**KATHOLIEKE UNIVERSITEIT LEUVEN**
FACULTEIT DER TOEGEPASTE WETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK
Kard. Mercierlaan 94 — 3001 Leuven (Heverlee)

# ALGORITHMS AND ARCHITECTURES FOR ADAPTIVE ARRAY SIGNAL PROCESSING

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

door

**Filiep VANPOUCKE**

U.D.C. 621.391

Februari 1995

*Voor Inge*

# Voorwoord

Bij de voltooiing van mijn doktoraat wil ik met veel genoegen een woord van dank richten tot iedereen die ertoe heeft bijgedragen. SISTA is zonder twijfel een aangename en stimulerende groep met internationale uitstraling. Dit klimaat is niet in het minst het werk van mijn promotor Prof. Vandewalle. Ik dank hem dan ook oprecht voor de vele kansen die ik in de voorbije jaren gekregen heb.

Prof. De Moor stond mee aan de wieg van dit doktoraat. Zijn dynamisme en actieve belangstelling stonden altijd garant voor een stevige zet in de rug.

Mijn speciale dank gaat naar Dr. Marc Moonen. Dit doktoraat sluit aan bij zijn vroegere werk en is gegroeid uit een intense samenwerking. Met zijn rijke ervaring heeft hij mij geïntroduceerd in de wetenschappelijke wereld. Welgemeend mag ik stellen dat ik me geen betere begeleider kon wensen.

Een gewaardeerde wetenschapper die ik via Marc heb leren kennen, is Prof. Deprettere van de Technische Universiteit Delft. De interactie met hem en de onderzoekers in zijn groep is een constante verrijking. Ik ben dan ook verheugd dat hij bereid was om deel te nemen aan het leescomité.

Ik wil Prof. Catthoor danken voor de aangename samenwerking op het vlak van architectuurontwerp. De visie op parallelle architecturen in dit werk is terdege beïnvloed door het NANA project waarvan hij de enthousiaste voortrekker is.

It is also a pleasure to have Prof. McWhirter from DRA Malvern as a member of the jury. He has always shown a sincere interest in my work. Therefore, I am grateful for his willingness to review this text.

Verder bedank ik ook Prof. Bultheel voor zijn wijde wetenschappelijke interesse en onmiddellijke bereidheid om in de jury te zetelen.

Als lid van de facultaire doktoraatscommissie had ik de kans om Prof. Dutré beter te leren kennen. Ik dank hem voor zijn vlotte medewerking als voorzitter van de jury.

During my work I have had the opportunity to pay two visits to Prof. Paulraj and Prof. Kailath at the Information Systems Laboratory of Stanford University. These magnificent stays were very rewarding. Part of this text is a direct outcome of a fruitful collaboration.

Alle huidige en vroegere leden van SISTA met naam vermelden is een delicate aangelegenheid geworden. Laat me enkel de volgende personen expliciet vernoemen: Bart en Lieven, met wie ik het langst aangename uren op één bureau gedeeld heb, en Jeroen voor zijn bereidheid om een eerste versie van dit proefschrift van commentaar te voorzien. Aan allen dank voor de vele ernstige en minder ernstige discussies over onderwerpen allerhande.

Ook gaat mijn dank naar Ingrid, Rita en Ann op het secretariaat en Johan Buelens en de mensen van de systeemmanagementploeg.

Bij het einde van mijn studies wil ik mijn ouders even in de bloemetjes zetten. Samen met mijn schoonouders, broers en zussen hebben zij zich dikwijls afgevraagd waarmee ik op de universiteit zo druk bezig was. Ik hoop dat deze tekst hun nieuwsgierigheid kan stillen.

Ik kan niet onder woorden brengen wat ik verschuldigd ben aan Inge. Ons huwelijk en dit doktoraat zijn ongeveer gelijktijdig gestart. Met warme genegenheid heeft ze me de volledige periode intens gesteund. Onder andere als compensatie voor de slapeloze nachten tijdens mijn talrijke afwezigheden draag ik dit werk aan haar op.

Tenslotte is dit werk enkel tot stand kunnen komen met de financiële steun van het Nationaal Fonds voor Wetenschappelijk Onderzoek en de ESPRIT BRA 3280 en 6632 projecten van de Europese Unie.

# Abstract

Antenna arrays sample propagating waves at multiple locations. They are employed *e.g.* in radar, sonar and wireless communication systems because of their capability of spatial selectivity and localization of radiating sources. Current model-based algorithms make use of computationally demanding orthogonal matrix decompositions such as the singular value decomposition (SVD). On the other hand the data rates are often extremely high. Therefore, real-time execution of complex algorithms often requires parallel computing. We study the simultaneous design of new algorithms and parallel architectures for subspace tracking, for robust adaptive beamforming and for direction finding of narrow-band and wide-band sources. By structuring all recursive algorithms in a similar way, they can be mapped efficiently onto the Jacobi architecture, which was originally developed for SVD updating. The numerical and architectural aspects of this algorithm are improved by the use of a minimal parameterization of the orthogonal matrix of short singular vectors. Finally, a new Fourier-based linear model for direction finding in colored ambient noise fields is proposed.

# Abstract

Roosterantennes bemonsteren propagerende golven op meerdere plaatsen. Ze worden o.a. in radar-, sonar- en radiocommunicatiesystemen gebruikt omwille van de mogelijkheid tot ruimtelijke selectiviteit en plaatsbepaling van signaalbronnen. De huidige modelgebaseerde algoritmen maken gebruik van rekenintensieve orthogonale matrixdecomposities zoals de singuliere waarden ontbinding (SWO). Anderszijds zijn de datasnelheden vaak heel hoog. Daarom vraagt uitvoering in reële tijd van deze complexe algoritmen vaak het gebruik van parallelle computers. We bestuderen het gelijktijdig ontwerp van nieuwe algoritmen en parallelle architecturen voor recursieve schatting van deelruimten, voor robuuste adaptieve straalvormers en voor richtingshoekbepaling van zowel smalbandige als breedbandige signalen. Door alle recursieve algoritmen op een gelijkaardige manier te structureren, kunnen ze efficiënt geïmplementeerd worden op de Jacobi architectuur die oorspronkelijk ontwikkeld werd voor recursieve SWO. De numerieke en architecturale aspecten van dit algoritme worden verbeterd door het gebruik van een minimale parameterisatie van de orthogonale matrix die de korte singuliere vectoren bevat. Tot slot wordt een nieuw Fourier-gebaseerd lineair model voorgesteld voor richtingshoekbepaling in gecorreleerde omgevingsruis.

# Glossary

## Symbols

| | | |
|---|---|---|
| $\left[\,\cdot\,\right]$ | : | general matrix or vector |
| $\left[\,\cdot\,\right]^T$ | : | transpose of a matrix or vector |
| $\left[\,\cdot\,\right]^*$ | : | conjugated matrix or vector |
| $\left[\,\cdot\,\right]^H$ | : | Hermitian transpose of a matrix or vector |
| $\lVert\cdot\rVert$ | : | Frobenius norm of a matrix or vector |
| $x_i$ | : | $i$th element of vector $x$ |
| $X_{ij}$ | : | element of matrix $X$ on row $i$ and column $j$ |
| $\mathrm{vec}(X)$ | : | column vector obtained by stacking the columns of $X$ |
| $\det(X)$ | : | determinant of matrix $X$ |
| $\mathrm{tr}(X)$ | : | trace of matrix $X$ |
| $X^{1/2}$ | : | Cholesky factor of square matrix $X$ |
| $Q^{i\|j}$ | : | matrix embedding of Givens rotation operating on rows (columns) $i$ and $j$ |
| $I_n$ | : | identity matrix of size $n \times n$ |
| $O_{m\times n}$ | : | zero matrix of size $m \times n$ |
| $1_{m\times n}$ | : | constant matrix of ones of size $m \times n$ |
| $\mathcal{O}(x)$ | : | order of $x$ operations |
| $\propto$ | : | proportional to |
| $\otimes$ | : | Kronecker product |
| $\odot$ | : | Khatri-Rao product |
| $\star$ | : | don't care value |
| $j$ | : | $\sqrt{-1}$ |
| $x_{[k]}$ | : | vector $x$ sampled at sampling time $k$ |

## Acronyms

| | | |
|---|---|---|
| 1-D | : | one dimensional |
| 2-D | : | two dimensional |
| ACMP | : | algebraically coupled matrix pencils algorithm |
| AOA | : | angle of arrival |
| ARMA | : | autoregressive moving average |
| CORDIC | : | coordinate rotation digital computer |
| CRB | : | Cramér-Rao bound |
| DF | : | direction finding |
| DG | : | dependence graph |
| DOA | : | direction of arrival |
| ESPRIT | : | estimation of signal parameters by rotational invariance techniques |
| EVD | : | eigenvalue decomposition |
| FDMA | : | frequency division multiple access |
| FIR | : | finite impulse response |
| GSC | : | generalized sidelobe canceler |
| GSM | : | global system for mobile communications |
| GSD | : | generalized Schur decomposition |
| LCMV | : | linearly constrained minimum variance beamformer |
| LMI | : | linear matrix inequality |
| MEMP | : | matrix enhancement and matrix pencil algorithm |
| ML | : | maximum likelihood |
| MSE | : | mean square error |
| MUSIC | : | multiple signal classification |
| QRD | : | QR decomposition |
| RLS | : | recursive least squares |
| RMSE | : | root mean square error |
| SFG | : | signal flow graph |
| SINR | : | signal to interference and noise ratio |
| SNR | : | signal to noise ratio |
| SOI | : | signal of interest |
| SVD | : | singular value decomposition |
| TDMA | : | time division multiple access |
| TDOA | : | time difference of arrival |
| TLS | : | total least squares |
| ULA | : | uniform linear array |
| VLSI | : | very large scale integration |
| WSF | : | weighted subspace fitting |

# Contents

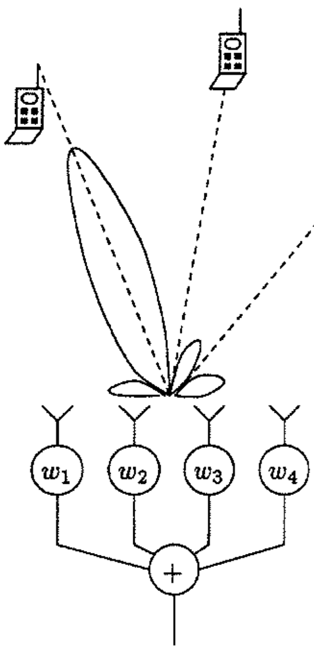
# Algoritmen en architecturen voor adaptieve signaalverwerking van roostersensoren

## Nederlandse samenvatting

### Inleiding

Het toepassingsgebied van dit proefschift is de verwerking van signalen afkomstig van roosterantennes of roostersensoren. Een roostersensor is een verzameling van sensoren die op meerdere plaatsen een golffront bemonsteren. Het gebruik van roostersensoren biedt tal van nieuwe mogelijkheden ten opzichte van enkelvoudige sensoren. Ze kunnen ruwweg opgedeeld worden in twee categorieën. Een eerste categorie beoogt spatiëel filteren. Door de uitgangen van de sensoren lineair te combineren wordt de ontvangst van de roosterantenne richtingsgevoelig. Deze structuur wordt een straalvormer (*E. beamformer*) genoemd en is afgebeeld in Figuur 0.1. De richtingsgevoeligheid wordt bepaald door de keuze van de filtercoëfficiënten. In vele applicaties, zoals satellietcommunicatie, wordt er gestreefd naar een maximale ontvangst uit de richting van het gewenste signaal terwijl signalen uit andere richtingen zoveel mogelijk onderdrukt worden. In andere toepassingen is de positie van ongewenste interferenties gekend, en dient er een nul geplaatst te worden in het ontvangstpatroon. Het verband met klassieke banddoorlaat FIR filters in het tijdsdomein ligt voor de hand.

