equiv \sum_{i=1}^{n/2}\sum_{l=0}^{N-1} C_i^2. \tag{3.54}$$

Since the different parameters $\hat{a}_i, \hat{p}_i, \hat{\omega}_i, \hat{d}_i, i = 1, \ldots, n/2$ each appear only once in a quadratic term C_l, it suffices to understand the shape of

$$g(\hat{a}, \hat{d}, \hat{\omega}, \hat{p}) = \sum_{l=0}^{N-1} (ae^{dl} sin(\omega l + p) - \hat{a}e^{\hat{d}l} sin(\hat{\omega} l + \hat{p}))^2 \tag{3.55}$$

in order to understand the shape of F_2 as a function of all the parameters. From (3.55) we can easily see that g is quadratic in $\hat{a}$. In appendix B.2, we show that g is a sinusoidal function of $\hat{p}$. Appendix B.3 explores the dependence of g on $\hat{\omega}$, using the frequency ω and the damping d of the component under consideration as parameters. The latter appendix shows that the valley containing the global minimum (located at ω) of (3.55) as a function of $\hat{\omega}$, gets narrower when the damping d increases.

To conclude we can say that the STLS cost function (for Hankel or Toeplitz matrices) as a function of an amplitude $\hat{a}$ of 1 component is quadratic, as a function of the phase $\hat{p}$ of 1 component it is sinusoidal and as a function of the frequency $\hat{\omega}$ of 1 component we get a global minimum (located at ω) contained in a narrow valley, surrounded by many local minima and that gets narrower when the damping d increases (see figure 3.4).

3.4 Additional properties

In this section we describe some additional algebraical, geometrical and statistical properties of the STLS problem.

3.4.1 Orthogonality property

Let us first consider the LS solution of the system

$$Ax \approx b.$$

It is well-known that for the LS problem the following is true

$$e \perp b + e,$$

where e is the residual vector at the solution $e = Ax_{LS}$, with x_{LS} the LS solution of x. We will now prove a similar property for the STLS problem using its projection interpretation.
From (3.19) we know that

$$\Delta\tilde{s} = -\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T(\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T)^{-1}\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{s},$$

and thus

$$\tilde{s} + \Delta\tilde{s} = (I_k - \tilde{\mathbf{H}}_{\mathbf{2}}(y)^T(\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{\mathbf{H}}_{\mathbf{2}}(y)^T)^{-1}\tilde{\mathbf{H}}_{\mathbf{2}}(y))\tilde{s}.$$

Using the formulas of $\Delta\tilde{s}$ and $s + \Delta\tilde{s}$ it is easy to see that:

$$\Delta\tilde{s}^T(\tilde{s} + \Delta\tilde{s}) = \Delta s^T W(s + \Delta s) = 0,$$

meaning that Δs and $s + \Delta s$ are orthogonal with respect to the metric W. Geometrically this follows immediately from the fact that the STLS can be interpreted as an orthogonal projection, as illustrated in figure 5.1.

3.4.2 Statistical interpretation

Up till now, the STLS problem was considered from an algebraic and geometrical point of view. Remember that in chapter 1, the STLS method was presented as a method for estimating the parameters $x \in \mathbb{R}^{n\times 1}$ given some noisy measurements. In order to discuss the statistical properties of STLS as an estimator, we first have to define the statistical "measurement" model that we consider. For the ordinary TLS case the corresponding model was shown to be the classical Errors-In-Variables (EIV) model (see [96]). The classical EIV model is described by:

$$\begin{aligned} & b_0 = A_0 x_0 \\ & \text{with } b = b_0 + \Delta b \text{ and } A = A_0 + \Delta A, \end{aligned} \tag{3.56}$$

where A_0 and b_0 contain the true unobservable quantities and A and b are their measured counterparts. Furthermore, the rows of the matrix $[\Delta A\ \Delta b]$ are assumed to be i.i.d. with common zero mean vector and common covariance

matrix $C = \sigma_\nu^2 I_{n+1}$, with σ_ν^2 unknown. To demonstrate the statistical properties, we consider a similar model, based on the same equations as in (3.56) but with different statistical assumptions on the elements of $\Delta S = [\Delta A\ \Delta b]$. In the model we consider it can very well happen that e.g. along the antidiagonals of ΔS we find the same random variable, thereby violating the independency assumption between the rows of the EIV model. In the model we consider, the random variables can be represented by a vector Δs that contains the k different random variables of ΔS. Thus if along the antidiagonals of ΔS we find the same random variable, each of the antidiagonals will correspond to one entry of $\Delta s \in \mathbb{R}^{k\times 1}$. The restriction we apply is that all the elements of ΔS contain linear combinations of the random variables in Δs. Furthermore the random variables in Δs are assumed to be i.i.d. with unknown variance σ_ν.
Having proposed the model and the assumptions on the "measurement" errors, we can now derive a ML estimator for the model parameters x. For ease of notation we let $y = [x^T\ \ -1]^T$. If the matrix $S = [A\ b]$ contains the measured values, we know that the unobservable true values should obey a linear relation or

$$(S - \Delta S)y = Sy + \mathbf{H_2}(y)\Delta s = 0$$

or

$$Sy = -\mathbf{H_2}(y)\Delta s \equiv e. \tag{3.57}$$

The e defined in the last equation can be seen as an "equation error". Since the elements of e are linear combinations of Gaussian random variables, they obey Gaussian distributions themselves. It is then well known that a ML estimate for y can be obtained by minimizing

$$e^T R^{-1} e = y^T S^T R^{-1} S y,$$

where R is the covariance matrix of e and thus

$$R = E(ee^T) = \mathbf{H_2}(y)E(\Delta s \Delta s^T)\mathbf{H_2}(y)^T = \mathbf{H_2}(y)\mathbf{H_2}(y)^T$$

where E is the expected value operator and the last equation followed from the noise assumptions we made in our model about Δs. Summarizing we see that for the proposed model, a ML estimate for y can be obtained from the observations in S by solving

$$\min_y y^T S^T (\mathbf{H_2}(y)\mathbf{H_2}(y)^T)^{-1} S y,$$

where $y = [x^T\ \ -1]^T$. The ML estimate is thus obtained by solving the STLS problem involving S, where W is set to I_k.

3.5 Conclusion

In this chapter we have presented properties and interpretations of the univariate STLS problem. We have considered the existence of a solution to the STLS problem. By introducing the definition of nongeneric STLS problems, it has been shown that under mild conditions, these nongeneric STLS problems fail to have a solution. The theory has been illustrated with some examples. Furthermore, the non-uniqueness of the solution has been illustrated by means of several examples.

A projection interpretation of the STLS problem has been developed. The latter interpretation has lead to a new definition of structure-dependent signal and noise subspaces, whereby the STLS approach can be seen as a subspace approach involving structure-dependent subspaces. Using the geometrical insight provided by the projection interpretation, an upperbound to the STLS objective function (weighted sum of squares) was derived. For reasons of completeness a lower bound to the same function has been added as well. The structured-dependent subspace interpretation was illustrated on a Hankel matrix example and on an unstructured matrix example. The usefulness of the derived bounds was demonstrated using 4 examples.

Furthermore, the invertibility of the matrix $\mathbf{D_2}(y)$, appearing in the projection interpretation as well as in many other places (e.g. RiSVD framework), has been addressed. The theory has been illustrated by several examples. For one of them (symmetric Toeplitz matrix), it has been proven in detail where the invertibility problem comes from and how it can be circumvented in STLS problems involving this particular structure.

Furthermore, we have treated the STLS problem involving Hankel matrices, by using a new parametrization of the set of rank deficient Hankel matrices. It has been shown how the difficulty of this particular STLS problem is related to the parameters of the new parametrization. The analysis can straightforwardly be extended to Toeplitz matrices, block Hankel matrices and block Toeplitz matrices.

Finally we have proposed some orthogonality properties of the STLS solution as wel as a ML interpretation of the STLS problem.

Chapter 4

Algorithms for solving the STLS problem

In this chapter we discuss algorithms for solving the STLS problem. The first section deals with 2 *well-known methods, namely Cadzow's method and the Iterative Quadratic Maximum Likelihood (IQML) method, that are often used to solve the STLS problem. Despite claims in literature, they are suboptimal as will be shown. The next three sections mainly contain an overview of existing algorithms, "capable" of providing optimal solutions to the STLS problem. We say "capable", since as was shown in previous chapters, the STLS problem is a non convex problem and as a consequence it can have many local minima. Taking into account the nature of the STLS problem (it is a nonlinear constrained optimization problem), it should come as no surprise that all the algorithms will be iterative. Section 4.2 contains algorithms for solving STLS problems within the CTLS framework. Section 4.3 describes algorithms for solving STLS problems within the RiSVD framework and section 4.4 deals with algorithms for solving STLS problems within the STLN framework. This is also the framework we will consider for further development of algorithms. We present an improved algorithm for this STLN framework.*
Section 4.5 describes a new initialization method for STLS problems, which is shown to outperform existing initialization methods. It goes without saying that it is impossible to develop 1 *algorithm capable of efficiently solving STLS problems for different structures. Whenever an STLS problem turns up in a specific application, an efficient version of one of the previously mentioned algorithms has to be developed by exploiting particular structures of the involved matrices or even by using specific parametrizations of the considered structured rank deficient matrices. Also in this chapter, it is assumed that the encountered STLS problems are generic, such that (3.1) and (3.3) are equivalent formulations of*

the STLS problem.

4.1 Suboptimal methods for solving the STLS problem

In this section we discuss some suboptimal methods that are often used for solving the STLS problem. In the first subsection we describe Cadzow's algorithm. The second subsection focuses on the Iterative Quadratic Maximum Likelihood method.

4.1.1 Cadzow's algorithm

The first algorithm we discuss is better known as Cadzow's algorithm [18]. Basically it is an heuristic algorithm that can be used for solving the STLS problem. Using the theory of composite property mapping [18], the STLS problem is solved by an iterative two step procedure. First the closest rank deficient matrix is determined, using the SVD. Since the latter approximation has no particular structure anymore, the second step proceeds by finding the closest appropriately structured matrix to the obtained rank deficient matrix. The algorithm outline is as follows.

Algorithm 4.1.1 (Cadzow's algorithm)

Input: extended data matrix $S \in \mathbb{R}^{m\times(n+1)}$, weighting matrix $W \in \mathbb{R}^{k\times k}$, structure that has to be preserved in the STLS problem, user-defined precision ϵ
Output: rank deficient extended data matrix $S + \Delta S$, the parameter vector x

Step 1: Initialization: $l = 0$, $S^{[l]} = S$, $[U^{[l]}\ \Sigma^{[l]}\ V^{[l]}] = svd(S^{[l]})$,
$\sigma_{n+1}^{[l]} = \Sigma^{[l]}(n+1, n+1)$
Step 2: while $\sigma_{n+1}^{[l]} > \epsilon$
Step 2.1: $\Sigma^{[l]}(n+1, n+1) = 0$, $S_a = U^{[l]}\Sigma^{[l]}V^{[l]T}$
Step 2.2: $S^{[l+1]}$ = closest (in the by W specified L_2 norm) matrix to S_a having the requested structure.
Step 2.3: $[U^{[l+1]}\ \Sigma^{[l+1]}\ V^{[l+1]}] = svd(S^{[l+1]})$,
$\sigma_{n+1}^{[l+1]} = \Sigma^{[l+1]}(n+1, n+1)$, $l = l + 1$
end

where the weighted L_2 norm of s, specified by W, is calculated as follows: $s^T W s$, where $s \in \mathbb{R}^{k\times 1}$ is a vector containing the k different elements of S. As proven in [18], algorithm 4.1.1 will converge to a rank deficient matrix $S + \Delta S$ having the requested structure. However, in the same paper it is also mentioned that the order of applying the property mapping can change the final result, thereby indicating the suboptimality of the approach. The latter is further confirmed in e.g. [67] and [31].

4.1.2 Iterative Quadratic Maximum Likelihood algorithm

Before starting the discussion on the Iterative Quadratic Maximum Likelihood (IQML) method, we remind that it is possible (see e.g. (2.5) or (2.13)) to replace (3.3) with the following equivalent problem:

$$y^T S^T \mathbf{D_2}(y)^{-1} S y,\ y^T y = 1, \tag{4.1}$$

where $\mathbf{D_2}(y) \in \mathbb{R}^{k\times k}$ is the matrix function described in e.g. (2.7). The Iterative Quadratic Maximum Likelihood (IQML) algorithm was initially [17] designed for estimating the parameters of superimposed complex damped exponentials in noise. For this purpose a cost function similar to (4.1) was derived [1]. The same algorithm as in [17] can thus be used for solving the STLS problem.

Algorithm 4.1.2 (IQML Algorithm)

Input: user-defined precision ϵ, structure that has to be preserved in the STLS problem (i.e. the formula for calculating $\mathbf{D_2}(y)$)
Output: the parameter vector x

Step 1: Initialize $y^{[0]} = arg\min_{y,\|y\|_2=1} y^T S^T S y$, $l = 0$
Step 2: $y^{[l+1]} = arg\min_{y,\|y\|_2=1} y^T S^T \mathbf{D_2}(y^{[l]})^{-1} S y$
Step 3: if $\|y^{[l+1]} - y^{[l]}\|_2 < \epsilon$ then $x = -y(1:n)/y(n+1)$
else $l = l + 1$ and goto Step 2

Let y_{IQML} represent the vector y obtained at convergence of algorithm 4.1.2. It is clear that y_{IQML} solves the following problem:

$$y_{IQML} = arg \min_{y,\|y\|_2=1} y^T S^T \mathbf{D_2}(y_{IQML})^{-1} S y.$$

This clearly means that y_{IQML} is the eigenvector of $S^T \mathbf{D_2}(y_{IQML})^{-1} S$ that corresponds to its smallest eigenvalue. We will now show that y_{IQML} is not

[1] as a matter of fact, writing out (4.1) for the STLS problem in case a Toeplitz structure has to be fixed and $W = I$, yields the cost function described in [17].

the optimal solution to (3.3). Therefore we apply the method of Lagrange multipliers to the equivalent problem (4.1). The Lagrangian of the latter problem is $L = y^T S^T \mathbf{D_2}(y)^{-1} S y - \lambda(y^T y - 1)$. Differentiating L with respect to y and λ gives the following necessary conditions for y to be a solution of (4.1):

$$2S^T \mathbf{D_2}(y)^{-1} S y - z_y = \lambda y \tag{4.2}$$

$$\text{with } z_y = \begin{bmatrix} y^T S^T \mathbf{D_2}(y)^{-1} \frac{\delta \mathbf{D_2}(y)}{\delta y(1)} \mathbf{D_2}(y)^{-1} S y \\ \vdots \\ y^T S^T \mathbf{D_2}(y)^{-1} \frac{\delta \mathbf{D_2}(y)}{\delta y(n+1)} \mathbf{D_2}(y)^{-1} S y \end{bmatrix}$$

$$y^T y = 1. \tag{4.3}$$

Let us represent the optimal solution of (4.1) by y_{STLS}. From (4.2) we clearly see that y_{STLS} is not an eigenvector of $S^T \mathbf{D_2}(y_{STLS})^{-1} S$. The latter implies that the the solution obtained with algorithm 4.1.2 is suboptimal. Furthermore, from lemma 2.3.4, we know that the cost function in (4.1) is scaling invariant and thus the constraint $y^T y = 1$ only serves to select one of the infinite number of solutions. The latter implies that $\lambda = 0$ in (4.2)-(4.3). Thus it is easy to derive a heuristic algorithm, similar to algorithm 4.1.2, simply by replacing Step 2 with

$$\text{Step 2: } y^{[l+1]} = -(2S^T \mathbf{D_2}(y^{[l]})^{-1} S)^{-1} z_{y^{[l]}}.$$

For the heuristic algorithm obtained in this way, we can say that if convergence takes place, the necessary conditions for a solution of (4.1) are satisfied [2].
Summarizing we can say that algorithm 4.1.2 yields a suboptimal solution to the STLS problem (3.3), because a term (namely z_y) in the differentiation w.r.t. y of the objective function in (4.1) is not taken into account [3]. An often made mistake (see e.g. [103]) is that by invoking ML arguments, the correct STLS formulation for the problem at hand is derived but then use is made of the suboptimal IQML algorithm 4.1.2. Evidently the statistical accuracy of the estimates can seriously be degraded by this improper choice of algorithm. This will be illustrated with a small example in the next paragraph.
The problem we consider is the following. A vector $s \in \mathbb{R}^{(m+n)\times 1}$ is given such that

$$s(l) = \sum_{i=1}^{n} a_i e^{jp_i} e^{j2\pi f_i (l-1)\Delta t} e^{d_i (l-1)\Delta t} + e(l), l = 1, \ldots, m+n,$$

where $a_i, i = 1, \ldots, n$ are the amplitudes, $p_i, i = 1, \ldots, n$ the phases, $f_i, i = 1, \ldots, n$ the frequencies and $d_i, i = 1, \ldots, n$ the dampings of the n complex

[2] tests we performed using this heuristic algorithm, showed that the algorithm converges to the optimal solution of the STLS problem; however, the algorithm is computationally intensive and needs further study in order to derive efficient versions.

[3] the reason for suboptimality suggested in [94], namely that the IQML constraint $y^T y = 1$ would be wrong, is misleading, since any constraint $y^T X y = 1$ with X positive definite does not change the solution of (2.6) or (3.3).

damped exponentials, Δt the sampling interval (set to 1 in the remainder of this section) and $e \in \mathbb{R}^{(m+n)\times 1}$ an i.i.d. Gaussian noise vector with standard deviation σ_e. The goal is to determine ML estimates of the parameters of the n complex damped exponentials. It is now briefly explained how this estimation can be carried out by solving an STLS problem.
First of all remember from section 3.3.1 that a rank deficient Hankel (or Toeplitz) matrix $S + \Delta S \in \mathbb{R}^{m\times(n+1)}$ can be parametrized by the parameters of n complex damped exponentials that sum up to $s + \Delta s \in \mathbb{R}^{(m+n)\times 1}$. Secondly, the STLS formulation (3.3) with $W = I_k$, in which the measured signal s is the sum of an unobserved signal s_{ex} (satisfying $\mathbf{M}(s_{ex})y_{ex} = 0$, where $\mathbf{M}$ defines the structure that needs to be preserved in the STLS problem) and i.i.d. Gaussian noise, can be shown (see chapter 3) to be a ML formulation leading to a ML estimate of y_{ex}. Since the frequencies $f_i, i = 1,\ldots,n$ and dampings $d_i, i = 1,\ldots,n$ of the complex damped exponentials are obtained from the roots of the polynomial constructed from the elements $y(i), i = 1,\ldots,n+1$ of the estimated y, they are ML estimates as well. Furthermore, the amplitudes $a_i, i = 1,\ldots,n$ and phases $p_i, i = 1,\ldots,n$ are obtained by solving a system containing the ML estimates of the frequencies and dampings and therefore they are ML estimates too.
Thus under the assumption that the signal s is a sum of n complex damped exponentials perturbed with additive i.i.d. Gaussian noise, ML estimates for the underlying parameters can be obtained by solving the STLS problem (3.3) in which S is a Toeplitz (or Hankel matrix) with s stored in its first row and column (or in its first column and last row), $W = I_n$ and the structure to be preserved is the Toeplitz (or Hankel) structure.
To illustrate the suboptimality of algorithm 4.1.2, we will solve the STLS problem under consideration with a second algorithm, algorithm 4.4.1 (i.e. the $STLN_1$algorithm), that is optimal and will be described in section 4.4. We first consider the following small example where the signal s contains only 2 complex damped exponentials perturbed with i.i.d. Gaussian noise:
$m = 4$, $n = 2$, $f_1 = 0.3$, $f_2 = -0.3$, $d_1 = -0.08$, $d_2 = -0.08$, $a_1 = 2$, $a_2 = 2$, $p_1 = 0.2$, $p_2 = -0.2$, $\sigma_e = 0.6$.
To illustrate the suboptimality of the algorithm 4.1.2 we compute $S^T\mathbf{D_2}(y)^{-1}Sy - z_y$ (this should be 0, see (4.2)) for both y_{IQML} and y_{STLN_1} (where y_{STLN_1} is the result obtained by algorithm 4.4.1) obtained for one realization of the noisy signal s. In the first case this yields $[0.0756\ -0.1193\ -0.0015]^T$ and with algorithm 4.4.1 we get $10^{-13}[-0.2626\ 0.0366\ 0.1998]^T$, which clearly illustrates the suboptimality of the IQML result. The result of algorithm 4.4.1, confirms that this algorithm solves the STLS problem in an optimal way.
To show the effect that this suboptimality can have on a larger example, we construct a new example, containing the two components of the previous example and two new ones characterized by the following parameters:
$f_3 = 0.35$, $f_4 = -0.35$, $d_3 = 0$, $d_4 = 0$, $a_3 = 1$, $a_4 = 1$, $p_3 = 0.1$, $p_4 = -0.1$.
Furthermore $m = 54$, $n = 4$ and again $\sigma_e = 0.6$. We do 500 Monte-Carlo simulations, with both the algorithm 4.1.2 and 4.4.1. Since we know the ex-

act parameters, we can calculate the "true" y vector, represented with y_{ex}. To get an idea of the statistical performance of both methods, we average the following relative errors over the 500 runs: $\|y_{IQML} - y_{ex}\|_2/\|y_{ex}\|_2$ and $\|y_{STLN_1} - y_{ex}\|_2/\|y_{ex}\|_2$. For algorithm 4.1.2 we obtain an error of 7.79 % and for algorithm 4.4.1 the error is 1.26 %. Note that this statistical suboptimality was already shown in [88], but only for a specific STLS problem and not for finite sample size.

Remark 4.1.3 Note that the so-called Bootstrapped Total Least Squares (BTLS) method described in [91] acts in a similar way as IQML, i.e. in every iteration $y^T S^T \mathbf{D_2}(y) Sy$ is minimized with $\mathbf{D_2}(y)$ kept constant during the iteration and updated after the iteration with the new y vector. Applied in this way this estimator is not consistent. However, in e.g. [78] it is described how by introducing a noise model and corresponding noise covariance matrix, it is possible to obtain a strongly consistent estimate. ◇

4.2 Algorithms in the CTLS framework

As shown in section 2.2.1 the CTLS framework recasts the STLS problem as an unconstrained optimization problem:

$$\min_x \begin{bmatrix} x \\ -1 \end{bmatrix}^T S^T(\mathbf{H}(x)W^{-1}\mathbf{H}(x)^T)^{-1}S \begin{bmatrix} x \\ -1 \end{bmatrix}, \tag{4.4}$$

with $\mathbf{H}(x) = \sum_{i=1}^{n} x(i)F_i - F_{n+1}$. In [3] Newton's method in its pure form is proposed to solve (4.4), by calculating analytically the first (gradient) and second order (Hessian) information. Although this method is known to have a high convergence rate (superlinear with order at least two), there are several drawbacks related to its application:

- in its pure form, Newton's method is not a descent method, meaning that function values of the function in (4.4) might increase in the course of the algorithm
- convergence can take place to a maximum as well as to a minimum
- problems may occur when the Hessian becomes ill-conditioned.

Even for cases where the previous items do not pose serious problems, Newton's method in its pure form has a prohibitively high numerical complexity. Per iteration this method requires roughly a function evaluation, a gradient evaluation and $O(n^3)$ operations to calculate a new search direction. Therefore

it is better to use alternative methods that try to reduce this large computational cost while not giving in too much on the convergence rate. There are 2 well-known alternatives: quasi-Newton methods and conjugate gradient methods. We will not discuss these methods in detail here, but we will indicate some factors that will determine the choice between these two methods. A comprehensive treatment of these methods can be found in [39, 44, 14].
There are several arguments that speak in favor of the quasi-Newton methods:

- the cost function in (4.4) is not quadratic, thereby implying that the conjugate gradient algorithm will have to restart periodically.
- quasi-Newton methods have been reported to be less sensitive to line search accuracy.

The quasi-Newton methods requires per iteration a function evaluation, a gradient evaluation and $O(n^2)$ flops to obtain a new search direction. The computational complexity of 1 iteration of the conjugate gradient method consists also of a function and a gradient evaluation, but only $O(n)$ computations for the calculation of a new search direction. The conjugate gradient method will thus evidentally be cheaper when the cost for evaluating the function and gradient is negligible with respect to the $O(n^2)$ operations required to calculated a new search direction in the quasi-Newton iteration. Looking at the cost function in (4.4) and keeping in mind that in general for STLS problems $m >> n$, it should be clear that the calculation of the inverse of $\mathbf{H}(x)W^{-1}\mathbf{H}(x)^T \in \mathbb{R}^{m\times m}$, being of order $O(m^3)$ is the major cost per iteration. If no efficient implementation of this inverse can be found (this has to be investigated for the particular STLS problem under consideration, since the matrix function $\mathbf{H}(x)W^{-1}\mathbf{H}(x)^T$ is structure dependent), the quasi-Newton method is preferred on the basis of the previous arguments. However, if also n would become large, the larger storage requirements for the quasi-Newton method ($O(n^2)$) compared to the conjugate gradient method ($O(n)$) might become a determinant factor.
Clearly, an advantage of the CTLS framework compared to the RiSVD and the STLN framework is that it transforms the STLS problem into an optimization problem with very few variables n. There are however some drawbacks related to this framework. First of all remember from section 2.2.4 that the CTLS framework can be seen as a nonlinear EVD decomposition framework. In the same section it was mentioned that another STLS framework –namely the RiSVD framework– can be interpretated as a nonlinear SVD framework. If we look at the TLS problem involving the matrix S, it is clear that although the standard way for solving the TLS problem is via the SVD, it could as well be solved by solving the EVD problem involving S^TS. In the TLS case the latter option is not withheld, because of the squaring of the data matrix which deteriorates the numerical accuracy. Something similar can be said about the STLS problem. In the CTLS framework (the EVD approach) we see that there is also a squaring of the data matrices (see $S^T(\mathbf{H}(x)W^{-1}\mathbf{H}(x)^T)^{-1}S$ in (4.4)).

There is even a second squaring "effect". To understand this note that from section 2.3 it should be clear that

$$(\mathbf{H}(x)W^{-1}\mathbf{H}(x)^T) = (\mathbf{H_2}([x^T \ -1]^T)W^{-1}\mathbf{H_2}([x^T \ -1]^T)) = \mathbf{D_2}([x^T \ -1]^T). \tag{4.5}$$

As can be seen from section 2.2 it is not necessary to do the squaring that appears in (4.5): the STLN framework avoids this squaring.
Furthermore it is also indicated in section 3.2.5, that the invertibility of $\mathbf{D_2}([x^T -1]^T)$ cannot be guaranteed, implying that the problem (4.4) is no longer equivalent to the STLS problem. For all these reasons, the CTLS framework was not withheld for development of new algorithms.

4.3 Algorithms in the RiSVD framework

The final formulation arrived at in the RiSVD formulation is (see section 2.2.2):

$$\begin{array}{l} \text{Find the triplet } (u,\tau,v) \text{ correponding to the smallest } \tau \text{ such that} \\ Sv = \mathbf{D_2}(v)u\tau \quad u^T\mathbf{D_2}(v)u = 1 \\ S^T u = \mathbf{D_1}(u)v\tau \quad v^T\mathbf{D_1}(u)v = 1, v^T v = 1. \end{array} \tag{4.6}$$

As explained in section 2.2.2, formulation (4.6) turns into a RSVD when $\mathbf{D_2}(v)$ and $\mathbf{D_1}(u)$ are kept constant. Since we are looking for the smallest singular value τ of the Riemannian SVD in (4.6), it makes sense to derive a heuristic inverse iteration algorithm , as proposed in [67], in which the matrices $\mathbf{D_1}(u)$ and $\mathbf{D_2}(v)$ are kept constant during one iteration and are updated at the end of each iteration. This yields the following algorithm.

Algorithm 4.3.1 (RiSVD1 Algorithm)

Input: extended data matrix $S \equiv \mathbf{M}(s)$, structure that has to be preserved in the STLS problem (determines the matrix functions $\mathbf{D_2}(v)$ and $\mathbf{D_1}(u)$)
Output: v, t (such that $\mathbf{M}(t) = S + \Delta S$, where $\mathbf{M}(s) = S$ and $\mathbf{M}(t)v = 0$)

Step 1: Initialization: $l = 0$, $[Q\ R] = qr(S)$,
$Q_1 = Q(:, 1 : n+1)$, $Q_2 = Q(:, n+2 : m)$, $R_1 = R(1 : n+1, :)$,
$[U\ \Sigma\ V] = svd(S)$, $u^{[0]} = U(:, n+1)$, $v^{[0]} = V(:, n+1)$,
$\tau^{[0]} = \Sigma(n+1, n+1)$
Step 2: $z^{[l+1]} = R_1^{-T}\mathbf{D_1}(u^{[l]})v^{[l]}\tau^{[l]}$
Step 3: $w^{[l+1]} = -(Q_2^T\mathbf{D_2}(v^{[l]})Q_2)^{-1}Q_2^T\mathbf{D_2}(v^{[l]})Q_1 z^{[l+1]}$
Step 4: $u^{[l+1]} = Q_1 z^{[l+1]} + Q_2 w^{[l+1]}$
Step 5: $v^{[l+1]} = R_1^{-1}Q_1^T\mathbf{D_2}(v^{[l]})u^{[l+1]}$
Step 6: $v^{[l+1]} = v^{[l+1]}/\|v^{[l+1]}\|_2$

Step 7: $\gamma^{[l+1]} = ((v^{[l+1]})^T \mathbf{D_1}(u^{[l+1]})v^{[l+1]})^{0.25}$
Step 8: $u^{[l+1]} = u^{[l+1]}/\gamma^{[l+1]}$
Step 9: $v^{[l+1]} = v^{[l+1]}/\gamma^{[l+1]}$
Step 10: $\tau^{[l+1]} = (v^{[l+1]})^T S^T u^{[l+1]}$
Step 11: $t^{[l+1]} = s - \begin{bmatrix} u^{[l+1]T} T_1 v^{[l+1]} \\ \vdots \\ u^{[l+1]T} T_k v^{[l+1]} \end{bmatrix} \tau^{[l+1]}$
Step 12: if $rank(\mathbf{M}(t^{[l+1]})) < n+1$ then stop
else goto Step 2

where $T_i, i = 1, \ldots, k$ represent the structure-dependent basis matrices as defined in section 2.2.2. In accordance with the inverse iteration interpretation, algorithm 4.3.1 displays a linear convergence rate. Furthermore, the above described algorithm is computationally intensive. Especially Step 3 will dominate the computations for the case $m >> n$. The latter step requires the inversion of the following $(m-n-1) \times (m-n-1)$ matrix:

$$Q_2^T \mathbf{D_2}(v^{[l]}) Q_2.$$

Notice that even if $\mathbf{D_2}(v^{[k]})$ is structured in some way, it will be very difficult to exploit this in the inversion, because the matrix is multiplied with S^T to the left and S to the right, thereby destroying any specific structure of $\mathbf{D_2}(v^{[l]})$. In [67] it is suggested that the computational efficiency can be improved by turning to the following equivalent EVD problem and applying the inverse iteration algorithm on it:

$$S^T \mathbf{D_2}(v)^{-1} S v = \mathbf{D_1}(u) v \tau^2 \;\; v^T \mathbf{D_1}(u) v = 1,$$

and $u = \mathbf{D_2}(v)^{-1} A v / \tau$. Although switching to this new formulation indeed allows the exploitation of the structure of $\mathbf{D_2}(v)$ (since now it is directly inverted), it also introduces the squaring of the data matrices S. Furthermore the algorithm will also have a linear convergence rate at best.

4.4 Algorithms in the STLN framework

As stated in section 2.2.3, the STLN framework sticks to the original optimization formulation of the STLS problem:

$$\begin{aligned} &\min_{\alpha,\beta,x} \alpha^T D_\alpha^2 \alpha + \beta^T D_\beta^2 \beta \\ &\text{such that} \;\; Ax + X\alpha = b + P_2\beta + P_1\alpha, \end{aligned} \tag{4.7}$$

for definitions of the involved matrices and vectors we refer to (2.10) in section 2.2.3. For ease of notation we also introduce the residual defined as $\hat{r} = b + \Delta b - (A + \Delta A)x$. The constraint in (4.7) can thus be expressed as $\hat{r} = 0$. The fact that the STLN framework is essentially the original optimization formulation of STLS constitutes also one of its advantages: there are no immediate problems related to the invertibility of $\mathbf{D_2}(y)$ as was the case in the CTLS and RiSVD framework. Also by keeping the original formulations the structure will better be preserved in the involved matrices, making it easier to exploit this.

In [95], several solution methods to solve (4.7) are described. In [95], [81] and [83] a weighting method is described. The latter corresponds to what in optimization theory is called a penalty function method [39][44], which transforms a constrained optimization problem into an unconstrained optimization problem. This is simply done by adding $w\hat{r}^T\hat{r}$ to the objective function in (4.7) and minimizing the newly obtained objective function, without any constraints. The w is a weight, which has to be taken large enough for the unconstrained and the constrained problem to be equivalent. However, the weight w should not be chosen too large, since this would make the Hessian of the new objective function too ill-conditioned, provoking inaccurate results. Since there is no rule by which we can easily choose a suitable w, it is almost impossible to implement a fully automated version of this algorithm. Even if we make use of refinements like starting with a small weight and gradually increasing the weight, it is well known that penalty function approaches provide low accuracy [38]. Therefore, we will not consider these methods in the remainder of the section.