De tweede categorie beoogt karakterisatie van de propagerende golven. Iedere puntbron wordt gekenmerkt door een beperkt aantal parameters,



Figuur 0.1: Een straalvormer in een cellulair mobilofoniesysteem. Drie mobiele zenders met verschillende locaties communiceren simultaan op dezelfde frequentie met het basisstation. De coëfficienten $w_i$ bepalen de richtingsgevoeligheid van de roosterantenne van het basisstation. Hier is de roosterantenne gericht naar de linkse mobiele zender. De twee overige signalen worden volledig onderdrukt.

zoals zijn positie (hoek en afstand) en het uitgezonden signaalvermogen. Om deze signaalparameters te kunnen schatten is het gebruik van roosterantennes noodzakelijk. Men kan geen informatie over de positie van een signaalbron afleiden als het golffront niet op meerdere plaatsen bemonsterd wordt.

Het domein van digitale signaalverwerking voor roosterantennes heeft sinds de jaren tachtig een sterke groei gekend. Veel nieuwe filter- en parameterestimatietechnieken zijn voorgesteld. Alle algoritmen hebben gemeen dat ze sterk steunen op numerieke lineaire algebra. Lineaire algebra vormt het natuurlijke kader om de problemen wiskundig te formuleren. Signalen als functie van de tijd kunnen voorgesteld worden door vectoren. Ook de uitgangen van de roosterantenne op een bepaald ogenblik vormen een vector. De algoritmen manipuleren dus logischerwijze vectoren en matrices. Uit de onderliggende geometrie van het datamodel volgt dat veel informatie vervat is in bepaalde invariante deelruimtes. Matrixdecomposities spelen dan ook een belangrijke rol. De twee meest courante matrixdecomposities zijn de QR decompositie (QRD) en de singuliere waardenontbinding (SWO). Het domein van de numerieke lineaire algebra biedt bovendien een rijk gamma robuuste algoritmen om onder andere matrixdecomposities te berekenen.

De prijs voor de verhoogde performantie van deze signaalverwerkingsalgoritmen is hun relatief hoge rekenkost. Per tijdsstap bedraagt de rekencomplexiteit typisch $\mathcal{O}(M^2)$ waarbij $M$ het aantal sensoren voorstelt. Daarbij komt dat de datadoorvoersnelheden in typische antennetoepassingen zoals radar-, sonar- of communicatiesystemen heel hoog liggen, in de ordegrootte van 10 kbit/s tot 1Mbit/s.

De combinatie van hoge bitsnelheden en een aanzienlijke rekenkost maakt uitvoering in reële tijd problematisch, zelfs met de huidige generatie van snelle processoren. Soms kunnen applicatie-specifieke processoren een oplossing bieden. Als ook dat niet het geval is, kan de benodigde rekenkracht enkel geleverd worden door parallelle computers. Hierbij stellen zich opnieuw een aantal uitdagingen. Uitvoering op parallelle computers versnelt de berekening enkel als er een goede afstemming is tussen het algoritme en de architectuur van de parallelle machine. Er moet een evenwicht heersen tussen berekening, communicatie tussen de verschillende processoren onderling en tussen processoren en geheugen.

Een belangrijk aandachtspunt van deze thesis is de afleiding van optimale architecturen voor uitvoering in reële tijd van de geavanceerde signaalverwerkingsalgoritmen. De motivatie hiervoor is dat simultaan ontwerp van algoritme en parallelle architectuur het optimale compromis tussen de verschillende systeemcomponenten (rekenmodules, geheugen en invoer/uitvoer) het dichtst kan benaderen. Omwille van haar kracht en bevattelijke visualisatie zal de grafische methode gebaseerd op signaalstroomgrafes (*E. signal flow graphs*) hierbij gebruikt worden. Een signaalstroomgrafe is een gerichte grafe waarin de knopen corresponderen met de berekeningen in het algoritme en de pijlen de uitgewisselde signalen voorstellen.

### Algemeen overzicht

Twee rode draden lopen door dit proefschrift. Een eerste rode draad is de ontwikkeling van nieuwe algoritmen voor de verwerking van antennesignalen. De nadruk ligt op recursieve algoritmen die geschikt zijn voor uitvoering in reële tijd. Zowel spatiaal filteren als invalshoekschatting worden behandeld. Veel algoritmen in de literatuur zijn gebaseerd op niet-lineaire optimaliseringstechnieken. Deze zijn echter moeilijk implementeerbaar op een parallelle computer. Daarom beperken we ons zoveel mogelijk tot matrixdecomposities in de keuze van bouwblokken voor recursieve algoritmen.

De tweede rode draad is toepassingsgerichte parallelle architecturen. Het is echter niet de bedoeling om per algoritme een totaal verschillende architectuur te poneren. Onze aanpak bestaat erin voor alle toepassingen structureel gelijkaardige algoritmen te ontwikkelen, die dan efficiënt implementeerbaar zijn op eenzelfde architectuur. Deze raamarchitectuur is de Jacobi architectuur die oorspronkelijk ontwikkeld werd voor recursieve singuliere waardenontbinding [66, 67]. De structuur van dit algoritme is een opeenvolging van matrix-vector vermenigvuldigingen, recursieve QRD en tweezijdige orthogonale transformaties. Door de signaalverwerkingsalgoritmen ook op die manier te formuleren, kan dezelfde Jacobi architectuur herbruikt worden. Bovendien wordt deze architectuur aantrekkelijker om een VLSI implementatie te ontwikkelen. Aan dit ontwerp wordt momenteel gewerkt [24, 118].

## Hoofdstuk 2. Concepten en bouwblokken

Het proefschrift steunt op drie peilers. Een eerste peiler is digitale signaalverwerking van meervoudige signalen met de nadruk op roosterantennes. Dit is het toepassingsdomein. Een tweede peiler is numerieke lineaire algebra. Dit domein vormt het natuurlijke kader voor de wiskundige probleemstelling en reikt een veelvoud van algoritmische bouwblokken aan. Het derde domein is grafische methodologieën voor simultaan ontwerp van algoritmen en architecturen.

Dit inleidend hoofdstuk bevat het nodige achtergrondmateriaal uit elk van deze drie domeinen. De opbouw van het hoofdstuk volgt de ontwikkeling van de signaalverwerkingsapplicaties. De bespreking van de nodige concepten uit lineaire algebra en grafische ontwerpmethodologieën gebeurt aan de hand van de toepassingen.

Eerst wordt het datamodel afgeleid. We beschouwen een basisbandmodel voor een scenario waarbij de $D$ signaalbronnen puntbronnen in het verre veld van de roosterantenne zijn. De uitgangen van de $M$ antennes in het rooster op $N$ bemonsteringstijden vormen een matrix $X \in \mathbb{C}^{M\times N}$. Elke (incoherente) puntbron geeft aanleiding tot een rang-1 bijdrage in $X$. Voor $D$ signalen voldoet de datamatrix aan de vergelijking

$$X = A \cdot S + W,$$

waarbij $A \in \mathbb{C}^{M\times D}$ de roosterwinstmatrix (*E. array gain matrix*) wordt genoemd, $S \in \mathbb{C}^{D\times N}$ de invallende basisbandsignalen bevat en $W \in \mathbb{C}^{M\times N}$ bestaat uit de additieve ruis. De informatie omtrent de invalshoeken van de signalen zit vervat in de kolommen van de roosterwinstmatrix. Uit dit elementair datamodel volgt onmiddellijk het belang dat matrixdecomposities zullen spelen in de opbouw van de algoritmen.

Het is bekend dat de bepaling van de coëfficiënten van een optimale straalformer kan geformuleerd worden als een kleinste kwadraten probleem met lineaire beperkingen.

$$\min_{w} w^H \cdot (X \cdot X^H) \cdot w \qquad \text{waarbij} \qquad C^H \cdot w = m.$$

De matrix $C \in \mathbb{C}^{M\times K}$ bevat de beperkingen en $m \in \mathbb{C}^K$ bevat de opgelegde winstvector. Een typische beperking legt de waarde van de ontvangst in de richting van het signaal $\theta_s$ vast.

$$a(\theta_s)^H \cdot w = \mu.$$

De vector $a(\theta_s) \in \mathbb{C}^M$ wordt de richtingsvector genoemd en bevat de amplitude en faze van een signaal uit richting $\theta_s$ voor elk van de sensoren. Kleinste kwadraten problemen kunnen elegant opgelost worden door gebruik te maken van de QR decompositie van de data matrix. De gewichtsvector $w$ volgt dan uit een bovendriehoekig stelsel lineaire vergelijkingen.

In een tijdsvariante signaal- en ruisomgeving moet de straalvormer zich voordurend aanpassen aan de veranderingen in zijn omgeving. De QRD oplossingsmethode leent zich uitstekend tot recursieve berekening. De nieuwe gewichtsvector kan berekend worden in $\mathcal{O}(M^2)$ operaties met de Givens rotatiemethode voor recursieve QRD [33]. Deze methode stelt een sequentie van $2 \times 2$ Givens rotaties op die de inkomende datavector gradueel nul maken. Deze Givens methode is erg geschikt voor parallelle implementatie omdat elke rotatie locaal berekend kan worden uit $2 \times 2$ submatrices. Een driehoekige signaalstroomgrafe (SSG) voor recursieve QRD is voorgesteld door Gentleman en Kung [31]. De complete signaalstroomgrafe van de straalvormer met lineaire beperkingen bestaat uit een rechthoekige grafe voor matrix-vector vermenigvuldiging gevolgd door de driehoekige grafe voor recursieve QRD. De topologie van deze grafe is eenvoudig. Alle data-afhankelijkheden zijn lokaal. Een knoop wisselt enkel signalen uit met zijn naaste buren. Bovendien is het afhankelijkheidspatroon identiek van knoop tot knoop. Daardoor is de omvorming van deze signaalstroomgrafe in een planaire architectuur haast triviaal. De omvorming behelst de toewijzing van een processor en een uitvoeringstijd aan elk stuk van de berekening. Omwille van haar uniformiteit leidt deze grafe bij maximale pijplijning tot een systolisch rooster. Een systolisch rooster is een synchrone parallelle architectuur waarin alle processoren locale en regelmatige interconnecties hebben. Omwille van hun regelmaat zijn ze uitermate geschikt voor VLSI implementatie van applicatie-specifieke processoren.

Naast spatiale filtering is ook scheiding en invalshoekschatting van meerdere signalen die invallen op de roosterantenne een belangrijk probleem. Deelruimte-algoritmen zoals MUSIC [96] en ESPRIT [92] worden hiervoor meest gebruikt. Omdat ze modelgebaseerd zijn, kunnen ze een beduidend hogere resolutie bereiken dan niet-modelgebaseerde technieken zoals de spatiale Fouriertransformatie. Alle algoritmen uit deze klasse berekenen eerst een singuliere waardenontbinding van de datamatrix. Op die manier bepalen ze de signaalruimte $\mathcal{S} = \text{Bereik}\{A\}$ en onderdrukken ze de additieve ruis. De tweede stap bestaat uit het vinden van de invalshoeken die de geschatte deelruimte zo goed mogelijk verklaren. De algoritmen onderscheiden zich in deze stap. Het MUSIC algoritme gebruikt niet-lineaire optimalisering. Het heeft dus een hoge rekenkost en is niet erg geschikt voor parallellisatie. Het ESPRIT algoritme biedt een oplossing voor beide problemen door de invalshoeken te schatten aan de hand van een veralgemeend eigenwaardenprobleem. Dit algoritme is enkel toepasbaar als de roosterantenne kan opgedeeld worden in twee identieke verschoven subroosters. In de praktijk is dit veelal het geval.

Een recursief Jacobi SWO algoritme met bijhorende signaalstroomgrafe is ontwikkeld in [66, 67]. Deze grafe is een combinatie van de grafe voor de adaptieve straalvormer (matrix-vectorvermenigvuldiging en recursieve QRD) met een grafe voor sequenties van tweezijdige Givens rotaties. Deze SSG is moeilijk parallelliseerbaar, omdat ze een lange bidirectionele keten van data-afhankelijkheden bevat. Een efficiënte pijplijning kan enkel door het opbreken van keten. In dit geval zijn standaard transformatietechnieken op grafen ontoereikend. Het algoritme zelf moet gewijzigd worden. De techniek van algoritmische transformaties die ontwikkeld werd voor recursieve SWO, is sindsdien ook succesvol gebleken in de afleiding van efficiënte architecturen voor andere recursieve signaalverwerkingsalgoritmen zoals recursieve kleinste kwadraten [65].

## Hoofdstuk 3. Gefactoriseerde Jacobi SVD updating

Dit hoofdstuk bevat de eerste originele bijdrage van dit proefschrift, nl. een variant op het originele recursieve Jacobi SWO algoritme met betere numerieke eigenschappen en VLSI implementeerbaarheid.

Orthogonale matrixdecomposities zijn een belangrijke hoeksteen van moderne signaalverwerking, identificatie- en controletheorie,... Voorbeelden zijn de QR decompositie, vaak gebruikt in kleinste kwadratenproblemen, en de singuliere waardenontbinding, vaak gebruikt voor problemen in verband met matrices van lage rang. Bovendien is er een rijk gamma van veralgemeningen en varianten op deze decomposities.

Orthogonale decomposities zijn niet enkel conceptueel belangrijk. Ze zijn ook geliefd omwille van hun goede numerieke eigenschappen. Fouten in de berekening kunnen niet aangroeien. Nochtans zijn orthogonale technieken niet helemaal veilig. Als voorbeeld beschouwen we het Jacobi algoritme voor recursieve SWO [66]. In elke iteratie wordt de matrix van korte singuliere vectoren $V \in \mathbb{R}^{M\times M}$ vermenigvuldigd met een orthogonale updating matrix $\Phi \in \mathbb{R}^{M\times M}$

$$V_{[k+1]} \leftarrow V_{[k]} \cdot \Phi_{[k]}. \qquad (0.1)$$

Accumulatie van afrondingsfouten in de opeenvolgende matrixvermenigvuldigingen leidt tot geleidelijk verlies van de orthogonaliteit van $V_{[k]}$.

Haast alle recursieve schema's voor orthogonale decomposities wapenen zich hiertegen door periodiek de betrokken matrices te herorthogonaliseren. Courante technieken zijn gebaseerd op een Gram-Schmidt orthogonalisatie (QRD). Belangrijke nadelen zijn de beduidende rekenkost ($\mathcal{O}(M^3)$) en de moeilijke parallellisatie.

In dit hoofdstuk stellen we een alternatief voor. Een willekeurige orthogonale matrix $V \in \mathbb{R}^{M\times M}$ met positieve determinant wordt uniek geparameteriseerd door een sequentie van Givens rotaties $Q^{i|j}$, elk gekenmerkt door één rotatiehoek $\alpha^{i|j}$

$$V = \prod_{i=1}^{M-1} \prod_{j=i+1}^{M} Q^{i|j}.$$

Zo'n Givens rotatie matrix $Q^{i|j}$ is een $2 \times 2$ rotatiematrix ingebed in een omvattende identiteitsmatrix. Door impliciet te rekenen met deze parameterisatie $\{Q^{i|j}\}$ in plaats van met de matrix zelf, blijft $V_{[k]}$ noodzakelijkerwijze binnen de verzameling orthogonale matrices. Afrondingsfouten manifesteren zich enkel nog op de hoeken. Deze fouten worden teniet gedaan door de terugkoppeling in het recursieve SVD algoritme. Waar reorthogonalisatiemethodes de afwijking van orthogonaliteit enkel binnen de perken houden, garandeert deze minimale parameterisatie de orthogonaliteit in elke iteratie. We bestuderen nu de implementatie van dit idee voor het recursieve Jacobi SWO algoritme.

Er zijn twee bewerkingen waarbij de matrix van korte singuliere vectoren $V_{[k]}$ betrokken is. Een eerste bewerking is een matrix-vector vermenigvuldiging met de inkomende data vector.

$$\tilde{x}_{[k]}^T = x_{[k]}^T \cdot V_{[k-1]}.$$

Deze matrix-vector vermenigvuldiging wordt nu vervangen door een sequentie van $M(M-1)/2$ Givens rotaties. Hierdoor wordt de SSG gewijzigd. De SSG van een matrix-vectorvermenigvuldiging is een rechthoekige grafe waarbij elke knoop een scalaire vermenigvuldiging en een optelling uitvoerde. Dit wordt nu vervangen door een driehoekige grafe waarbij elke knoop een Givens rotatie uitvoert. De impact hiervan op een hardware implementatie is niet onbelangrijk. Het volledige recursieve Jacobi SWO algoritme bestaat nu exclusief uit eenzijdige en tweezijdige $2 \times 2$ rotaties. Voor snelle en accurate uitvoering van zulke rotaties is een CORDIC processor [143] de ideale hardware component. De vervanging van vermenigvuldigingen door rotaties maakt een parallelle Jacobi architectuur mogelijk die enkel bestaat uit CORDIC-gebaseerde processoren. Dit vereenvoudigt het ontwerp en de programmering van een toepassingsgerichte parallelle computer aanzienlijk.

De tweede en laatste operatie op de matrix $V_{[k]}$ is gegeven door vergelijking (0.1). De orthogonale matrix $\Phi_{[k]}$ is het product van $M-1$ rotaties op opeenvolgende kolommen

$$\Phi_{[k]} = \prod_{i=1}^{M-1} \Phi_{[k]}^{i|i+1}.$$

De originele bijdrage van dit hoofdstuk is een schema om de rotaties $Q_{[k]}^{i|j}$ rechtstreeks aan te passen zonder expliciete berekening van $V_{[k]}$. De $\Phi_{[k]}^{i|i+1}$ rotaties moeten geleidelijk in de sequentie van $Q_{[k]}^{i|j}$ verwerkt worden. De transformaties op de hoeken hangen af van de kolomindices van de twee interagerende rotaties. Drie types van rotaties zijn noodzakelijk.

1. Als de indexparen totaal verschillen, commuteren de rotaties en is er geen berekening.
2. Als de indexparen identiek zijn, is de samenstelling van de rotaties de rotatie over de som van de hoeken.
3. Als de indexparen één index gemeen hebben, moet een derde rotatie in beschouwing genomen worden om de volgorde van de indices te wijzigen.

De gecombineerde SSG die zowel de matrix-vector vermenigvuldiging als het updatingschema bevat, blijft driehoekig. De grafe bevat net als het oorspronkelijke Jacobi algoritme lange afhankelijkheidsketens. Daarom moet bij het pijplijnen opnieuw gebruik gemaakt worden van dezelfde algoritmische transformaties om tot een efficiënte parallelle (systolische) implementatie te komen.

## Hoofdstuk 4. Een gefactoriseerde sferische deelruimtevolger

In dit hoofdstuk passen we de minimale parameterisatie van orthogonale matrices toe op een tweede algoritme voor recursieve deelruimteschatting, nl. de sferische deelruimte volger. Een variant op het originele algoritme wordt voorgesteld en twee systolische architecturen worden afgeleid.