A second kind of method, briefly mentioned in [95] and applied e.g. in [24], is the following. The optimization problem (4.7) is again solved iteratively. In each iteration a Quadratic Programming (QP) problem is solved, namely the quadratic objective function remains the same, but the constraints are linearized around the current solution point. Schematically, this can be represented as follows (for ease of notation we introduce a new vector containing all the variables of the problem: $v_{tot} = [\alpha^T \ \beta^T \ x^T]^T$)

Algorithm 4.4.1 (STLN1 Algorithm)

Input: extended data matrix $S \in \mathbb{R}^{m\times(n+1)}$, weighting matrices D_α, D_β, user-defined precision ϵ.
Output: rank deficient extended data matrix $[S + \Delta S]$, the parameter vector x

Step 1: Initialization: $k = 0$, $v_{tot}^{[k]} = [\alpha^{[k]T} \ \beta^{[k]T} \ x^{[k]T}]^T$
Step 2: while stopping criterion not satisfied

$$\text{Step 2.1: } H = \begin{bmatrix} D_\alpha^2 & 0 & 0 \\ 0 & D_\beta^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \ g = \begin{bmatrix} D_\alpha^2 \alpha^{[k]} \\ D_\beta^2 \beta^{[k]} \\ 0 \end{bmatrix}$$

$J(i,j) = \frac{\delta \hat{r}(i)}{\delta v(j)}(v_{tot}^{[k]}),\ \forall i = 1,\dots,m,\ \forall j = 1,\dots,m+2n$
Step 2.2: solve the following QP problem:
$\min_{\Delta v} f(v_{tot}^{[k]}) + g(v_{tot}^{[k]})^T \Delta v + \frac{1}{2}\Delta v^T H \Delta v,$
such that $\hat{r} + J\Delta v = 0$
Step 2.3: $v_{tot}^{[k+1]} = v_{tot}^{[k]} + \Delta v$
Step 2.4: Extract $\alpha^{[k]}$, $\beta^{[k]}$ and $x^{[k]}$ from $v_{tot}^{[k]}$
Step 2.5: Compute $\hat{r}$ using $\alpha^{[k]}$, $\beta^{[k]}$ and $x^{[k]}$.
end

with $f(v_{tot}) = 0.5(\alpha^T D_\alpha^2 \alpha + \beta^T D_\beta^2 \beta)$.
Remarks:

- the Hessian H used in algorithm 4.4.1 contains no information on the curvature of the constraints vector $\hat{r}$.
- the discussion on initialization in Step 1 is deferred to the next section, where improved initialization methods are presented.
- the actual stop criterion has to be chosen as a function of the application. In the applications considered in this thesis, we will always use the criterion $\hat{r} < \epsilon$. In other, e.g. signal enhancement (where the condition on rank deficiency is not so critical) the following stop criterion could be used: $\Delta v < \epsilon$.

We propose to solve the QP programming problem in Step 2.2 by solving the following equality constrained LS (ECLS) problem:

Step 2.2: Solve the following ECLS problem:
$min_{\Delta v} \|H^{0.5}\Delta v + H^{0.5} v_{tot}\|_2$
such that $\hat{r} + J\Delta v = 0$.

To see that this is equivalent to Step 2.2 of algorithm 4.4.1, just write out the new objective function to see that its minimum is reached at the same Δv as for the objective function presented in Step 2.2 of algorithm 4.4.1. The latter choice allows to use efficient and robust ECLS solvers that are available (in our simulations we will use the LAPACK [7] routine DGGLSE (implemented as a MEX-file in Matlab)). Algorithm 4.4.1 has a linear convergence rate. The reason for this can be explained as follows. First remember that algorithm 4.4.1 is an iterative one, in which in each iteration the constraints of (4.7) are linearized around the current iterate but the objective function remains unchanged. This linearization is a heuristic procedure that allows to solve an approximate QP problem in each iteration, but comes with no guarantee as far as the convergence rate is concerned. In this respect, we will now apply Newton's method to the Lagrangian of problem (4.7). If we represent the Lagrangian[4] of (4.7) by $L(\beta,\alpha,x,\lambda) = \frac{1}{2}\alpha^T D_\alpha^2 \alpha + \frac{1}{2}\beta^T D_\beta^2 \beta - \sum_{i=1}^{i=m} \lambda(i)\hat{r}(i)$, application of Newton's method to this Lagrangian, gives the Karush-Kuhn-Tucker equations:

$$K \begin{bmatrix} \Delta v \\ \Delta\lambda \end{bmatrix} \equiv \begin{bmatrix} H & -J^T \\ -J & 0 \end{bmatrix} \begin{bmatrix} \Delta v \\ \Delta\lambda \end{bmatrix} = \begin{bmatrix} -g(v_{tot}) + J^T\lambda \\ \hat{r} \end{bmatrix} \tag{4.8}$$

J is the Jacobian of $\hat{r}$, $\Delta v = [\Delta\alpha^T\ \Delta\beta^T\ \Delta x^T]^T$, $g = \nabla_{v_{tot}} f$ and $H = \nabla^2_{v_{tot}} f - \sum_{i=1}^m \lambda(i)\nabla^2_{v_{tot}}\hat{r}(i)$, $2f$ being the objective function in (4.7), with $\nabla_{v_{tot}} f$ the gradient of the scalar function f with respect to the parameters in v_{tot} and $\nabla^2_{v_{tot}} f$ is the Hessian of the scalar function f with respect to the parameters in v_{tot}. It can be shown that solving the Karush-Kuhn-Tucker equations is equivalent to solving the following QP problem [39]:

$$f(v_{tot}) + g^T\Delta v + \frac{1}{2}\Delta v^T H \Delta v \ \text{ such that } \ J(v_{tot})\Delta v + \hat{r}(v_{tot}) = 0 \tag{4.9}$$

with f, g, and H similar as in (4.8). Equation (4.9) clearly shows why algorithm 4.4.1 has a convergence rate that is linear: the QP problem solved in algorithm 4.4.1 does not take the constraints properly into account, since the Hessian H does not contain the term $-\sum_{i=1}^m \lambda(i)\nabla^2_{v_{tot}}\hat{r}(i)$. It is thus possible to use the QP problem derived in (4.9) to develop a new algorithm that is similar to algorithm 4.4.1 but with a new step 2.1:

Algorithm 4.4.2 (STLN2 Algorithm)

Input: extended data matrix $S \in \mathbb{R}^{m\times(n+1)}$, weighting matrices D_α, D_β, user-defined precision ϵ.
Output: rank deficient extended data matrix $[S + \Delta S]$, the parameter vector x

Step 1: Initialization: $k = 0$, $v_{tot}^{[k]} = [\alpha^{[k]T}\ \beta^{[k]T}\ x^{[k]T}]^T$
Step 2: while stopping criterion not satisfied

Step 2.1: $H = \begin{bmatrix} D_\alpha^2 & 0 & 0 \\ 0 & D_\beta^2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \sum_{i=1}^m \lambda(i)\nabla^2_{v_{tot}}\hat{r}(i)$,

$g = \begin{bmatrix} D_\alpha^2\alpha^{[k]} \\ D_\beta^2\beta^{[k]} \\ 0 \end{bmatrix}$

$J(i,j) = \frac{\delta\hat{r}(i)}{\delta v(j)}(v_{tot}^{[k]})$, $\forall i = 1,\dots,m$, $\forall j = 1,\dots,m+2n$

Step 2.2: solve the following QP problem:

[4] Again for ease of notation, we include a factor 0.5 in the objective function.

$\min_{\Delta v} f(v_{tot}^{[k]}) + g(v_{tot}^{[k]})^T \Delta v + \frac{1}{2}\Delta v^T H \Delta v,$
such that $\hat{r} + J\Delta v = 0$
Step 2.3: $v_{tot}^{[k+1]} = v_{tot}^{[k]} + \Delta v$
Step 2.4: Extract $\alpha^{[k]}$, $\beta^{[k]}$ and $x^{[k]}$ from $v_{tot}^{[k]}$
Step 2.5: Compute $\hat{r}$ using $\alpha^{[k]}$, $\beta^{[k]}$ and $x^{[k]}$.
end

Remarks:

- Now the constraints are properly taken into account, such that the incremental step $[\Delta v_{tot}^T \ \Delta\lambda^T]^T$ converges in a quadratic fashion and Δv in a superlinear fashion, as opposed to algorithm 4.4.1. A theoretical explanation can be found in [39].
- Solving the QP subproblem only makes sense when the reduced Hessian $\tilde{H} = Z^T H Z$ (where the columns of Z form a basis for the nullspace of the Jacobian J). There are several ways to work around this problem. One of them is e.g. to perturb the eigenvalues of $\tilde{H}$ such that they are all positive. Such steps can be taken in the course of the algorithm itself by e.g. doing a modified Cholesky decomposition.
- It is straightforward to use a quasi-Newton method for solving the QP subproblem [39, 44]. This could alleviate memory problems, since those methods do not evaluate the analytically derived Hessian, but only perform rank 1 corrections to an initial estimate of the Hessian of the Lagrangian.
- note that the new expression for the Hessian does not entail much extra work compared to the STLN1 algorithm. It is however obvious that the extra terms introduced in the Hessian can complicate the structure, making exploitation of the particular structure possibly more difficult.

The difference in convergence rate will now briefly be illustrated on the following small example

Example 4.4.3 Let $A = [6\ 5\ 4\ 3\ 2]^T$ and $b = [5\ 4\ 3\ 2\ 1]^T$. In the STLS problem (3.1) the Hankel structure needs to be preserved and $W = diag([1\ 2\ 2\ 2\ 2\ 1])$. $\square$

As mentioned in chapter 1, the exact solution for this problem can be determined analytically and is: $x_{opt} = 0.7629230150743218$,
$\alpha_{opt} = [1.640185885241734e{-}01$
$-1.635164416774599e{-}01$ $-2.820841654040049e-01$ $-1.267483597492385e-01$
$3.876060152360497e-01]^T$ and $\beta_{opt} = 6.509795862337404e-01$. Figure 4.1 shows $\|v_{tot} - v_{opt}\|_2$ as a function of the iteration number. The figure clearly shows that both algorithms find the exact solution, but algorithm 4.4.1 has a linear convergence rate, whereas algorithm 4.4.2 converges in a superlinear fashion.

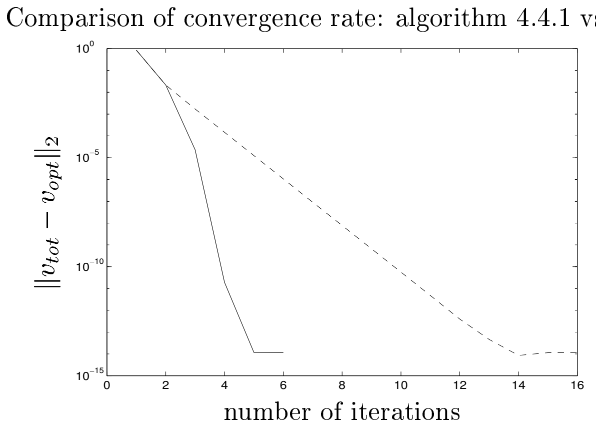


Figure 4.1: *This figure shows* $\|v_{tot} - v_{opt}\|_2$ *as a function of the iteration number for the two algorithms: algorithm 4.4.1 (dashed line) and algorithm 4.4.2 (full line). The start value for* v_{tot} *is* $v_{start} = [0\ 0\ 0\ 0\ 0\ 0\ 7.644999395502275e-01]^T$. *The example shown is example 4.4.3*

4.5 Improved initialization methods

The STLS problem has been shown to be a non convex optimization problem that can suffer from many local minima. Therefore it is obvious that the choice of a good starting value for the iterative algorithm to solve it, is of utmost importance. We will first propose a method to obtain better initial estimates for the case S is Hankel (or (block) Toeplitz or block Hankel) and where the structure to be preserved in the STLS problem is the Hankel (or (block) Toeplitz, Hankel) structure. This new way of initializing STLS problems will be compared to other existing methods by investigating the effect they have on the final result of algorithm 4.4.1 applied to a speech compression example. Finally we suggest a new initialization that can be used for every type of structure and that is based on the projection interpretation of the STLS problem.

4.5.1 Initialization method for Hankel/Toeplitz matrices

One initialization method that is often used (see e.g. [83]) as step 1 in the algorithms STLN1 and STLN2 is the following:

$$\text{Step 1: } v_{tot} \equiv [\alpha^T \; \beta^T \; x^T]^T = [0 \; 0 \; x_{LS}^T]^T,$$

with x_{LS} the LS solution of $Ax \approx b$. Another possibility would be to use:

$$\text{Step 1: } v_{tot} \equiv [\alpha^T \; \beta^T \; x^T]^T = [0 \; 0 \; x_{TLS}^T]^T,$$

with x_{TLS} the TLS solution of $Ax \approx b$. The previous two initialization methods started from a zero estimate for α and β. As will be shown in the speech compression example described in the next section, good initial estimates of α, β *and* x are necessary in order to avoid the many local minima of the STLS problem and thereby getting qualitatively better results. Moreover, the better the initial estimates, the fewer iterations (one iteration refers to the execution of Step 2 of algorithm algorithm 4.4.1 (or 4.4.2), so 1 iteration requires the same number of computations, independent of the initialization method used) are needed to converge. The previous initialization methods still do not yield a feasible initial estimate (i.e. $\hat{r} \neq 0$ after step 1).
The initialization procedure we propose, uses the HTLS subspace algorithm [93]. This subspace algorithm is a TLS variant of Kung et al.'s method described in [57]. It computes a rank-deficient approximating Hankel matrix $[A + \Delta A_{HTLS} \quad b + \Delta b_{HTLS}]$, which is suboptimal in the sense that it is not the closest rank-deficient Hankel matrix to $[A \quad b]$. The subspace algorithm proceeds as follows.

Algorithm 4.5.1 (HTLS Algorithm)

Input: extended Hankel data matrix $[A \quad b] \in \mathbb{R}^{m\times(n+1)}$ $(m > n)$ of full rank $n + 1$.
Output: extended Hankel noise data matrix $[\Delta A_{HTLS} \quad \Delta b_{HTLS}]$ and parameter vector x_{HTLS}, such that $[A + \Delta A_{HTLS} \quad b + \Delta b_{HTLS}]$ is a rank-deficient (of rank n or lower implying that $[A + \Delta A_{HTLS} \quad b + \Delta b_{HTLS}]x_{HTLS} = 0$) Hankel matrix.

Step 1: $y = [A(:,1)^T \quad A(m,2:n) \quad b(m)]^T$
Step 2: $M = ceil((m+n)/2)$
Step 3: $H = hankel(y(1:m+n-M+1), y(m+n-M+1:m+n))$
Step 4: Calculate the right singular vectors $V(:,i), i = 1,\ldots,n$ of H, corresponding to the n largest singular values.
Step 5: Calculate the TLS solution of the following system: $V(1:M-1,1:n)E^H \approx V(2:M,1:n)$. The eigenvalues of E are the estimated signal poles $\hat{z}_l, l = 1,\ldots,n$.
Step 6: Compute the LS solution of the following system of equations

$y(k) \approx \sum_{l=1}^{n} \hat{c}_l \hat{z}_l^k, k = 1, \ldots, m+n,$
to obtain the complex amplitudes $\hat{c}_l, l = 1, \ldots, n$.
Step 7: $\hat{y}(k) = \sum_{l=1}^{n} \hat{c}_l \hat{z}_l^k, k = 1, \ldots, m+n$
Step 8: $[\Delta A_{HTLS} \;\; \Delta b_{HTLS}] = hankel(\hat{y}(1:m), \hat{y}(m:m+n)) - [A \;\; b]$
Step 9: Solve the square system
$A(1:n, 1:n) + \Delta A_{HTLS}(1:n, 1:n))x_{HTLS} = b(1:n) + \Delta b_{HTLS}(1:n)$

Note that $ceil(x)$ rounds x to the nearest integer towards infinity and $hankel(c, r)$ is a function that creates a Hankel matrix whose first column is c and whose last row is r. In steps 1 to 3 we reshape the original rectangular matrix $[A \;\; b]$ into a square (or almost square) matrix H in order to obtain a higher statistical accuracy (see [93]).
The output arguments of algorithm 4.5.1 are very similar to the ones of algorithm 4.4.1. The difference lies in the fact that algorithm 4.5.1 does not yield the *best* rank-deficient structured approximation of the given extended data matrix $[A \;\; b]$. However, as we will see further on, algorithm 4.5.1 is able to provide very good starting values. Note also that algorithm 4.5.1 can easily be converted so that it can handle Toeplitz matrices, since the Toeplitz case can be transformed into a Hankel case by simple permutation of the rows of the given extended data matrix $[A \;\; b]$. To recover the solution of the Toeplitz case, we only have to rearrange the solution of the corresponding Hankel case. As shown in [22], the HTLS approach can also be used with block Hankel and block Toeplitz matrices and for solving multivariate STLS problems $AX \approx B$. However the HTLS-based initialization method is limited to these types of structures.
Also note that the only difference between algorithm 4.5.1 and Kung's method (which is also known as HSVD in NMR spectroscopy [93]), is the way the system of equations in Step 5 of algorithm 4.5.1 is solved. In Kung's approach, this system is solved in a LS sense, whereas in HTLS it is solved in a TLS sense. Intuitively this should yield better results since there is no reason to assume that only the right-hand side of this system of equations is perturbed. Tests on the example presented in the next section have shown that the quality of the initial estimates obtained with algorithm 4.5.1 is in general better (better or same quality of final result in as many or fewer iterations) than with Kung's HSVD method. In the particular example (low rank approximation) we describe, there is only a minor increase in the number of computations by using algorithm 4.5.1 instead of the HSVD-based algorithm. This combined with the fact that an HSVD starting value leads to a considerable increase in the number of iterations that have to be performed in algorithm 4.4.1 (these iterations are a lot more expensive than 1 HSVD or HTLS calculation performed on 1 frame), explains why we choose the TLS variant.
The initialization step using algorithm 4.5.1 thus looks as follows:

Step 1: Apply algorithm 4.5.1 to the extended Hankel data matrix $[A\ b]$,
$v \equiv [\alpha^T \beta^T x^T]^T = [\alpha^T_{HTLS} \beta^T_{HTLS} x^T_{HTLS}]^T$,
with $\alpha_{HTLS} = [\Delta A_{HTLS}(:,1)^T \quad \Delta A_{HTLS}(m, 2:n)]^T$ and
$\beta_{HTLS} = \Delta b_{HTLS}(m)$.

4.5.2 Numerical example

In this paragraph we apply the previously proposed initialization procedures on a speech compression application. We will briefly describe the application. Further details can be found in [59][60]. We start from a speech signal s containing 15000 samples sampled at 8 kHz. We process the speech signal on a frame by frame basis. Each frame is 500 samples long ($N = 500$). We stack each frame in a 488×13 Hankel matrix and use algorithm 4.4.1 from the previous section to obtain the closest rank-deficient Hankel matrix (note that we take $D_\alpha, D_\beta = I$, since each data point should be weighted equally, if we want each frame of the signal $s + \Delta s$ to be as close as possible (in L_2 norm) to the corresponding frame of s). As stopping criterion we use the following rule: the change in $\alpha^T D^2_\alpha \alpha + \beta^T D^2_\beta \beta$ between two consecutive iterations should be smaller than $1e-4$, $\hat{r}(v)$ has to be smaller than $1e-4$ and the number of iterations is limited to 30. Each frame of the resulting signal $s + \Delta s$ (that can be read from the first column and last row of the rank-deficient Hankel matrix calculated for that particular frame) can be stored using only 24 parameters. This follows immediately from the fact that the signal in the obtained rank-deficient Hankel matrix satisfies a linear prediction equation. The latter signal can thus be reconstructed starting from the first 12 elements on the first row of the calculated rank-deficient Hankel matrix together with the 12 linear prediction coefficients (which are derived from the elements of the null vector of the calculated Hankel matrix). In the actual implementation of the speech coder, we do not use the first 12 samples of the frame and the corresponding prediction coefficients because this procedure can become numerically unstable. Instead we apply again HTLS but then to the rank-deficient Hankel matrix obtained at the end of algorithm 4.4.1. The compressed frame is then represented by the calculated amplitudes, phases, frequencies and dampings (see chapter 6 for further details).

Important in the compression stage is the quality of the result and the speed at which it is obtained. Calculations are mainly dominated by the iterations in Step 2 of algorithm 4.4.1. A good starting value should therefore yield a good quality (i.e. the frames of the signal $s + \Delta s$ we obtain at the end of algorithm 4.4.1, have to be as close as possible to the corresponding frames of the original signal s) in a minimal number of iterations (i.e. the number of times Step 2 of algorithm 4.4.1 is repeated). Figure 4.2 shows the quality of the solutions obtained with algorithm 4.4.1, using the different initial estimates. We show a plot of the Signal-to-Noise Ratio (SNR) versus the frame number. The SNR of

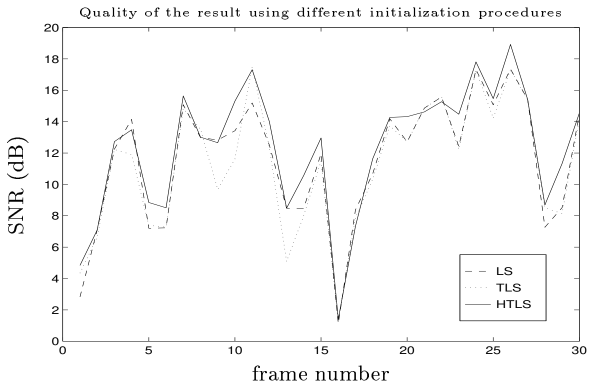


Figure 4.2: *This figure shows the quality of the solution obtained with algorithm 4.4.1, using different initialization procedures.*

the ith frame is defined as follows:

$$20log\frac{\|s(1+(i-1)500:i500)\|_2}{\|\Delta s(1+(i-1)500:i500)\|_2}.$$

In figure 4.2, we clearly see that the HTLS starting values generally yield a better quality than the LS starting values. Especially for the frames where LS yields a low SNR, HTLS is able to improve the quality considerably. The TLS initialization does not really outperform the LS starting value. The explanation is obvious: the vector x_{TLS} is obtained from the framework in which both left and right hand side of the equations $Ax \approx b$ are perturbed. The initialization method based on the TLS solution can not use the perturbation on A and b, since that perturbation no longer has a Hankel structure. In the LS case it is only the right hand side that is being perturbed. Since $\|Ax_{LS} - b\|_2 < \|Ax_{TLS} - b\|_2$, it is obvious that with the LS initialization described in the previous section, the equality constraints are better satisfied at the beginning than with the TLS initialization, providing an intuitive explanation for the bad performance of the TLS based (as compared to the LS based) initialization method. Figure 4.3 shows the number of iterations algorithm 4.4.1 needs in order to converge (i.e. satisfy the stopping criterion). Here we see a clear advantage for the HTLS based initialization method.

4.5.3 Initialization method for other structured matrices

The previous initialization method was only applicable to Hankel structures with the obvious extensions towards block Hankel and (block) Toeplitz matrices.

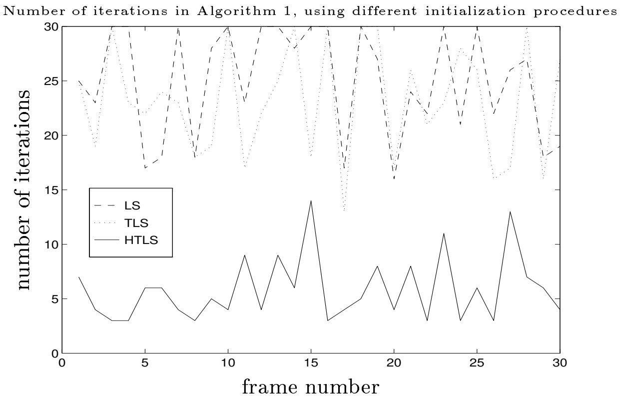


Figure 4.3: *This figure shows the number of iterations performed by algorithm 4.4.1, using different initialization procedures.*

We will now describe an initialization method based on the projection interpretation of the STLS problem. The goal is to find an initial estimate that is feasible (i.e. $\hat{r} = 0$). From formula (3.19) it is clear that the vector $\Delta\tilde{s} \equiv W^{0.5}s$ can be used to obtain a feasible estimate since by filling in the $\Delta\tilde{s}$ from (3.18) we find that:

$$-\Delta S y = -\tilde{\mathbf{H}}_{\mathbf{2}}(y)\Delta\tilde{s} = -\tilde{\mathbf{H}}_{\mathbf{2}}(y)\tilde{s} = Sy.$$

This still leaves open the choice for y, but a good choice for y seems to be the right singular vector corresponding to the smallest singular value of S. In this way, we obtain the solution right away in case the matrix S would already be rank deficient or in case it would be unstructured. Hence the following initialization step is proposed:

$$\text{Step 1: } v \equiv [\alpha^T\ \beta^T\ x^T]^T = [\Delta s^T\ x_{SVD}^T]^T,$$

where Δs is obtained from (3.18) with y replaced by y_{SVD}, the right singular vector corresponding to the smallest singular vector of S. The latter vector is then normalized to yield the parallel vector $[x_{SVD}^T\ \ -1]^T$.

4.6 Conclusion

In this chapter we have discussed algorithms for solving the STLS problem. Two well-known methods, namely Cadzow's method and the Iterative Quadratic Maximum Likelihood (IQML) method, that are often used to solve the STLS problem, have been reviewed. Despite claims in literature, they are suboptimal. An overview of existing algorithms, capable of providing optimal solutions to the STLS problem has been given. An improved algorithm for the STLN framework has been proposed, the main improvements being the replacement of the penalty function approach by an equality constrained LS approach and the introduction of a new initialization method. The new initialization method for STLS problems is shown to outperform existing initialization methods.

Chapter 5

Efficient implementation of the deconvolution problem

In this chapter we develop a fast algorithm for the basic deconvolution problem. Deconvolution problems occur in many areas such as reflection seismology, telecommunications, medical applications [20, 76, 8, 43],... to name just a few. The basic deconvolution problem is depicted in figure 5.1, where $u(k)$ represents the input and $y(k)$ the output at time k; $\Delta u(k)$ and $\Delta y(k)$ are i.i.d. white Gaussian measurement noise added respectively to the input and to the output. The system, represented by its transfer function $X(z)$, is a linear time-invariant system with impulse response $x \in \mathbb{R}^{n\times 1}$. The basic deconvolution problem can now be formulated as follows:

> Given the measurements $u(k) + \Delta u(k), k = 1, \ldots, m$ and $y(k) + \Delta y(k)$, $k = 1, \ldots, m$, of a system, find a Maximum Likelihood (ML) estimate for the system impulse response $x(i), i = 1, \ldots, n$.

In the first section we show that the ML estimator for the basic deconvolution problem is an STLS problem. Section 5.2 describes a fast algorithm for solving the kernel problem in each iteration of the STLN approach applied to the basic deconvolution problem: a Least Squares (LS) problem involving structured matrices. As will be shown, the algorithm is based on the low displacement rank of the involved matrices. Due to the zero initial conditions that appear in the example described in section 5.4, we consider a modified version of the basic deconvolution problem in section 5.3. Section 5.4 describes an example of a typical deconvolution problem. A simulation experiment based on a medical application (whose kernel problem is the modified deconvolution problem) in renography is described. This example is used to demonstrate the improved efficiency of the new algorithm compared to other algorithms for solving this

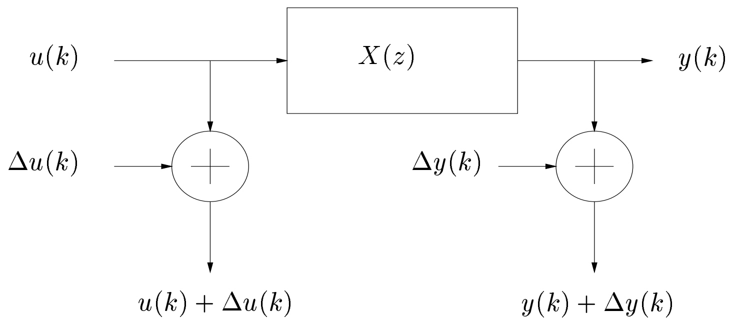


Figure 5.1: *This figure shows a schematic outline of the basic deconvolution problem. The goal is to estimate the impulse response x starting from the noisy input $(u + \Delta u)$ and the noisy output $(y + \Delta y)$.*

type of STLS problem. By means of a Monte-Carlo simulation, using several noise levels, we show the improved statistical accuracy of the deconvolution results obtained with the STLS estimator as compared to other estimators such as the TLS estimator that do not impose a structure on the correction matrix $[\Delta A \;\; \Delta b]$.

5.1 The basic deconvolution problem

Starting from the problem formulation of the basic deconvolution problem as described in the introduction to this chapter, it is straightforward to show that a ML estimate can be found as the solution of the following problem (for a proof, see chapter 3):

$$\begin{aligned} &\min_{\Delta A, x} \alpha^T \alpha + \beta^T \beta \\ &\text{s.t. } (A + \Delta A)x = y + \beta, \end{aligned} \tag{5.1}$$

with

$$A = \begin{bmatrix} u(n) & u(n-1) & \ldots & u(1) \\ u(n+1) & u(n) & \ldots & u(2) \\ \vdots & \ddots & & \vdots \\ u(m+n-1) & u(m+n-2) & \ldots & u(m) \end{bmatrix} \in \mathbb{R}^{m \times n},$$

$$\Delta A = \begin{bmatrix} \alpha(n) & \alpha(n-1) & \ldots & \alpha(1) \\ \alpha(n+1) & \alpha(n) & \ldots & \alpha(2) \\ \vdots & \ddots & & \vdots \\ \alpha(m+n-1) & \alpha(m+n-2) & \ldots & \alpha(m) \end{bmatrix} \in \mathbb{R}^{m \times n},$$

$$\beta = (A + \Delta A)x - y \in \mathbb{R}^{m \times 1}$$

with ΔA the correction applied to A, β the correction applied to y, $y \in \mathbb{R}^{m \times 1}$ the output and $x \in \mathbb{R}^{n \times 1}$ the impulse response. Problem (5.1) clearly is an STLS problem. As already mentioned in the introduction, we will apply the STLN approach, implying that we solve (5.1) as an optimization problem. Following the lines of section 4.4 the outline of the basic deconvolution algorithm is then as follows:

Algorithm 5.1.1 (Basic Deconvolution Algorithm)

Input: extended data matrix $[A \;\; b] \in \mathbb{R}^{m \times (n+1)}$ $(m > n)$ of full rank $n + 1$.
Output: correction vector α and parameter vector x s.t. $\alpha^T \alpha + \beta^T \beta$ is as small as possible and $\beta = (A + \Delta A)x - y$.

Step 1: $\alpha = 0$
$x = A \backslash b$
Step 2: while stop criterion not satisfied
Step 2.1: $\min_{\Delta x, \Delta \alpha} \| M \begin{bmatrix} \Delta \alpha \\ \Delta x \end{bmatrix} + \begin{bmatrix} \beta \\ \alpha \end{bmatrix} \|_2$
with $M = \begin{bmatrix} X & A + \Delta A \\ I & 0 \end{bmatrix} \in \mathbb{R}^{(2m+n-1) \times (m+2n-1)}$
Step 2.2: $x = x + \Delta x$,
$\alpha = \alpha + \Delta \alpha$
end

with $X \in \mathbb{R}^{m\times(m+n-1)}$ defined such that $X\alpha = \Delta A x$. Taking ΔA as defined previously in the deconvolution problem (5.1), X becomes

$$X = \begin{bmatrix} x(n) & x(n-1) & \cdots & x(1) & 0 & \cdots & \cdots & 0 \\ 0 & x(n) & x(n-1) & \cdots & x(1) & 0 & & \vdots \\ \vdots & & \ddots & \ddots & & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & x(n) & x(n-1) & \cdots & x(1) \end{bmatrix}.$$

Note that $A\backslash b$ in Step 1 is a shorthand notation for the LS solution of the overdetermined system of equations $Ax \approx b$. As described in section 4.5, more advanced initialization steps are possible. They yield better starting values in the sense that convergence takes place in fewer iterations and to a better local minimum. However the price to be paid is an increase in computational complexity of the initialization. Due to the nature of the problem we consider, the simple LS estimate will turn out to be sufficient in the considered application. We use the following stop criterion in our implementation of the algorithm:

$$\|[\Delta\alpha^T \;\; \Delta x^T]\|_2 < 10^{-6}.$$

Furthermore, we note that algorithm 5.1.1 is basically a special case of algorithm 4.4.1 applied to the basic deconvolution problem. To understand this, simply calculate the matrix H and J as described in algorithm 4.4.1. This yields $H = \begin{bmatrix} I_{2m+n-1} & 0_{(2m+n-1)\times n} \\ 0_{n\times(2m+n-1)} & 0_{n\times n} \end{bmatrix}$ and $J = [-X \; I_m \; -(A+\Delta A)]$, since $\hat{r} = \beta + y - (A+\Delta A)x$. Filling in H and J in step 2.2 (the ECLS version) of algorithm 4.4.1, we obtain:

$$\begin{aligned} &\min_{\Delta v} \left\| \begin{bmatrix} \alpha + \Delta\alpha \\ \beta + \Delta\beta \end{bmatrix} \right\|_2 \\ &\text{such that } -X\Delta\alpha + \Delta\beta - (A+\Delta A)\Delta x = 0. \end{aligned} \tag{5.2}$$

It is obvious in this particular STLS problem, that by using the constraint in (5.2), $\Delta\beta$ can easily be substituted in the objective function of (5.2) thereby yielding the unconstrained optimization problem presented in algorithm 5.1.1.

5.2 Development of a fast algorithm

In this section we describe a fast algorithm for solving the LS problem in Step 2.1 of the basic deconvolution algorithm described in the previous section. It basically consists of a fast triangularization of the matrix M followed by the solution of the normal equations

$$M^T M[\Delta\alpha^T \;\; \Delta x^T]^T = R^T R[\Delta\alpha^T \;\; \Delta x^T]^T = -M^T[\beta^T \;\; \alpha^T]^T,$$

with $M = Q[R^T \quad 0]^T$ the QR factorization of M. The triangularization of the matrix M can be obtained by means of the QR decomposition of M with a computational complexity of $O((m+n)^3)$. The algorithm considered in [83] requires $O(mn^2 + m^2)$ flops. We propose an algorithm for computing the matrix R based on the displacement representation [54, 25] of the matrix $M^T M$ that requires $O(mn + n^2)$ flops. First we describe briefly the displacement representation of a matrix (see [25, 54] for more details).

Let $Z_k \in \mathbb{R}^{k\times k}$ be the lower shift matrix, that is

$$Z_k = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 1 & 0 & \cdots & 0 \\ & \ddots & \ddots & \\ & & 1 & 0 \end{bmatrix}.$$

and $x \in \mathbb{R}^k$. We denote by $\mathbf{L_k}(x)$ the so–called Krylov matrix generated by x :

$$\mathbf{L_k}(x) = \left[x, Z_k x, Z_k^2 x, \ldots, Z_k^{k-1} x\right].$$

The following lemma holds.