Een belangrijke applicatie van recursieve SWO algoritmen is het volgen van een traag tijdsvariante deelruimte. Een voorbeeld is het schatten van invalshoeken van signalen met behulp van roosterantennes. Als de bronnen bewegen, moet het algoritme in staat zijn deze evolutie te volgen. Het recursieve Jacobi SWO algoritme berekent de SWO op elk ogenblik slechts benaderend. De reden hiervoor is dat per iteratie bij exacte berekening de rekenkost $\mathcal{O}(M^3)$ zou bedragen. In vele applicaties is deze kost te hoog. Ook de $\mathcal{O}(M^2)$ kost van het Jacobi algoritme kan nog problemen stellen. Daarom zijn verscheidene ruwere benaderingen voor recursieve SWO ontwikkeld. Het adaptieve sferische deelruimte algoritme [22] verlaagt de rekenkost extreem tot $\mathcal{O}(M \cdot D)$. Dit wordt bereikt door twee technieken. Een eerste techniek bestaat erin om op elk ogenblik de singuliere waarden in de signaaldeelruimte en de ruisdeelruimte apart uit te middelen. Dit kan in applicaties waar de exacte kennis van de singuliere waarden zelf niet cruciaal is. Een voorbeeld is invalshoekschatting waarbij enkel de scheiding van signaal- en ruisdeelruimte cruciaal zijn. Een deelruimte waarvan de geassocieerde singuliere waarden uitgemiddeld zijn, wordt een sferische deelruimte genoemd omdat elke orthogonale basis voor deze deelruimte een basis van singuliere vectoren vormt. Een tweede techniek bestaat in het bijhouden van enkel de kleinste deelruimte, bvb. de signaaldeelruimte als $D < M$. Indien kennis van de ruisdeelruimte vereist is, kan die op elk ogenblik berekend worden als het orthogonale complement.

Het originele algoritme is afgeleid als een benaderend recursief eigenwaardenalgoritme. Om de vergelijking met het recursieve Jacobi SWO algoritme expliciet te maken, hebben we ervoor geopteerd om hier het algoritme opnieuw af te leiden als een benaderend SWO algoritme. Uit de afleiding blijkt duidelijk dat het sferische algoritme steunt op hetzelfde werkingsprincipe als het Jacobi algoritme om de deelruimte recursief te schatten. In elke iteratie wordt een orthogonale matrix die de deelruimte omspant, aangepast door $2 \times 2$ kolomrotaties. Daarom is er ook hier weer gevaar voor geleidelijk verlies van orthogonaliteit van deze matrix. We stellen daarom voor om dezelfde orthogonale parameterisatie met hoeken te gebruiken. Daardoor wordt het algoritme perfect numeriek stabiel.

Een nieuw aspect is dat nu enkel een deelmatrix van een orthogonale matrix moet geparameteriseerd worden. Dit stelt geen probleem. Aangezien de berekening van de rotaties $Q^{i|j}$ kolomsgewijze gebeurt, volstaat het deze parameterisatie stop te zetten na $D$ kolommen.

Bovendien tonen we aan dat het bewaren van een (parameterisatie van een) orthogonale basis voor de ruisruimte niet hoeft. Het volstaat in elke iteratie de projectie van de inkomende datavector op de ruisruimte te berekenen. In het nieuwe schema met hoeken kan dit zelfs veel eleganter dan in het oude schema waar de matrices expliciet bewaard worden. Daar vergt de berekening van de geprojecteerde vector een sequentie van twee matrix-vectorvermenigvuldigingen en een normalisatie. In de SSG geeft dit aanleiding tot tegengestelde datastromen. Met de parameterisatie is de berekening veel eenvoudiger en vergt geen tweezijdige datastroom. De rotatieknopen in de laatste kolom berekenen hun hoek door één coördinaat nul te maken.

De SSG bestaat opnieuw uit twee delen. Het bovenste trapezoïdaal gedeelte voert een matrix-vector vermenigvuldiging uit. Het onderste gedeelte is niet langer driehoekig. Eén rij van rotatieknopen volstaat. Vertrekkend vanuit deze nieuwe SSG worden twee systolische architecturen afgeleid. De eerste architectuur is een lineaire rij van $D+1$ processoren en heeft een pijplijningsperiode van $2M-1$ cycli. Deze architectuur kan eenvoudig afgeleid worden door alle knopen in eenzelfde kolom van de SSG aan dezelfde processor toe te wijzen. De uitvoeringstijd volgt dan uit de data-afhankelijkheden.

De tweede architectuur is planair. Elke knoop in de SSG wordt aan één enkele processor toegewezen. De pijplijning van deze architectuur is meer complex. Algoritmische transformaties zijn opnieuw onontbeerlijk om het kritische pad van lengte $\mathcal{O}(M)$ verder op te delen. Het eindresultaat is minder elegant dan voor het recursieve Jacobi SWO algoritme. De algoritmische transformaties introduceren nieuwe rotaties die in dit geval niet eenvoudig toegewezen kunnen worden aan bestaande knopen zonder de regelmaat en de localiteit van de SSG te verstoren. Dit is de prijs voor de verhoogde doorvoersnelheid (1 cyclus) van de planaire architectuur.

## Hoofdstuk 5. Een robuuste adaptieve LCMV straalvormer

In dit hoofdstuk stellen we een robuuste adaptieve straalvormer voor. De parallelle implementatie van dit algoritme kan heel efficiënt gebeuren op de Jacobi architectuur voor recursieve SWO.

Een belangrijk praktijkgericht probleem van adaptieve straalvormers is hun hoge gevoeligheid voor perturbaties van de elementen in de gelijkheidsbeperkingen. Vooral wanneer de signaal-ruishouding aan de ingang van de antennes hoog is, is de verslechtering van de signaal-ruisverhouding aan de uitgang aanzienlijk. Eén van deze vectoren – en vaak ook de enige – is noodzakelijkerwijze de richtingsvector van het gewenste signaal. In de praktijk zal de echte richtingsvector altijd wat afwijken van de richtingsvector gebruikt in de beperkingsmatrix door onnauwkeurigheden in de positionering van de sensoren, amplitude- en fazefouten in de antennes,...

Om deze vector zo accuraat mogelijk te kennen, stellen we voor om deze vector continu te schatten uit de data [10]. Dit kan bijvoorbeeld in communicatietoepassingen waar vaak een referentiesignaal voorhanden is. Op voorwaarde dat ruis- en eventuele interferentiesignalen ongecorreleerd zijn met het referentiesignaal, is de kruiscorrelatievector van uitgangen en referentiesignaal evenredig met de echte richtingsvector. Mits de schatting voldoende nauwkeurig gebeurt, wordt de robuustheid van de straalvormer sterk verhoogd. Een tweede voordeel is dat de straalvormer in staat is om een bewegende bron te volgen zonder het traject vooraf te kennen.

De wiskundige formulering leidt tot een kleinste kwadratenprobleem met een tijdsafhankelijke beperking en eventueel nog constante beperkingen. In het inleidend hoofdstuk hebben we al een parallelle architectuur besproken voor adaptieve straalvorming. Het bovengedeelte is een matrix die de beperkingen implementeert en het ondergedeelte is een adaptieve driehoekige kleinste-kwadraten-processor. Bovenop deze structuur moet nu een adaptatieschema geïmplementeerd worden. Het adaptatieschema zorgt ervoor dat de geschatte kruiscorrelatievector tussen uitgangen en referentie altijd in de laatste kolom behouden blijft. Dit kan opnieuw gebeuren door tweezijdige rotaties. De kolomrotaties zorgen voor de alignering van de kolommen. Ze moeten ook toegepast worden op de matrix $R$ van de QR decompositie. Hierdoor verliest deze matrix zijn driehoeksvorm. Rijrotaties zijn noodzakelijk om de ingevulde elementen onder de diagonaal weer weg te werken.

Dit algoritme is structureel haast identiek aan het recursieve Jacobi SWO algoritme. Het bestaat ook uit een sequentie van een matrix-vector-vermenigvuldiging, een recursieve QRD stap en tweezijdige orthogonale rotaties. Daarom kan het onmiddellijk afgebeeld worden op de Jacobi architectuur. Het verdient ook de aandacht dat de factorisatie van orthogonale matrices uit vorige hoofdstukken ook hier haar nut kan bewijzen.

## Hoofdstuk 6. Schatting van tweedimensionale spectraallijnen

In de laatste hoofdstukken van dit proefschrift gaat de aandacht naar het schatten van invalshoeken. In dit hoofdstuk stellen we een nieuwe efficiënte methode voor voor het schatten van invalshoeken van smalbandige signalen waarbij hun draagfrequenties niet gekend en verschillend kunnen zijn. Dit is een toepassing van tweedimensionale (2-D) spectraalschatting. De koppels $(\phi_i, \psi_i)$ zijn gerelateerd tot de invalshoek $\theta_i$ en de draagfrequentie van het signaal $f_i$

$$\begin{aligned}\phi_i &= \exp(j2\pi f_i \Delta \sin(\theta_i)/c)\\ \psi_i &= \exp(j2\pi f_i T_s).\end{aligned}$$

De efficiëntie van het voorgestelde algoritme is te danken aan twee eigenschappen. Een eerste eigenschap is de separabiliteit van het datamodel. Hierdoor kan het 2-D estimatieprobleem opgelost worden als twee 1-D estimatieproblemen. Een complicatie is wel het combineren van de twee verzamelingen van 1-D schattingen. We tonen aan dat extra berekeningen vermeden kunnen worden door een juiste keuze van transformaties. Een tweede eigenschap is de Vandermonde structuur van de datamatrix. Hierdoor kunnen efficiënte matrixdecomposities gebruikt worden om de frequenties te schatten.

De datamatrix $Z \in \mathbb{C}^{N\times M}$ is sterk gestructureerd

$$Z = X_N \cdot A \cdot Y_M^T + W.$$

De matrices $X_N \in \mathbb{C}^{N\times D}$ en $Y_M \in \mathbb{C}^{M\times D}$ zijn Vandermonde matrices in $\phi_i$ en $\psi_i$ respectievelijk. $A \in \mathbb{C}^{D\times D}$ is een diagonale matrix met de amplitude en faze van elke component. De matrix $W \in \mathbb{C}^{N\times M}$ bevat additieve ruis.