Lemma 5.2.1 *For an arbitrary matrix $A \in \mathbb{R}^{k\times k}$,*

$$A - Z_k A Z_k^T = \sum_{i=1}^{\hat{\delta}} x_i y_i^T \text{ if and only if } A = \sum_{i=1}^{\hat{\delta}} \mathbf{L_k}(x_i)\mathbf{L_k}(y_i)^T.$$

Proof: See [25]. □

The matrix pair $G_{\hat{\delta}}(A) = \{X, Y\}$, where $X = \left[x_1, \ldots, x_{\hat{\delta}}\right]$ and $Y = \left[y_1, \ldots, y_{\hat{\delta}}\right]$, is called a *generator* of A. Generators are clearly not unique and can be of different lengths. The smallest possible length is called the *displacement rank* of A and denoted by $\delta(A)$.

Remark 5.2.2 A symmetric matrix A has a symmetric generator, in the sense that $x_i = \pm y_i, i = 1, \ldots, \hat{\delta}$. Hence, its displacement representation has the symmetric form

$$A = \sum_{i=1}^{p} \mathbf{L_k}(x_i)\mathbf{L_k}(x_i)^T - \sum_{i=p+1}^{\hat{\delta}} \mathbf{L_k}(x_i)\mathbf{L_k}(x_i)^T.$$

◇

Given a displacement representation of $M^T M$, it is possible to construct a factorization procedure with a computational complexity proportional to the displacement rank of M^TM. Since M^TM is a block–Toeplitz–like matrix it is more natural to consider its displacement representation with respect to the block–shift matrix $Z = Z_{m+n-1} \oplus Z_n$ [53], where $\oplus$ is defined as follows.

Definition 5.2.3 *Let $P \in \mathbb{R}^{m\times m}$ and $Q \in \mathbb{R}^{n\times n}$ then $P \oplus Q$ is defined by*

$$P \oplus Q = \begin{bmatrix} P & 0_{m\times n} \\ 0_{n\times m} & Q \end{bmatrix}.$$

5.2.1 Generators for M^TM

For the sake of brevity, in the following we indicate the Krylov matrices $\mathbf{L_{m+2n-1}}(x)$ by $\mathbf{L}(x)$. The displacement rank of M^TM is 5, that is $\delta(M^TM) = 5$.

Lemma 5.2.4 *The following vectors form a generator of M^TM.*

$$\begin{aligned}
&x_1 = M(1,:)^T \\
&x_2 = e_1 \\
&x_3 \\
&x_4 \\
&x_5 = [0\ M(m,1:m+n-2)\ 0\ M(m,m+n:m+2n-2)]^T,
\end{aligned}$$

where $e_1 = [1\ \underbrace{0\ldots0}_{m+2n-2}\ ]^T$.

Let $w = M(2:2m+n-1,m+n)/\|M(2:2m+n-1,m+n)\|_2$ *and* $t = M(2:2m+n-1,2:m+2n-1)^T w$. *The generators x_3 and x_4 are then defined in the following way:*

$$x_3(i) = [0\ t^T]^T, \quad x_4 = [0\ t^T(1:m+n-2)\ 0\ t^T(m+n:m+2n-2)]^T.$$

Then

$$M^TM = \sum_{i=1}^{3} \mathbf{L}(x_i)\mathbf{L}(x_i)^T - \sum_{i=4}^{5} \mathbf{L}(x_i)\mathbf{L}(x_i)^T \tag{5.3}$$

Proof: Construct M^TM and ZM^TMZ^T, with $Z = Z_{m+n-1} \oplus Z_n$, then straightforward manipulations show that $M^TM - ZM^TMZ^T$ can be expressed as a sum of 5 rank one matrices. $\square$

Following the technique described in [25, 74] we can easily construct an algorithm for the computation of R that requires only $O(m^2 + mn + n^2)$ flops.

In the following section we consider an algorithm with computational complexity of $O(mn + n^2)$ taking into account the *"sparsity"* of the vectors $x_i, i = 1, \ldots, 5$.

5.2.2 Description of the algorithm

Starting from the vectors $x_i, i = 1, \ldots, 5$, and following the same steps of the method proposed in [25, 74], (see also appendix C.1), we transform these vectors in the following way,

$$\begin{bmatrix} x_i^T \\ x_j^T \end{bmatrix} = Q \begin{bmatrix} x_i^T \\ x_j^T \end{bmatrix}, \tag{5.4}$$

where Q is either a Givens rotation (updating) if $L(x_i)$ and $L(x_j)$ have the same sign in the sum (5.3) or a hyperbolic rotation (downdating) if these terms have opposite sign in the sum (5.3). We perform the downdating step by means of a stabilized hyperbolic rotation [87], since the latter is more stable. Furthermore, at the kth iteration, the matrix Q is chosen to annihilate the kth entry of the resulting vector x_j. At the end of the kth iteration, we have $x_j(k) = 0, j \neq 1$. Then $x_1(k : m + 2n - 1)$ is the kth row of R and we set $x_1(k + 1 : m + 2n - 1) = x_1(k : m + 2n - 2)$.

We divide the algorithm in 4 phases:

1. initialization: $i = 1$,
2. the iterations for $i = 2 : m$,
3. the iterations for $i = m + 1 : m + n - 1$,
4. the iterations for $i = m + n : m + 2n - 1$.

Initialization: $i = 1$
The only vectors with the first entry different from 0 are x_1 and x_2. The new vectors $\tilde{x}_1$ and $\tilde{x}_2$ are computed as

$$\begin{bmatrix} \tilde{x}_1^T \\ \tilde{x}_2^T \end{bmatrix} = G \begin{bmatrix} x_1^T \\ x_2^T \end{bmatrix},$$

where G is the Givens rotation chosen to annihilate the first element of x_2. The first row of R is $\tilde{x}_1^T$. In the following we denote by $x_k^{(i)}, k = 1, \ldots, 5$, the vectors at the ith iteration. Then we define

$$\begin{aligned}
x_1^{(1)} &= [0 \; \tilde{x}_1(1 : m + n - 2)^T \; 0 \; \tilde{x}_1(m + n : m + 2n - 2)^T] \\
x_2^{(1)} &= \tilde{x}_2 \\
x_3^{(1)} &= x_3 \\
x_4^{(1)} &= x_4 \\
x_5^{(1)} &= x_5.
\end{aligned}$$

The number of flops for this phase is $4n$.
Iterations for $i = 2 : m$
In each iteration of this phase $\mathbf{L}(x_1^{(i-1)})$ is updated with $\mathbf{L}(x_2^{(i-1)})$ and $\mathbf{L}(x_3^{(i-1)})$, $\mathbf{L}(x_4^{(i-1)})$ is updated with $\mathbf{L}(x_5^{(i-1)})$ and $\mathbf{L}(x_1^{(i-1)})$ is downdated with $\mathbf{L}(x_4^{(i-1)})$. Since $x_5^{(1)} = [\underbrace{0\ldots0}_{m}\ x_n\ \ldots x_1\ \alpha_{m+n-1}\ldots\alpha_{m+1}]^T$, $\mathbf{L}(x_5^{(1)})$ does not contribute to this phase.

Moreover, the structure of the vectors $x_k^{(i-1)}, k = 1, 2$, at the beginning of the ith iteration is

$$\begin{aligned} x_1^{(i-1)} &= [\underbrace{0\ldots0}_{i-1}\underbrace{*\cdots*}_{n}\underbrace{0\ldots0}_{m-i+1}\underbrace{*\cdots*}_{n-1}]^T \\ x_2^{(i-1)} &= [\underbrace{0\ldots0}_{i-1}\underbrace{*\cdots*}_{n-1}\underbrace{0\ldots0}_{m-i+1}\underbrace{*\cdots*}_{n}]^T, \end{aligned}$$

where the entries $*$ are in general different from 0. Then updating $L(x_1^{(i-1)})$ with $L(x_2^{(i-1)})$ does not modify the sparsity of the updated $x_1^{(i-1)}$. To complete the ith iteration, $L(x_1^{(i-1)})$ must be updated with $L(x_3^{(i-1)})$ and downdated with $L(x_4^{(i-1)})$. To explain this computation, we describe the iteration for $i = 2$, recalling that the vectors $x_3^{(1)}$ and $x_4^{(1)}$ are equal, except for the $(m + n)$th element (the entries of these vectors are generally different from 0). Let

$$x_1^{(1)} = [0\ \xi_2\ldots\xi_{n+1}\underbrace{0\ldots0}_{m-2}\xi_{m+n}\ldots\xi_{m+2n-1}]^T \tag{5.5}$$

$$\begin{aligned} x_3^{(1)} &= [0\ \zeta_2\ldots\zeta_{m+n-1}\ \zeta_{m+n}\ldots\zeta_{m+2n-1}]^T \\ x_4^{(1)} &= [0\ \zeta_2\ldots\zeta_{m+n-1}\ \mu_{m+n}\ldots\zeta_{m+2n-1}]^T. \end{aligned}$$

The Givens rotation used during the updating is

$$G = \begin{bmatrix} c_G^{(1)} & s_G^{(1)} \\ -s_G^{(1)} & c_G^{(1)} \end{bmatrix}, \text{ with } c_G^{(1)} = \frac{\xi_2}{\sqrt{\xi_2^2+\zeta_2^2}} \text{ and } s_G^{(1)} = \frac{\zeta_2}{\sqrt{\xi_2^2+\zeta_2^2}}.$$

The updated vectors $\tilde{x}_1^{(1)}$ and $\tilde{x}_3^{(1)}$ are

$$\tilde{x}_1^{(1)} = c_G^{(1)}x_1^{(1)} + s_G^{(1)}x_3^{(1)} \tag{5.6}$$

$$\tilde{x}_3^{(1)} = -s_G^{(1)}x_1^{(1)} + c_G^{(1)}x_3^{(1)}, \tag{5.7}$$

with $\tilde{x}_1^{(1)} = [0\ \tilde{\xi}_2\ldots\tilde{\xi}_{m+2n-1}]^T$ $\left(\tilde{\xi}_2 = \sqrt{\xi_2^2+\zeta_2^2}\right)$. Moreover, we point out that, for (5.5),

$$\tilde{x}_3^{(1)}(n+2:n+m-1) = c_G^{(1)}x_3^{(1)}(n+2:n+m-1). \tag{5.8}$$

To finish the iteration for $i = 2$, $L(\tilde{x}_1^{(1)})$ must be downdated with $L(x_4^{(1)})$ by means of the stabilized hyperbolic rotation

$$H = \begin{bmatrix} 1 & 0 \\ \rho & \sqrt{1-\rho^2} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{1-\rho^2}} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & \rho \\ 0 & 1 \end{bmatrix},$$

where ρ is such that $H[\tilde{\xi}_2, \zeta_2]^T = [\hat{\xi}_2, 0]^T$. Taking (5.6) into account, it is straightforward to see that

$$\rho = -s_G^{(1)} \text{ and } \sqrt{1-\rho^2} = c_G^{(1)}.$$

The downdated vectors $\hat{x}_1^{(1)}$ and $\tilde{x}_4^{(1)}$ are

$$\hat{x}_1^{(1)} = \frac{\tilde{x}_1^{(1)} - s_G^{(1)} x_4^{(1)}}{c_G^{(1)}} = x_1^{(1)} + \frac{s_G^{(1)}(x_3^{(1)} - x_4^{(1)})}{c_G^{(1)}} \tag{5.9}$$

and

$$\tilde{x}_4^{(1)} = -s_G^{(1)} \hat{x}_1^{(1)} + c_G^{(1)} x_4^{(1)} \tag{5.10}$$

Hence,

$$\begin{aligned} \tilde{x}_4^{(1)} &= -s_G^{(1)} \hat{x}_1^{(1)} + c_G^{(1)} x_4^{(1)} \\ &= -s_G^{(1)} x_1^{(1)} + \frac{{c_G^{(1)}}^2 x_4^{(1)} - {s_G^{(1)}}^2 (x_3^{(1)} - x_4^{(1)})}{c_G^{(1)}} \\ &= -s_G^{(1)} x_1^{(1)} + \frac{x_4^{(1)} - (1 - {c_G^{(1)}}^2) x_3^{(1)}}{c_G^{(1)}} \end{aligned} \tag{5.11}$$

From (5.7) and (5.11), $\tilde{x}_3^{(1)}$ and $\tilde{x}_4^{(1)}$ continue to be equal, except for the $(m+n)$th entry. Furthermore, from (5.9), we observe that $x_1^{(1)}$ and $\hat{x}_1^{(1)}$ differ in their $(n+m)$th entry. $\hat{x}_1^{(1)}$ now becomes the 2nd row of R, and, for the next iteration, the updated vectors are

$$\begin{array}{l} x_1^{(2)} = [0\; 0\; \hat{x}_1^{(1)}(2:m+n-2)^T\; 0\; \hat{x}_1^{(1)}(m+n:m+2n-2)^T]^T \\ x_5^{(2)} = x_5^{(1)} \\ x_2^{(2)} = x_2^{(1)} \\ x_3^{(2)} = \tilde{x}_3^{(1)} \\ x_4^{(2)} = [\tilde{x}_3^{(1)}(1:m+n-1)\; \gamma\; \tilde{x}_3^{(1)}(m+n+1:m+2n-1)]^T \end{array},$$

where $\gamma = -s_G^{(1)} \hat{x}_1^{(1)}(m+n) + c_G^{(1)} x_4^{(1)}(m+n)$. To reduce the computational cost of this phase, we observe that it is not necessary to calculate $x_3^{(2)}(n+3:m+n-1)$ since at the next iteration the corresponding entries of the vector

$x_1^{(2)}$ are equal to 0. Hence for the vector $x_3^{(3)}(n+4:m+n-1)$ the following relation holds,

$$x_3^{(3)}(n+4:m+n-1) = c_G^{(2)} c_G^{(1)} x_3^{(1)}(n+4:m+n-1)$$

and, at the ith iteration

$$x_3^{(i-1)}(n+i:m+n-1) = c_G^{(i-2)} \cdots c_G^{(2)} c_G^{(1)} x_3^{(1)}(n+i:m+n-1).$$

Hence it is sufficient to store the partial product

$$c_G^{(i-2)} \cdots c_G^{(2)} c_G^{(1)} \tag{5.12}$$

into a temporary variable, and multiply $x_3^{(1)}(n+i)$ with this variable at the ith iteration. At the end of each iteration we set $R(i, i: m+2n-1) = x_1^{(i-1)}(i: m+2n-1)$, $x_1^{(i)}(i+1:m+2n-1) = x_i^{(i-1)}(i:m+2n-2)$, $x_1^{(i)}(m+n) = 0$. Hence the number of flops of this phase is $18mn$.

<u>Iterations for $i = m+1 : m+n-1$</u>

The only difference of this phase with the previous one is that also $L(x_5^{(i)})$ must be downdated from $L(x_1^{(i)})$. We set $R(i, i: m+2n-1) = x_1^{(i-1)}(i: m+2n-1)$, $x_1^{(i)}(i+1:m+2n-1) = x_i^{(i-1)}(i:m+2n-2)$, $x_1^{(i)}(m+n) = 0$. The number of flops of this phase is $24n^2$.

<u>Iterations for $i = m+n : m+2n-1$</u>

This phase is similar to the previous one. The only difference is that also the vector $x_4^{(i)}$ must be computed, since now it differs from $x_3^{(i)}$. Also after each iteration of this phase we set $R(i, i: m+2n-1) = x_1^{(i-1)}(i: m+2n-1)$, $x_1^{(i)}(i+1:m+2n-1) = x_i^{(i-1)}(i:m+2n-2)$, $x_1^{(i)}(m+n) = 0$ The number of flops of this phase is $12.5n^2$.

A Matlab-like program describing this algorithm can be found in appendix C.2.

5.3 Modified problem

We will now consider a slightly modified problem (5.1). This modification is necessary for the renography application we consider further on, due to the zero initial conditions in that application. Thus the difference with the basic deconvolution problem formulation is the introduction of an error-free zero upper triangular part in the matrix A and by consequence ΔA also has a zero upper triangular part. The latter means that A and ΔA are as follows:

$$A = \begin{bmatrix} u(1) & 0 & \dots & 0 \\ u(2) & u(1) & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & u(1) \\ \vdots & & & \vdots \\ u(m) & u(m-1) & \dots & u(m-n+1) \end{bmatrix} \in \mathbb{R}^{m \times n},$$

$$\Delta A = \begin{bmatrix} \alpha(1) & 0 & \dots & 0 \\ \alpha(2) & \alpha(1) & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \alpha(1) \\ \vdots & & & \vdots \\ \alpha(m) & \alpha(m-1) & \dots & \alpha(m-n+1) \end{bmatrix} \in \mathbb{R}^{m \times n}.$$

By consequence, X becomes (remember that X is defined by $X\alpha = \Delta A x$):

$$X = \begin{bmatrix} x(1) & & & & \\ x(2) & x(1) & & & \\ \vdots & \ddots & \ddots & & \\ x(n) & \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \ddots & \ddots \\ & & x(n) & x(n-1) & \cdots & x(1) \end{bmatrix} \in \mathbb{R}^{m \times m}.$$

Again we define M as

$$M = \begin{bmatrix} X & A + \Delta A \\ I & 0 \end{bmatrix} \in \mathbb{R}^{2m \times (m+n)},$$

with $I \in \mathbb{R}^{m \times m}$ the identity matrix and $0 \in \mathbb{R}^{m \times n}$ the null matrix. The displacement rank of $M^T M$ with respect to the block shift matrix $Z = Z_m \oplus Z_n$ is 5. Let $w_1 = M(:,1)/\|M(:,1)\|_2$ and $t_1 = M^T w_1$. Let $w_2 = M(:,m+1)/\|M(:,m+1)\|_2$ and $t_2 = M^T w_2$. Then the generators of $M^T M$ with respect to Z are

$$\begin{aligned} x_1 &= t_1 \\ x_2 &= [0, t_2^T(2:m+n)]^T \\ x_3 &= [0, t_2^T(2:m), 0, t_2^T(m+2:m+n)]^T \\ x_4 &= [0, t_1^T(2:m+n)]^T \\ x_5 &= [0, M(m,1:m-1)^T, 0, M(m,m+1:m+n-1)^T]^T \end{aligned}$$

and

$$M^T M = \sum_{i=1}^{2} L(x_i)L(x_i)^T - \sum_{i=3}^{5} L(x_i)L(x_i)^T.$$

Also in this case, taking into account the sparsity of the vectors, x_1, x_4, x_5, and since x_2 and x_3 differ only in the $(m+1)$th entry, following the same technique described for the original deconvolution problem (previous subsection) , it is possible to construct an algorithm for the fast triangularization of M with the same computational complexity as the previous one ($O(mn+n^2)$). We omit the description of the algorithm since it is very similar to the one described for the basic deconvolution problem. A Matlab-like description can be found in appendix C.3.

5.4 Numerical experiments

In this section we illustrate, by means of a deconvolution problem that occurs in renography [37], the efficiency of the algorithm described in section 5.2 and the increased statistical accuracy by using the STLS estimator as compared to the TLS estimator used previously in [97]. The goal is to determine via deconvolution the so-called renal retention function of the kidney, which in system theoretic terms corresponds to the impulse response x of the system in figure 5.1. This retention function visualizes the mean whole kidney transit time of one unit of a tracer, injected into the patient and enables a physician to evaluate the renal function and renal dysfunction severity after transplantation. In order to obtain this impulse response, the following experiment is conducted. A radioactive tracer is injected in an artery of the patient. The input of the system (u in figure 5.1) is the arterial concentration of the radioactive tracer as a function of time. This concentration is measured by means of a gamma camera and thus in discretized time $u(k)$ represents the number of counts registered in the vascular region at the entrance of the kidney under study during the kth sampling interval. The output $y(k)$ (the so-called renogram) represents the number of counts registered in the whole kidney region by the gamma camera in the kth sampling interval. Deconvolution analysis of the renogram is based on modelling the kidney as a linear time-invariant system with zero-initial state. This is why we consider the modified problem, described in section 5.3. In a first paragraph we will compare the efficiency of the new fast algorithm with that of the standard STLN approach (a straightforward implementation of algorithm 4.4.1, which does not exploit the low displacement rank structure of the kernel LS problem) and the improved version presented in [81]. A second paragraph shows the better statistical accuracy of the STLS estimator as compared to the other estimators. Since we want to evaluate some statistical properties of the STLS estimator, we will use the same simulation example as described in [97]. The noiseless input is described as follows:

$$u_0(k+1) = q_1 e^{(-a_1 k \Delta t)} + q_2 e^{(-a_2 k \Delta t)} + q_3 e^{(-a_3 k \Delta t)}, k = 0, 1, 2, \ldots, (t_{obs}/\Delta t) - 1$$

with $q_1 = 40.3$, $q_2 = 45.2$, $q_3 = 15.2$, $a_1 = 1.8$, $a_2 = 0.43$ and $a_3 = 0.035$. Δt represents the sampling interval, expressed in minutes, and t_{obs} is the total observation time. The exact impulse response is characterized by the following function:

$$x_0(k+1) = \begin{cases} 1 & k = 0, 1, 2, \ldots, (t_1/\Delta t) - 1 \\ (k\Delta t - t_2)/(t_1 - t_2) & k = t_1/\Delta t, (t_1/\Delta t) + 1, \ldots, (t_2/\Delta t) - 1 \end{cases},$$

with $t_1 = 3$ minutes, $t_2 = 5$ minutes and $t_{obs} = 20$ minutes. The noiseless output y_0 is then obtained by convolution of the noiseless input u_0 with the proposed impulse response x_0. In matrix format we have that

$$Ax_0 = y_0,$$

where

$$A = \begin{bmatrix} u_0(1) & 0 & \ldots & 0 \\ u_0(2) & u_0(1) & \ldots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & u_0(1) \\ \vdots & \ddots & \ddots & \vdots \\ u_0(m) & u_0(m-1) & \ldots & u_0(m-n+1) \end{bmatrix} \in \mathbb{R}^{m \times n}.$$

For $\Delta t = 1/3$ minutes, we have that $m \equiv 3t_{obs} = 60$ and $n \equiv 3t_2 = 15$. As described in [36], these functions and constants are a realistic simulation of real in-vivo measurements.

Computational efficiency

In this subsection we give the computational cost of the deconvolution algorithm for the modified problem described in section 5.3. Given the iterative nature of algorithm 5.1.1, the total number of floating point operations (flops) is a multiple of the flops necessary to execute step 2.1. Therefore, we will analyze this step in further detail and see how our implementation of step 2.1 compares to standard algorithms for solving step 2.1.

In step 2.1 we solve a LS problem by solving the corresponding normal equations

$$R^T(R \begin{bmatrix} \Delta\alpha \\ \Delta x \end{bmatrix}) = -M^T \begin{bmatrix} \beta \\ \alpha \end{bmatrix}, \tag{5.13}$$

where R is the triangular factor of the QR factorization of M. From (5.13), the description of the fast QR algorithm in section 5.3 and the specific structure of M and R, we obtain the following number of computations for Step 2.1:

1. Calculation of the generators: $4mn + 4m + 2n^2 - 2n + 12$.
2. Construction of right hand side (see (5.13)): $4mn + 4m - n^2 - n + 8$.
3. Fast QR: $O(mn + n^2)$.

4. Solving lower triangular system (see (5.13)): $4mn + 3m + n + 1$.

5. Solving upper triangular system (see (5.13)): $4mn + 5m + n + 6$.

This means that overall the algorithm is $O(mn)$, when $m >> n$ as is mostly the case. We consider 3 different cases. The first case is the simulation example we already described: $m = 60$ and $n = 15$. For the second case, we consider $t_{obs} = 60$ minutes and $t_2 = 5$ minutes, and obtain a data matrix $A \in \mathbb{R}^{m \times n}$, with $m = 180$ and $n = 15$. The third case results from taking $t_{obs} = 20$ and $t_2 = 10$ minutes, yielding a data matrix $A \in \mathbb{R}^{m \times n}$, with $m = 60$ and $n = 30$. In table 5.1 we give the number of flops used by our efficient implementation of the different parts of Step 2.1, as well as their sum. Flops are calculated using the Matlab command *flops*. From the table we clearly can see that the computational cost of Step 2.1 is $O(mn)$.

Table 5.1: *This table shows the number of flops for the different parts of Step 2.1, as well as their sum. The number of flops is measured in Matlab using the flops command.*

$m \times n$	part 1	part 2	part 3	part 4	part 5	total
60×15	4458	3608	22175	3796	3921	37958
180×15	12738	11288	58895	11356	11721	105998
60×30	8583	6518	49640	7411	7536	79688

To illustrate the better computational efficiency of the newly presented efficient implementation (referred to as alg_1) of algorithm 5.1.1 (remember that this is itself an efficient implementation of the earlier presented algorithm 4.4.1, applied to the basic deconvolution problem) , we compare its efficiency with that of the "unefficient" version of algorithm 5.1.1 (referred to as alg_2 and basically the algorithm 5.1.1 using the standard QR for solving step 2.1) described in section 4.4) and the faster STLN algorithm for Toeplitz structured TLS problems described in [81] (referred to as alg_3). As example we take the basic deconvolution problem considered in [95]. We use an example different from the previous paragraph since the algorithm alg_3 solves the basic deconvolution problem and not the modified one. As a consequence alg_1 is the algorithm described in section 5.2. Table 5.2 clearly shows the computational advantage of alg_1 over alg_2 and alg_3, for different sizes $m \times n$ of the matrix A.

Statistical accuracy

We compare the statistical accuracy of the STLS estimator with that of the TLS estimator which in [97] was shown to outperform by far other estimators. To this end, we perform for each noise standard deviation σ_ν a Monte-Carlo simulation consisting of 100 runs. In every run, we add a different realization of i.i.d. white Gaussian noise with standard deviation σ_ν to the noiseless input u_0 and the noiseless output y_0 of the previously described medical simulation

Table 5.2: *This table shows the following ratios: total number of flops of* alg_2 *w.r.t. total number of flops of* alg_1 ($flops_{alg_2}/flops_{alg_1}$) *and total number of flops of* alg_3 *w.r.t. total number of flops of* alg_1 ($flops_{alg_3}/flops_{alg_1}$), *for matrices* A *of dimension* $m \times n$. *The number of flops is measured in Matlab using the flops command.*

$m \times n$	$flops_{alg_2}/flops_{alg_1}$	$flops_{alg_3}/flops_{alg_1}$
90×20	117.2	5.5
180×20	368.6	6.2
90×10	162.5	3.5

example. The obtained noisy vectors u and y are used as input to the modified deconvolution algorithm described at the beginning of section 5.3. To compare the performance of both estimators at a noise level σ_ν, we average for both estimators the following relative error $\frac{\|x-x_0\|_2}{\|x_0\|_2}$ over the different runs. Table 5.3 shows that in the case of the STLS estimator, the relative errors are 9% to 14% lower than in the case of the TLS estimator, confirming the statistical superior performance of the STLS estimator.

Table 5.3: *This table shows the relative error* $\frac{\|x-x_0\|_2}{\|x_0\|_2}$ *for the TLS and STLS estimator, averaged over* 100 *runs, for different noise standard deviations* σ_ν.

σ_ν	TLS	STLS	σ_ν	TLS	STLS
0.05	0.0010	0.0009	0.5	0.0104	0.0091
0.071	0.0015	0.0013	0.707	0.0138	0.0123
0.087	0.0018	0.0016	0.866	0.0177	0.0153
0.1	0.0021	0.0018	1	0.0204	0.0180
0.158	0.0032	0.0029	1.58	0.0313	0.0279
0.224	0.0046	0.0041	2.24	0.0454	0.0402
0.274	0.0056	0.0049	2.74	0.0576	0.0513
0.316	0.0064	0.0057	3.16	0.0660	0.0601

Chapter 6

Speech compression scheme

In this chapter we describe a new subband based coding scheme. In a first step a frame of the incoming signal is fed to a lowpass filter. This yields the low-frequency (LF) part. By subtracting the latter from the incoming signal, we obtain the high-frequency (HF) , non-smoothed part. The motivation behind splitting the speech signal into two bands is that the LF part of a speech signal can efficiently be modeled by means of a linear model (Auto Regressive (AR) model). Instead of determining a classical AR model, leading to a vector of model parameters and a residual vector [63], we apply a minimal correction to the measured data such that the corrected measurements exactly satisfy an AR model. It will be shown that the latter problem is an STLS problem. Moreover, the STLS problem corresponds to an STLS problem in which the Hankel structure of the extended data matrix S needs to be preserved, thereby revealing its link with data fitting by means of a sum of complex damped exponentials. Once this link is established, it should be clear that the LF part of our scheme is related to sinusoidal coding based coders [55]. The main difference is that sinusoidal coders use a sum of undamped *sinusoids to fit a frame of the signal, whereas our scheme uses* exponentially damped *sinusoids.*
As for the HF part of the speech signal, where linear models have poor performance, we use quantization methods and in particular waveform vector quantization (VQ) [42]. Due to the low bit rate of the HF branch of our scheme, quantization noise is quite noticable and it is removed using adaptive postfiltering.
The chapter is organized as follows. Section 6.1 describes our new scheme, starting with the LF part (section 6.1.1) and ending with the HF part (section 6.1.2). In section 6.2 the new scheme is applied to a speech signal. A comparison of the quality of the reconstructed signals is made with the Federal Standard 1016 4800 bps CELP (code excited linear prediction) vocoder [19], both in terms of spectral properties of the reconstruction and in terms of the SNR. Finally,

we conclude in section 6.3.

6.1 Description of the scheme

A schematic view of the algorithm is shown in figure 6.1. The algorithm processes the incoming signal on a frame by frame basis. At the sender side, a frame of the incoming signal is fed to a lowpass filter. We use an 8th order Chebyshev lowpass filter with 0.2 dB ripple in the pass band and cut-off frequency $f_{c.o.}$. The resulting "linear" LF signal is first downsampled by a factor D. Note that the down-sampling is performed by retaining 1 sample every D samples of the LF part of the frame. If the signal was originally sampled at frequency f_{sample}, there is no information loss due to downsampling and consecutive upsampling of the LF part of the signal, if the following inequality is satisfied:

$$2f_{c.o.} < \frac{f_{sample}}{D}. \tag{6.1}$$

The downsampled signal is then fed to the STLS analysis algorithm which yields a bit rate of d_{STLS} bits per second. The "non-linear" HF signal is obtained by subtracting the LF part from the incoming signal. The HF part is preemphasized (the filter used is the following: $1 - 0.95z^{-1}$) in order to increase the importance of the highest frequencies with respect to the lower frequencies in the HF subband. The resulting signal serves as input to the waveform vector quantization (VQ) algorithm, leading to a bit rate of d_{VQ} bits per second. The quantized results of the STLS analysis algorithm and the VQ algorithm are multiplexed, yielding an overall bit rate of $d_{STLS} + d_{VQ}$ bits per second. At the receiver side, the incoming bits are demultiplexed. The STLS synthesis algorithm reconstructs the approximation of the downsampled LF part of the original signal, followed by an upsampling operation. The interpolation filter used is a symmetric one that allows the original data to pass through unchanged and interpolates between so that the mean squared error between the interpolated values and the ideal values is minimized (i.e. we use the Matlab function interp.m). The only approximation introduced in the LF branch is the one induced by the STLS analysis algorithm. The approximation of the HF part of the signal is reconstructed using the code book (CB), followed by an adaptive postfilter routine, in order to reduce the quantization noise. LF and HF reconstructed signals are added to obtain the output signal. The 2 branches (LF and HF branch) as depicted in figure 6.1 will now be explained in more detail.