Eerst beschouwen we het geval waarin alle horizontale respectievelijk verticale frequenties verschillen. In het ruisloze geval is de matrix $Z$ van lage rang ($D$). Met ruis is de eerste stap van het algoritme een SWO om de kolom- en rijruimten te schatten. Omwille van de Vandermondestructuur van $X_N$ en $Y_M$ kan de stap van deelruimte naar frequentieschattingen opnieuw gebeuren door matrixdecomposities. Het ESPRIT algoritme [92] is hiervoor de aangewezen kandidaat. Uitvoering van dit algoritme volgens de horizontale en verticale richting geeft twee verzamelingen schattingen. Het paren van de juiste frequenties vergt in principe $\mathcal{O}(D^2)$ bewerkingen. Deze extra berekeningen kunnen vermeden worden door een algebraïsche koppelingsmethode [115]. De eigentransformaties van zowel het horizontale als verticale eigenwaardenprobleem kunnen identiek gemaakt worden door keuze van bepaalde deelmatrices van $Z$. De winst in berekeningstijd is beduidend. Er moet maar één eigenwaardenprobleem meer opgelost worden. Toepassing van dezelfde eigentransformaties op de tweede matrix geeft de complementaire frequenties in de juiste volgorde.

In het geval dat een bepaalde component in meerdere frequentieparen voorkomt, faalt het beschreven algoritme. In de corresponderende Vandermondematrix zijn twee kolommen identiek. Hierdoor zakt de rang tot $D-1$. Dit probleem kan opgelost worden door uitmiddeling. Dit kan door op basis van de oorspronkelijke data een grote matrix $J$ te construeren die opnieuw van volle rang $D$ is. Bovendien moet $J$ nog altijd separabel zijn en een Vandermonde-achtige structuur behouden. In het hoofdstuk construeren we een 'dubbele' blokhankelmatrix die aan beide voorwaarden voldoet. Deze techniek heeft de eigenschap dat de dimensies van de matrix enkel vermenigvuldigd worden met de maximale meervoudigheid van een frequentiecomponent. Dit is een verder voordeel ten opzichte van bestaande methodes.

## Hoofdstuk 7. Localisatie van breedbandige signalen met toestandsruimtemodellen

In dit hoofdstuk bestuderen we een verdere uitbreiding van het datamodel naar breedbandige stochastische signalen. We stellen een klasse van deelruimte-algoritmen voor die gestoeld zijn op toestandsruimtemodellen van de breedbandige signalen. Deze aanpak laat toe om opnieuw op een vrij eenvoudige manier een efficiënt parallel recursief algoritme te ontwikkelen.

Over het breedbandige probleem is veel minder literatuur verschenen dan over het smalbandige. Nochtans zijn er ook belangrijke toepassingen zoals localisatie van akoestische bronnen. De meeste methodes steunen op de opsplitsing van het breedbandige signaal in meerdere smalbandige signalen. De schattingen van al deze deelproblemen worden dan gecombineerd. Een alternatief is de breedbandige signalen te modelleren met behulp van toestandsmodellen geëxciteerd door onafhankelijke witte ruissignalen. De signalen worden dan ontbonden in modes in plaats van frequenties. Op voorwaarde dat de bemonstering uniform is en dat de roosterantenne uit twee verschoven deelroosters bestaat, kan voor elk van deze modes de invalshoek geschat worden met behulp van matrixdecomposities.

In een eerste stap van het deelruimte-algoritme worden blokhankelmatrices geconstrueerd met de uitgangen van de twee deelroosters. Elk van deze datamatrices is een som van een term die lineair is in de toestand van het totale systeem, een term die bijdragen van de ingangen bevat en een ruisterm. Enkel de toestandsterm bevat informatie over de ligging van de systeempolen en de locatie van de bronnen. Daarom moeten de ingangs- en ruistermen afgescheiden worden. Uit het stochastisch karakter van de signalen volgt dat dit kan gebeuren door een projectie op verleden uitgangen. Met de geprojecteerde matrices kunnen, precies zoals in het vorige hoofdstuk, twee algebraïsch gekoppelde eigenwaardenproblemen opgesteld worden. Uit de eigenwaardeparen kan tenslotte de invalshoek van elk van de breedbandige bronnen bepaald worden.

Dit deelruimte-algoritme bestaat uit een opeenvolging van een projectiestap en meerdere eigenwaardedecomposities op vier matrices. Omwille van numerieke stabiliteit en precisie is het goed de projectiestap uit te voeren via een QR decompositie. Eigenwaarden kunnen ook berekend worden door een andere orthogonale transformatie, nl. de Schur decompositie. Omdat het hier in feite om veralgemeende eigenwaardenproblemen met matrixparen gaat, is de veralgemeende Schurdecompositie nodig. Uiteindelijk kan het volledige algoritme opgebouwd worden met enkel orthogonale transformaties.

Het belang van deze opbouw schuilt in de afleiding van een recursief algoritme. Het uiteindelijke resultaat is een Jacobi schema dat simultaan rij- en kolomrotaties op de vier matrices uitvoert. De datastroom blijft echter identiek aan het recursieve Jacobi SWO algoritme. Daarom vergt de implementatie op de Jacobi architectuur enkel herprogrammering van de processoren.

## Hoofdstuk 8. Een parametrische methode voor localisatie van bronnen in onbekende omgevingsruis

Het laatste hoofdstuk introduceert een nieuwe aanpak voor een belangrijk praktijkgericht probleem. Alle hoge-resolutiemethoden voor het schatten van invalshoeken van smalbandige signalen gaan uit van de veronderstelling dat de ruis van sensor tot sensor ongecorreleerd is. In de praktijk is dat nooit zo. Als alle sensoren dezelfde karakteristieken hebben, is dit een goed model voor ruis intern gegenereerd in de sensoren. Daartegenover staat dat omgevingsruis die invalt op de roosterantenne altijd gecorreleerd is.

Men kan corrigeren voor de ruiscorrelatie door de signalen eerst te filteren met de inverse van de ruiscorrelatiematrix. Daarom zijn methodes om goede schattingen van de ruiscorrelatiematrix te vinden van groot belang. Onze aanpak is modelgebaseerd. We ontwikkelen een model voor de ruiscorrelatiematrix dat gebaseerd is op een Fourieranalyse van het ruisvermogen in functie van de invalshoek. Volgens dit model is de ruiscorrelatiematrix een lineaire combinatie van een aantal basismatrices met onbekende coëfficiënten. De basismatrices corresponderen met een term in de Fourierexpansie. Ze kunnen vooraf berekend worden op voorwaarde dat de transfertfunctie van de roosterantenne gekend is. De lineaire parameters zijn de Fouriercoëfficiënten. Dit model is aantrekkelijk omwille van drie aspecten. Ten eerste is het fysisch betekenisvol. De coëfficiënten zijn niet zomaar fittingparameters, maar bevatten informatie over de plaatsafhankelijkheid van de ruis. Ten tweede is dit model toepasbaar op antennes met een willekeurige geometrie. De meeste ruismodellen in de literatuur zijn enkel geldig voor regelmatige roosterantennes. Tenslotte is er de eenvoud van de parameterisatie. De elementen van de ruiscorrelatiematrix zijn lineaire functies van de parameters. Dit vergemakkelijkt de algoritmen.

Een belangrijk aspect is de uniciteit van de parameterisatie. Het is echter moeilijk onder- of bovengrenzen op het aantal uniek identificeerbare Fouriercoëfficiënten te geven. Uit een studie van de topologie van het probleem volgt dat het aantal oplossingen gelijk is aan het aantal matrices van lage rang die aan twee lineaire matrixongelijkheden voldoen. Het is bekend dat dit probleem in zijn algemeenheid NP-compleet is [16].

Om de parameters te schatten worden twee methodes besproken. De eerste aanpak steunt op de theorie van maximale waarschijnlijkheid (*E. maximum likelihood*). Uit de afleiding volgt dat de optimisatie van ruis- en signaalparameters niet gescheiden kan worden. De oplossing vergt daarom een niet-lineaire optimisatie in een hoogdimensionale ruimte. Dit is rekentechnisch niet interessant. Daarom opteren we voor een tweede lichtjes suboptimale aanpak. Eerst worden de ruisparameters via niet-lineaire optimisatie geschat. Met deze optimale ruiscorrelatiematrix worden de uitgangen gefilterd. Tenslotte kunnen de signaalparameters zoals invalshoek en signaalvermogen geschat worden aan de hand van de klassieke hoge-resolutie-algoritmen.

Het nieuwe optimisatiecriterium volgt uit de theorie van maximale waarschijnlijkheid voor een vereenvoudigd datamodel waarin de structuur van de richtingsvectoren buiten beschouwing gelaten wordt. Zolang het aantal bemonsteringen groot genoeg is, blijft het verschil met optimale maximale waarschijnlijkheid klein. Het criterium is een maat voor de gelijkheid van de kleinste eigenwaarden van de ruiscorrelatiematrix. Het is de verhouding van het geometrisch tot het arithmetisch gemiddelde van deze eigenwaarden. Deze verhouding komt ook voor in detectiecriteria voor de bepaling van het aantal bronnen zoals MDL en AIC [147]. Aan de hand van simulaties wordt aangetoond dat dit algoritme een goede performantie en robuustheid bezit.

## Hoofdstuk 9. Conclusies en open vragen

In dit proefschrift ging de aandacht naar recursieve algoritmen voor digitale signaalverwerking van roostersignalen in reële tijd. Numerieke aspecten waren zeer belangrijk. Omwille van de combinatie van aanzienlijke berekeningskost en hoge datasnelheden hebben we de implementatie van deze algoritmen op parallelle architecturen bestudeerd.

De (recursieve) singuliere waardenontbinding vormt de ruggegraat van heel wat voorgestelde algoritmen. In een eerste luik hebben we een numeriek meer betrouwbare variant op het Jacobi algoritme voor recursieve SWO afgeleid. Dit algoritme bood inspiratie voor de afleiding en parallellisatie van volledige signaalverwerkingsalgoritmen zoals robuuste straalvormers en localisatie-algoritmen. Voor invalshoekschatting hebben we algoritmen voor smalbandige signalen uitgebreid naar het geval van breedbandige signalen. Tenslotte is ook een niet-recursief algoritme voorgesteld voor invalshoekschatting in omgevingen met onbekende gekleurde ruis.