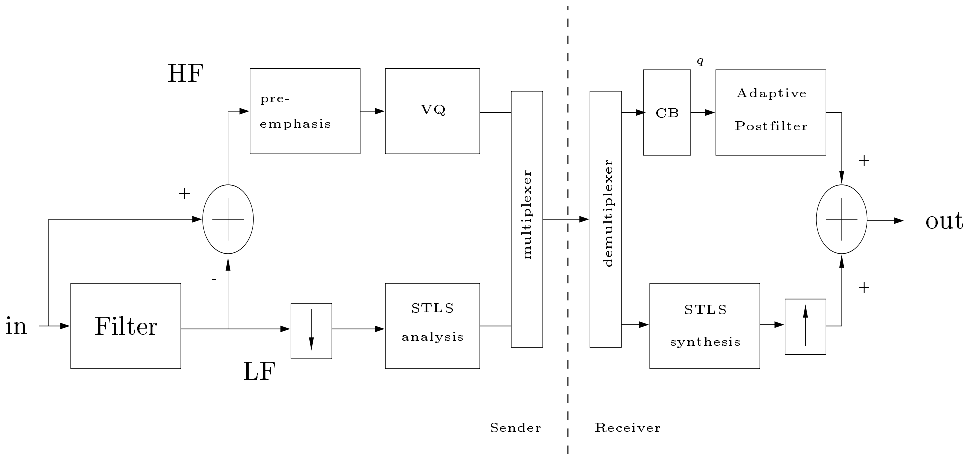


Figure 6.1: *This figure gives an overview of the new scheme.*

6.1.1 LF part

Before we start off with the description of the STLS analysis algorithm, we explain why we use downsampling at the sender side followed by upsampling at the receiver side. As mentioned before there is no quality loss as a result of these operations (as long as (6.1) is satisfied). Moreover downsampling reduces the number of computations that have to be performed in the STLS analysis algorithm, since the number of samples we have to deal with is divided by a factor D: $N_{LFD} \equiv N_{LF}/D$ where N_{LF} is the number of samples per frame before downsampling.
As already explained in the introduction, the output of the lowpass filter of figure 6.1 is very well suited for modeling by a linear AR model (the same is true of course for the downsampled output of the filter).
Let s denote the downsampled output of the lowpass filter. The classical AR approach applied to frame i can then be written as follows:

$$\min_{\substack{e(j),j=1+(i-1)N_{LFD},\ldots,iN_{LFD},\\ x(l),l=1,\ldots,L}} \sum_{j=1+(i-1)N_{LFD}}^{iN_{LFD}} (e(j))^2 \tag{6.2}$$

$$\text{such that } s(l) = \sum_{k=1}^{n} x(k)s(l-k)+e(l),$$

$$l = n+1+(i-1)N_{LFD},\ldots,iN_{LFD},$$

with N_{LFD} the number of samples per downsampled frame , i the frame number, $x \in \mathbb{R}^{n\times 1}$ the vector of model parameters of frame i and n the model

order.
If we use the convention that the vector $s(1+(i-1)N_{LFD} : iN_{LFD})$ can be read from the first column and the last row of the Hankel matrix $S = [A\ b] \in \mathbb{R}^{(N_{LFD}-n)\times(n+1)}$, by starting in the upper left corner and ending in the lower right corner, the constraints of (6.2) can be written as

$$Ax = b + e. \tag{6.3}$$

Together with the objective function in (6.2), we see that (6.3) defines a Least Squares (LS) problem for solving the inconsistent set of equations

$$Ax \approx b.$$

Using this LS approach it is not sufficient to transmit the vector x containing the model parameters together with the first n samples of $s(1+(i-1)N_{LFD} : iN_{LFD})$, in order to reconstruct the signal. At the receiver side, we also need to know the residual vector e of the corresponding frame. This of course limits the attainable compression ratio, since also the residual vector e needs to be quantized and transmitted.
Our approach of the LF part is different. The problem formulation we solve for the ith frame, is the following:

$$\min_{\substack{\Delta s(j), j=1+(i-1)N_{LFD},\ldots,iN_{LFD},\\ x(k), k=1,\ldots,n}} \sum_{j=1+(i-1)N_{LFD}}^{iN_{LFD}} (\Delta s(j))^2 \tag{6.4}$$
$$\text{such that } s(l)+\Delta s(l) = \sum_{k=1}^{n} x(k)(s(l-k)+\Delta s(l-k)),$$
$$l = n+1+(i-1)N_{LFD},\ldots,iN_{LFD}.$$

If we use the convention that the vector $\Delta s(1+(i-1)N_{LFD} : iN_{LFD})$ can be read from the first column and the last row of the Hankel matrix $\Delta S = [\Delta A\ \Delta b] \in \mathbb{R}^{(N_{LFD}-n)\times(n+1)}$, by starting in the upper left corner and ending in the lower right corner, (6.4) can be reformulated as follows:

$$\min_{\substack{\Delta s(j), j=1+(i-1)N_{LFD},\ldots,iN_{LFD},\\ x(k), k=1,\ldots,n}} \sum_{j=1+(i-1)N_{LFD}}^{iN_{LFD}} (\Delta s(j))^2 \tag{6.5}$$
$$\text{such that } (A+\Delta A)x = b + \Delta b$$
$$\text{and } \Delta S \text{ has same structure as } S.$$

The latter clearly defines an STLS problem. Moreover, since it is an STLS problem in which the Hankel structure of the extended data matrix needs to be preserved, we know from section 3.3.1 that problem (6.5) can be reparametrized as follows:

$$\min_{\substack{\Delta s(j), j=1+(i-1)N_{LFD},\ldots,iN_{LFD},\\ x(k),k=1,\ldots,n}} \sum_{j=1+(i-1)N_{LFD}}^{iN_{LFD}} (\Delta s(j))^2 \tag{6.6}$$

$$\text{such that } s(l+(i-1)N_{LFD}) + \Delta s(l+(i-1)N_{LFD}) = \sum_{k=1}^{n} c_k z_k^l,$$

$$l = 1,\ldots,N_{LFD}$$

with $c_k = a_k e^{jp_k}$ the complex amplitude of the kth component and $z_k = e^{j2\pi f_k + d_k}$ the complex pole of the kth component. Furthermore we have that a_k is the amplitude, p_k the phase, d_k the damping and f_k the frequency of the kth component. Note that by using the above signal model, we exclude the non-generic case in which 2 or more signal poles would coincide (this would lead to additional polynomial terms in t, see [77]). Since we only consider real data, the signal poles

$$e^{d_k + j2\pi f_k}, k = 1,\ldots,n$$

appear in complex conjugate pairs and the same goes for the complex amplitudes

$$a_k e^{jp_k}, k = 1,\ldots,n$$

The latter means that when n is odd, there will at least be 1 component k such that $f_k = 0$ and $p_k = 0$ (we say “at least” since there can be an even number of real exponentials when n is even and an odd number of real exponentials when n is odd). As a result, the sum of complex damped exponentials in (6.6) becomes a sum of exponentially damped sinusoids.

Summarizing we can say that the particular STLS problem that needs to be solved for each frame of the LF signal corresponds to finding the sum of n complex damped exponentials that fits as close (in L_2 norm) as possible the LF frame. In this sense the LF part of our scheme is related to other sinusoidal coders, the main difference being that in our scheme damped sinusoids are employed instead of undamped sinusoids. The fact that classical sinusoidal coders only use undamped sinusoids obviously implies that the number of sinusoids needed to fit 1 frame in a satisfactory way is quite high compared to fitting the same frame with damped sinusoids. For this reason classical sinusoidal coders use heuristic methods to reduce the number of parameters, such as e.g. the fitting of an envelope that passes through the obtained amplitudes (see e.g. spectral envelope estimation vocoder (SEEVOC) [75]). Other implementations try to exploit the properties of the speech production mechanism even more (see e.g. the sinusoidal transform coder (STC) [56]).

The STLS analysis algorithm used in the LF branch of the scheme to solve the STLS problem (6.6) uses the HTLS initialization method (algorithm 4.5.1) described in section 4.5 and algorithm 4.4.1 described in section 4.7. The outline of the STLS analysis algorithm is as follows:

STLS Analysis Algorithm

Input: downsampled ith frame of the LF part of the speech signal: $s(j+(i-1)N_{LFD})$, $j=1,\ldots,N_{LFD}$, with N_{LFD} the number of samples per frame, n the order of the AR filter
Output: $f_k^q, d_k^q, a_k^q, p_k^q, k=1,\ldots,n/2$, representing the quantized versions of f_k, d_k, a_k, p_k,
$k=1,\ldots,n/2$ the frequencies, dampings, amplitudes and phases of the complex damped exponentials, satisfying $\sum_{k=1}^{n/2} c_k z_k^j + \overline{c}_k \overline{z}_k^j = s(j+(i-1)N_{LFD}) + \Delta s(j+(i-1)N_{LFD})$, $j=1,\ldots,N_{LFD}$.

Step 1: Apply algorithm 4.4.1 to (6.6) using HTLS starting values (algorithm 4.5.1)
Step 2: Apply HTLS to the result of Step 1
$s(1+(i-1)N_{LFD} : iN_{LFD})+$
$\Delta s(1+(i-1)N_{LFD} : iN_{LFD})$,
to extract $f_k, d_k, a_k, p_k, k=1,\ldots,n/2$
Step 3: Quantize the parameters to obtain the following quantized parameters:
$f_k^q, d_k^q, a_k^q, p_k^q, k=1,\ldots,n/2$

With $c_k = a_k e^{(\sqrt{-1}p_k)}$, $z_k = e^{2\sqrt{-1}\pi f_k + d_k}$ (c_k^q and z_k^q are their counterparts with the parameters replaced by the quantized parameters) and $\overline{x}$ indicating the complex conjugate of x. We note that, as described in [23], it is possible to merge Step 1 and Step 2 of the STLS analysis algorithm. This increases the efficiency by replacing the SVD of Step 2 by a QR factorization. However, the overall effect on the number of computations would be marginal, since the contribution of Step 2 in the total amount of work is minor compared to the iterative steps inside algorithm 4.4.1 in step 1 of the STLS analysis algorithm. The quantization in Step 3 of the analysis algorithm is a simple bit allocation scheme. This will be further explained in section 6.2.
The synthesis part of the LF branch is very simple. We only have to sum the exponentially damped sinusoids using the transmitted quantized parameters $f_k^q, d_k^q, a_k^q, p_k^q, k=1,\ldots,n/2$. This can be summarized as follows:

STLS Synthesis Algorithm

Input: $f_k^q, d_k^q, a_k^q, p_k^q, k=1,\ldots,n/2$, representing the quantized frequencies, dampings, amplitudes and phases of the complex damped exponentials.
Output: signal r, which up to quantization errors equals $s(1+(i-1)N_{LFD} : iN_{LFD}) + \Delta s(1+(i-1)N_{LFD} : iN_{LFD})$, the rank-deficient signal that lies closest to $s(1+(i-1)N_{LFD} : iN_{LFD})$ in L_2 norm.

Step 1: $r = \sum_{k=1}^{n/2} c_k (z_k^q)^j + \overline{c}_k (\overline{z}_k^q)^j$,

$j = 1, \ldots, N_{LFD}$.

Finally, we have to upsample the obtained signal r with the same factor D used to downsample the incoming frame at the sender side.
We note that since the parameters of a frame are determined using only that particular frame, their might be an abrupt transition going from one reconstructed frame to the next one. This can be remedied by introducing an overlap between 2 consecutive frames, say f_1 and f_2. So e.g. instead of taking at the receiver side frames of length N_{LF}, we take frames of length $N_{LF}+N_{ov}$, in such a way that the last N_{ov} samples of frame f_1 overlap with the first N_{ov} samples of frame f_2, i.e. $f_1(N_{LF} - N_{ov} + i)$ "corresponds" to $f_2(i), i = 1, \ldots, N_{ov}$. Eventually we have to replace those overlapping samples by N_{ov} new samples that make up the small frame f_{12}. This is done in the following way:
for $i = 1, \ldots, N_{ov}$
$$f_{12}(i) = \frac{N_{ov}-i}{N_{ov}} f_1(N_{LF} - N_{ov} + i) + \frac{i}{N_{ov}} f_2(N_{ov} + 1)$$
end

6.1.2 HF part

At the sender side, the high-frequency part of the original signal is also processed on a frame by frame basis (N_{HF} samples per frame). First of all the HF part of the incoming frame is preemphasized in order to put more stress on the highest frequencies in this band. Each frame is encoded using a straightforward waveform VQ algorithm, based on a CB generated by the well known LBG algorithm [42]. The codevectors in the CB are w bits long, leading to 2^w different codevectors. At the receiver side decoding is performed by selection of the appropriate waveform from the CB. To reduce the quantization noise of the resulting signal q, we apply an adaptive postfiltering algorithm as explained below.

Adaptive Postfiltering

The idea behind the method proposed here is that quantization noise can be reduced using low rank approximations of the observation matrix of the quantized data. This is achieved by a "successive projections" algorithm which alters the singular value distribution of the signal observation matrix in successive steps simplifying the signal structure and thus removing quantization noise by selecting the most important spectral bands adaptively.

More precisely the algorithm consists in adjusting the distribution of the singular values of the signal Hankel matrix Q, constructed from the signal q, which is the HF reconstructed signal at the receiver side:

$$Q = \begin{bmatrix} q(1) & q(2) & q(3) & \cdots & q(N-p) \\ q(2) & q(3) & q(4) & \cdots & q(N-p+1) \\ \vdots & \vdots & \vdots & & \vdots \\ q(p+1) & q(p+2) & q(p+3) & \cdots & q(N) \end{bmatrix}, \tag{6.7}$$

by projecting it onto the space of "reduced rank matrices" by means of the SVD. Given the fact that this operation destroys the Hankel nature of the matrix, the latter is restored by projecting it onto the space of Hankel matrices by averaging along the cross diagonals [18].
It was shown in [30] that one iteration of the successive projections procedure can be realized through a zero phase filtering operation using the appropriate eigenfilters. Let $H_k(z)$ be the z-transform of the filter associated to the kth left singular vector u_k of the Hankel matrix Q in (6.7),

$$H_k(z) = \sum_{l=1}^{p+1} u_k(l)\, z^{-l}, \tag{6.8}$$

let also $H_k^*(z)$ be the self-reversive polynomial of $H_k(z)$

$$H_k^*(z) = \sum_{l=1}^{p+1} u_k(p+2-l)\, z^{-l}. \tag{6.9}$$

The transfer function

$$Z_k(z) = H_k(z) H_k^*(z). \tag{6.10}$$

preserves the phase of the input signal, ensuring zero phase discrepancy between input and output signals. One single iteration of the successive projections algorithm may be expressed as follows,

$$Q_{m+1}(z) = \left[\sum_{k=1}^{r} Z_k(z) \right] Q_m(z), \tag{6.11}$$

where $Q_m(z)$ and $Q_{m+1}(z)$ are the z-transforms of the signals at the mth and $(m+1)$th iterations of the successive projection algorithm, whereas r stands for the number of singular values σ_k of Q_m which are retained at each iteration of the successive projection. Obviously the above expression implies a zero-phase filtering operation. Consequently this algorithm may be seen as an iterative adaptive zero-phase filtering procedure, where each iteration corresponds to a zero-phase filter with filters being updated continuously.

Iterative Signal Enhancement algorithm

The Iterative Signal Enhancement (ISE) algorithm [29] is based on the successive projections algorithm and consists of two nested loops. The inner loop is an iterative procedure which computes an enhanced signal z from a noisy input signal y. Since the output of this inner loop still contains noise (with lower energy than in the input signal), the outer loop consists of repeating the inner loop a number of times. The enhanced signal z from the inner loop is used as the input signal y for the next execution of the inner loop.
The iterative procedure (inner loop) proceeds as follows : the noisy input signal y is filtered using the zero-phase eigenfilter which is associated to the largest singular value. The output d is associated to the most energetic spectral band of the input signal. The residual signal r is then computed by subtracting d from the input signal y and the procedure is started over again using the residual signal as input for the next iteration. The enhanced signal z is obtained by summing the intermediate decompositions d over all iterations. The inner loop is ended when the residual signal r contains mainly noise. This enhanced signal z can then be used again as input signal (outer loop). To indicate the difference between inner and outer loop we adopt the following notation: quantity c in the jth iteration of the inner loop, in the kth iteration of the outer loop is denoted by $c^{[k,j]}$. The adaptive postfilter routine applied to the ith frame of the synthesized HF result q can thus be summarized as follows:

ISE Algorithm

Input: $q(1+(i-1)N_{HF} : iN_{HF})$, user-supplied integers p_1 and p_2
Output: Enhanced signal z

Step 1: Initialize $y^{[0,0]}$ with $q(1+(i-1)N_{HF} : iN_{HF})$
Step 2: for $k = 0 \dots p_1 - 1$, *(outer loop)*
  Step 2.1: for $j = 0 \dots p_2 - 1$, *(inner loop)*
    Step 2.1.1: compute Hankel matrix $Y^{[k,j]}$ from the input signal $y^{[k,j]}$
    Step 2.1.2: compute first singular value σ_1 and corresponding left singular vector u_1 of $Y^{[k,j]}$
    Step 2.1.3: compute rank-1 signal approximation $d^{[k,j]}$ of $y^{[k,j]}$
    Step 2.1.4: compute residual $r^{[k,j]}$: $r^{[k,j]} = y^{[k,j]} - d^{[k,j]}$
    Step 2.1.5: $y^{[k,j+1]} = r^{[k,j]}$
  end
  Step 2.2: enhanced signal after p_2 iterations of inner loop: $z^{[k]} = \sum_{l=0}^{p_2-1} d^{[k,l]}$
  Step 2.3: $y^{[k+1,0]} = z^{[k]}$ *(new input signal)*
end
Step 3: Enhanced signal : $z = y^{[p_1,0]}$

Note that the proposed algorithm can be implemented very efficiently because it only requires the computation of the largest singular value and of the corresponding singular vector, for which efficient numerical algorithms exist. $Y^{[k,j]}$ is constructed from $y^{[k,j]}$ as was done in (6.7) for Q and q ($p = 10$ in this case). The user-supplied integers p_1 and p_2 are set to 3 and 12 respectively. It has to be mentioned however, that the result is not overly sensitive to this choice.

6.2 Experimentation-Testing

In this section we compare our scheme to the CELP standard algorithm. We apply both schemes to a speech signal sampled at 8kHz, using 8 bits per sample. The signal contains 15000 samples (approximately 2 seconds of speech) and is a phonetically balanced French sentence, uttered by a male speaker. The sentence is an enumeration of geographical places:

Paris, Bordeaux, Le Mans, Saint-Leu, Léon, Loudun

which has the following phonetic transcription (according to the International Phonetic Association's rules [58]):

paʁi, bɔʁdo, ləmɑ̃, sɛ̃ lø, leɔ̃, ludœ̃

For the CELP algorithm, we use a Fortran implementation of the Federal Standard 1016 4800 bps CELP vocoder [19]. We will now discuss the choice of the different parameters in our scheme. The cut-off frequency $f_{c.o.}$ plays the most important role in this scheme, since it determines the amount of work the LF and the HF branches will have to perform. A high $f_{c.o.}$ e.g., will be very demanding for the LF branch of our scheme, implying the use of a high value for n (in order to obtain a good quality), the number of complex damped exponentials to be used in the STLS analysis algorithm. Moreover, the higher the frequency $f_{c.o.}$, the lower the downsampling factor D should be if we want no information loss (see (6.1)) due to downsampling and upsampling. As mentioned in 6.1.1, this means that the workload of the STLS analysis algorithm increases. Using the following bit allocation scheme: 12 bits per frequency, 7 bits per damping, 8 bits per amplitude and 5 bits per phase, we derive the following bit rate for the LF branch

$$d_{STLS} \equiv \frac{2n(parameters)\ \ 8(bits/parameter)}{D\ \ N_{LFD} - N_{ov}(samples)}.$$

As explained at the end of section 6.1.1, we use overlapping frames. The latter means that before downsampling we have frames of $N_{LF} \equiv DN_{LFD}$ samples.

At the receiver side, after upsampling, only the first $N_{LF} - N_{ov}$ samples of the reconstructed frame are used. The last N_{ov} samples of the reconstruction will be used to smooth the transitions with the next frame. This explains the denominator of d_{STLS}. Since the total bit rate d is the sum of the bit rates of the LF (d_{STLS}) and HF (d_{VQ}) branch, it is obvious that for a constant total bit rate d the choice of $f_{c.o.}$ not only influences d_{STLS} (through n), but also d_{VQ}. The bit rate d_{VQ} can be calculated as follows:

$$d_{VQ} \equiv \frac{w \ \ bits}{N_{HF} \ \ samples},$$

with 2^w the size of the CB and N_{HF} the dimension of the codevectors. In our application we took the following parameters: $f_{c.o.} = 400$ Hz, $n = 4$, $N_{LFD} = 39$, $D = 8$ (so $N_{LF} = 312$), $N_{ov} = 10$, $w = 11$, and $N_{HF} = 28$. Filling in these values we obtain a bit rate of $d_{STLS} + d_{VQ} = 0.2119 + 0.3929 = 0.6048$ bits/sample or at a sampling frequency of 8 kHz: 4838 bit/sec. We thus have a compression ratio similar to CELP. Since $f_{c.o.} = 400$ Hz, we can downsample the frames with a factor of 8 ($D = 8$) without loosing any information.
To assess the quality of the compressed speech, we use the following segmental SNR definition:

$$SNR_{seg} \equiv 10log_{10} \frac{1}{F} \sum_{j=1}^{F} \frac{\sum_{i=1}^{p} (in_j(i))^2}{\sum_{i=1}^{p} (in_j(i) - out_j(i))^2}, \tag{6.12}$$

where F represents the number of frames, p is the frame length used in the computation of SNR_{seg}, $in_j = in(1+(j-1)p{:}jp)$, $out_j = out(1+(j-1)p{:}jp)$. Here p is chosen equal to 302. Since the segmental SNR is a mathematical measure for the accuracy of fit between the signals in and out in (6.12), phase issues play an important role. Since pre- and postprocessing can severely alter the (segmental) SNR, we compare the preprocessed signal at the sender side with the non-postprocessed signal at the receiver side. For the CELP scheme, we compare the highpass filtered incoming signal (i.e. the preprocessed signal) with the non-postfiltered output of the scheme (i.e. the signal at the receiver side, before the adaptive postfilter routine to reduce the perceptual coder noise is applied). This yields the following segmental SNR: $SNR_{seg} = 12.16$dB.

In the case of our scheme, we consider the sum of the LF part of the incoming signal and the preemphasized HF part of the incoming signal on the one hand and the output of the scheme on the other hand. Based on these two signals, the new method offers a segmental SNR of 12.33dB. After this quantitative measure, a qualitative one follows to show how well both schemes can fit the frequencies in the range 0 to 4000 Hz. We do so by comparing for both schemes the Fast Fourier Transform (FFT) magnitude spectrum of a frame at the input of the scheme with the FFT magnitude spectrum of the same frame at the output. The frame we propose is $s(3323 : 3624)$. Figure 6.2 shows the FFT magnitude spectrum of the preprocessed CELP input (full line) together with the FFT magnitude spectrum of the non-postprocessed CELP output (dashed line). Figures 6.3 and 6.4 refer respectively to the signals in the LF branch and the signals in the HF branch of our scheme. Figure 6.3 illustrates the FFT magnitude spectrum of the LF part of the incoming signal (full line) together with the FFT magnitude spectrum of the reconstructed LF part at the receiver side (dashed line). Figure 6.4 is the HF counterpart of figure 6.3. It shows the FFT magnitude spectrum of the preemphasized HF part of the incoming signal (full line) together with the FFT magnitude spectrum of the reconstructed HF part (before postprocessing, since postprocessing will alter the frequency content) at the receiver side (dashed line).

Comparison of figures 6.2 to 6.4 learns that for the LF part as well as for the HF part of the spectrum our scheme gives results that are of similar quality. Summarizing we can conclude that both the quantitative and the qualitative measures we have shown, indicate a result that is in quality similar to the CELP result. Note however that the new scheme can be used over a wide range of signals (speech as well as music) and compression ratios. Furthermore, it allows progressive speech compression with a variable degree of analysis/synthesis at the emitter/receiver depending on the availability of channel capacity and/or the requirements of the specific application. We note that in some applications (e.g. music) it might be necessary to use more exponentially damped sinusoids for modelling frames of the LF part of the signal. In those cases, the high complexity of the STLS analysis algorithm, might pose a problem. The kernel problem of the STLS analysis algorithm is an equality constrained LS problem, so that each iteration of the STLS algorithm (remember that algorithm 4.4.1, as described in section 4.4, has a Sequential Quadratic Programming-like structure) requires approximately $4mn^2$ flops, with $m \times n$ the dimensions of the matrix S. However, it is possible to speed up the kernel problem, by exploiting the Hankel structure of S in the same way as was done for the deconvolution problem in chapter 5. This leads to algorithms having a complexity of $O(mn)$ flops per iteration.
On the other hand it is also possible to speed up our scheme by simplifying the STLS analysis algorithm. This can easily be done, by terminating after the initialization of algorithm 4.4.1 in step 1 of the STLS analysis algorithm and transmitting the quantized frequencies, dampings, amplitudes and phases obtained by HTLS. As already indicated in section 4.5, the advantage is that it yields initial estimates that satisfy the equality structure constraints of (6.5). This initial estimate is however not optimal in the sense that the objective function in (6.5) can further be decreased. The latter means that this shortened scheme is faster but leads to a lower quality.

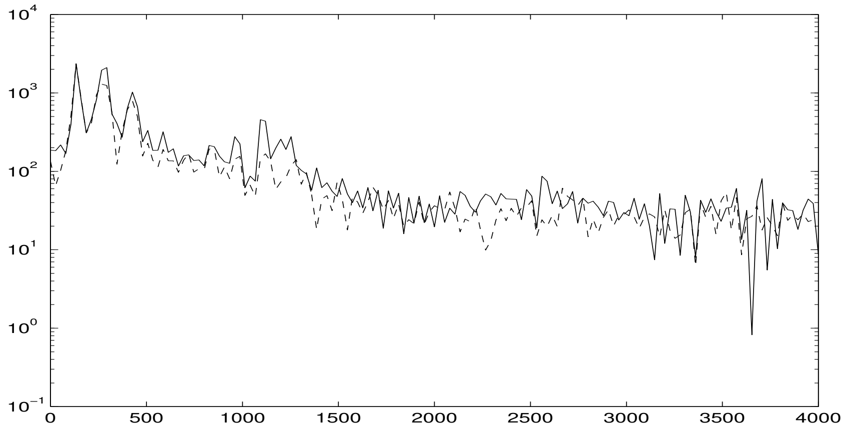


Figure 6.2: *This figure shows the FFT magnitude spectrum of the preprocessed CELP input (full line) together with the FFT magnitude spectrum of the non-postprocessed CELP output (dashed line). The depicted frame corresponds to* $s(3323 : 3624)$ *(s being the speech signal described in section 6.2).*

6.3 Conclusions

In this chapter we have described a new low bit rate subband speech compression scheme. The LF part of the scheme is related to sinusoidal coders. The

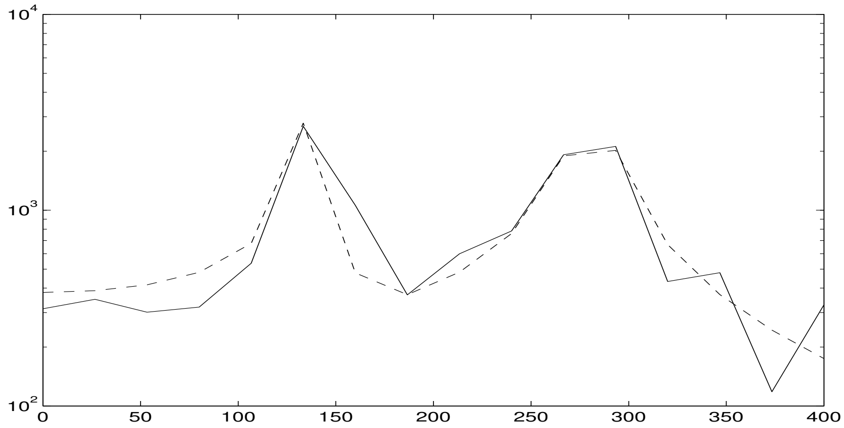


Figure 6.3: *This figure illustrates the FFT magnitude spectrum of the LF part of the incoming signal (full line) together with the FFT magnitude spectrum of the reconstructed LF part at the receiver side (dashed line) of our newly proposed scheme. The depicted frame corresponds to* $s(3323 : 3624)$ *(s being the speech signal described in section 6.2).*

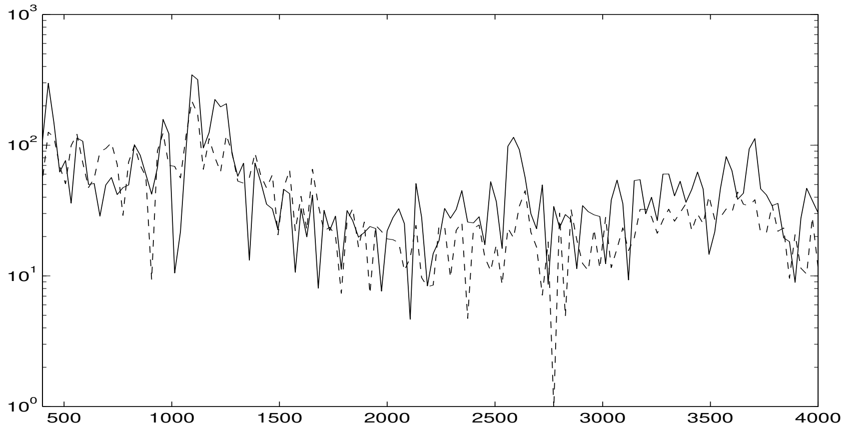


Figure 6.4: *This figure shows the FFT magnitude spectrum of the preemphasized HF part of the incoming signal (full line) together with the FFT magnitude spectrum of the reconstructed HF part (before postprocessing) at the receiver side (dashed line) of our newly proposed scheme. The depicted frame corresponds to* $s(3323 : 3624)$ *(s being the speech signal described in section 6.2).*

difference however is that we use sums of exponentially damped sinusoids to model the frames of the LF part of the signal, instead of sums of undamped sinusoids. The frames of the HF part of the signal are quantized using waveform VQ.

We proposed a parameter choice for our scheme, leading to a bit rate of 4838 bits/sec. The latter bit rate is similar to that of the Federal Standard 1016 4800 bps CELP vocoder, allowing us to compare both schemes. By means of a speech example, we show in a quantitative (using the segmental SNR) as well as in a qualitative way (using the FFT magnitude spectrum) that our new scheme, although more flexible, has properties comparable to the CELP scheme. The flexibility resides in the fact that the new LF scheme allows to do progressive speech compression.

Chapter 7

Misfit versus latency

Mathematical models are the heart of many applications in areas such as control system design, prediction, simulation, fault detection,... In this chapter a new framework [69] for modeling dynamic linear time-invariant (LTI) systems is developed. It combines the ideas of the prediction error modeling (PEM) approach [63, 86] and the behavioral modeling approach [100, 79, 80, 71]. Both approaches –the PEM approach and the behavioral modeling approach– aim to fit a model of the proposed model set to the observed data. The PEM method minimizes a cost function involving the so-called prediction errors, which are computed using the model and the observed data. These prediction errors can be thought of as "unobserved" signals, with certain pre-assumed properties, such as for instance a typical stationary zero mean white Gaussian noise *assumption. Such signals will be labelled* latent. *Summarizing we can say that in the PEM approach the observed data can be explained by a model belonging to the proposed model set thanks to the introduction of latent variables.*
The behavioral modeling framework approaches the modeling problem from a set theoretic point of view: the set of all conceivable input/output time series is called the behavior of the system. A model is then found by selecting –according to some user-defined distance measure– the time series of the behavior closest to the observed data. The latter distance is termed "misfit". Summarizing, we can say that in the behavioral modeling approach the observed data is explained by a model of the proposed model set, through the modification of the observed data in such a way that they become compatible with a model of the model set. The latter modification will be referred to as "misfit". In this chapter, we develop the idea of obtaining mathematical models that "explain" observed data, by modifying the data ("misfit") or introducing new "unobserved" additional data ("latency") or by applying a combination of misfit and latency [69]. The chapter is structured as follows. The following section introduces the idea of misfit versus latency for static LTI models. Section 7.2 extends this idea to a

general set of dynamic LTI models. We motivate the use of this framework in system identification and rewrite the associated model selection procedure in a matrix framework. Using the technique of Lagrange multipliers, this procedure can be recast as a set of Riemannian SVD equations, similar to the Riemannian SVD equations obtained in the RiSVD framework for STLS problems. Furthermore we discuss the importance of using the correct constraints that have to be imposed on the model parameters. Section 7.3 derives some properties and interpretations of the new framework. Section 7.4 shows how the new framework can be recast into an optimization framework with only the model parameters as variables. Not only does this allow us to perform some experiments (see section 7.5), it also provides additional insight into the model selection procedure for the new framework.

7.1 Static case

In this section we address static LTI models of the form:

$$Ax = b, \tag{7.1}$$

with $A \in \mathbb{R}^{m \times n}, m \geq n, b \in \mathbb{R}^{m \times 1}$ the data and $x \in \mathbb{R}^{n \times 1}$ the parameter vector. In general, equation (7.1) is not satisfied for measured data as it implies the rank deficiency of the matrix $[A \quad b] \in \mathbb{R}^{m \times (n+1)}$. In those cases, the rank of $[A \quad b]$ equals $n+1$ instead of n and therefore we will have to determine *approximate* models. We will now describe three different approximate models and associated model selection procedures. The first method consists of introducing a so called "latent variable" $e \in \mathbb{R}^{m \times 1}$, such that $Ax = b + e$, and the appropriate model is selected by the following optimization problem:

$$min_{e,x} \|e\|_2^2 \text{ such that } Ax = b + e, \tag{7.2}$$

where $\|.\|_2$ indicates the vector $2-$norm. The latter approach is called the "latency" approach and corresponds to the well-known LS solution of the overdetermined system of equations $Ax \approx b$. As can be seen from (7.2), the inconsistency of the equations is explained by the introduction of a new variable e. The minimization of $\|e\|_2$ can heuristically be explained by the desire to find the "minimum energy" e so that $Ax = b + e$.
A second method does not introduce new variables, but it assumes a misfit $[\Delta A \quad \Delta b]$ between the observed data $[A \quad b]$ and the true data, such that $(A + \Delta A)x = b + \Delta b$.

The model selection procedure can be formulated as follows:

$$\begin{aligned} &\min_{\Delta A, \Delta b, x} \|[\Delta A \;\; \Delta b]\|_F^2 \\ &\text{such that } \;(A + \Delta A)x = b + \Delta b, \end{aligned} \tag{7.3}$$

where $\|.\|_F$ stands for the Frobenius norm. Formulation (7.3) obviously corresponds to the classical EIV approach, which solves the overdetermined system of equations $Ax \approx b$ in a Total Least Squares (TLS) sense. It is well known [46] that the solution to (7.3) follows from the SVD of $[A \;\; b]$. Further on, this type of approach will be referred to as the misfit approach.
The third way is to combine both the latency and the misfit approach, so that we obtain the following general model:

$$(A + \Delta A)x = b + \Delta b + e.$$

To select the model x in this case, we propose to solve the following optimization problem:

$$\begin{aligned} &\min_{\Delta A, \Delta b, e, x} \;\; \alpha \|[\Delta A \;\; \Delta b]\|_F^2 + \beta \|e\|_2^2 \\ &\text{such that} \;\; (A + \Delta A)x = b + \Delta b + e, \end{aligned} \tag{7.4}$$

with α and β user-specified positive scalar weights that are inversely proportional to the importance assigned to the misfit respectively the latency contributions. Use of the method of Lagrange multipliers shows that the solution of the latter optimization problem is the solution of a weighted SVD problem: $x = -V(1:n, n+1)/V(n+1, n+1)$ with $[A \;\; b]W^{-1/2} = U\Sigma Z^T$, the SVD of $[A \;\; b]W^{-1/2}$, $V = W^{-1/2}Z$, $W = \begin{bmatrix} \frac{1}{\alpha} I_q & 0 \\ 0 & \frac{1}{\alpha} + \frac{1}{\beta} \end{bmatrix}$ under the assumption that $V(n+1, n+1) \neq 0$. We see that the approach (7.4) is a generalization of the

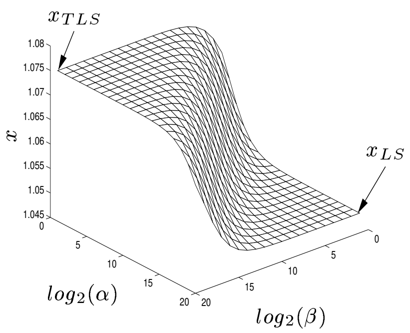


Figure 7.1: *This figure shows the solution x of (7.4) as a function of $log_2(\alpha)$ and $log_2(\beta)$, applied to example 7.1.1. The higher α, the less "misfit" we allow; the higher β, the less latency we allow.*

approaches described in (7.2) and (7.3). By taking e.g. $\alpha = \infty$ and $\beta = 1$ in (7.4), we obtain (7.2). In Figure 7.1, we show the solution x of problem (7.4) as a function of the weights α and β for the following example.

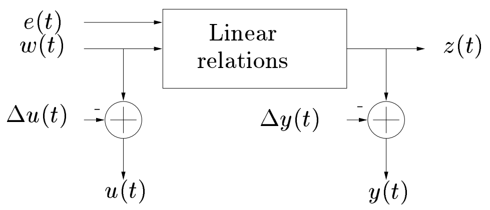


Figure 7.2: *This figure illustrates the new model set of LTI dynamical systems. The observed data are $u(t)$ and $y(t)$. They are assumed to be corrupted versions of exact data, such that $w(t) = u(t) + \Delta u(t)$ and $z(t) = y(t) + \Delta y(t)$. There are also "unobserved" inputs $e(t)$, that "help explain" the data $z(t)$.*

Example 7.1.1 $A = [3 \;\; 4 \;\; 4.3 \;\; 3.4 \;\; 2 \;\; 1.09 \;\; -3 \;\; -2.54 \;\; -2.09 \;\; 0.89]^T$ and $b = [3.57 \;\; 4.80 \;\; 4.33 \;\; 3.93 \;\; 2.50 \;\; 2.05 \;\; -2.25 \;\; -1.99 \;\; -1.20 \;\; 1.52]^T$. □

It is clear that by changing the values of the weights, we can go in a continuous way from the LS solution ($x_{LS} = 1.04909$, $\alpha = \infty$, $\beta = 1$) to the TLS solution ($x_{TLS} = 1.0757$, $\alpha = 1$, $\beta = \infty$).
Finally, from a statistical point of view, it is obvious that these three approaches correspond to the ML solution under different noise assumptions.

7.2 Dynamic case

In a way similar to the static case, we can create a new framework for dynamic LTI systems by combining the dynamic "latency" approach (i.e. the prediction error approach containing models such as AR, ARX, ARMA, ... in which "white noise residuals" are introduced to explain the observed input/output data) and the dynamic "misfit" approach (i.e. the behavioral approach containing models such as dynamic TLS or errors-in-variables,...). The motivation for this framework is shown in Figure 7.2. There is no reason why the presence of a latency input $e(t)$ would exclude a noise misfit of the measurements at the input ($\Delta u(t)$) and output ($\Delta y(t)$) of the system or vice versa. Since the statistical properties of latency inputs and noise misfit can be very different, it makes sense to make a distinction in the problem formulation and the consecutive model selection procedure (using the weights α, β and γ in (7.6) we can vary the importance of the latency terms vs. the misfit terms). Models in this framework are of the form

$$A(q)z(t) = B(q)w(t) + C(q)e(t), \tag{7.5}$$

with $z(t) = y(t) + \Delta y(t)$ and $w(t) = u(t) + \Delta u(t)$, where $\Delta y(t)$ and $\Delta u(t)$ represent the so-called misfit of the observed output $y(t)$ and the observed input

$u(t)$. Further, $e(t)$ is the so-called latent variable, which can be interpreted as an "unobserved" input. $A(q), B(q)$ and $C(q)$ are polynomials of appropriate degree in the delay operator q^{-1}, with $A(q) = a_0 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a}$, $B(q) = b_1 + b_2 q^{-1} + \ldots + b_{n_b} q^{-n_b+1}$ and $C(q) = c_0 + c_1 q^{-1} + \ldots + c_{n_c} q^{-n_c}$, where we assume that n_a, n_b and n_c are user-defined. To select the best model from this new model set (7.5), we propose to minimize the following cost function:

$$J = \alpha J_{\text{output}} + \beta J_{\text{input}} + \gamma J_{\text{latency}} \ , \tag{7.6}$$

with $J_{\text{output}} = \sum_{t=t_1}^{N} (\Delta y(t))^2$, $J_{\text{input}} = \sum_{t=t_2}^{N} (\Delta u(t))^2$, $J_{\text{latency}} = \sum_{t=t_3}^{N} e(t)^2$, N being the number of data points, and α, β, γ real positive scalars, specified by the user; let $M = max(n_a, n_b - 1, n_c) + 1$ and $m = N - M + 1$, then $t_1 = M - n_a$, $t_2 = M - n_b + 1$ and $t_3 = M - n_c$.
It is clear that, e.g., a classical ARMAX model fits into this framework by selecting $\alpha \to \infty, \beta \to \infty$ and $\gamma = 1$. There are 27 different cases that can be obtained by the particular choices of α, β and γ (they can be finite, infinite, or their corresponding term can be omitted in the model (7.5)). Table 7.1 shows the different possibilities. We see that the framework covers many existing linear dynamic modeling techniques, but in addition it proposes several new models (e.g. case 23 in table 7.1).

We now discuss two cases, in order to clarify the table:

- case 17 ($\alpha = 1$, $\beta = \infty$ and $\gamma = 1$): to interpret the values of the weightings α, β and γ, we have to look at the model equations (7.5) and the cost function of the model selection procedure (7.6). Setting $\alpha = 1$ we see from (7.6) that we allow a misfit term $\Delta y(t)$ on the output. The fact that $\beta = \infty$ implies that J_{input} in (7.6) has to be zero and thus there can be no misfit on the input. Finally, $\gamma = 1$ means that the cost function (7.6) contains a latency variable. As a result the model equation (7.5) becomes

  $$A(q)z(t) = B(q)u(t) + C(q)e(t),$$

  since there is misfit on the output (this explains why $z(t)$ is used instead of the measured $y(t)$, see figure 7.2), there is no misfit on the input (in that case $u(t) = w(t)$, so we use the measured quantity $u(t)$) and a latency variable is included.

- case 11 ($\alpha = 1$, $\beta = *$ and $\gamma = 1$): again using the cost function (7.6), we see that there is misfit on the output ($\alpha = 1$), there is no input ($\beta = *$; this "$*$" indicates that the input term in (7.6) and (7.5) is simply omitted) and a latency term is included. Obviously this leads to the following model equation:

  $$A(q)z(t) = C(q)e(t).$$

Now it should also be clear why this model is called ARMA with noisy output: the only difference with the classical ARMA model

$$A(q)y(t) = C(q)e(t)$$

is the fact that we take a misfit on the output $(y(t) \to z(t))$ into account.

We can write equations (7.5) for $t = M, \ldots, N$ in the following matrix format: $Za - Wb - Ec = 0$, with $a = [a_{n_a}, \ldots, a_0]^T, b = [b_{n_b}, \ldots, b_1]^T, c = [c_{n_c}, \ldots, c_0]^T$, $Z \in \mathbb{R}^{m \times (n_a+1)}$ a Hankel matrix constructed from $\tilde{z} \equiv z(t_1 : N)$ (Y is defined in a similar way w.r.t. $\tilde{y} \equiv y(t_1 : N)$), $W \in \mathbb{R}^{m \times n_b}$ a Hankel matrix constructed from $\tilde{w} \equiv w(t_2 : N-1)$ (U is defined in a similar way w.r.t. $\tilde{u} \equiv u(t_2 : N-1)$) and $E \in \mathbb{R}^{m \times (n_c+1)}$ a Hankel matrix constructed from $\tilde{e} \equiv e(t_3 : N)$ (all these Hankel matrices are constructed by putting the corresponding vector in its first column and last row, starting in the upper left corner). The model selection procedure can thus be rephrased as:

$$\begin{aligned} &\min_{\Delta y, \Delta u, e, a, b, c} \; \alpha J_{output} + \beta J_{input} + \gamma J_{latency} \\ &\text{such that } \; Za - Wb - Ec = 0. \end{aligned} \tag{7.7}$$

To avoid the trivial solution $a = 0$, $b = 0$ and $c = 0$, we need to impose extra constraints. We propose a set of linear constraints, which lead to ML solutions for the special cases like AR, ARMA, ... The constraints imposed are: $a_0 = 1$ and $c_0 = 1$. At the end of this section we will revisit the choice of the constraints and show that e.g. quadratic constraints on c lead to results having different undesired properties. By applying the technique of Lagrange multipliers to the optimization problem in (7.7) we will now derive a set of equations that characterizes the stationary points of problem (7.7). The Lagrangian of the latter problem is

$$\begin{aligned} L(a, b, c, \tilde{w}, \tilde{z}, \tilde{e}, l, \kappa, \lambda) \;\; &= \;\; \alpha \sum_{t=t_1}^{N} (\Delta y(t))^2 + \beta \sum_{t=t_2}^{N} (\Delta u(t))^2 + \gamma \sum_{t=t_3}^{N} (e(t))^2 + \\ & \quad\;\; l^T (Za - Wb - Ec) + \kappa(a_0 - 1) + \lambda(c_0 - 1). \end{aligned}$$

Taking the derivatives of L with respect to a, b, c, $\tilde{w}$, $\tilde{z}$, $\tilde{e}$, l, κ, λ and setting

No	(α, β, γ)	Case	Data misfit	Model	M/L
1	(*,*,*)	No meaning			
2	(*,*,1)	No meaning			
3	(*,*,∞)	See Case 1			
4	(*,1,*)	Noisy input realization	$U \to W$	$Wb = 0$	M
5	(*,1,1)	ARMA with noisy inputs	$U \to W$	$Wb + Ec = 0$	M+L
6	(*,1,∞)	See Case 4			
7	(*,∞,*)	Input=impulse response		$Ub = 0$	E
8	(*,∞,1)	ARMA for inputs		$Ub + Ec = 0$	L
9	(*,∞,∞)	See Case 7			
10	(1,*,*)	Noisy output realization	$Y \to Z$	$Za = 0$	M
11	(1,*,1)	ARMA with noisy outputs	$Y \to Z$	$Za + Ec = 0$	M+L
12	(1,*, ∞)	See Case 10			
13	(1,1,*)	Dynamic Errors-in-variables	$Y \to Z$ $U \to W$	$Za + Wb = 0$	M
14	(1,1,1)	ARMAX with noisy in/outputs	$Y \to Z$ $U \to W$	$Za + Wb + Ec = 0$	M +L
15	(1,1,∞)	See case 13			
16	(1,∞,*)	Output error	$Y \to Z$	$Za + Ub = 0$	M
17	(1,∞,1)	ARMAX with noisy output	$Y \to Z$	$Za + Ub + Ec = 0$	M+L
18	(1,∞,∞)	See case 16			
19	(∞,*,*)	Output=impulse response		$Ya = 0$	E
20	(∞,*,1)	ARMA		$Ya + Ec = 0$	L
21	(∞,*,∞)	See Case 19			
22	(∞,1,*)	Input error	$U \to W$	$Ya + Wb = 0$	M
23	(∞,1,1)	ARMAX with noisy input	$U \to W$	$Ya + Wb + Ec = 0$	M+L
24	(∞,1,∞)	See Case 22			
25	(∞,∞,*)	Linear System		$Ya + Ub = 0$	E
26	(∞,∞,1)	ARMAX		$Ya + Ub + Ec = 0$	L
27	(∞,∞,∞)	See Case 25			

Table 7.1: *This table shows the different LTI dynamic systems of our general framework. Depending on the triplet (α, β, γ) we obtain 1 of the 27 model sets (note that when e.g. $\beta = *$, this means that there is no input term present in (7.6), nor in (7.5)). Some of them correspond to existing model sets, others are new. The last column indicates whether we have a misfit-like model (M), a latency-like model (L) or a model that applies to exact data (E) only.*

these equal to 0, we obtain the following equations:

$$Z^T l + \kappa[0 \ \dots \ 0 \ 1]^T = 0 \tag{7.8}$$

$$-W^T l = 0 \tag{7.9}$$

$$-E^T l + \lambda[0 \ \dots \ 0 \ 1]^T = 0 \tag{7.10}$$

$$\beta(\tilde{w} - \tilde{u}) - \mathbf{P_2}(b)^T l = 0 \tag{7.11}$$

$$\alpha(\tilde{z} - \tilde{y}) + \mathbf{P_1}(a)^T l = 0 \tag{7.12}$$

$$\gamma\tilde{e} - \mathbf{P_3}(c)^T l = 0 \tag{7.13}$$

$$Za - Wb - Ec = 0 \tag{7.14}$$

$$a_0 = 1 \tag{7.15}$$

$$c_0 = 1, \tag{7.16}$$

where $\mathbf{P_1}$, $\mathbf{P_2}$ and $\mathbf{P_3}$ are matrix functions defined as follows:

$$\mathbf{P_1} : \mathbb{R}^{(n_a+1)\times 1} \rightarrow \mathbb{R}^{m\times(N-M+n_a+1)} : a \rightarrow \mathbf{P_1}(a), \text{ defined by } \mathbf{P_1}(a)\tilde{z} = Za$$
$$\mathbf{P_2} : \mathbb{R}^{n_b\times 1} \rightarrow \mathbb{R}^{m\times(N-M+n_b)} : b \rightarrow \mathbf{P_2}(b), \text{ defined by } \mathbf{P_2}(b)\tilde{w} = Wb$$
$$\mathbf{P_3} : \mathbb{R}^{(n_c+1)\times 1} \rightarrow \mathbb{R}^{m\times(N-M+n_c+1)} : c \rightarrow \mathbf{P_3}(c), \text{ defined by } \mathbf{P_3}(c)\tilde{e} = Ec.$$

For later use we also define the following matrix functions

$$\mathbf{P_4} : \mathbb{R}^{m\times 1} \rightarrow \mathbb{R}^{(n_a+1)\times(N-M+n_a+1)} : l \rightarrow \mathbf{P_4}(l), \text{ defined by } \mathbf{P_4}(l)\tilde{z} = Z^T l$$
$$\mathbf{P_5} : \mathbb{R}^{m\times 1} \rightarrow \mathbb{R}^{n_b\times(N-M+n_b)} : l \rightarrow \mathbf{P_5}(l), \text{ defined by } \mathbf{P_5}(l)\tilde{w} = W^T l$$
$$\mathbf{P_6} : \mathbb{R}^{m\times 1} \rightarrow \mathbb{R}^{(n_c+1)\times(N-M+n_c+1)} : l \rightarrow \mathbf{P_6}(l), \text{ defined by } \mathbf{P_6}(l)\tilde{e} = E^T l.$$

Note also that in the equations (7.8) to (7.16) the factor 2 that should have appeared in (7.11), (7.12) and (7.13), has been absorbed in the Lagrange multipliers l, λ, and κ. In the remainder of this section we will also make use of the following lemma, called the exchange lemma.

Lemma 7.2.1 (Exchange lemma) *For the previously defined matrix functions* $\mathbf{P_1}$, $\mathbf{P_2}$, $\mathbf{P_3}$, $\mathbf{P_4}$, $\mathbf{P_5}$ *and* $\mathbf{P_6}$, *we have that*

$$\begin{aligned} \mathbf{P_1}(a)^T l &= \mathbf{P_4}(l)^T a \\ \mathbf{P_2}(b)^T l &= \mathbf{P_5}(l)^T b \\ \mathbf{P_3}(c)^T l &= \mathbf{P_6}(l)^T c, \end{aligned}$$

for all $l \in \mathbb{R}^{m\times 1}$, $a \in \mathbb{R}^{(n_a+1)\times 1}$, $b \in \mathbb{R}^{n_b\times 1}$ *and* $c \in \mathbb{R}^{(n_c+1)\times 1}$.

Proof: Since the proof is similar for the 3 equalities, only the proof of the first equality is presented here. For an arbitrary $\tilde{z} \in \mathbb{R}^{(N-M+n_a+1)\times 1}$ we find that

$$\begin{aligned} \tilde{z}^T \mathbf{P_4}(l)^T a &= l^T Z a \\ &= l^T \mathbf{P_1}(a)\tilde{z} \\ &= \tilde{z}^T \mathbf{P_1}(a)^T l, \end{aligned}$$

where the first equality follows from the definition of $\mathbf{P_4}$ and the second equality is obtained by the definition of $\mathbf{P_1}$. The above is valid for arbitrary $\tilde{z}$ and therefore

$$\mathbf{P_4}(l)^T a = \mathbf{P_1}(a)^T l.$$

□

Multiplying (7.8) with a^T to the left, (7.9) with b^T to the left and (7.10) with c^T to the left and adding the results, we find by using (7.14) that

$$-\kappa a_0 = \lambda c_0$$

or by taking (7.15) and (7.16) into account:

$$\lambda = -\kappa.$$

Therefore, from now on κ will be replaced with $-\lambda$. Using (7.9), the definition of $\mathbf{P_5}$, (7.11) and the exchange lemma, we find:

$$\begin{aligned} U^T l &= -(W-U)^T l \\ &= -\mathbf{P_5}(l)(\tilde{w}-\tilde{u}) \\ &= -\frac{\mathbf{P_5}(l)\mathbf{P_2}(b)^T}{\beta} l \\ &= -\frac{\mathbf{B_5}(l)}{\beta} b, \end{aligned} \tag{7.17}$$

where $\mathbf{B_5}(l) \equiv \mathbf{P_5}(l)\mathbf{P_5}(l)^T$. Combining (7.8), the definition of $\mathbf{P_4}$, (7.12) and the exchange lemma, we obtain:

$$\begin{aligned} Y^T l &= \lambda[0\ldots 0\ 1]^T - \mathbf{P_4}(l)(\tilde{z}-\tilde{y}) \\ &= \lambda[0\ldots 0\ 1]^T + \frac{\mathbf{P_4}(l)\mathbf{P_1}(a)^T}{\alpha} l \\ &= \lambda[0\ldots 0\ 1]^T + \frac{\mathbf{P_4}(l)\mathbf{P_4}(l)^T}{\alpha} a \\ &= \lambda[0\ldots 0\ 1]^T + \frac{\mathbf{B_4}(l)}{\alpha} a, \end{aligned} \tag{7.18}$$

where $\mathbf{B_4}(l) \equiv \mathbf{P_4}(l)\mathbf{P_4}(l)^T$.
Using (7.10), the definition of $\mathbf{P_6}$, (7.13) and the exchange lemma, we find:

$$\begin{aligned} \lambda[0\ldots 0\ 1]^T &= \mathbf{P_6}(l)\tilde{e} \\ &= \frac{\mathbf{P_6}(l)\mathbf{P_3}(c)^T}{\gamma} l \\ &= \frac{\mathbf{P_6}(l)\mathbf{P_6}(l)^T}{\gamma} c \\ &= \frac{\mathbf{B_6}(l)}{\gamma} c, \end{aligned} \tag{7.19}$$

where $\mathbf{B_6}(l) \equiv \mathbf{P_6}(l)\mathbf{P_6}(l)^T$.
Starting with (7.14), using the definitions of $\mathbf{P_1}$, $\mathbf{P_2}$ and $\mathbf{P_3}$, formulas (7.11), (7.12) and (7.13) we find:

$$\begin{aligned} Ya - Ub &= \mathbf{P_3}(c)\tilde{e} - \mathbf{P_1}(a)(\tilde{z}-\tilde{y}) + \mathbf{P_2}(b)(\tilde{w}-\tilde{u}) \\ &= \left(\frac{\mathbf{B_3}(c)}{\gamma} + \frac{\mathbf{B_1}(a)}{\alpha} + \frac{\mathbf{B_2}(b)}{\beta}\right) l, \end{aligned} \tag{7.20}$$

where $\mathbf{B_1}(l) \equiv \mathbf{P_1}(l)\mathbf{P_1}(l)^T$, $\mathbf{B_2}(l) \equiv \mathbf{P_2}(l)\mathbf{P_2}(l)^T$ and $\mathbf{B_3}(l) \equiv \mathbf{P_3}(l)\mathbf{P_3}(l)^T$. According to (7.17), (7.18), (7.19) and (7.20) the stationary points of problem (7.7) are given by the following set of Riemannian SVD equations:

$$[Y \quad -U] \begin{bmatrix} a \\ b \end{bmatrix} = (\frac{\mathbf{B_1}(a)}{\alpha} + \frac{\mathbf{B_2}(b)}{\beta} + \frac{\mathbf{B_3}(c)}{\gamma})l \tag{7.21}$$

$$\begin{bmatrix} Y^T \\ -U^T \end{bmatrix} l = \begin{bmatrix} \frac{\mathbf{B_4}(l)}{\alpha} + G_{(n_a+1)}\lambda & 0 \\ 0 & \frac{\mathbf{B_5}(l)}{\beta} \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} \tag{7.22}$$

$$\frac{\mathbf{B_6}(l)}{\gamma} c = \lambda G_{(n_c+1)} c, \tag{7.23}$$

where $a_0 = 1$ (it then follows from (7.21)-(7.23) that $c_0 = 1$), $G_q \in \mathbb{R}^{q \times q}$ is 0 everywhere except $G_q(q,q) = 1$. The equations (7.21)-(7.23) are a set of $m+n_a+n_b+n_c+2$ nonlinear equations in as many variables, with $l \in \mathbb{R}^{m\times 1}$ and $\lambda \in \mathbb{R}$ being Lagrange multipliers. From (7.21) and (7.22) the analogy with the Riemannian SVD equations (2.7) in the RiSVD framework for STLS problems should be clear. Indeed, the above equations are also similar to ordinary SVD equations for $[Y \quad -U]$ but they contain additional "weighting matrices" (e.g. in (7.21) $(\frac{\mathbf{B_1}(a)}{\alpha} + \frac{\mathbf{B_2}(b)}{\beta} + \frac{\mathbf{B_3}(c)}{\gamma})$) that in turn depend upon the "singular vectors" $[a^T \quad b^T]^T$ and l. (7.23) can be viewed as a nonlinear constraint imposed on the Riemannian SVD equations (7.21)-(7.22).
In this paragraph it is shown that the choice of the constraints –that have to be imposed to the model vectors a and c of problem (7.7) in order to avoid the trivial solution– is not an obvious one. To this end we first consider a simple AR model, since this is a special case of the framework described in (7.5). The model selection procedure formulated in (7.7) then becomes:

$$\min_{a,\tilde{e}} \tilde{e}^T\tilde{e} \tag{7.24}$$
$$\text{such that } Ya = \tilde{e}.$$

Clearly if we do not want to end up with the trivial solution $a = 0$, $\tilde{e} = 0$ we have to impose an extra constraint on a. The classical PEM approach adopts the constraint $a_0 = 1$. We now investigate what happens when the latter constraint is replaced by $a^Ta = 1$. Problem (7.24) thus becomes:

$$\min_{a} \frac{a^TY^TYa}{m} \tag{7.25}$$
$$\text{such that } a^Ta = 1.$$

We inserted m (the number of rows of Y) in the cost function of (7.25), since this allows us to let $m \to \infty$ while the cost function stays bounded. Suppose that the latent variables in $\tilde{e}$ are i.i.d. zero-mean white Gaussian noise. For stable AR systems it is well-known [63] that the output will then be stationary too. As a consequence, the matrix $\frac{Y^TY}{m}$ becomes Toeplitz in the limit for

$m \to \infty$. Since it is already symmetric by construction, it belongs to the class of centrosymmetric (see definition 3.2.4) matrices. It should be clear that the a that results from (7.25) is the eigenvector corresponding to the smallest eigenvalue of $\frac{Y^TY}{m}$. Therefore, by applying theorem 3.2.6 to problem (7.25) we can conclude that the model vector a is centrosymmetric or skew centrosymmetric and thus we find that the poles of the AR system, i.e. the roots of the following equation in ζ

$$a_0\zeta^{n_a} + a_1\zeta^{n_a-1} + \ldots + a_{n_a-1}\zeta + a_{n_a} = 0,$$

come in pairs $(\zeta, 1/\zeta)$ (i.e. symmetric w.r.t. the unit circle). Summarizing we have shown that by selecting a constraint on a different from the classical one, we obtain a model selection procedure that excludes minimum phase models. From a system identification point of view, this "new" model selection procedure does not make much sense of course.
Next, if we reconsider the generalized framework in (7.7) and impose the constraints $a^Ta = 1$ and $c^Tc = 1$ instead of (7.15) and (7.16), the Lagrange multiplier equations (7.8), (7.10), (7.15), (7.16) change into:

$$Z^Tl + \kappa a = 0 \tag{7.26}$$

$$-E^Tl + \lambda c = 0 \tag{7.27}$$

$$a^Ta = 1 \tag{7.28}$$

$$c^Tc = 1. \tag{7.29}$$

Using (7.27), the definition of $\mathbf{P_6}$, (7.13), the exchange lemma and the definition of $\mathbf{B_6}$, we obtain:

$$\begin{aligned} \lambda c &= E^Tl \\ &= \mathbf{P_6}(l)\tilde{e} \\ &= \frac{\mathbf{P_6}(l)\mathbf{P_3}(c)^T}{\gamma}l \\ &= \frac{\mathbf{B_6}(l)}{\gamma}c. \end{aligned} \tag{7.30}$$

It is straightforward to check that since E is a Hankel matrix, $\mathbf{B_6}(l)$ will be a symmetric Toeplitz matrix. Note that (7.30) has now become one of the equations (compare it to (7.23), the third equation determining stationary points in case linear constraints on a and c are used) that determines stationary points of the new (i.e. with the linear constraints replaced by new quadratic ones) model selection procedure. From (7.30) we see that c is an eigenvector of a symmetric Toeplitz (and thus centrosymmetric) matrix, namely $\mathbf{B_6}(l)$. Thus –for the same reasons as mentioned before for the AR model vector a– we can conclude that the zeroes of the Moving Average (MA) model c are symmetric w.r.t. the unit circle. Therefore, also in the generalized framework it does not make sense to use quadratic constraints instead of the proposed linear ones.

7.3 Properties and interpretations

In this section we discuss some properties and new interpretations of the new misfit versus latency framework. In the first subsection some orthogonality properties between the misfit on the input, the misfit on the output and the latent variables are discussed. The second subsection proves that the misfit on the input, the misfit on the output and the latent variables satisfy themselves linear systems. Finally subsection 7.3.3 shows that it is possible to interprete the misfit versus latency framework discussed in this chapter, as a pure latency framework applied to corrected input and output data matrices U and Y.

7.3.1 Orthogonality

The first orthogonality property we prove concerns the "corrected" input $\tilde{w}$ and the input misfit $\tilde{w} - \tilde{u}$.

Property 7.3.1 *The "corrected" input $\tilde{w}$ and the input misfit $\tilde{w} - \tilde{u}$ obtained as the result of the model selection procedure (7.7) are orthogonal:*

$$\tilde{w}^T(\tilde{w} - \tilde{u}) = 0.$$

Proof: Using (7.11), the definition of $\mathbf{P_2}$ and (7.9), we find

$$\begin{aligned} \tilde{w}^T(\tilde{w} - \tilde{u}) &= \tilde{w}^T \mathbf{P_2}(b)^T l/\beta \\ &= b^T W^T l/\beta = 0, \end{aligned}$$

thereby proving the first orthogonality property. □

The next orthogonality property is on the output misfit and the latent variables.

Property 7.3.2 *Let* $K = \begin{bmatrix} \alpha I_{N-M+n_a+1} & 0 \\ 0 & \gamma I_{N-M+n_c+1} \end{bmatrix}$. *The vector that contains the "corrected" output and the latent variables is K-orthogonal to the vector containing the output misfit and the latent variables:*

$$[(\tilde{z} - \tilde{y})^T \ \tilde{e}^T] K \begin{bmatrix} \tilde{z} \\ \tilde{e} \end{bmatrix} = 0.$$

Proof: Consecutive application of (7.12), the definition of $\mathbf{P_1}$, (7.8) and (7.15) yields:

$$\begin{aligned} \tilde{z}^T(\alpha I_{N-M+n_a+1})(\tilde{z} - \tilde{y}) &= -\tilde{z}^T \mathbf{P_1}(a)^T l \\ &= -a^T Z^T l \\ &= \kappa. \end{aligned} \tag{7.31}$$

On the other hand, using (7.13), the definition of $\mathbf{P_3}$, (7.10) and (7.16), we obtain:

$$\begin{aligned} \tilde{e}^T(\gamma I_{N-M+n_c+1})\tilde{e} &= \tilde{e}^T\mathbf{P_3}(c)^T l \\ &= c^T E^T l \\ &= \lambda. \end{aligned} \tag{7.32}$$

Since $\kappa = -\lambda$ the property follows by adding (7.31) and (7.32). □

7.3.2 LTI systems for misfit and latency

Remember the rank 1 property of the misfit $[\Delta A \; \Delta b]$ in the static case. Similarly, the latent variables, the input misfit and the output misfit are structured too in the sense that under certain conditions on the model orders, they satisfy themselves model equations of LTI dynamic systems (whose parameter vectors can easily be derived from the parameter vectors obtained in (7.7)).

Property 7.3.3 *If $n_a + 1 = n_b$, the input misfit and the output misfit satisfy the following LTI dynamic system equations:*

$$-\beta(W - U)a(n_a + 1 : -1 : 1) = \alpha(Z - Y)b(n_b : -1 : 1). \tag{7.33}$$

If $n_a = n_c$, the output misfit and the latent variables satisfy the following LTI dynamic system equations:

$$-\alpha(Z - Y)c(n_c + 1 : -1 : 1) = \gamma E a(n_a + 1 : -1 : 1). \tag{7.34}$$

If $n_b = n_c + 1$, the input misfit and the latent variables satisfy the following LTI dynamic system equations:

$$\beta(W - U)c(n_c + 1 : -1 : 1) = \gamma E b(n_b : -1 : 1). \tag{7.35}$$

Proof: First note that the following properties are readily checked if $n_a + 1 = n_b$, simply by using the definitions of $\mathbf{P_1}$ and $\mathbf{P_2}$:

$$\mathbf{P_1} = \mathbf{P_2} \tag{7.36}$$

$$\mathbf{P_1}(b(n_b : -1 : 1))\mathbf{P_1}(a)^T = \mathbf{P_1}(a(n_a + 1 : -1 : 1))\mathbf{P_1}(b)^T. \tag{7.37}$$

Using (7.11), (7.12), (7.36) we find

$$\beta\mathbf{P_1}(a(n_a + 1 : -1 : 1))(\tilde{w} - \tilde{u}) = \mathbf{P_1}(a(n_a + 1 : -1 : 1))\mathbf{P_1}(b)^T l \text{ and}$$

$$\alpha\mathbf{P_1}(b(n_b : -1 : 1))(\tilde{z} - \tilde{y}) = -\mathbf{P_1}(b(n_b : -1 : 1))\mathbf{P_1}(a)^T l.$$

From the latter 2 equations, the definition of $\mathbf{P_1}$ and (7.37) we obtain:

$$-\beta(W - U)a(n_a + 1 : -1 : 1) = \alpha(Z - Y)b(n_b : -1 : 1).$$

Similarly, if $n_a = n_c$ the following is easily verified by using the definitions of $\mathbf{P_1}$ and $\mathbf{P_3}$:

$$\mathbf{P_1} = \mathbf{P_3} \tag{7.38}$$
$$\mathbf{P_1}(c(n_c+1:-1:1))\mathbf{P_1}(a)^T = \mathbf{P_1}(a(n_a+1:-1:1))\mathbf{P_1}(c)^T. \tag{7.39}$$

Combining (7.12), (7.13), (7.38) it is clear that:

$$-\alpha\mathbf{P_1}(c(n_c+1:-1:1))(\tilde{z}-\tilde{y}) = \mathbf{P_1}(c(n_c+1:-1:1))\mathbf{P_1}(a)^T l \text{ and}$$
$$\gamma\mathbf{P_1}(a(n_a+1:-1:1))\tilde{e} = \mathbf{P_1}(a(n_a+1:-1:1))\mathbf{P_1}(c)^T l.$$

Using the latter 2 equations, the definition of $\mathbf{P_1}$ and (7.39) we find:

$$-\alpha(Z-Y)c(n_c+1:-1:1) = \gamma Ea(n_a+1:-1:1).$$

To obtain the third LTI dynamic system of this property, note that if $n_b = n_c+1$ straightforward application of the definitions of $\mathbf{P_2}$ and $\mathbf{P_3}$ gives:

$$\mathbf{P_2} = \mathbf{P_3} \tag{7.40}$$
$$\mathbf{P_2}(c(n_c+1:-1:1))\mathbf{P_2}(b)^T = \mathbf{P_2}(b(n_b:-1:1))\mathbf{P_2}(c)^T. \tag{7.41}$$

Using (7.11), (7.13), (7.40) yields:

$$\beta\mathbf{P_2}(c(n_c+1:-1:1))(\tilde{w}-\tilde{u}) = \mathbf{P_2}(c(n_c+1:-1:1))\mathbf{P_2}(b)^T l$$
$$\gamma\mathbf{P_2}(b(n_b:-1:1))\tilde{e} = \mathbf{P_2}(b(n_b:-1:1))\mathbf{P_2}(c)^T l.$$

Combining the latter 2 equations, the definition of $\mathbf{P_2}$ and (7.41), it follows that:

$$\beta(W-U)c(n_c+1:-1:1) = \gamma Eb(n_b:-1:1).$$

□

If in the previous property, the conditions on the orders are not satisfied, it is still possible to prove similar properties. If e.g. $n_b > n_a+1$, a formula similar to (7.33) can be proven by defining a new $\check{\mathbf{P}}_\mathbf{1}(a) \in \mathbb{R}^{m\times(N-M+n_b)}$, which is simply $\mathbf{P_1}(a)$ padded with $n_b - n_a - 1$ zero columns.

7.3.3 Pure latency interpretation

In this subsection we demonstrate how the misfit versus latency framework can be interpreted as a pure latency framework applied to corrected input and output data matrices.

Property 7.3.4 *The misfit versus latency framework characterized by the Riemannian SVD equations (7.21)-(7.23), with the additional constraint $a_0 = 1$ can be interpreted as a pure latency framework whose stationary points are determined by the following equations:*

$$[Y_{corr} \quad -U_{corr}] \begin{bmatrix} a \\ b \end{bmatrix} = \frac{\mathbf{B_3}(c)}{\gamma} \tag{7.42}$$

$$\begin{bmatrix} Y_{corr}^T \\ -U_{corr}^T \end{bmatrix} l = \begin{bmatrix} G_{n_a+1}\lambda a \\ 0 \end{bmatrix} \tag{7.43}$$

$$\frac{\mathbf{B_6}(l)}{\gamma} c = \lambda G_{n_a+1} c, \tag{7.44}$$

with $a_0 = 1$, $Y_{corr} = Y - \frac{\mathbf{P_1}(a)\mathbf{P_4}(l)^T}{\alpha}$ and $U_{corr} = U + \frac{\mathbf{P_2}(b)\mathbf{P_5}(l)^T}{\beta}$.