Tot slot sommen we enkele grote lijnen op voor verder onderzoek. De grote krachtlijn hierbij is een verdere oriëntering naar de specifieke eigenschappen van de toepassing. Dit zal een belangrijke verdere optimalisering van de performantie van de algoritmen toelaten. De algoritmen zijn nu vooral gegroeid vanuit de theoretische studie van simultaan ontwerp van algoritmen en architecturen en vanuit de digitale signaalverwerkingstheorie. In de praktijk betekent dit dat een aantal veronderstellingen waarop de algoritmen steunen maar gedeeltelijk voldaan zijn. Een voorbeeld zijn de cellulaire mobilofoniesystemen. Het signaal van de mobiele zender bereikt het basisstation via meerdere paden met verschillende sterktes en fazeverschuivingen. Door dit coherente multipad verschijnsel verliest de antenneresponsievector zijn spatiale structuur en kan er geen hoek meer eenduidig mee geassocieerd worden. Het zenden vanuit het basisstation naar de mobiele zender stelt ook specifieke moeilijkheden. Het zend- en ontvangstkanaal verschillen in frequentie (45MHz in GSM). Hierdoor zijn er ook verschillen in de responsievectoren. Daarom kan de ontvangstresponsievector niet zomaar gebruikt worden voor de berekening van de zendgewichtsvectoren.

Op het domein van de mobiele telefonie-applicatie is er nood aan een algemeen aanvaard, realistisch, maar mathematisch hanteerbaar model voor het propagatiekanaal van mobiele zender naar roosterantenne. Voorlopig zijn heel weinig metingen beschreven [76]. Men kan het model wel theoretisch uitbreiden vertrekkend van bestaande modellen voor één antenne.

Op het domein van de algoritmen zijn er twee grote lijnen. Ten eerste is een statistische gevoeligheidsanalyse van de algoritmen noodzakelijk. Kleine verschillen in de transfertfunctie van de ontvangstapparatuur of onnauwkeurigheden in de plaats van de antenne-elementen zorgen voor afwijkingen tussen het model en het werkelijke systeem. In dit proefschrift lag de nadruk op simultaan ontwerp van algoritmen en architecturen. Testen van de algoritmen gebeurde enkel op basis van simulaties. Een nadeel van deze – zij het algemeen aanvaarde – methode is de moeilijke veralgemeenbaarheid van de resultaten. Een asymptotische analyse in eerste orde heeft een algemenere geldigheid.

Anderzijds kan er nog in belangrijke mate gesleuteld worden aan de algoritmen zelf. In communicatietoepassingen is men niet zozeer geïnteresseerd in de exacte locatie van de zender, maar in de kwaliteit van de signaalreconstructie. Omwille van het stochastisch karakter van het propagatiekanaal is de spatiale structuur van de signalen niet goed gedefinieerd. Maar communicatiesignalen zijn veelal sterk gestructureerd. Vele signalen bezitten een constante modulus of hebben een digitale structuur. Deze signaalstructuur wordt vrij goed behouden na propagatie. Recent is er onderzoek verricht naar algoritmen die op basis van de partiële kennis van de signaalstructuur het propagatiekanaal schatten [109, 117]. Dit is nog maar een aanzet. Verder onderzoek op deze matrices van lage rang met zowel een (niet-lineaire) structuur op kolomruimte als op de rijruimte zal naar alle verwachting een beduidende verbetering van de performantie brengen.

# Chapter 1

# Introduction

## 1.1 Motivation

*Array signal processing* is the branch of digital signal processing which studies propagating wavefronts sampled in time and space by sensor arrays. The most common sensor types are antennas, listening to electromagnetic radiation, and microphones, registering acoustic pressure waves. The use of arrays of sensors has a long history. Important applications include radar and sonar systems, radio astronomy and satellite communication [44]. However, nowadays they attract a lot of attention for terrestrial communication [2, 106]. The main reason is the expansion of the market for wireless communication systems, such as cellular mobile telephony. The success of the pan-European digital GSM system (Global System for Mobile communications [72]) operational in Belgium since January 1994 is illustrative. Also wireless data networks gain importance [4]. The huge demand for user capacity requires economical use of the scarce spectral resources. Several techniques are in use to maximize the capacity.

First, GSM is a cellular system as illustrated in Figure 1.1. The propagation environment for mobile communication is such that electromagnetic power evolves inversely proportional to $r^{3\cdots4}$ [52], where $r$ is the distance to the emitter. Because of this fast decay, frequencies can be reused as soon as a distant signal has become too weak to disturb the communication with a nearby user. Therefore, the area to be serviced can be divided into non-overlapping cells. Inside each cell mobile users, such as phones installed in cars or hand sets held by pedestrians, communicate with a central base station. When a user leaves a cell, the communication is switched to a new base station by a hand-over procedure.

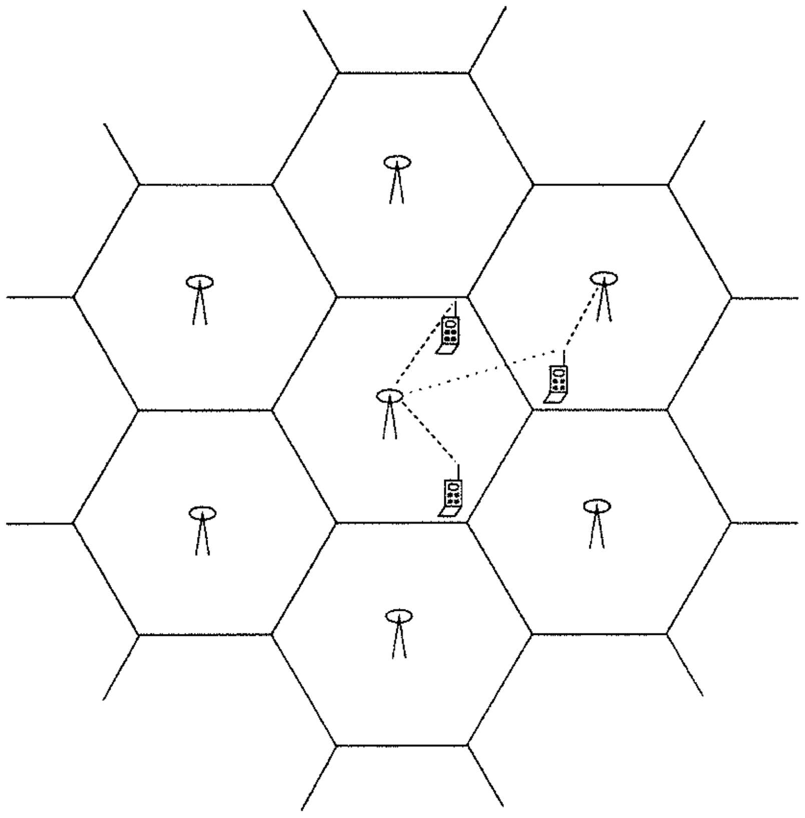


Figure 1.1: A cellular communication system. The service area is covered with cells. All mobile users inside a cell communicate with a central omnidirectional base station on different channels (dashed lines). Users from neighboring cells on a same channel are perceived as interferences (dotted line).

Secondly, GSM makes use of time and frequency division multiple access (TDMA/FDMA). Two frequency bands of 25 MHz, one for the uplink and one for the downlink, have been allocated around 900 MHz. Inside a band there are 8 time slots and 124 frequency slots of 200 kHz available. In each cycle of 8 slots, a typical channel carrying digitized voice occupies one time slot on one frequency slot. Therefore, in principle 992 users could be serviced simultaneously inside a cell. The actual number is much lower. First, in all countries there are (or will be) several system operators who run competing GSM systems in parallel. Therefore, an operator only gets part of the frequency slots. Also, frequencies cannot be reused in neighboring cells. As seen in Figure 1.1 a user acts as an interference source in surrounding cells. To keep the interference level sufficiently low, frequencies are only reallocated to the cells of the second tier (layer). This further divides the capacity with a factor 7. In an area where 2 operators coexist, the resulting maximal capacity per cell is roughly 70 users.

It is clear that this upper limit can often be reached in busy areas. A long-term solution is the installation of a new GSM-like system (DCS1800) in the 1800 MHz band where two 75 MHz band will be available. However, this requires a complete redesign of the system, whereas congestion often occurs in a limited number of busy cells. These local saturation problems can currently only be handled by subdividing a cell into smaller cells, in the limit leading to microcells only covering a few street blocks. This approach has important disadvantages. The investment in expensive base stations becomes prohibitive and frequent hand-overs have to be performed.

Recently, antenna arrays have been recommended to provide a cheap local solution to such a local capacity problem [2, 73, 106]. Currently, the omnidirectional base station has no knowledge of the position of the mobile user. The communication messages are broadcast omnidirectionally over the cell. They are received by all mobile users and each user selects its own messages. Broadcasting requires substantial power and possibly creates severe interference in the neighboring cells. In fact, it is more optimal to establish several point to point communication links. This can be achieved by directional transmitting and receiving by means of antenna arrays at the base station. This has several advantages. For the same reception level at the mobile user, the power emitted by the base station can go down. This in turn lowers also co-channel interference. And last but not least, array signal processing adds the possibility of spatial division multiple access (SDMA). Several users can be put onto the same frequency and time slots and still be discriminated according to their location. If three users with different positions are multiplexed onto the same channel, then the capacity of a cell is tripled. Only in cells suffering from saturation a more complex base station benefiting from antenna array technology has to be installed. Cells where no congestion occurs, remain unaffected. Also, the antenna array and the additional signal processing equipment are entirely part of the base station hardware.

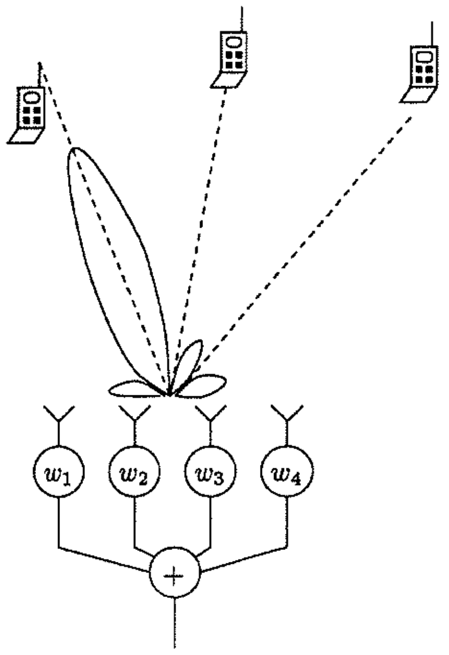


Figure 1.2: Spatial division multiple access. For each co-channel user a weighted linear combination of the antennas is computed. The associated directivity pattern of the weight vector $w$, indicating the directional array gain, exhibits a beam (maximal reception) pointing towards the desired user and a null (zero gain) along the directions of co-channel users.

The mobile transceivers are not altered.