Proof: Substituting the formulas for Y_{corr} and U_{corr} in (7.42), and through the use of the exchange lemma, the definition of $\mathbf{B_1}$ and $\mathbf{B_2}$ we obtain (7.21). On the other hand if Y_{corr} and U_{corr} are substituted in (7.43) and use is made of the exchange lemma and the definition of $\mathbf{B_4}$ and $\mathbf{B_5}$, we recover (7.22). Furthermore, since (7.44) is the same as (7.23) and we also imposed $a_0 = 1$, it is proven that the misfit versus latency framework (represented by the Riemannian SVD equations (7.21)-(7.23)) is equivalent to the pure latency framework applied to the corrected data matrices Y_{corr} and U_{corr} (and represented by the Riemannian SVD equations (7.42)-(7.44)). □

7.4 Optimization approach

In this subsection we derive an optimization approach for the misfit versus latency framework. The final result obtained will be a nonlinear cost function that only depends on the model parameters a, b and c, with some extra constraints on a and c. Since this optimization problem is derived from the Lagrange multipliers equations (7.8)-(7.16), it yields the same solution as the original misfit versus latency model selection procedure (7.7) with the constraints $a_0 = 1$, $c_0 = 1$.
Using (7.11) and the definition of $\mathbf{B_2}$, we find

$$(\tilde{w} - \tilde{u})^T(\tilde{w} - \tilde{u}) = l^T \mathbf{B_2}(b) l / \beta^2. \tag{7.45}$$

Similarly, we find by combining (7.12) with the definition of $\mathbf{B_1}$ that

$$(\tilde{z} - \tilde{y})^T(\tilde{z} - \tilde{y}) = l^T \mathbf{B_1}(a) l / \alpha^2, \tag{7.46}$$

and finally, proper use of (7.13) and the definition of $\mathbf{B_3}$ reveals that

$$\tilde{e}^T \tilde{e} = l^T \mathbf{B_3}(c) l / \gamma^2. \tag{7.47}$$

From (7.14), the definition of $\mathbf{P_1}$, $\mathbf{P_2}$ and $\mathbf{P_3}$, we know that

$$\mathbf{P_1}(a)\tilde{z} - \mathbf{P_2}(b)\tilde{w} - \mathbf{P_3}(c)\tilde{e} = 0.$$

Using (7.11)-(7.13), the definitions of $\mathbf{P_1}$, $\mathbf{P_2}$, $\mathbf{B_1}$, $\mathbf{B_2}$ and $\mathbf{B_3}$, the last equation becomes

$$Ya - Ub = (\frac{\mathbf{B_1}(a)}{\alpha} + \frac{\mathbf{B_2}(b)}{\beta} + \frac{\mathbf{B_3}(c)}{\gamma})l \text{ or}$$

$$l = (\frac{\mathbf{B_1}(a)}{\alpha} + \frac{\mathbf{B_2}(b)}{\beta} + \frac{\mathbf{B_3}(c)}{\gamma})^{-1}(Ya - Ub). \tag{7.48}$$

Combining (7.45), (7.46), (7.47) and (7.48), the cost function in the model selection procedure can be rewritten as follows:

$$[a^T \; b^T] \begin{bmatrix} Y^T \\ -U^T \end{bmatrix} (\frac{\mathbf{B_1}}{\alpha} + \frac{\mathbf{B_2}}{\beta} + \frac{\mathbf{B_3}}{\gamma})^{-1} [Y \;\; -U] \begin{bmatrix} a \\ b \end{bmatrix}. \tag{7.49}$$

Note that (7.49) only depends on the model parameters a, b and c. Careful inspection of the Lagrange multiplier equations (7.8)-(7.16) that have been used to arrive at (7.49) reveals that only constraint independent multiplier equations were used. The latter means that if the constraints on a and c (see equations (7.15) and (7.16)) were changed, we would still find the same cost function (7.49). The cost function (7.49) is scaling invariant in the parameter vector $[a^T \; b^T \; c^T]^T$ (this is easily verified by noting the quadratic dependence of $\mathbf{B_1}(a)$ on a, $\mathbf{B_2}(b)$ on b and $\mathbf{B_3}(c)$ on c), implying that we have to impose extra constraints on the parameter vectors in order to single out a solution. Note that this scaling invariance property only holds if none of the misfit terms is missing. If we write down (7.49) e.g. for an ARMAX model (i.e. a special case of our new framework), it becomes obvious that (7.49) no longer is scaling invariant, because the terms containing $\mathbf{B_1}(a)$ and $\mathbf{B_2}(b)$ have disappeared. In those cases the cost function is no longer scaling invariant and the constraints that we impose do not just single out one of the solutions along a line of solutions, but they can lead to solutions lying in different directions. To obtain the same results as in our misfit versus latency framework, we also impose the constraints $a_0 = 1$ and $c_0 = 1$. Summarizing, we have derived an optimization formulation that yields the same solution as the model selection procedure (7.7):

$$\min_{a,b,c} [a^T \; b^T] \begin{bmatrix} Y^T \\ -U^T \end{bmatrix} (\frac{\mathbf{B_1}(a)}{\alpha} + \frac{\mathbf{B_2}(b)}{\beta} + \frac{\mathbf{B_3}(c)}{\gamma})^{-1} [Y \;\; -U] \begin{bmatrix} a \\ b \end{bmatrix} \tag{7.50}$$
$$\text{such that } a_0 = 1, c_0 = 1.$$

The advantage of this approach over the Riemannian SVD equations (7.21)-(7.23) is the existence of software packages for solving the nonlinear optimization problem (7.50). Solving the set of Riemannian SVD equations (7.21)-(7.23) turns out to be a lot more complicated (note that one approach for solving this set of equations could be the development of an inverse iteration algorithm similar to the one derived for the STLS problem in the RiSVD framework).

7.5 Experiments

In this section we describe 2 experiments to give an idea of the potential of the misfit versus latency framework. In the first experiment we simulate an ARMAX model. The experiment itself consists in determining models of the new framework using the selection procedure (7.50). Models are determined for several combinations of the triplet (α, β, γ) in such a way that we obtain a "root locus" with on one end the PEM approach and on the other end the behavioral modeling approach.
The second experiment shows how knowledge of the statistical properties of the misfit and latency variables can be exploited in the new framework, thereby leading to an improved statistical accuracy of the estimated parameter vectors. The results are compared to that of the classical PEM approach.

7.5.1 Experiment 1

For this experiment we start from the following exact model parameters:

$$\begin{aligned} a &= [0.81 \ -1.2541 \ 1]^T \\ b &= [1 \ 2.3 \ 0.3]^T \\ c &= 1. \end{aligned}$$

Using an i.i.d. Gaussian noise sequence $e(t)$ with standard deviation 1, a deterministic input sequence $w(t)$ and initial conditions $z(1) = z(2) = 0$, an output $z(t)$ is generated using to the following ARMAX model equation:

$$A(q)z(t) = B(q)w(t) + C(q)e(t), t = 3, \ldots, N.$$

Setting $N = 50$ we obtain figure 7.3, which shows the above mentioned sequences. Since we consider the case that there is no misfit on the input nor on the output, we have that $z(t) = y(t)$ and $w(t) = u(t)$ (see figure 7.2). Using $\tilde{y} = y(1:50)$, $\tilde{u} = u(1:50)$, $\tilde{e} = e(3:50)$, we can construct the corresponding matrices Y, U and E and apply the model selection procedure (7.50). The optimization is performed using the function leastsq in Matlab. The procedure is repeated for the following triplets (α, β, γ):

- case 1: $(\alpha, \beta, \gamma) = (10^5, 10^5, 1)$
- case 2: $(\alpha, \beta, \gamma) = (10^3, 10^3, 10^2)$
- case 3: $(\alpha, \beta, \gamma) = (1, 1, 10^5)$.

Case 1 clearly corresponds to the ARMAX case (see case 26 of table 7.1). Case 3 clearly corresponds to the dynamic TLS model (see case 15 of table

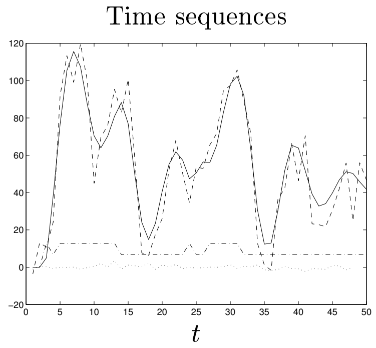


Figure 7.3: *This figure shows the deterministic input $w(t)$ (dash-dotted line), the i.i.d. Gaussian noise input $e(t)$ (dotted line) of experiment 1. The full line is the sequence $z(t)$, obtained by using the ARMAX model and the initial conditions described in the first experiment. The dashed line is the sequence $y(t)$ as used in the second experiment. It is obtained by adding a noise "misfit" (in this particular case i.i.d. Gaussian noise with standard deviation 10) to the sequence $z(t)$.*

7.1). Case 2 is a model that lies in between an ARMAX model and a dynamic TLS model. To give an idea of the effect of the different choices of the triplet (α, β, γ), we plot in figure 7.4 the poles (zeroes of the polynomial based on the estimated a): poles of case 1 are indicated by a "x", those of case 2 by a "o" and for case 3 we used a "+". The left hand side shows the location of the poles w.r.t. the unit circle. The right hand side zooms in on the region of interest. We clearly see the "root locus" for the poles of the different models, i.e. the continuous evolution of the poles when going from a PEM approach ("x") towards a behavioral modeling approach ("+").

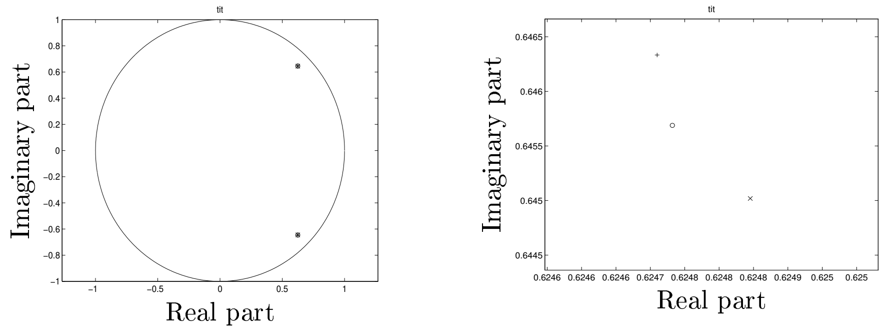


Figure 7.4: *These figures give a geometrical interpretation of the different methods for solving a system of overdetermined equations.*

7.5.2 Experiment 2

In this experiment we use the same ARMAX model as in the first experiment. The output z is thus generated in the same way as in the first experiment, but in this case we add noise to z such that $z \neq y$ as was the case in experiment 1. The noise added to z is i.i.d. Gaussian noise with standard deviation 10. The sequence y that is obtained in this way is the dashed line in figure 7.3. We now perform 200 Monte-Carlo simulations in which each time a different realization of the latent variable e and the noise misfit on the output $(y - z)$ is generated. In each run we determine 2 models based on the available measurements y and u:

1. a model determined using (7.50) with $\alpha = 1$, $\beta = 10^5$ and $\gamma = 100$.

2. a classical ARMAX model (i.e. the PEM approach) using the identification toolbox in Matlab.

The choice of α, β and γ is obviously based on the knowledge of the statistical properties of the latent variable and the output misfit. First of all, $\gamma/\alpha = 100$ since the variance on the output misfit is 100 times larger than the variance on the latent variable. Furthermore, $\beta = 10^5$, since no misfit is present on the input (note that the 10^5 is thus used as an approximation of ∞).
The goal of the experiment is in fact to see whether we can get better statistical results with our new framework, if the statistical properties of noise misfit and latent variables are properly taken into account. To get an idea of the statistical accuracy of our new framework and the associated selection procedure (7.50), we calculate the following relative errors for the "misfit versus latency" model obtained with (7.50) and the classical PEM ARMAX result:

$$\begin{aligned} err_{mvl} &= \sum_{i=1}^{200} \frac{\|a_{mvl}^{(i)} - a\|_2}{\|a\|_2} \\ err_{PEM} &= \sum_{i=1}^{200} \frac{\|a_{PEM}^{(i)} - a\|_2}{\|a\|_2}, \end{aligned}$$

where a is the exact AR parameter vector, $a_{mvl}^{(i)}$ is the AR parameter vector obtained using (7.50) in the ith Monte Carlo run and $a_{PEM}^{(i)}$ is the AR parameter vector obtained using the classical PEM approach in the ith Monte Carlo run. The relative accuracies obtained in this way are:

$$err_{mvl} = 1.84\% \text{ and } err_{PEM} = 35.75\%.$$

This clearly shows the potential of the new framework.

7.6 Conclusion

In this chapter we have presented the misfit versus latency framework that was introduced in [69]. The latter framework combines the ideas of the PEM approach and the behavioral modeling approach. We have shown that by using the technique of Lagrange multipliers, it is possible to rewrite the model selection procedure of the new framework as a set of Riemannian SVD equations that is very similar to the set of equations obtained in the RiSVD framework for the STLS problem. The importance of the constraints imposed on the model parameters has been demonstrated. Some orthogonality properties and interpretations are proven.
A new optimization approach of the misfit versus latency framework is presented and proofs are given. Using this new optimization approach we have been able to do some experiments in the new framework. First of all it has been demonstrated –by means of a root locus plot– how the specific values of the weights α, β and γ influence the location of the poles of the selected model. Secondly, it has been shown that knowledge of the misfit and latency variables can greatly enhance the statistical accuracy of the obtained model parameter vectors, compared to the classical PEM approach.

Chapter 8

Conclusions and open problems

In the first section of this chapter the conclusions are summarized. A second section describes open problems and outlines suggestions for further research.

8.1 Conclusions

In this thesis we have treated the analysis, algorithms and applications of Structured Total Least Squares problems. We now present the conclusions and original contributions.

- The different existing approaches for Structured Total Least Squares problems, the Constrained Total Least Squares, Structured Total Least Norm and Riemannian SVD approach have been proven to be equivalent (section 2.3) despite previous claims of inequality in literature. The Structured Total Least Squares problems have been situated in a larger framework of so-called Structured Total Approximation Problems. This extended framework envisages other cost functions than the sum of squares as well as nonlinearly structured matrices instead of only considering affinely structured matrices. The Riemannian SVD framework has been extended for nonlinearly structured matrices. The theoretical derivation of this nonlinear Structured Total Least Squares approach has been presented in section 2.4.2 and was illustrated with a numerical example.
- The Structured Total Least Squares problem is shown to be a nonconvex

problem, suffering from many local minima. Using existing theoretical results, a new definition for nongeneric Structured Total Least Squares problems has been introduced and it has been shown that under certain conditions due to this nongenericity, the Structured Total Least Squares problem fails to have a solution. Theory and definitions have been illustrated with new examples.
Furthermore it has been shown by means of examples that besides the local minima problem, the Structured Total Least Squares problem might also have several *optimal* solutions.

- A new projection interpretation of the Structured Total Least Squares problem has been derived in section 3.2.1. Using the latter interpretation, it is possible to view Structured Total Least Squares problems as a kind of structured subspace methods, based on newly defined (structure-dependent) signal and noise subspaces (see figure 3.2). The newly defined signal and noise subspaces have been illustrated on two structures: the Hankel matrix and the unstructured matrix.

- Based on the projection interpretation an upper bound for the objective function in the STLS problem (3.1) has been derived. For reasons of completeness we have also derived a lower bound for the latter objective function. However, it is mainly the upper bound that turns out to be useful in applications as described in section 3.2.

- Invertibility issues of the matrix $\mathbf{D_2}(y)$ have been discussed, since this matrix plays an important role in the projection interpretation as well as in some other formulations. It has been shown that these invertibility problems can be divided into two categories. The example of symmetric Toeplitz matrices has been analyzed in some detail, thereby proving why this structure leads to invertibility problems. Furthermore it has been proven that the problems in this particular case can be avoided by using a matrix function $\mathbf{D_2}(y)$ that corresponds to a different matrix structure, without of course changing the solution of the Structured Total Least Squares problem.

- The nature of Structured Total Least Squares problems involving Hankel (Toeplitz) matrices has been analyzed in detail in section 3.3, by using a specific parametrization of the data in these Hankel (Toeplitz) matrices. The difficulty of the Structured Total Least Squares problem has been shown to depend on the values of the parameters of this new parametrization at the solution of the Structured Total Least Squares problem.

- In chapter 4 existing algorithms and alternative algorithms have been discussed for the Constrained Total Least Squares and the Riemannian SVD framework. For the Structured Total Least Norm framework a new algorithm has been developed. The main improvement consists in the replacement of the previously used penalty function approach by the numerically robust equality constrained least squares approach. Another important improvement, given the problem of the multiple local minima, has been the use of subspace based initialization methods. For Hankel (Toeplitz) matrices this has been realized by an ESPRIT-type approach, but also for other structures a subspace based initialization method, based on the projection interpretation of the Structured Total Least Squares problem, has been derived (see end of section 4.5).

- Using a deconvolution example in a medical application called renography, it has been demonstrated how efficient algorithms can be developed for a particular Structured Total Least Squares problem. This has been done by exploiting the low displacement rank of the matrices involved in the kernel routine.
- A novel speech compression algorithm has been proposed. It is a subband based scheme in which the compression of the low frequency band is essentially a Structured Total Least Squares problem. It has been shown that this new scheme is related to the class of sinusoidal coders, the main difference being that damped sinusoids are used instead of undamped sinusoids. This way of encoding allows variable rate speech compression. The latter means that without having to re-encode the data, it is possible to shift to a lower bit rate –depending on the available bandwidth and/or the available computational resources– with a gradual decrease of the audible quality.
- A previously proposed framework that incorporates both the prediction error models and behavioural models has been shown to be very similar to the Structured Total Least Squares problem. A novel optimization approach for this framework has been derived (see formula (7.50). The latter approach has provided many insights, e.g. that the constraints imposed on the model parameters have a severe impact on the obtained result. Using this new approach it has been possible to demonstrate the potential (e.g. increased statistical accuracy of the estimated parameter vectors when the noise conditions are known) of the new framework (see section 7.5).

8.2 Open problems and further research

Structured Total Least Squares problems are still relatively new. In this section we summarize the open problems that we encountered in this thesis, as well as possible subjects for further research.

- In this thesis we have mainly considered the use of the L_2 norm in formulation (2.16). As mentioned in section 2.4, the use of L_p norms for $p \neq 2$ is appropriate under certain noise circumstances. These STAP problems as well as algorithms for solving them are in large part still unexplored. An even more general approach consists in replacing $\|\Delta s\|$ in (2.16) by f where f stands for the probability distribution function of the elements of Δs.

- As mentioned at the end of section 2.4.2, the statistical interpretation of the nonlinear extension of STLS problems is another open problem in the STAP framework. A link with higher order statistics seems very likely.

- An interesting extension of the STLS problem is the formulation in which a regularization term (e.g. $\lambda\|x\|$) is added to the objective function in (2.16). Interesting results in this respect have already been presented in [33, 34, 102] for the so-called regularized Constrained Total Least Squares problem.

- In the speech compression scheme of chapter 6 better perceptual results can be obtained by replacing the objective function (sum of squares) of the involved STLS problem by a cost function that better reflects the perceptual quality of the obtained approximate frame. Another option is the use of pre- and postprocessing algorithms that exploit the knowledge of the human auditory system. Finally, the replacement of the simple bit allocation scheme by a more sophisticated quantization scheme undoubtably will yield better quality at the same bit rate or a lower bit rate at the same quality.

- In chapter 7, it has been demonstrated that the misfit versus latency framework is able to improve the statistical accuracy of the estimated model parameters greatly. However, the choice of the weights α, β and γ was mainly intuitive. To further explore the potential of the new framework, a thorough statistical analysis is necessary (for some results on a number of special cases of the generalized framework see [101]).
  Besides this statistical analysis, it would be interesting to investigate the following subjects:

  - Does it make sense to choose the weights α, β and γ in such a way that a stable (of course other properties that vary along the "root locus" can be selected for this purpose) system is obtained?
  - Can the determination of models for different triplets (α, β, γ) give us information on the (unknown) noise circumstances?

Appendix A

Derivations

A.1 Proof of Lemma 2.3.1

Proof: Applying the method of Lagrange multipliers to (2.10). The Lagrangian for this problem equals:

$$L(\alpha, \beta, x, l) = \alpha^T D_\alpha^2 \alpha + \beta^T D_\beta^2 \beta + l^T (b + P_2 \beta + P_1 \alpha - Ax - X\alpha), \quad \text{(A.1)}$$

with $l \in \mathbb{R}^{m \times 1}$ the vector of Lagrange multipliers. Putting the partial derivatives of (A.1) w.r.t. α, β, x and l equal to 0 yields:

$$\frac{\delta L}{\delta \alpha} = 2D_\alpha^2 \alpha - (X - P_1)^T l = 0 \quad \text{(A.2)}$$

$$\frac{\delta L}{\delta \beta} = 2D_\beta^2 \beta + P_2^T l = 0 \quad \text{(A.3)}$$

$$\frac{\delta L}{\delta x} = -(A + \Delta A)^T l = 0 \quad \text{(A.4)}$$

$$\frac{\delta L}{\delta l} = b + P_2 \beta + P_1 \alpha - Ax - X\alpha = 0. \quad \text{(A.5)}$$

From (A.2) and (A.3) we find that

$$\alpha = 0.5 D_\alpha^{-2} (X - P_1)^T l \ \text{ en } \ \beta = -0.5 D_\beta^{-2} P_2^T l. \quad \text{(A.6)}$$

After substituting (A.6) in (A.5) we obtain

$$l = -2(P_2 D_\beta^{-2} P_2^T + (X - P_1) D_\alpha^{-2} (X - P_1)^T)^{-1} [A \;\; b] \begin{bmatrix} x \\ -1 \end{bmatrix}. \quad \text{(A.7)}$$

Substituting (A.6) and (A.7) in the objective function of (2.10), using the definition $y \equiv -y(n+1)[x^T \;\; -1]^T$ and the definitions of $\tilde{X}$, $\tilde{P}_1$ and $\tilde{P}_2$, we obtain the following problem equivalent to (2.10):

$$\begin{aligned}&\min_y y^T S^T (\tilde{P}_2 D_\beta^{-2} \tilde{P}_2^T + (\tilde{X} - \tilde{P}_1) D_\alpha^{-2} (\tilde{X} - \tilde{P}_1)^T)^{-1} S y \\ &y(n+1) = -1\end{aligned}$$

□

A.2 Proof of Lemma 2.3.2

Proof: This Lemma could also be proven by using the method of Lagrange multipliers (see [61]). Since the formulation (2.7) is itself already the result of applying the method of Lagrange multipliers to (2.6), we can start from the equations in (2.7):

$$\begin{aligned}&\mathbf{D_2}(v)^{-1} S v = u\tau \Rightarrow \\ &S^T \mathbf{D_2}(v)^{-1} S v = S^T u \tau \Rightarrow \\ &v^T S^T \mathbf{D_2}(v)^{-1} S v = \tau^2\end{aligned}$$

Noting that in (2.7) we are looking for the triplet (u, τ, v) that corresponds to the smallest τ we see from the last equation and the constraints in (2.7) that (2.7) can be reformulated as follows:

$$\begin{aligned}&\min_v v^T S^T \mathbf{D_2}(v)^{-1} S v \\ &v^T v = 1\end{aligned} \tag{A.8}$$

Knowing that $y = v$ (see (2.9)), the proof is completed. □

Note that at the end of the proof we might as well have opted for the constraint $v^T \mathbf{D_1}(u) v = 1$ (or $u^T \mathbf{D_2}(v) u = 1$; however if 1 of both is satisfied the other is also satisfied, see remark 2.2.2). Although this looks a bit more complicated (due to the appearance of u), this constraint still allows the use of Lemma 2.3.4 for proving the equivalence of the three approaches.

A.3 Proof of Lemma 2.3.3

Proof: We begin with the first equality. What we have to prove for the first equality to be true is

$$\begin{aligned}\mathbf{H_2}(y)W^{-1}\mathbf{H_2}(y)^T &= \tilde{P}_2 D_\beta^{-2}\tilde{P}_2^T + (\tilde{X}-\tilde{P}_1)D_\alpha^{-2}(\tilde{X}-\tilde{P}_1)^T \\ &= [\tilde{X}-\tilde{P}_1 \;\; \tilde{P}_2]\begin{bmatrix} D_\alpha^2 & 0 \\ 0 & D_\beta^2 \end{bmatrix}^{-1}\begin{bmatrix} (\tilde{X}-\tilde{P}_1)^T \\ \tilde{P}_2^T \end{bmatrix}\end{aligned}$$

Using the definitions of the symbols in subsections 2.2.1 and 2.2.3, we can choose Δs such that $\Delta s = [\alpha^T \; \beta^T]^T$. Furthermore, from the above mentioned subsections it is clear that

$$P_2\beta - P_1\alpha = F_{n+1}\Delta s \tag{A.9}$$

$$X\alpha = (F_1 x(1) + \ldots + F_n x(n))\Delta s. \tag{A.10}$$

With $y \in \mathbb{R}^{(n+1)\times 1}$, $y = -y(n+1)[x^T \;\; -1]^T$, $\mathbf{H_2}(y)$ defined as in (2.11) and using the defintions of $\tilde{X}$, $\tilde{P}_1$ and $\tilde{P}_2$, (A.9)-(A.10) becomes:

$$[\tilde{X}-\tilde{P}_1 \; \tilde{P}_2]\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \mathbf{H_2}(y)\Delta s.$$

Since Δs is a minimal representation of the elements of ΔS, we can conclude that

$$[\tilde{X}-\tilde{P}_1 \; \tilde{P}_2] = \mathbf{H_2}(y). \tag{A.11}$$

Furthermore, since we chose $\Delta s = [\alpha^T \; \beta^T]^T$, it is clear from (2.4) and (2.10) that in case the errors Δs and $[\alpha^T \; \beta^T]^T$ are weighted in the same way the following equality holds:

$$\begin{bmatrix} D_\alpha^2 & 0 \\ 0 & D_\beta^2 \end{bmatrix} = W. \tag{A.12}$$

Combining (A.11) and (A.12) proves the first equality of this Lemma.
To prove the second equality we calculate ΔSy using the CTLS notation:

$$\begin{aligned}\Delta Sy &= [F_1\Delta s \; F_2\Delta s \ldots F_{n+1}\Delta s]y \\ &= F_1\Delta s y(1) + \ldots + F_{n+1}\Delta s y(n+1) \\ &= (F_1 y(1) + \ldots + F_{n+1}y(n+1))\Delta s \end{aligned} \tag{A.13}$$

and using the RiSVD notation we obtain

$$\begin{aligned}\Delta Sy &= \sum_{j=1}^{k} T_j \Delta s(j) y \\ &= [T_1 y \ldots T_k y]\Delta s. \end{aligned} \tag{A.14}$$

From (A.13) and (A.14) and the minimal dimension of Δs, we conclude that

$$F_1 y(1) + \ldots + F_{n+1} y(n+1) = [T_1 y \ldots T_{n+1} y]. \tag{A.15}$$

Thus using (A.15) and the definition of $\mathbf{H_2}(y)$ and $\mathbf{D_2}(y)$ (see subsection 2.2.2), we obtain:

$$\begin{aligned}
\mathbf{H_2}(y) W^{-1} \mathbf{H_2}(y)^T &= (\sum_{i=1}^{n+1} F_i y(i)) W^{-1} (\sum_{i=1}^{n+1} F_i y(i))^T \\
&= [T_1 y \ldots T_k y] W^{-1} [T_1 y \ldots T_k y]^T \\
&= \sum_{j=1}^{k} W(j,j)^{-1} (T_j y)(T_j y)^T \\
&= \mathbf{D_2}(y),
\end{aligned}$$

thereby proving the second equality of Lemma 2.3.3. □

Appendix B

Proofs of chapter 3

B.1 Ratio r

We first consider the case when i and j are the only signal components. Starting from the definition of C_i and using some elementary goniometry (see (B.27)), we can derive the following formula for the cross terms (which is also valid for the quadratic terms by putting $i = j$):

$$
\begin{aligned}
2C_i(l)C_j(l) &= a_i a_j e^{(d_i+d_j)l} cos((\omega_i - \omega_j)l + p_i - p_j) && \text{(B.1)}\\
&\quad -a_i a_j e^{(d_i+d_j)l} cos((\omega_i + \omega_j)l + p_i + p_j) && \text{(B.2)}\\
&\quad -a_i \hat{a}_j e^{(d_i+\hat{d}_j)l} cos((\omega_i - \hat{\omega}_j)l + p_i - \hat{p}_j) && \text{(B.3)}\\
&\quad +a_i \hat{a}_j e^{(d_i+\hat{d}_j)l} cos((\omega_i + \hat{\omega}_j)l + p_i + \hat{p}_j) && \text{(B.4)}\\
&\quad -\hat{a}_i a_j e^{(\hat{d}_i+d_j)l} cos((\hat{\omega}_i - \omega_j)l + \hat{p}_i - p_j) && \text{(B.5)}\\
&\quad +\hat{a}_i a_j e^{(\hat{d}_i+d_j)l} cos((\hat{\omega}_i + \omega_j)l + \hat{p}_i + p_j) && \text{(B.6)}\\
&\quad +\hat{a}_i \hat{a}_j e^{(\hat{d}_i+\hat{d}_j)l} cos((\hat{\omega}_i - \hat{\omega}_j)l + \hat{p}_i - \hat{p}_j) && \text{(B.7)}\\
&\quad -\hat{a}_i \hat{a}_j e^{(\hat{d}_i+\hat{d}_j)l} cos((\hat{\omega}_i + \hat{\omega}_j)l + \hat{p}_i + \hat{p}_j) && \text{(B.8)}
\end{aligned}
$$

Looking at the different terms (B.1) to (B.8), we see that high-frequency terms (i.e. terms with angular frequencies $\omega_i + \omega_j, \omega_i + \hat{\omega}_j, \hat{\omega}_i + \omega_j$ and $\hat{\omega}_i + \hat{\omega}_j$) alternate with low(er)-frequency terms (i.e. terms with angular frequencies $\omega_i - \omega_j, \omega_i - \hat{\omega}_j, \hat{\omega}_i - \omega_j$ and $\hat{\omega}_i - \hat{\omega}_j$). When the dampings satisfy the condition

$$\frac{2\pi}{\overset{(\wedge)}{\omega}_i + \overset{(\wedge)}{\omega}_j} < \frac{1}{|\overset{(\wedge)}{d}_i + \overset{(\wedge)}{d}_j|}, \qquad \text{(B.9)}$$

where $\overset{(\wedge)}{\omega}$ (respectively $\overset{(\wedge)}{d}$) stands for ω or $\hat{\omega}$ (respectively d or $\hat{d}$), (this condition says that the time constant of decay of the terms (B.1)-(B.8) is larger than a period of a high-frequency term), it is easy to understand that the sum (over time, from $l = 0$ to $l = N-1$) of the high-frequency terms is negligible with respect to the sum (over time, from $l = 0$ to $l = N-1$) of the low-frequency terms. To see this just think of the sum as a discretization of the integral. To get an idea of the order of magnitude of $\sum_{l=0}^{N-1}$(A-x), $x = 1, \ldots, 8$ integrate (A-x), $x = 1, \ldots, 8$ over 1 half period. It then becomes clear that the sums $\sum_{l=0}^{N-1}$(B.1),..., $\sum_{l=0}^{N-1}$(B.8) are more or less inversely proportional to the corresponding frequency, thereby explaining what was stated before. To get a qualitative idea of what happens to the ratio

$$\frac{\sum_{l=0}^{N-1} 2C_i(l)C_j(l)}{\sum_{l=0}^{N-1} C_i^2(l)}$$

we can take the ratio of the corresponding (B.1) terms. Using the formula (B.22) from appendix B.4, this yields the following indicative ratio r:

$$r \equiv 2\frac{a_j}{a_i}\frac{num_1}{den_1}\frac{num_2}{den_2} \tag{B.10}$$

with

$$\begin{aligned}
num_1 &\equiv 1 - 2e^{2d_i} + e^{4d_i} \\
den_1 &\equiv 1 - 2e^{(d_i+d_j)}cos(\omega_i - \omega_j) + e^{2(d_i+d_j)} \\
num_2 &\equiv cos(p_i - p_j) - e^{(d_i+d_j)}cos(p_i - p_j - (\omega_i - \omega_j)) - \\
&\quad e^{(d_i+d_j)N}cos((\omega_i - \omega_j)N + p_i - p_j) + \\
&\quad e^{(d_i+d_j)(N+1)}cos((\omega_i - \omega_j)(N-1) + p_i - p_j) \\
den_2 &\equiv 1 - e^{2d_i} - e^{2d_iN} + e^{2d_i(N+1)}
\end{aligned}$$

Influence of the amplitudes

From (B.10) we can draw several qualitative conclusions. First of all it is clear that when $a_i >> a_j$, the sum of quadratic terms in i will be much larger than the sum of cross terms in i and j. This gives rise to a small r.

Influence of the difference between the angular frequencies

The impact of the difference $\omega_i - \omega_j$ can be deduced from the factors $\frac{1}{den_1}$ and num_2 in (B.10). The first factor is close to 0, unless $|\omega_i - \omega_j|$ approaches 0, since then this factor reaches its maximum. This is shown in figure B.1 for the example (3.51). In this and the following figures we choose $i = 1$, $j = 2$, $N = 301$ and take the parameters as described in (3.51), except for the parameter under consideration, which is of course variable. The factor num_2 consists of 4 terms. The first one is a constant (with respect to the angular frequencies), the second one is a low-frequency cosinus, compared to the last 2 terms. This is shown in figure B.2. The left hand side of the latter figure shows the

scenario	d_1	d_2	other parameters
old scenario	see (3.51)	see (3.51)	see (3.51)
scenario 1	$5e-3$	$5e-3$	see (3.51)
scenario 2	$-5e-3$	$-5e-3$	see (3.51)

Table B.1: *This table summarizes the different scenarios for the dampings d_1 and d_2.*

different terms of the factor num_2 (see the formula in (B.10)) as a function of ω_i, whereas the right hand side shows the sum of the different terms depicted on the left hand side, i.e. the factor num_2 is plotted. The effect of multiplying $\frac{1}{den_1}$ with num_2 is shown in figure B.3. As could be expected, the only effect of the factor num_2 is the splitting of the sharp peak produced by $\frac{1}{den_1}$ into 2 main lobes and many smaller side lobes around $|\omega_i - \omega_j| = 0$. This means that our indicator (B.10) reaches a peak when ω_i approaches ω_j, thereby increasing the importance of the sum of cross terms $\sum_{l=0}^{N-1} 2C_i(l)C_j(l)$ with respect to the corresponding sum of quadratic terms $\sum_{l=0}^{N-1} C_i^2(l)$.

Influence of the dampings

The factors num_1 and den_2 only depend on the damping d_i. The left part of figure B.4 shows num_1 as a function of d_i and the right part shows den_2 as a function of d_i. We see that both num_1 and den_2 increase with increasing absolute value of the damping.