Two operations are needed for directional air links. First combinations of the antenna elements have to be computed such that a focusing in space is attained. As shown in Figure 1.2 this is achieved by linearly combining the antenna outputs. The weights are determined such that the array is maximally receptive to the signal from the desired user, while all signals impinging from other directions are suppressed or at least sufficiently attenuated. This spatial filtering structure is often called a beamformer. If there is only one mobile user with a known position, then the weight vector can be computed by least squares estimation techniques. If there are multiple users with unknown locations, then eigendecomposition based techniques have to be used to determine the number of users and estimate their location. This position information is impossible to obtain with a single omnidirectional antenna.

As an alternative a similar spatial selectivity can be obtained by a single sensor with a large continuous aperture. Examples of such sensors are the parabola antennas used in radio astronomy [44]. However, this approach is often undesirable. The technical difficulties and the equipment cost increase more than proportional with the aperture of such a large sensor. Moreover, array processing offers the user important flexibility. It is well known that a single antenna with a continuous aperture is only optimal in a spatially white noise environment [153]. Digital processing of array signals allows to take into account arbitrary noise correlation. It also adds the capability to steer notches (zeros) to cancel known interferences and jamming signals. And the adaptation of the weights is done electronically, without physically moving the sensors. Therefore, the array can continuously adapt to changes in its environment, such as moving users or short time interferences. Finally, arrays allow to attain a resolution which is higher than the one of continuous aperture antennas of comparable size.

The last decade has witnessed a large activity in the field of digital array signal processing. New concepts have been brought forward and many advanced algorithms have been developed. The main break through is the fact that recent algorithms heavily rely on a mathematical model for the antenna outputs. The geometry of the data model is expressed in terms of concepts from *linear algebra*. The outputs of the array at a certain sampling time form a vector of observations called a snapshot. Similarly the signals from a single sensor observed over time are collected in vectors. The observations of the sensor array over a time interval are stored in matrices. As mentioned earlier, optimal weight vectors can be computed based on solving a set of linear equations (least squares estimation) and the position of the users is hidden in certain invariant subspaces of the data matrix. Therefore, matrix decompositions, such as the QR decomposition (QRD) and the singular value decomposition (SVD), play an important role.

In addition to being the natural framework for modeling the data, linear algebra also provides the algorithm designer with a whole set of robust algorithmic building blocks. Advanced algorithms offering improved performance can often be built up as a sequence of matrix decompositions. The trade off is the increase in computation with respect to simpler, not model-based algorithms. Therefore, execution of these algorithms in real time necessitates the use of powerful computers.

Moreover, throughput rates in the front end processing of antenna array systems are often extremely high. In the GSM system the bit duration is only 3.7 $\mu$sec, which yields a bit rate of 270.3 kbit/s. At this speed the signal reconstruction has to be performed. It consists of filtering the antenna outputs and performing equalization in time. The adaptation of the filter coefficients can be done at a lower speed. The position of users only changes very slowly compared to the bit rate. However, the propagation environment changes rapidly. The mobile user evolves at low height – his antenna is typically at 2 to 3 m – in an environment full of large structures, such as tall buildings or hills. Due to reflection and diffraction, a signal from a mobile reaches the antenna array via various paths with different gains and delays. In a mountainous area these multipath components may have delays up to 15 $\mu$sec ($\pm 4$ bit periods) which corresponds to a difference in path length of 4500 m. In urban areas maximum delays of 10 $\mu$sec are observed [15, 97]. Therefore, equalization is needed. But the interference of the multipath components also causes fast fading of the signal power. The fast fading may vary substantially in as little as half a wavelength (16.6 cm at 900 MHz) [5]. For a car driving at 120 km/h, it takes 5 ms to travel this distance. This number can be taken as a measure for the rate of change of the mobile channel. A cycle (sequence of 8 time slots) in GSM lasts 4.65 ms. Therefore, the GSM system recomputes the $\pm$ 5 equalization filter taps in each cycle, based on a sequence of 26 training bits in the middle of a burst.

At these speeds, even today's fast digital processors have a hard time performing complicated digital signal processing tasks on multiple channels in real time. Single dedicated or application specific processors may be able to provide the required computing power. If this is not the case, parallel computing is the natural way to further increase the execution speed. This is not an easy issue. In order to obtain a good speed up, one has to carefully balance computation, communication between processors and memory access. Of course, the constraints imposed by the architecture of the particular parallel machine under consideration have to be taken into account. Important distinctions can be drawn between synchronous and asynchronous machines and between global memory and distributed memory machines [26]. This double distinction leads to four classes of architectures with a different style of programming. Networks of workstations belong to the class of asynchronous distributed memory machines. Application specific systolic arrays are clearly a member of the class of synchronous architectures with distributed memory. However, these features are only important at a later stage of the mapping from algorithm to architecture.

Whatever the target architecture may be, modern parallel design methodologies begin with a data flow analysis of the algorithm [50]. The flavor of these methodologies is mainly graphical. First a dependence graph (DG) of the algorithm is constructed. This is an acyclic graph uncovering the order in which signals are generated and consumed by operations. An alternative, more compact representation is the signal flow graph (SFG) which contains loops, *i.e.,* memory. Various transformations can already be applied to these graphs to make them more suitable for mapping onto a parallel machine. In a second phase a processor and an execution time have to be associated with each node (operation) in the graph. Assigning nodes to a physical processor is called the placement step. This step depends of course on the architecture of the particular machine under consideration. Determining an execution time for each node is called the scheduling step. Here the important issue is to respect the ordering imposed by the dependencies. A signal cannot be used in the computation before it has been calculated.

A large class of matrix algorithms is well suited for parallel execution because of their intrinsic regularity. Vectors and matrices are regular data objects. Also the processing is often shift invariant, *i.e.,* the same operations are to be performed to neighboring entries of the matrix. Moreover, sometimes the operations on a certain entry only require knowledge of the previous value of the entry and its neighbors. This property gives rise to local dependencies in the DG. Such algorithms are ideal candidates for implementation on systolic arrays. A systolic array is a special purpose synchronous architecture in which the processors are interconnected to nearest neighbors in a regular pattern. This type of architecture is attractive because of the extreme degree of pipelining and its regularity which eases a VLSI implementation. However, it should be noted that many implementations of algorithms for which a systolic architecture exists will not be systolic, *e.g.,* asynchronous communication or global memory may be used. A first reason is that it is unlikely that an algorithm has to be executed at such high data rates that the ultimate pipelining is needed. In general, the throughput requirements can already be met at a lower cost by allocating large parts of the computation to a few processors. A second reason is that a systolic array is parameterized by the problem size, *e.g.,* the size of the antenna array. Often problems with different sizes have to be solved on the same architecture. The implemented architecture with a fixed size must then have an additional control to allocate computation over time and processors. Therefore, the attribute 'systolic' is more a property of the algorithm, indicating that its SFG is homogeneous and regular with local dependencies such that a fully systolic architecture is feasible. In this sense, one should realize that the (systolic) architectures which will be presented later on, are not really a specification of a physical parallel machine, but a virtual parameterized architecture from which an optimal physical parallel machine can be derived.

Currently formal methodologies and interactive software tools are under development to assist the designer in developing a special purpose array processor for his algorithm [23, 62, 84]. This simultaneous design of the signal processing algorithm and the parallel architecture on which it is to be executed, offers the largest freedom to obtain an optimal fit between algorithm and architecture. The corresponding discipline of deriving stable versions of algorithms which are well suited for parallel implementation by transformations on the SFG is often labeled *algorithmic engineering*[1] [57, 58, 83].

Economic considerations determine whether the development cost of an application specific parallel architecture is affordable or not for a particular application. In the near future the advent of semi-automated parallelization software may cut this cost considerably. In any case, whether an algorithm is to be executed on an application specific or on a general purpose parallel machine, reconsidering existing algorithms to increase their parallelism and regularity is an important issue. In a design it is generally the rule that the largest optimizations can be obtained at the highest level, *i.e.,* the algorithmic level.

## 1.2 General survey

As shown in Figure 1.3 two themes recur throughout this text. The first theme is the development of new algorithms for array processing applications. The second one is parallel architectures. Chapters proposing algorithms for similar applications are ordered sequentially. First chapter 2 introduces the necessary background material. Then chapter 3 and 4 pro-

[1]McWhirter and co-workers use the term 'algorithmic engineering' for transformations on the SFG on a high level. Here the term denotes the complete field of simultaneous design of algorithms and architectures.

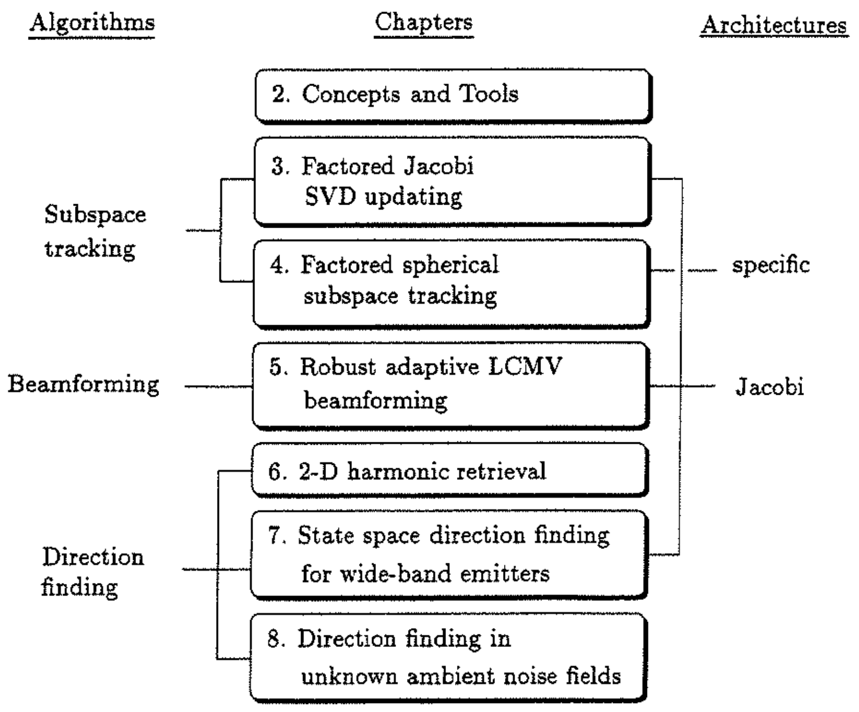

Algorithms
Chapters
Architectures
2. Concepts and Tools
3. Factored Jacobi SVD updating
4. Factored spherical subspace tracking
5. Robust adaptive LCMV beamforming
6. 2-D harmonic retrieval
7. State space direction finding for wide-band emitters
8. Direction finding in unknown ambient noise fields
Subspace tracking
Beamforming
Direction finding
specific
Jacobi

Figure 1.3: Relations between chapters 2 to 8

pose algorithms for *subspace tracking.* Tracking the dominant subspace of the data matrix will be a key component of recursive direction finding algorithms. The difference between the algorithms in chapter 3 and 4 is their computational cost and the accuracy with which they track the subspace. Chapter 5 introduces an adaptive beamformer with increased robustness to errors in the constraints. These constraints influence for instance the position of the main lobe of the beamformer. An example is a mobile user whose position estimate is subject to estimation errors. Insensitivity of the beamformer to this type of errors is an important asset. Finally, chapters 6 to 8 treat the direction finding problem under various assumptions. In the standard narrow-band data model there are multiple signals modulated onto the same known carrier frequency. In chapter 6 a subspace algorithm is proposed for the situation in which the carrier frequencies are not known. Chapter 7 presents a further generalization of the data model. Now the signals are assumed to be wide-band signals.