However, den_1 and num_2 are also influenced by the specific value of the dampings d_i and d_j. To explore the latter dependence, we revisit figure B.1 and the right hand side of figure B.2 and create a new figure using 2 new scenarios for the dampings (d_i, d_j). The first scenario consists of large(r) positive values for the dampings. The second scenario consists of large(r) (in absolute value) negative values for the dampings. The new scenarios together with the example described in (3.51) are summarized in table B.1. The results are shown in figure B.5. The left hand side shows $\frac{1}{den_1}$ and the right hand side shows num_2. The full line represents the old scenario, the dotted line scenario 1 and the dashed line scenario 2. As can be deduced from the formulas for den_1 in (B.10) $\frac{1}{den_1}$ is less peaked in scenario 1 and 2, compared to the old scenario. The effect of scenario 1 and 2 on num_2 can easily be deduced from the formula for num_2 in (B.10). Scenario 1 gives a large weight (compared to the old scenario) to the last two terms in num_2, leading to a very sinusoidal looking function of high amplitude. In scenario 2, the last two terms of num_2 are of minor importance compared to the first 2 terms. This leads to the rippled (corresponding to the high-frequency terms) slowly decaying line (corresponding to the low-frequency terms) depicted on the right hand side of figure B.5.

The question that remains to be answered is: what is the overall influence of the dampings on the ratio r and on the ratio $2\frac{\sum_{l=0}^{l=N-1} C_i C_j}{\sum_{l=0}^{l=N-1} C_i^2}$? In figure B.6 we

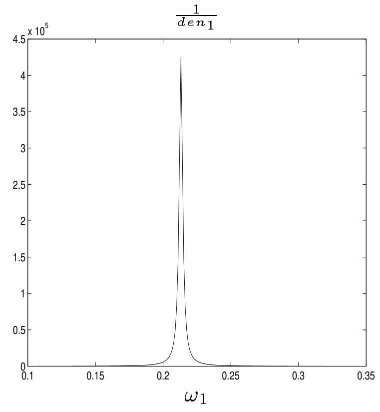


Figure B.1: *The figure illustrates the behaviour of the factor* $\frac{1}{den_1}$ *of (B.10), applied to example (3.51).*

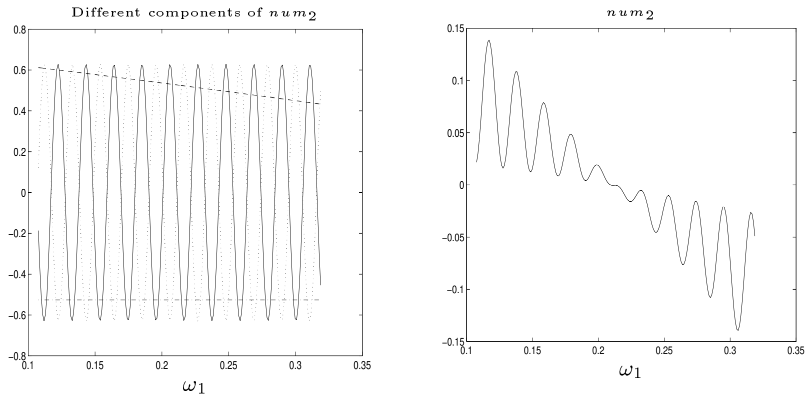


Figure B.2: *The figure on the left illustrates the different components (i.e. terms) of factor* num_2 *in formula (B.10), as a function of* ω_1. *The dash-dotted line represents the first term, the dashed line the second term, the dotted line the third term and the full line the fourth term of* num_2. *The figure on the right illustrates the sum of the terms depicted on the left, i.e. the factor* num_2 *of (B.10) as a function of* ω_1.

show the latter expression on the right hand side. The left hand side shows r. As we can see from these figures, the effect of the dampings on the ratios is mainly determined by the dependence of num_2 on the dampings. A second order expansion of the exponentials indicates that num_2 increases with increasing absolute values of the damping. E.g. positive damping values with a high absolute value (scenario 1) will lead to almost periodically changing importance of the sum of cross terms w.r.t. the sum of quadratic terms. So we

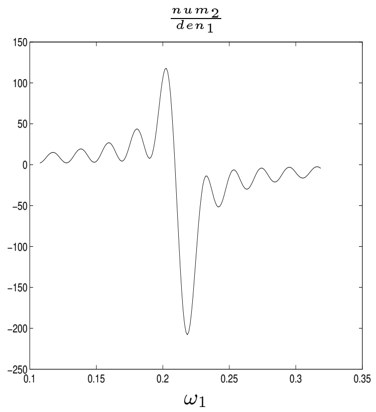


Figure B.3: *The figure shows* $\frac{num_2}{den_1}$ *of (B.10), applied to example (3.51).*

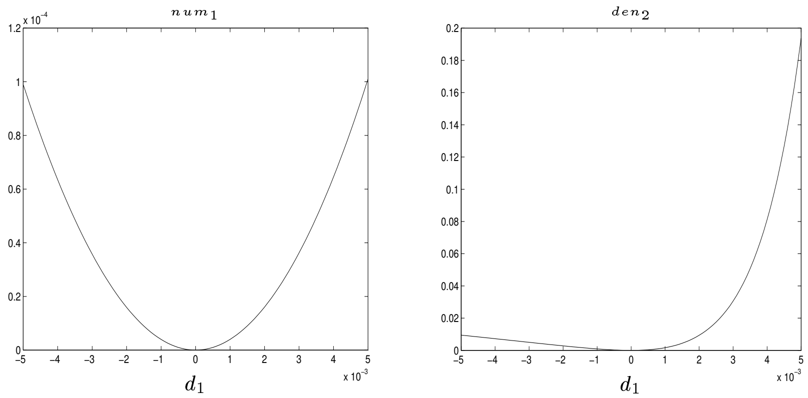


Figure B.4: *The left hand side of the figure shows the factor* num_1 *of (B.10) as a function of* d_1*, applied to example (3.51). The right hand side shows the factor* den_2 *as a function of* d_1 *applied to the same example.*

can conclude that the sum of cross terms becomes more important w.r.t. the corresponding sum of quadratic terms, when the angular frequencies of two adjacent components get closer. The higher the value of the damping, the faster the cross terms become important.
When considering more than 2 components, each sum of cross terms of adjacent components (this is a sufficient condition since as we demonstrated before, the importance of the sum of cross terms w.r.t. the corresponding sum(s) of quadratic terms increases when the angular frequencies get closer) has to be

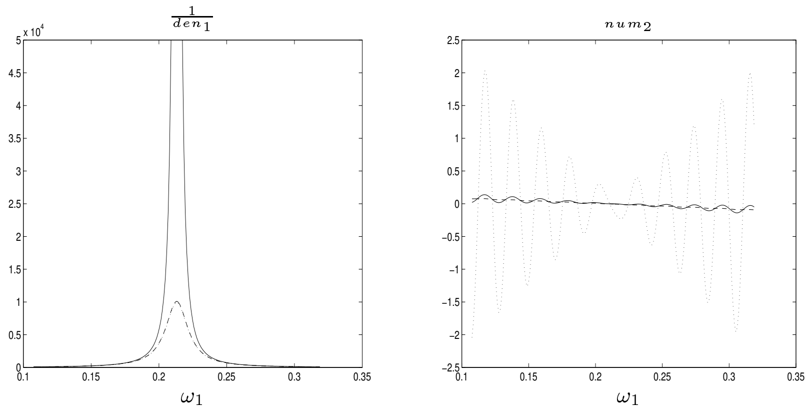


Figure B.5: *The left hand side illustrates the behaviour of the factor* $\frac{1}{den_1}$ *of (B.10) as a function of* ω_1 *for the 3 scenarios described in table B.1. The right hand side illustrates the behaviour of the factor* num_2 *of (B.10) as a function of* ω_1 *for the 3 scenarios described in table B.1. The full line represents the old scenario, the dotted line scenario* 1 *and the dashed line scenario* 2 *(note that on the left hand figure, the lines for scenario* 1 *and* 2 *almost coincide).*

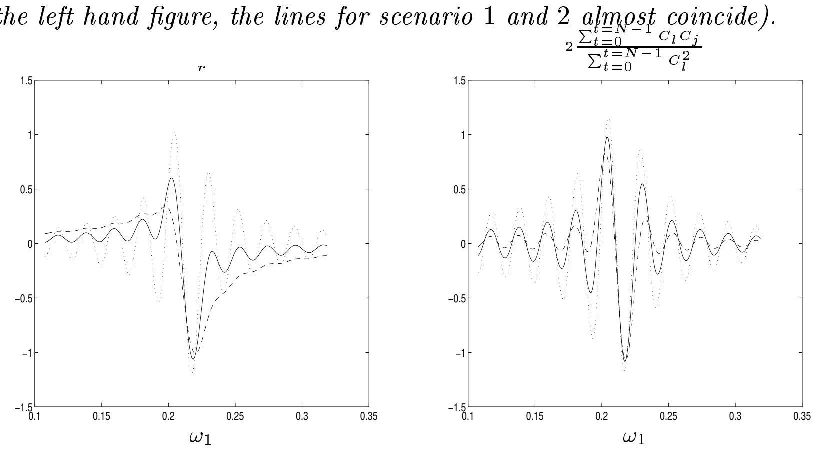


Figure B.6: *The left hand side illustrates the behaviour of the ratio* r *of (B.10) as a function of* ω_1 *for the 3 scenarios described in table B.1. The right hand side illustrates the behaviour of the ratio* $2\frac{\sum_{l=0}^{l=N-1} C_i C_j}{\sum_{l=0}^{l=N-1} C_i^2}$ *as a function of* ω_1 *for the 3 scenarios described in table B.1. The full line represents the old scenario, the dotted line scenario* 1 *and the dashed line scenario* 2.

smaller than the corresponding sums of quadratic terms.

B.2 Basic function

In this appendix we show that (3.55) depends sinusoidally on $\hat{p}$. By applying some elementary goniometry (see (B.27)) on (3.55), we can show that:

$$
\begin{aligned}
g(\hat{a},\hat{d},\hat{\omega},\hat{p}) \quad &= \quad \sum_{l=0}^{N-1} a^2 e^{2dl} sin^2(\omega l + p) &\quad \text{(B.11)}\\
&- \quad \sum_{l=0}^{N-1} a\hat{a} e^{(d+\hat{d})l} cos((\omega - \hat{\omega})l + p - \hat{p}) &\quad \text{(B.12)}\\
&+ \quad \sum_{l=0}^{N-1} a\hat{a} e^{(d+\hat{d})l} cos((\omega + \hat{\omega})l + p + \hat{p}) &\quad \text{(B.13)}\\
&+ \quad \sum_{l=0}^{N-1} 0.5\hat{a}^2 e^{2\hat{d}l} &\quad \text{(B.14)}\\
&- \quad \sum_{l=0}^{N-1} 0.5\hat{a}^2 e^{2\hat{d}l} cos(2\hat{\omega}l + 2\hat{p}). &\quad \text{(B.15)}
\end{aligned}
$$

First of all, we see that the terms (B.11) and (B.14) are of no importance here, since they do not depend on $\hat{p}$. Of all the other terms, it is term (B.12) that determines the variation of g around the constant value determined by the sum of (B.11) and (B.14). This is explained by the fact that (B.12) is a summation of a low-frequency sinus, the lobes of which are a lot wider than the lobes of a high-frequency sinus (this argument is similar to the one used in Appendix A; we therefore implicitly assume that the absolute value of the dampings is not too large with respect to the angular frequencies (see condition (B.9))). Since (B.12) is sinusoidal in $\hat{p}$ (see sum formulas (B.22) for sines), we can say that g depends sinusoidally on $\hat{p}$. Figure B.7 illustrates this dependency for the example (3.51).

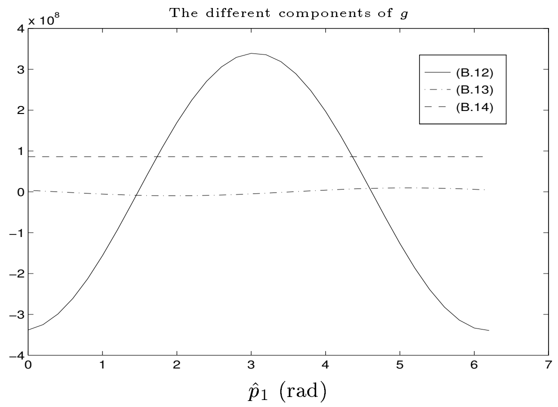


Figure B.7: *This figure shows the different components of g as a function of $\hat{p}_1$, applied to (3.51)*

B.3 Dependence on $\hat{\omega}$

In this appendix, we explore the dependence of (3.55) on $\hat{\omega}$ as a function of the parameters ω and d. Therefore we calculate $\frac{\delta g}{\delta \hat{\omega}}$ using (B.25):

$$\frac{1}{2}\frac{\delta g}{\delta \hat{\omega}} = -\sum_{l=0}^{N-1} la\hat{a}e^{(d+\hat{d})l} sin((\omega+\hat{\omega})l + p + \hat{p}) \tag{B.16}$$

$$+\sum_{l=0}^{N-1} l\hat{a}^2 e^{2\hat{d}l} sin(2(\hat{\omega}l + \hat{p})) \tag{B.17}$$

$$-\sum_{l=0}^{N-1} la\hat{a}e^{(d+\hat{d})t} sin((\omega-\hat{\omega})l + p - \hat{p}), \tag{B.18}$$

where ω is the angular frequency obtained at the end of an appropriate estimation algorithm and $\hat{\omega}$ is an estimate of ω obtained during one of the consecutive iterations. The following analysis will be for $\hat{\omega}$ near ω, since we are interested in the behaviour of (3.55) near its solution (e.g. the width of the valley containing the (local) minimum ω). We study $\frac{\delta g}{\delta \hat{\omega}}$ since the zero crossings of the latter function in the neighbourhood of $\hat{\omega} = \omega$ indicate the width of the valley that contains the solution ω. Figure B.8 illustrates the shape of the following type of functions:

$$\sum_{l=0}^{N-1} le^{dl} sin(\lambda l). \tag{B.19}$$

This shows that (B.19) only differs considerably from 0 in the neighbourhood of $\lambda = 0$. This means that for $\hat{\omega} \approx \omega$ the term (B.18) dominates both terms

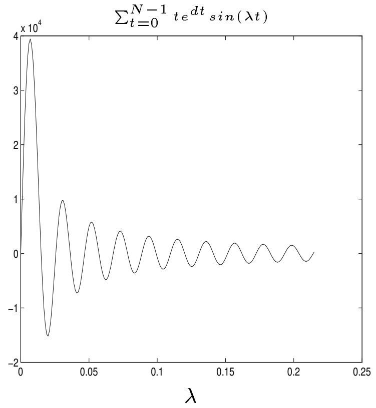


Figure B.8: *This figure shows the shape of functions of the form (B.19).*

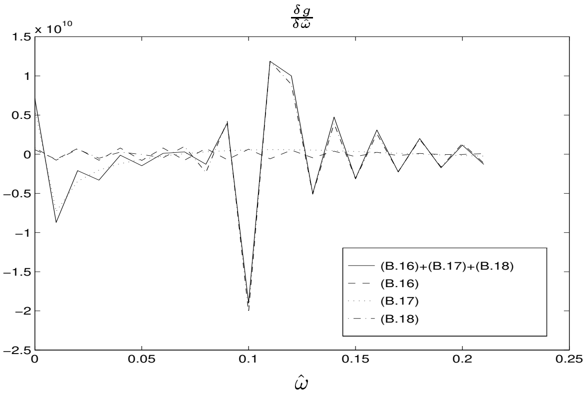


Figure B.9: *This figure shows $\frac{\delta g}{\delta \hat{\omega}}$ and its different components, applied to the first component of the example described in (3.51).*

(B.16) and (B.17). Since (B.19) is very steep near the origin, the location of the zeros of (B.18) near the origin ($\hat{\omega} \approx \omega$) is not very sensitive to the small contributions of the terms (B.16) and (B.17). Therefore the zeros of $\frac{\delta g}{\delta \hat{\omega}}$ for $\hat{\omega} \approx \omega$ can be determined by determining the zeros of (B.18). The latter is illustrated in figure B.9, where the different components of $\frac{\delta g}{\delta \hat{\omega}}$ as a function of $\hat{\omega}$ are shown. The figure shows (B.16), (B.17) and (B.18) applied to the first component of the example described in (3.51).

By using the sum formulas for geometric series (see (B.28)) we can write

	$\|d\|$ small	$\|d\|$ large
$d > 0$	case 1 ($d = 6e-4$)	case 2 ($d = 5e-3$)
$d < 0$	case 3 ($d = -6e-4$)	case 4 ($d = -5e-3$)

Table B.2: *This table summarizes the different cases studied in the analysis of* $\frac{\delta g}{\delta \hat{\omega}}$.

(B.18) as follows:

$$\sum_{l=0}^{N-1} la\hat{a}e^{(d+\hat{d})l} sin((\omega - \hat{\omega})l + p - \hat{p}) = P_1 + P_2 \qquad \text{(B.20)}$$

with

$$P_1 = \frac{(N-1)e^{\tilde{d}(N+1)} sin(\tilde{\omega}(N-1) + \tilde{p}) - (N-1)e^{(\tilde{d}N)} sin(\tilde{\omega}N + \tilde{p})}{e^{2d} - 2e^{d} cos(\tilde{\omega}) + 1}$$

$$P_2 = \frac{-e^{\tilde{d}N} sin(\tilde{\omega}N + \tilde{p}) + e^{\tilde{d}} sin(\tilde{\omega} + \tilde{p}) + 2e^{\tilde{d}(N+1)} sin(\tilde{\omega}(N-1) + \tilde{p}) - 2e^{2\tilde{d}} sin(\tilde{p})}{(e^{2d} - 2e^{d} cos(\tilde{\omega}) + 1)^2}$$

$$-\frac{e^{\tilde{d}(N+2)} sin(\tilde{\omega}(N-2) + \tilde{p}) + e^{3d} sin(\tilde{\omega} - \tilde{p})}{(e^{2d} - 2e^{d} cos(\tilde{\omega}) + 1)^2}$$

with $\tilde{\omega} \equiv \omega - \hat{\omega}, \tilde{d} \equiv d + \hat{d}, \tilde{p} = p - \hat{p}$. We consider 4 different cases depending on the size of the damping and its sign. Table B.2 enumerates the dampings used in the different cases and gives the numerical values we will use to illustrate these 4 cases. The other parameters p (amplitude, frequency and phase) are those of the first component described in (3.51), whereas their estimated counterparts $\hat{p}$ are chosen close to their final values p, since we want to study the behaviour near the solution of the problem.

Case 1 and 3

The analysis of case 1 and 3 is similar. By first order approximation (Taylor series expansion around 1), it is easy to show that near $\tilde{\omega} = 0$ (i.e. $\hat{\omega} \approx \omega$), both P_1 and P_2 are of similar magnitude but opposite sign. This can be seen in figure B.10 where we illustrate the 4 different cases. The full line corresponds to (B.20)=$P_1 + P_2$, the dashed line shows P_1 and the dotted line represents P_2. When $\hat{\omega}$ departs from ω (i.e. $\tilde{\omega}$ departs from 0), P_1 becomes the most important part of (B.20). This can easily be understood by looking at the denominators of P_1 and P_2. From appendix A, we know that a denominator like in P_1, will cause a large peak around $\tilde{\omega} = 0$. Since the denominator of P_2 is the square of the denominator in P_1, the peak caused by the denominator of P_2 will even be larger. As indicated before, P_1 and P_2 are of similar magnitude around $\tilde{\omega} = 0$. Therefore it is obvious that due to the difference in denominators, P_2 will rapidly decrease compared to P_1, once we depart from $\tilde{\omega} = 0$. Therefore, to find the zero-crossings of $\frac{\delta g}{\delta \tilde{\omega}}$ (in cases 1 and 3), it suffices to find the roots of the numerator of P_1, by looking for the roots of

$$e^{\tilde{d}} sin(\tilde{\omega}(N-1)+\tilde{p}) - sin(\tilde{\omega}N+\tilde{p}). \tag{B.21}$$

In figure B.11 we show (B.21) for cases 1, 2 and 3. From the latter figure we clearly see that by increasing the damping from a small negative value (case 3) to a small positive value (case 1), the width of the valley containing the global minimum decreases.

Case 2 and 4

First of all we mention that due to the large (absolute) values of the dampings, a first order Taylor series expansion around $\tilde{\omega}=0$ is no longer valid, explaining the different (as compared to cases 1 and 3) behaviour of $\frac{\delta g}{\delta \tilde{\omega}}$ in case 2 and 4 in the region $\tilde{\omega} \approx 0$. We treat both cases together since the denominator of P_1 (and thus the same goes for P_2) is of similar magnitude in both cases. However, as can be seen from figure B.10, in case 2 P_1 dominates, whereas in case 4 P_2 dominates. Hence, the difference between the 2 cases lies in the numerators of P_1 and P_2. Indeed, graphical analysis has shown that the importance of the numerator of P_1 grows when switching from a large (in absolute value) negative damping towards a large positive damping, explaining the differences between cases 2 and 4 in figure B.10. From figure B.11, we see that an even larger positive damping leads to a smaller valley. Notice that case 4 is not indicated in the latter figure since P_2 is the major term in case 4. However, from figure B.10, it is easy to see that large (in absolute value) negative dampings lead to very wide valleys containing the global minimum (just look at the zero-crossings of (B.20)).

We can conclude by saying that increasing the damping of a component from large (in absolute value) negative dampings towards large positive dampings yields a (local) minimum located in a valley that is getting narrower.

B.4 Some formulas

$$\sum_{t=0}^{N-1} ae^{dt}cos(\omega t+\phi) = \tag{B.22}$$
$$a\frac{cos(\phi) - e^{d}cos(\phi-\omega) - e^{dN}cos(\omega N+\phi) + e^{d(N+1)}cos(\omega(N-1)+\phi)}{1-2e^{d}cos(\omega)+e^{2d}}$$

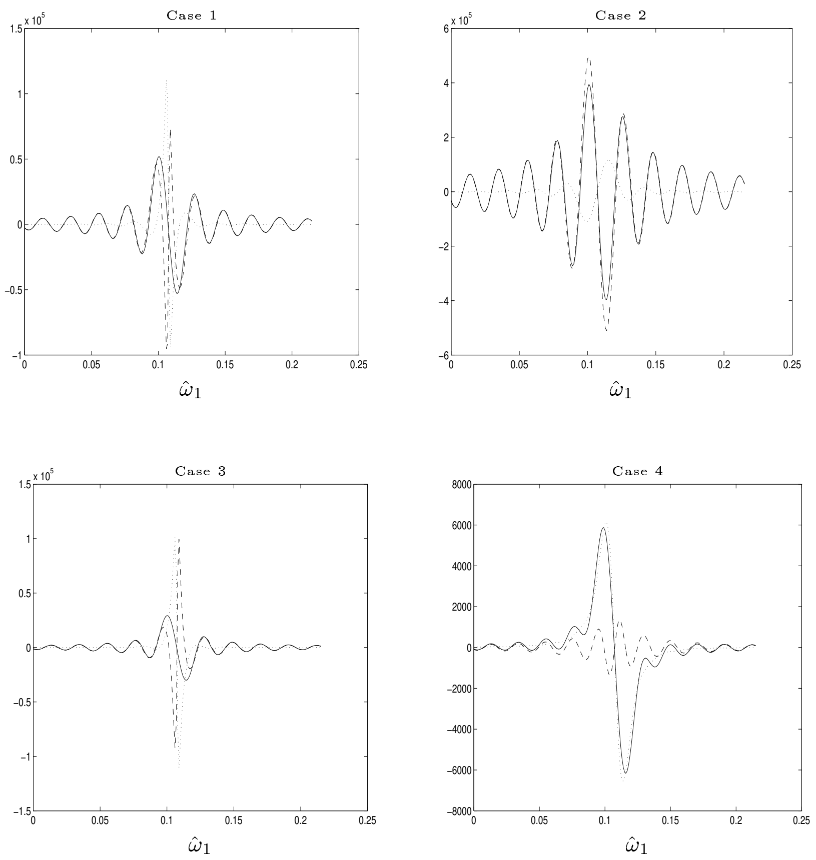


Figure B.10: *This figure illustrates (B.20) (full line) and its different components P_1 (dashed line) and P_2 (dotted line) as a function of $\hat{\omega}_1$ for the 4 different cases described in table B.2.*

$$\begin{aligned}
sin(x \pm y) &= sin(x)cos(y) \pm sin(y)cos(x) && \text{(B.23)}\\
cos(x \pm y) &= cos(x)cos(y) \mp sin(x)sin(y) && \text{(B.24)}\\
sin(x) \pm sin(y) &= 2sin[\frac{1}{2}(x \pm y)]cos[\frac{1}{2}(x \mp y)] && \text{(B.25)}\\
cos(x) + cos(y) &= 2cos[\frac{1}{2}(x + y)]cos[\frac{1}{2}(x - y)] && \text{(B.26)}\\
cos(x) - cos(y) &= 2sin[\frac{1}{2}(x + y)]sin[\frac{1}{2}(y - x)] && \text{(B.27)}
\end{aligned}$$

$$\sum_{t=0}^{N-1} te^{(j\omega+d)t} = \frac{(N-1)e^{(j\omega+d)N}}{e^{(j\omega+d)}-1} - \frac{e^{(j\omega+d)}(e^{(j\omega+d)(N-1)}-1)}{(e^{(j\omega+d)}-1)^2} \tag{B.28}$$

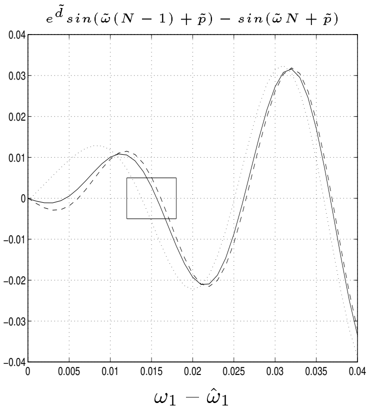


Figure B.11: *This figure shows (B.21) as a function of* $\tilde{\omega}_1 \equiv \omega_1 - \hat{\omega}_1$ *for case* 1 *(full line), case* 2 *(dotted line) and case* 3 *(dashed line). The amplitude, phase and frequency are those of the first component of the example described in (3.51). The area indicated by the small rectangle shows the region containing the zero-crossings that determine the width of the valley of (3.55).*

Appendix C

Displacement rank

C.1 Principles of low displacement rank algorithms

Let $A \in \mathbb{R}^{n\times n}$ be a symmetric positive definite matrix with displacement representation

$$A = \sum_{i=1}^{m} L(x_i)L(x_i)^T - \sum_{i=m+1}^{p} L(x_i)L(x_i)^T, \quad p \leq n, \tag{C.1}$$

where $x_i \in \mathbb{R}^n, i = 1, \dots, p$. We can write (C.1) in the following way

$$\begin{aligned} A &= [L(x_1), \dots L(x_m), L(x_{m+1}), \dots, L(x_p)] \begin{bmatrix} I & & & & & \\ & \ddots & & & & \\ & & I & & & \\ & & & -I & & \\ & & & & \ddots & \\ & & & & & -I \end{bmatrix} \begin{bmatrix} L(x_1)^T \\ \vdots \\ L(x_m)^T \\ L(x_{m+1})^T \\ \vdots \\ L(x_p)^T \end{bmatrix} \\ &= [L(x_1), \dots L(x_m), L(x_{m+1}), \dots, L(x_p)] J \begin{bmatrix} L(x_1)^T \\ \vdots \\ L(x_m)^T \\ L(x_{m+1})^T \\ \vdots \\ L(x_p)^T \end{bmatrix} \end{aligned}$$

where I is the identity matrix of order n. We say that a matrix Q is J–orthogonal if $J = QJQ^T$. To compute the Cholesky factor of A, we have to construct a J–orthogonal matrix Q such that

$$Q \begin{bmatrix} L(x_1)^T \\ \vdots \\ L(x_m)^T \\ L(x_{m+1})^T \\ \vdots \\ L(x_p)^T \end{bmatrix} = \begin{bmatrix} R \\ O \\ \vdots \\ \vdots \\ O \end{bmatrix},$$

where $R, O \in \mathbb{R}^{n \times n}$, R is upper triangular and O is the null matrix.

As an example, we describe briefly how to obtain the matrix R in case $A \in \mathbb{R}^{5\times 5}$ and its displacement representation is given by:

$$A = L(x_1)L(x_1)^T + L(x_2)L(x_2)^T - L(x_3)L(x_3)^T - L(x_4)L(x_4)^T.$$

Then

$$J = \begin{bmatrix} I & & & \\ & I & & \\ & & -I & \\ & & & -I \end{bmatrix},$$

where I is the identity matrix of order 5, and

$$L = \begin{bmatrix}
x_1(1) & x_1(2) & x_1(3) & x_1(4) & x_1(5) \\
 & x_1(1) & x_1(2) & x_1(3) & x_1(4) \\
 & & x_1(1) & x_1(2) & x_1(3) \\
 & & & x_1(1) & x_1(2) \\
 & & & & x_1(1) \\
x_2(1) & x_2(2) & x_2(3) & x_2(4) & x_2(5) \\
 & x_2(1) & x_2(2) & x_2(3) & x_2(4) \\
 & & x_2(1) & x_2(2) & x_2(3) \\
 & & & x_2(1) & x_2(2) \\
 & & & & x_2(1) \\
x_3(1) & x_3(2) & x_3(3) & x_3(4) & x_3(5) \\
 & x_3(1) & x_3(2) & x_3(3) & x_3(4) \\
 & & x_3(1) & x_3(2) & x_3(3) \\
 & & & x_3(1) & x_3(2) \\
 & & & & x_3(1) \\
x_4(1) & x_4(2) & x_4(3) & x_4(4) & x_4(5) \\
 & x_4(1) & x_4(2) & x_4(3) & x_4(4) \\
 & & x_4(1) & x_4(2) & x_4(3) \\
 & & & x_4(1) & x_4(2) \\
 & & & & x_4(1)
\end{bmatrix}. \tag{C.2}$$

Denote by $G_{i,j}, H_{i,j}$ the Givens and hyperbolic rotation of order 20 respectively, where $G_{i,j}, H_{i,j}$ are the identity matrices except for the following 4 entries

$$\begin{array}{l} G_{i,j}(i,i) = G_{i,j}(j,j) = c;\ G_{i,j}(i,j) = s;\ G_{i,j}(j,i) = -s; \\ H_{i,j}(i,i) = H_{i,j}(j,j) = c;\ H_{i,j}(i,j) = -s;\ H_{i,j}(j,i) = -s; \end{array}$$

the following matrices are J–orthogonal:

$$\begin{array}{ll} G_{i,j}, & 1 \leq i,j \leq 10, \text{or} \;\; 11 \leq i,j \leq 20 \\ H_{i,j}, & 1 \leq i \leq 10, \text{and} \;\; 11 \leq j \leq 20. \end{array}$$

Now we describe briefly how to annihilate the matrices $L(x_i)^T,\ i = 2,\ldots,4$, to obtain R. At the first step, we consider the Givens rotation $G_{1,6}$ chosen to annihilate the entries $x_2(1)$ of L. We can construct the Givens matrices $G_{i,i+5}$, $i = 2,\ldots,5$, such that the multiplication of these matrices with L annihilates the diagonal elements of the matrix $L(x_2)^T$ without any computation, because of the block Toeplitz structure of L. We remark that these matrices are J–orthogonal. Let $Q_{1,2} = \prod_{i=1}^{5} G_{i,i+5}$. Then

$$Q_{1,2}L = \begin{bmatrix}
\tilde{x}_1(1) & \tilde{x}_1(2) & \tilde{x}_1(3) & \tilde{x}_1(4) & \tilde{x}_1(5) \\
 & \tilde{x}_1(1) & \tilde{x}_1(2) & \tilde{x}_1(3) & \tilde{x}_1(4) \\
 & & \tilde{x}_1(1) & \tilde{x}_1(2) & \tilde{x}_1(3) \\
 & & & \tilde{x}_1(1) & \tilde{x}_1(2) \\
 & & & & \tilde{x}_1(1) \\
0 & \tilde{x}_2(2) & \tilde{x}_2(3) & \tilde{x}_2(4) & \tilde{x}_2(5) \\
 & 0 & \tilde{x}_2(2) & \tilde{x}_2(3) & \tilde{x}_2(4) \\
 & & 0 & \tilde{x}_2(2) & \tilde{x}_2(3) \\
 & & & 0 & \tilde{x}_2(2) \\
 & & & & 0 \\
x_3(1) & x_3(2) & x_3(3) & x_3(4) & x_3(5) \\
 & x_3(1) & x_3(2) & x_3(3) & x_3(4) \\
 & & x_3(1) & x_3(2) & x_3(3) \\
 & & & x_3(1) & x_3(2) \\
 & & & & x_3(1) \\
x_4(1) & x_4(2) & x_4(3) & x_4(4) & x_4(5) \\
 & x_4(1) & x_4(2) & x_4(3) & x_4(4) \\
 & & x_4(1) & x_4(2) & x_4(3) \\
 & & & x_4(1) & x_4(2) \\
 & & & & x_4(1)
\end{bmatrix}.$$

We call the multiplication of $Q_{1,2}$ times L an *update* between $L(x_1)$ and $L(x_2)$. Then we set $L = Q_{1,2}L$. We remark that the newly computed matrices $L(x_1)$ and $L(x_2)$ continue to have the Toeplitz structure. To compute this update it is sufficient to update the vectors x_1 and x_2 (instead of $L(x_1)$ and $L(x_2)$),

$$\begin{pmatrix} \tilde{x}_1^T \\ \tilde{x}_2^T \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} {x_1}^T \\ {x_2}^T \end{pmatrix},$$

where c and s are the same "Givens coefficients" of the matrix $G_{1,6}$. In the same way, we compute the Givens rotation $G_{11,16}$ chosen to annihilate the $(16,1)$ entry of the new L and the corresponding Givens matrices $G_{i+10,i+15}$, $i = 2, \ldots, 5$. Let $Q_{3,4} = \prod_{i=1}^{5} G_{i+10,i+15}$. Then

$$Q_{3,4}L = \left[\begin{array}{ccccc} \tilde{x}_1(1) & \tilde{x}_1(2) & \tilde{x}_1(3) & \tilde{x}_1(4) & \tilde{x}_1(5) \\ & \tilde{x}_1(1) & \tilde{x}_1(2) & \tilde{x}_1(3) & \tilde{x}_1(4) \\ & & \tilde{x}_1(1) & \tilde{x}_1(2) & \tilde{x}_1(3) \\ & & & \tilde{x}_1(1) & \tilde{x}_1(2) \\ & & & & \tilde{x}_1(1) \\ 0 & \tilde{x}_2(2) & \tilde{x}_2(3) & \tilde{x}_2(4) & \tilde{x}_2(5) \\ & 0 & \tilde{x}_2(2) & \tilde{x}_2(3) & \tilde{x}_2(4) \\ & & 0 & \tilde{x}_2(2) & \tilde{x}_2(3) \\ & & & 0 & \tilde{x}_2(2) \\ & & & & 0 \\ \tilde{x}_3(1) & \tilde{x}_3(2) & \tilde{x}_3(3) & \tilde{x}_3(4) & \tilde{x}_3(5) \\ & \tilde{x}_3(1) & \tilde{x}_3(2) & \tilde{x}_3(3) & \tilde{x}_3(4) \\ & & \tilde{x}_3(1) & \tilde{x}_3(2) & \tilde{x}_3(3) \\ & & & \tilde{x}_3(1) & \tilde{x}_3(2) \\ & & & & \tilde{x}_3(1) \\ 0 & \tilde{x}_4(2) & \tilde{x}_4(3) & \tilde{x}_4(4) & \tilde{x}_4(5) \\ & 0 & \tilde{x}_4(2) & \tilde{x}_4(3) & \tilde{x}_4(4) \\ & & 0 & \tilde{x}_4(2) & \tilde{x}_4(3) \\ & & & 0 & \tilde{x}_4(2) \\ & & & & 0 \end{array} \right]. \tag{C.3}$$

We call the multiplication of $Q_{3,4}$ times L an update between $L(x_3)$ and $L(x_4)$ and define $L = Q_{3,4}L$. Also in this case, because of the Toeplitz structure of these matrices, in order to compute the new L it is sufficient to update the vectors x_3 and x_4 only, i.e. $\begin{pmatrix} \tilde{x}_3^T \\ \tilde{x}_4^T \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} {x_3}^T \\ {x_4}^T \end{pmatrix}$, where c and s are the same "Givens coefficients" of the matrix $G_{11,16}$. To complete an iteration we consider the hyperbolic rotaton $H_{1,11}$ constructed to annihilate the $(11,1)$ entry of the matrix (C.3). Without any computation we obtain from $H_{1,11}$ also the associated hyperbolic rotations $H_{i,i+10}$, $i = 2, \ldots, 5$. Let

$S_{1,3} = \prod_{i=1}^{5} H_{i,i+10}$. Then

$$S_{1,3}L = \begin{bmatrix} \hat{x}_1(1) & \hat{x}_1(2) & \hat{x}_1(3) & \hat{x}_1(4) & \hat{x}_1(5) \\ & \hat{x}_1(1) & \hat{x}_1(2) & \hat{x}_1(3) & \hat{x}_1(4) \\ & & \hat{x}_1(1) & \hat{x}_1(2) & \hat{x}_1(3) \\ & & & \hat{x}_1(1) & \hat{x}_1(2) \\ & & & & \hat{x}_1(1) \\ 0 & \tilde{x}_2(2) & \tilde{x}_2(3) & \tilde{x}_2(4) & \tilde{x}_2(5) \\ & 0 & \tilde{x}_2(2) & \tilde{x}_2(3) & \tilde{x}_2(4) \\ & & 0 & \tilde{x}_2(2) & \tilde{x}_2(3) \\ & & & 0 & \tilde{x}_2(2) \\ & & & & 0 \\ 0 & \hat{x}_3(2) & \hat{x}_3(3) & \hat{x}_3(4) & \hat{x}_3(5) \\ & 0 & \hat{x}_3(2) & \hat{x}_3(3) & \hat{x}_3(4) \\ & & 0 & \hat{x}_3(2) & \hat{x}_3(3) \\ & & & 0 & \hat{x}_3(2) \\ & & & & 0 \\ 0 & \tilde{x}_4(2) & \tilde{x}_4(3) & \tilde{x}_4(4) & \tilde{x}_4(5) \\ & 0 & \tilde{x}_4(2) & \tilde{x}_4(3) & \tilde{x}_4(4) \\ & & 0 & \tilde{x}_4(2) & \tilde{x}_4(3) \\ & & & 0 & \tilde{x}_4(2) \\ & & & & 0 \end{bmatrix}. \tag{C.4}$$

We observe that the matrix $S_{1,3}Q_{3,4}Q_{1,2}$ is J–orthogonal. We call the multiplication of $S_{1,3}$ times L a *downdate* between $L(x_1)$ and $L(x_3)$. Also in this case, since these matrices have a Toeplitz structure, it is sufficient to downdate the vectors x_1 and x_3 in order to compute the multiplication $S_{1,3}L$,

$$\begin{pmatrix} \hat{x}_1^T \\ \hat{x}_3^T \end{pmatrix} = \begin{pmatrix} c & -s \\ -s & c \end{pmatrix} \begin{pmatrix} \tilde{x}_1^T \\ \tilde{x}_3^T \end{pmatrix},$$

where c and s are the same "hyperbolic coefficients" of the matrix $H_{1,11}$. Then, the first row of (C.4) is the first row of R. To compute the other rows of R, we apply the previous procedure to the matrix $L(2:20, 2:5)$.

C.2 Matlab-like code for algorithm of section 5.2.2

The algorithm is summarized in the function `FTriang`. `generator` is a Matlab function that, given the matrix M, computes the generator $x_i, i = 1\ldots, 5$ of M^TM. `givens` and `hyp` compute the Givens and hyperbolic rotation, respectively. The variables t_1, t_2, t_3, t_4 are temporary variables like *temp* in which the partial product (5.12) is stored.

`function`$[R]$=`FTriang`(M, m, n)

$(x_1, x_2, x_3, x_4, x_5) = \texttt{generator}(M);$
$temp = 1; mn1 = m + 2n - 1; mn2 = m + 2n - 2;$
% Initialization
$(c, s) = \texttt{givens}(x_1(1), x_2(1));$
$$\begin{pmatrix} x_1^T([1:n, m+n:mn1]) \\ x_2^T([1:n, m+n:mn1]) \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_1^T([1:n, m+n:mn1]) \\ x_2^T([1:n, m+n:mn1]) \end{pmatrix}$$
$R(1, 1:mn1) = x_1^T;\ x_1^T(2:mn1) = x_1^T(1:mn2); x_1(m+n) = 0;$
% Phase 1
$\texttt{for}\ i = 2:m,$
  $(c, s) = \texttt{givens}(x_1(i), x_2(i));$
$$\begin{pmatrix} x_1^T([i:n+i-1, m+n:mn1]) \\ x_2^T([i:n+i-1, m+n:mn1]) \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_1^T([i:n+i-1, m+n:mn1]) \\ x_2^T([i:n+i-1, m+n:mn1]) \end{pmatrix}$$
  $t_3 = x_3(n+m); t_4 = x_4(n+m);$
  $(c, s) = \texttt{givens}(x_1(i), x_3(i));$
$$\begin{aligned} x_3^T([i+1:n+i-1, m+n:mn1]) = \ & -s x_1^T([i+1:n+i-1, m+n:mn1]) \\ & + c x_3^T([i+1:n+i-1, m+n:mn1]); \end{aligned}$$
  $x_1(m+n) = x_1(m+n) + s(t_3 - t_4)/c;$
  $x_4(m+n) = -s x_1(m+n) + c t_4;$
  $temp = temp \times c;$
  $\texttt{if}\ \ i < m,$
    $x_3(n+i) = temp \times x_3(n+i);$
  $\texttt{end}$
  $R(i, i:mn1) = x_1(i:mn1);$
  $x_1(i+1:mn1) = x_1(i:mn2);\ x_1(m+n) = 0;$
$\texttt{end}$
% Phase 2
$\texttt{for}\ i = m+1 : m+n-1,$
  $(c, s) = \texttt{givens}(x_1(i), x_2(i));$
$$\begin{pmatrix} x_1^T(i:mn1) \\ x_2^T(i:mn1) \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_1^T(i:mn1) \\ x_2^T(i:mn1) \end{pmatrix}$$
  $t_3 = x_3(n+m); t_4 = x_4(n+m);$
  $(c, s) = \texttt{givens}(x_1(i), x_3(i));$
  $x_3^T(i+1:mn1) = -s x_1^T(i+1:mn1) + c x_3^T(i+1:mn1);$
  $x_1(m+n) = x_1(m+n) + s(t_3 - t_4)/c;$
  $x_4(m+n) = -s x_1(m+n) + c t_4;$
  $s = x_5(i)/x_1(i);$
  $c_1 = \sqrt{x_1(i)^2 - x_5(i)^2};$
  $c = c_1/x_1(i);$
  $x_1^T(i+1:m+n) = \left(x_1^T(i+1:m+n) - s x_5^T(i+1:m+n)\right)/c;$
  $x_5^T(i+1:m+n) = -s x_1^T(i+1:m+n) + c x_5^T(i+1:m+n);$
  $x_1(i) = c_1;$
  $R(i, i:mn1) = x_1^T(i:mn1);\ x_1^T(i+1:mn1) = x_1^T(i:mn2);\ x_1(m+n) = 0;$
$\texttt{end}$
$x_4(n+m) = t_4; x_4(m+n+1:mn1) = x_3(m+n+1:mn1);$
% Phase 3

for $i = m+n : mn1$,
$(c, s) = \texttt{givens}(x_1(i), x_2(i));$
$$\begin{pmatrix} x_1^T(i:mn1) \\ x_2^T(i:mn1) \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_1^T(i:mn1) \\ x_2^T(i:mn1) \end{pmatrix}$$
$(c, s) = \texttt{givens}(x_1(i), x_3(i));$
$$\begin{pmatrix} x_1^T(i:mn1) \\ x_3^T(i:mn1) \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_1^T(i:mn1) \\ x_3^T(i:mn1) \end{pmatrix}$$
$(c, s) = \texttt{givens}(x_5(i), x_4(i));$
$$\begin{pmatrix} x_5^T(i:mn1) \\ x_4^T(i:mn1) \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_5^T(i:mn1) \\ x_4^T(i:mn1) \end{pmatrix}$$
$s = x_5(i)/x_1(i);$
$c_1 = \sqrt{x_1(i)^2 - x_5(i)^2};$
$c = c_1/x_1(i);$
$x_1^T(i+1:m+n) = \left(x_1^T(i+1:m+n) - s x_5^T(i+1:m+n)\right)/c;$
$x_5^T(i+1:m+n) = -s x_1^T(i+1:m+n) + c x_5^T(i+1:m+n);$
$x_1(i) = c_1;$
$R(i, i:mn1) = x_1^T(i:mn1);\ x_1^T(i+1:mn1) = x_1^T(i:mn2);$
end

C.3 Matlab-like code for algorithm of section 5.3

$\texttt{function}[R]\texttt{=Ftriang2}(M, m, n)$
$(x_1, x_2, x_3, x_4, x_5) = \texttt{generator2}(M);$
% Initialization
$temp = 1;$
$R(1, 1:m+n) = x_1^T;\ x_1^T(2:m+n) = x_1^T(1:m+n-1);\ x_1(m+1) = 0;$
% Phase 1
for $i = 2 : m-n+1$,
$t_2 = x_2(m+1);\ t_3 = x_3(m+1);$
$(c, s) = \texttt{givens}(x_1(i), x_2(i));$
$$x_2^T([i+1:n+i-1, m+1:m+n]) = -s x_1^T([i+1:n+i-1, m+1:m+n]) + c x_2^T([i+1:n+i-1, m+1:m+n])$$
$x_1(m+1) = s * (t_2 - t_3)/c;$
$x_3(m+1) = c * t_3 - s * x_1(m+1);$
$temp = temp * c;$
if $i < m-n+1$,
$x_2(n+i) = temp * x_2(n+i);$
end
$s = x_4(i)/x_1(i);$
$c_1 = \sqrt{x_1(i)^2 - x_4(i)^2};$
$c = c_1/x_1(i);$

$$x_1^T([i+1:n+i-1,m+1:m+n]) = \left(x_1^T([i+1:n+i-1,m+1:m+n]) - sx_4^T([i+1:n+i-1,m+1:m+n])\right)/c;$$
$$x_4^T([i+1:n+i-1,m+1:m+n]) = -sx_1^T([i+1:n+i-1,m+1:m+n]) + cx_4^T([i+1:n+i-1,m+1:m+n]);$$

$x_1(i) = c_1;$
$R(i, i:m+n) = x_1(i:m+n)^T;$
$x_1(i+1:m+n) = x_1(i:m+n-1);\ x_1(m+1) = 0;$

`end`

% Phase 2

`for` $i = m-n+2:m,$

$t_2 = x_2(m+1);\ t_3 = x_3(m+1);$
$(c,s) = \mathtt{givens}(x_1(i), x_2(i));$
$x_2^T(i+1:m+n) = -sx_1^T(i+1:m+n) + cx_2^T(i+1:m+n);$
$x_1(m+1) = x_1(m+1) + s*(t_2 - t_3)/c;$
$x_3(m+1) = c*t_3 - s*x_1(m+1);$
$(c,s) = \mathtt{givens}(x_4(i), x_5(i));$

$$\begin{pmatrix} x_4^T(i:m+n) \\ x_5^T(i:m+n) \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_4^T(i:m+n) \\ x_5^T(i:m+n) \end{pmatrix}$$

$s = x_4(i)/x_1(i);$
$c_1 = \sqrt{x_1(i)^2 - x_4(i)^2};$
$c = c_1/x_1(i);$
$x_1^T(i+1:m+n) = \left(x_1^T(i+1:m+n) + -sx_4^T(i+1:m+n)\right)/c;$
$x_4^T(i+1:m+n) = -sx_1^T(i+1:m+n) + cx_4^T(i+1:m+n);$
$x_1(i) = c_1;$
$R(i, i:m+n) = x_1(i:m+n)^T;$
$x_1(i+1:m+n) = x_1(i:m+n-1); x_1(m+1) = 0;$

`end`

$x_3(m+2:m+n) = x_2(m+2:m+n);$

% Phase 3

`for` $i = m+1:m+n,$

$(c,s) = \mathtt{givens}(x_1(i), x_2(i));$

$$\begin{pmatrix} x_1^T(i:m+n) \\ x_2^T(i:m+n) \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_1^T(i:m+n) \\ x_2^T(i:m+n) \end{pmatrix}$$

$(c,s) = \mathtt{givens}(x_4(i), x_3(i));$

$$\begin{pmatrix} x_4^T(i:m+n) \\ x_3^T(i:m+n) \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_4^T(i:m+n) \\ x_3^T(i:m+n) \end{pmatrix}$$

$(c,s) = \mathtt{givens}(x_4(i), x_5(i));$

$$\begin{pmatrix} x_4^T(i:m+n) \\ x_5^T(i:m+n) \end{pmatrix} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} x_4^T(i:m+n) \\ x_5^T(i:m+n) \end{pmatrix}$$

$s = x_4(i)/x_1(i);$
$c_1 = \sqrt{x_1(i)^2 - x_4(i)^2};$
$c = c_1/x_1(i);$
$x_1^T(i+1:m+n) = \left(x_1^T(i+1:m+n) + -sx_4^T(i+1:m+n)\right)/c;$
$x_4^T(i+1:m+n) = -sx_1^T(i+1:m+n) + +cx_4^T(i+1:m+n);$
$x_1(i) = c_1;$

$R(i, i : m+n) = x_1(i : m+n)^T$; $x_1(i+1 : m+n) = x_1(i : m+n-1)$;
```
end
```

Bibliography

[1] T. J. Abatzoglou and R. H. Hackman. Constrained total least squares based extraction of acoustic resonances of elastic targets. In *Proceedings of IEEE International Conf. on Acoustics, Speech & Signal Processing*, pages 3385–3388, Toronto, Ontario, Canada, may 1991.

[2] T. J. Abatzoglou and J. M. Mendel. Constrained total least squares. In *Proceedings of IEEE International Conf. on Acoustics, Speech & Signal Processing*, pages 1485–1488, Dallas, 1987.

[3] T. J. Abatzoglou, J. M. Mendel, and G. A. Harada. The constrained total least squares technique and its applications to harmonic superresolution. *IEEE Transactions on Signal Processing*, 39:1070–1086, 1991.

[4] T.J. Abatzoglou and L.K. Lam. Direction finding using uniform arrays and the constrained total least squares method. In San Jose State University Ray R. Chen, editor, *Proceedings of IEEE International Conf. on Acoustics, Speech & Signal Processing*, pages 573–578, Pacific Grove, California, November 1991. IEEE Computer Society Press.

[5] T.J. Abatzoglou and B.F. Rice. Application of ctls for estimating the direction of arrival of cyclostationary signals. In San Jose State University Ray R. Chen, editor, *Proceedings of 25th Asilomar Conference on Signals, Systems & Computers*, pages 247–251, Pacific Grove, California, November 1991. IEEE Computer Society Press.

[6] R. J. Adcock. A problem in least squares. *The Analyst*, 5:53–54, 1878.

[7] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide.* SIAM, Philadelphia, 1995.

[8] M.T. Bajen, R. Puchal, A. Gonzalez, J.M. Grinyo, A. Castelao, J. Mora, and J. M. Comin. Mag3 renogram deconvolution in kidney transplantation: utility of the measurement of initial tracer uptake. *J. Nucl. Med.*, 38(8):1295–1299, August 1997.

[9] I. Barrodale and F. Roberts. An improved algorithm for discrete l_1 linear approximation. *SIAM J. Num. Anal.*, 10:839–848, 1972.

[10] R.H. Bartels, A.R. Conn, and J.W. Sinclair. Minimization techniques for piecewise differentiable functions: the l_1 solution to an overdetermined linear system. *SIAM J. Numer. Anal.*, 15:224–240, 1978.

[11] S. G. Bartels. *Totale Approximationsprobleme.* PhD thesis, Fakultät für Wirtschaftswissenschaften der Universität Fridericiana zu Karlsruhe, 1995.

[12] S. G. Bartels. The structured total approximation problem. Technical report, Technische Universität Berlin, 1996. submitted to SIAM J. Matrix Anal. Appl.

[13] M. Berry, S. Dumais, and G.O. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37(4):573–595, 1995.

[14] D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, Belmont, Massachusetts, 1995.

[15] K. Blom. Information retrieval using the singular value decomposition and Krylov subspaces. Technical report, Department of Mathematics, Chalmers University of Technology, Göteborg, Sweden, 1999.

[16] D. Boss and K. D. Kammeyer. Blind estimation of ARMA systems. In *Proceedings of EUSIPCO-94*, volume III, pages 1105–1108, Edinburgh, September 1994.

[17] Y. Bresler and A. Macovski. Exact maximum likelihood parameter estimation of superimposed exponential signals in noise. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 34(5):1081–1089, October 1986.

[18] J. A. Cadzow. Signal enhancement: A composite property mapping algorithm. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 36(1):49–62, January 1988.

[19] J. P. Campbell Jr., T. E. Tremain, and V. C. Welch. The federal standard 1016 4800 bps celp voice coder. *Digital Signal Processing, Academic Press*, 1(3):145–155, 1991.

[20] A. Capderou, D. Douguet, T. Similowski, A. Aurengo, and M. Zelter. Non-invasive assessment of technetium-99m albumin transit time distribution in the pulmonary circulation by first-pass angiocardiography. *Eur. J. Nucl. Med.*, 24(7):745–753, July 1997.

[21] M. Cedervall. *Subspace and Maximum Likelihood estimation for systems and signals modelling.* PhD thesis, School of Engineering, Systems and control group, Uppsala University, 1997.

[22] H. Chen, S. Van Huffel, A. van den Boom, and Paul van den Bosch. Extended htls methods for parameter estimation of multiple data sets modeled as sums of exponentials. In *Proc. of the 13th International Conference on Digital Signal Processing*, pages 1035–1038, Santorini, Greece, July 1997.

[23] H. Chen, S. Van Huffel, and D. Van Ormondt. Application of the structured total least norm technique in spectral estimation. In *Proc. of the 8th European Signal Processing Conference (EUSIPCO'96)*, pages 706–709, Trieste, Italy, September 1996.

[24] H. Chen, S. Van Huffel, and J. Vandewalle. Exponential data fitting using the structured total least norm technique. Technical Report 95-18, ESAT-SISTA, Dept. of Electr. Eng., Katholieke Universiteit Leuven, Belgium, 1995.

[25] J. Chun, T. Kailath, and H. Lev-ari. Fast parallel algorithms for qr and triangular factorization. *SIAM J. Sci. and Stat. Comp.*, 8(6):899–913, 1987.

[26] G. Cirrincione. *A neural approach to the structure from motion problem.* PhD thesis, Institut National Polytechnique de Grenoble, 1998.

[27] R. de Beer and D. van Ormondt. *NMR Basic Principles and Progress*, volume 26, chapter Analysis of NMR data using time-domain fitting procedures, pages 201–248. Springer-Verlag, Berlin, 1992.

[28] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6), 1990.

[29] S. Doclo, I. Dologlou, and M. Moonen. A novel iterative signal enhancement algorithm for noise reduction in speech. In *Proc. of the 5th International Conference on Spoken Language Processing (ICSLP'98)*, Sydney, Australia, December 1998.

[30] I. Dologlou and G. Carayannis. Physical interpretation of signal reconstruction from reduced rank matrices. *IEEE Trans. on Acoust., Speech and Signal Proc.*, 39:1681–1682, 1991.

[31] I. Dologlou, S. Van Huffel, and D. Van Ormondt. Improved signal enhancement procedures applied to exponential data modeling. *IEEE TRansactions on Signal Processing*, 45(3):799–803, 1997.

[32] R.R. Ernst, G. Bodenhausen, and A. Wokaun. *Principles of Nuclear Magnetic Resonance in One and Two Dimensions.* Clarendon Press, Oxford, 1987.

[33] X. Fan. *The constrained total least squares with regularization and its use in ill-conditioned signal restoration.* PhD thesis, Mississippi State University, 1992.

[34] X. Fan, N.H. Younan, and C.D. Taylor. A perturbation analysis of the regularized constrained total least squares. *IEEE Trans. Circuits Systems II: Analog Digital Signal Process.*, 43(2):140–142, February 1996.

[35] R.A. Fisher. On the mathematical foundations of theoretical statistics. *Phil. Trans.*, 1921.

[36] J.S. Fleming. Measurement of hippuran plasma clearance using a gamma camera. *Ibid.*, 22:526–530, 1977.

[37] J.S. Fleming and B.A. Goddard. A technique for the deconvolution of the renogram. *Phys. Med. Biol.*, 19:546–549, 1974.

[38] R. Fletcher. Methods for nonlinear constraints. In M.J.D. Powell, editor, *Nonlinear Optimization*, Nato Conference Series, pages 185–211. Cambridge, Great Britain.

[39] R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, New York, 1987.

[40] D.G. Gadian. *Nuclear Magnetic Resonance and its applications to living systems.* Clarendon Press, Oxford, 1982.

[41] K. F. Gauss. *Teoria Motus Corporum Coelestium in Sectionibus Conicis Solem Ambientium. Reprinted translation: theory of the Motion of the Heavinly Bodies Moving about the Sun in Conic Sections.* Dover New York, 1963.

[42] A. Gersho and R. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, Boston, 1992.

[43] R. Howman Giles, A. Moase, K. Gaskin, and R. Uren. Hepatobiliary scintigraphy in a pediatric population: determination of hepatic extraction fraction by deconvolution analysis. *J. Nucl. Med.*, 34(2):214–221, February 1993.

[44] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* Academic Press, London, 1981.

[45] G. H. Golub and C.F. Van Loan. An analysis of the total least squares problem. *SIAM J. Numer. Anal.*, 1980.

[46] G.H. Golub and C.F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, 1989.

[47] G.H. Golub and V. Pereyra. The differentiation of pseudo-inverses and nonlinear least squares problems whose variables separate. *SIAM J. Numer. Anal.*, 10(2):413–432, 1973.

[48] M. Haardt and Josef A. Nossek. Structured least squares to improve the performance of ESPRIT-type high-resolution techniques. In *Proc. of ICASSP 1996*, volume V, Atlanta, Georgia, May.

[49] M. Haardt, M.D. Zoltowski, C.P. Mathews, and J. Ramos. *The digital signal processing handbook*, chapter ESPRIT and closed-form 2-D angle estimation with planar arrays. CRC Press and IEEE Press, 1996.

[50] S. Haykin. *Adaptive Filter Theory.* Prentice-Hall International, Inc., 1991.

[51] E. P. Jiang. *Information retrieval and filtering using the Riemannian SVD.* PhD thesis, Dept. of Computer Science, University of Tennessee, Knoxville, TN, 1998.

[52] E. P. Jiang and M. W. Berry. Information filtering using the riemannian svd (r-svd). In *Proceedings of the Fifth International Symposium on: Solving Irregularly Structured Problems in Parallel, Lecture Notes in Computer Science 1457*, pages 386–395, 1998.

[53] T. Kailath and J. Chun. Generalized displacement structure for block–toeplitz, toeplitz–block and toeplitz–derived matrices. *SIAM J. Matrix Anal. Appl.*, 15:114–128, 1994.

[54] T. Kailath, S. Kung, and M. Morf. Displacement ranks of matrices and linear equations. *J. Math. Anal. Appl.*, 68:395–407, 1979.

[55] W.B. Kleijn and K.K. Paliwal. *Speech Coding and Synthesis.* Elsevier, 1995.

[56] M.A. Kohler and L.M. Supplee. Progress towards a new government standard 2400 bps voice coder. In *Proc. Int. Conf. Acoust., Speech and Signal Proc.*, Detroit, Michigan, 1995.

[57] S.Y. Kung, K.S. Arun, and D.V. Bhaskar Rao. State-space and singular-value decomposition-based approximation methods for the harmonic retrieval problem. *J. Opt. Soc. Am.*, 73(12):1799–1811, 1983.

[58] P. Ladefoged. *A Course in Phonetics.* Harcourt Brace Jovanovich, New York, 1975.

[59] P. Lemmerling, I. Dologlou, and S. Van Huffel. Speech compression based on exact modeling and structured total least norm optimization. In *Proceedings of ICASSP 98*, volume I, pages 353–356, Seattle, Washington, U.S.A., May 1998.

[60] P. Lemmerling, I. Dologlou, and S. Van Huffel. Variable rate speech compression based on exact modeling and waveform vector quantization. In *Signal Processing Symposium (SPS 98)*, pages 127–130, Katholieke Universiteit Leuven, Leuven, March 1998. IEEE Benelux Signal Processing Chapter.

[61] P. Lemmerling, B. De Moor, and S. Van Huffel. On the equivalence of constrained total least squares and structured total least squares. *IEEE Trans. on Signal Processing*, 44(11):2908–2910, November 1996.

[62] H. Liu, G. Xu, L. Tong, and T. Kailath. Recent developments in blind channel equalization: From cyclostationarity to subspaces. *Signal Processing*, (50):83–99, 1996.

[63] L. Ljung. *System identification: theory for the user.* Prentice-Hall, Inc., 1987.

[64] The MathWorks Inc. *MATLAB User's Guide. High-Performance Numeric Computation and Visualization Software*, August 1992.

[65] V. Z. Mesarović, N. P. Galatsanos, and A. K. Katsaggelos. Regularized constrained total least squares image restoration. *IEEE Transactions on Image Processing*, 4(8), August 1995.

[66] B. De Moor. Structured total least squares and l_2 approximation problems. *Linear Algebra and its Applications, Special Issue on Numerical Linear Algebra Methods in Control, Signals and Systems, (Van Dooren, Ammar, Nichols, and Mehrmann, eds.)*, 188-189:163–207, July 1993.

[67] B. De Moor. Total least squares for affinely structured matrices and the noisy realization problem. *IEEE Transactions on Signal Processing*, 42:3004–3113, November 1994.

[68] B. De Moor and G.H. Golub. The restricted singular value decomposition: properties and applications. *SIAM Journal on Matrix Analysis and Applications*, 12(3):401–425, July 1991.

[69] B. De Moor and P. Lemmerling. A new trade-off in linear system identification: misfit versus latency. In *Proceedings of the MTNS98*, Padova, Italy, June 1998.

[70] B. De Moor, P. Van Overschee, and G. Schelfhout. h_2 model reduction for siso systems. In *Proc. of the 12th IFAC World Congress*, pages 227–230, Sydney, Australia, July 1993.

[71] B. De Moor and B. Roorda. l_2-optimal linear system identification: Dynamic total least squares for siso systems. In *Proc. of 33rd IEEE CDC*, pages 2874–2879, Florida, December 1994.

[72] L. Ng and V. Solo. Errors-in-variables modelling in optical flow problems. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP 98)*, pages 2773–2776, Seattle, Washington, USA, May.

[73] M.R. Osborne. *Finite algorithms in optimization and data analysis*. Wiley Series in Probability and Mathematical Statistics. Wiley, 1985.

[74] H. Park and L. Eldén. Stability analysis and fast triangularization of toeplitz matrices. *Numer. Math*, 76:383–402, 1997.

[75] D.B. Paul. The spectral envelope estimation vocoder. *IEEE Trans. on Acoust., Speech and Signal Proc.*, 29, 1981.

[76] J.J. Pedroso de Lima. Nuclear medicine and mathematics. *Eur. J. Nucl. Med.*, 23(6):705–719, June 1996.

[77] W.W.F. Pijnappel, R. de Beer, and D. van Ormondt. State space modelling of medical nmr signals. In J.P. Veen and M.J. de Ket, editors, *Proc. PRORISC*, pages 225–230, Utrecht, 1992. STW.

[78] R. Pintelon, P. Guillaume, G. Vandersteen, and Y. Rolain. Analyses, development, and applications of TLS algorithms in frequency domain system identification. *SIAM J. Matrix Anal. Appl.*, 19(4):983–1004, October 1998.

[79] B. Roorda. Algorithms for global total least squares modelling of finite multi-variable time series. *Automatica*, 31(3), 1995.

[80] B. Roorda and C. Heij. Global total least squares modelling of multivariable time series. *IEEE Transcations on Automatic Control*, 40(1):50–63, 1995.

[81] J. Rosen, H. Park, and J. Glick. Total least norm formulation and solution for structured problems. Technical Report 94-041, University of Minnesota, Army High Performance Computing Research Center, July 1994.

[82] J. B. Rosen, H. Park, and J. Glick. Structured total least norm for nonlinear problems. *SIAM Journal on Matrix Analysis and Applications*.

[83] J.B. Rosen, H. Park, and J. Glick. Total least norm formulation and solution for structured problems. *SIAM Journal on Matrix Anal. Appl.*, 17(1):110–128, 1996.

[84] G. Schelfhout. *Model Reduction for Control Design*. PhD thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, 1996.

[85] G. Schelfhout and B. De Moor. Frequency weighted h2 and hilbert-schmidt-hankel norm model reduction. In *Proc. of the 33rd IEEE Conference Decision Control*, pages 3215–3216, Lake Buena Vista, Florida, December 1994.

[86] T. Söderström and P. Stoica. *System Identification*. Prentice Hall, 1989.

[87] M. Stewart and P. Van Dooren. Stability issues in the factorization of structured matrices. *SIAM J. Matrix Anal. Appl.*, 18:104–118, 1997.

[88] P. Stoica, J. Li, and T. Söderström. On the inconsistency of IQML. *Signal Processing*, (56):185–190, 1997.

[89] P. Stoica, R. L. Moses, B. Friedlander, and T. Söderström. Maximum likelihood estimation of the parameters of multiple sinusoids from noisy measurements. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37:378–391, 1989.

[90] L. Tong, G. Xu, and T. Kailath. Blind identification and equalization based on second-order statistics: a time domain approach. *IEEE Transactions on Information Theory*, 40(2):340–349, March 1994.

[91] H. Van hamme. *Identification of linear systems from time-or frequency-domain measurements*. PhD thesis, Vrije Universiteit Brussel, Dept. Electrical Engineering, ELEC, 1992.

[92] S. Van Huffel. *Analysis of the total least squares problem and its use in parameter estimation*. PhD thesis, Dept. of Electrical Eng., Katholieke Universiteit Leuven, Belgium, 1987.

[93] S. Van Huffel, C. Decanniere, H. Chen, and P. Van Hecke. Algorithm for time-domain nmr data fitting based on total least squares. *J. Magn. Reson.*, A110:228–237, 1994.

[94] S. Van Huffel, B. De Moor, and H. Chen. Relationships between structured and constrained tls, with applications to signal enhancement. In Helmke U., Mennicken R., and Saurer J., editors, *Systems and Networks: Mathematical Theory and Applications*, pages 899–904, Berlin, 1994. Akademie Verlag.

[95] S. Van Huffel, H. Park, and J. B. Rosen. Formulation and solution of structured total least norm problems for parameter estimation. *IEEE Trans. on Signal Processing*, 44(10):2464–2474, 1996.

[96] S. Van Huffel and J. Vandewalle. *The Total Least Squares Problem: computational aspects and analysis*, volume 9. SIAM, Philadelphia, 1991.

[97] S. Van Huffel, J. Vandewalle, M. Ch. De Roo, and J.L. Willems. Reliable and efficient deconvolution technique based on total linear least squares for calculating the renal retention function. *Med. & Biol. Eng. & Comput.*, 25, 1987.

[98] G. A. Watson. On two methods for discrete l_p approximation. *Computing*, 18:263–266, 1977.

[99] G.A. Watson. On a class of matrix nearness problems. *Const. Approx.*, 7:299–314, 1991.

[100] J. C. Willems. From time series to linear system, part i: Finite dimensional linear time invariant systems; part ii: Exact modelling; part iii: Approximate modelling. *Automatica*, 22/23:561–580,675–694,87–115, 1986,1987.

[101] A. Yeredor. *The Extended Least-Squares Criterion for Discriminating Measurement Errors from Model Errors: Algorithms, Applications, Analysis.* PhD thesis, Dept. of Electrical Eng., Tel-Aviv University, Israel, June 1997.

[102] N.H. Younan and X. Fan. Signal restoration via the regularized constrained total least squares. *Signal Processing*, 71:85–93, 1998.

[103] G. Zhu, W. Y. Choy, and B.C. Sanctuary. Spectral parameter estimation by an iterative quadratic maximum likelihood method. *Journal of Magnetic Resonance*, 135:37–43, 1998.