Such a model is needed for the processing of acoustic signals, such as speech or audio. Finally, chapter 8 returns to the standard narrow-band model for the signals, but proposes a extension of the noise model. The often unrealistic assumption that the sensor noise is spatially white, is relaxed to the assumption that the noise correlation matrix belongs to a well chosen linear model class. We also discuss an algorithm to estimate the additional noise parameters.

We will be especially interested in recursive algorithms for real time execution. Because of our concern for parallel implementation, the algorithms are composed of matrix decompositions. More irregular operations, such as nonlinear optimization methods, are avoided as much as possible. An exception is the colored noise algorithm of chapter 8 for which nonlinear optimization is necessary. But there, we will not be looking for a recursive algorithm either.

In the text four recursive algorithms will be mapped onto a parallel architecture. Our approach is not to come up with a totally different architecture for each new algorithm. Instead three of the four algorithms will be mapped efficiently onto the same architecture, *i.e.*, the Jacobi array which was originally developed for SVD updating [66, 67]. The fourth algorithm is mapped onto a related but simplified architecture. The wide applicability of the Jacobi architecture is at first sight surprising. However, several recursive signal processing algorithms for seemingly widely different applications can be formulated as Jacobi algorithms. Here we take the freedom to call an algorithm a Jacobi algorithm[2] if its structure is identical to the structure of the Jacobi SVD updating algorithm to be discussed in chapter 2, *i.e.*, it consists of a sequence of matrix-vector multiplication, QRD updating and a series of two-sided Givens rotations. Their data flow is identical. Only the node descriptions specifying the input output mapping, may differ. Examples are recursive algorithms for SVD and its generalizations, such as quotient SVD, product SVD [60] and the URV decomposition [68, 102]. Also other signal processing algorithms such as inverse recursive least squares (RLS) [61] and narrow-band direction finding [71] have been formulated as Jacobi algorithms. Here we extend this class by three more algorithms: a stable Jacobi SVD updating algorithm without reorthogonalization, a beamforming algorithm and a direction finding algorithm for wide-band sources. The fact that a rela-

[2]Originally the term Jacobi algorithm refers to a symmetric eigenvalue solver using two sided rotations [42]. Later the term was also adopted to denote the Kogbetlianz algorithm which uses two sided rotations to compute the SVD [48].

tively large class of relevant signal processing algorithms can be mapped onto the same Jacobi architecture, makes it much more attractive for implementation in hardware. The only requirement is that this architecture is not fully hard wired, but instead slightly programmable. It provides a limited set of operations, mainly rotations, and the user has to program certain actions, such as the way the rotation angles are determined. The design of such an array is currently taking place [24, 118].

## 1.3 Chapter overview

Below we give a detailed overview of the text. The main contributions of each chapter are summarized.

### Chapter 2. Concepts and tools

In this introductory chapter we give an outline of the background material from the fields of *array processing*, *linear algebra* and *algorithmic engineering*. These are the three major areas on which this text is based. The chapter gradually introduces the field of array processing. First the data model for narrow-band sources impinging on an array is developed. Next some algorithms for beamforming and direction-of-arrival estimation are presented. In later chapters we will propose modifications and extensions to these algorithms. The concepts of linear algebra and algorithmic engineering are introduced as they are needed.

### Chapter 3. Factored Jacobi SVD updating

This chapter presents a first algorithm for stable subspace tracking. As will be explained in chapter 2, the dominant subspace of the data matrix is determined by the position of the mobile users. When the mobiles are moving, this subspace changes. These variations have to be captured in order to track the position of the mobiles. SVD updating is well suited for subspace tracking. We propose a modification to the Jacobi algorithm for SVD updating. The original version of this algorithm requires periodic reorthogonalization in order to remain numerically stable.

The contribution of this chapter is the introduction of a minimal factorization for orthogonal matrices. An arbitrary orthogonal matrix $\mathbb{R}^{M\times M}$ can be factored as a sequence of $M(M-1)/2$ Givens rotations. This factorization is applied to the matrix of short singular vectors in the Jacobi SVD updating algorithm. This ensures the numerical stability of the algorithm, by excluding accumulation of round off errors in finite precision arithmetic. Moreover, this approach leads to a new Jacobi systolic architecture which solely consists of rotation nodes. This resemblance of the functionality of all nodes simplifies the hardware design of such a processor node.

The emphasis in this chapter is mainly on numerical linear algebra and architectures. However, updating the SVD is a key to the array processing algorithms to follow.

The material from this chapter is published in [135]. More concise descriptions are given in [136, 137].

### Chapter 4. Parallel spherical subspace tracking

The spherical SVD algorithm is a second subspace tracker. It differs from the Jacobi algorithm in that it requires substantially less computation. This is due to the fact that it only keeps track of the dominant subspace of the SVD and that the exact signal and noise singular values are averaged. In contrast to the Jacobi SVD algorithm, the spherical subspace tracker is non-iterative. The trade off is a slower convergence speed.

The original algorithm suffers from the same error accumulation as the Jacobi SVD updating algorithm. Therefore, we propose to apply the same factorization idea as in the previous chapter. This leads to a novel factored spherical subspace tracker. Based on its SFG, we obtain by a linear placement and a piecewise linear schedule a new linear architecture. Finally, a planar architecture is developed using algorithmic transformations. Unfortunately, due to some irregularities this planar array is less efficient than its counterpart for the Jacobi algorithm.

The derivation of the algorithm and the mapping onto a linear array are described in [128].

### Chapter 5. Robust adaptive LCMV beamforming

In chapter 3 we already proposed a modified Jacobi algorithm for SVD updating. This chapter introduces a second Jacobi algorithm for robust beamforming. The adaptive linearly constrained minimum variance (LCMV) beamformer is a popular choice for updating the weight vector of an antenna array, when a single signal-of-interest (SOI) with a known position is received in an unknown, possibly time-varying, noise and interference environment.

An application is a cellular communication system in which the base stations are equipped with antenna arrays, but no spatial multiplexing is applied. Once the position of the user is estimated, a beamformer can be computed such that interferences coming from neighboring cells are attenuated. Ideally in such a scheme the capacity could be multiplied by a factor close to 7 by reusing all frequencies in all cells [85]. The base station then has to cancel the co-channel interference from neighboring cells.

However, the LCMV beamformer is known to be very sensitive to errors in the constraint matrix caused for instance by inaccuracies in the location estimate. We propose to increase the robustness of the beamformer by recursively estimating the array response vector of the SOI. This is possible in applications such as GSM where a periodic training signal is available. The mathematical formulation leads to a recursive least squares problem with a time-varying constraint. Based on a known structure for LCMV beamforming with fixed constraints, *i.e.*, the generalized sidelobe canceler (GSC), a Jacobi-type algorithm can be formulated. Again, it can be mapped efficiently onto the parallel Jacobi architecture.

This chapter is based on [129]. In this reference also a second application specific systolic architecture is derived. Parts of the work are reported in [124, 125, 138]. An alternative approach to increase the robustness, based on SVD regularization, is given in [70].

## Chapter 6. 2-D Harmonic retrieval

The thesis is concluded with three chapters on direction-of-arrival (DOA) estimation of multiple signals. Each chapter makes different assumptions on the data or noise model. In the standard narrow-band model the sources have a known carrier frequency in common. In this chapter we assume that the carrier frequencies are not longer known. They may or may not differ and have to be estimated simultaneously with the DOAs.

An example is a broadband antenna array system for controlling the regulations on spectral allocation. In every nation there is a governmental institute distributing licenses for radio transmission. Each licensee has to comply with the regulations on power levels, bandwidth,... As shown in Figure 1.4 airborne array systems can be used to check if the regulations are observed in the field. Such a system has to scan the frequency band of interest, determine the location of all emitters and measure their spectral characteristics.

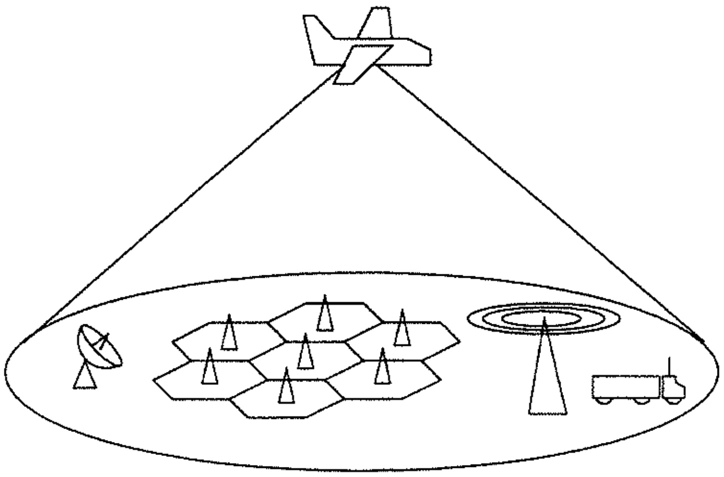


Figure 1.4: Airborne surveillance system monitoring all radio sources in a given spectral band, such as radio and TV broadcasting and wireless communication systems.

In this chapter we restrict our attention to a non-recursive algorithm for off-line processing. No architectural issues are discussed. The new 2-D harmonic retrieval algorithm is cheaper than existing algorithms. This is due to the fact that the 2-D problem is separated into two related 1-D estimation problems. For each 1-D estimation problem a matrix pencil has to be diagonalized. In fact, by careful selection of the matrices in the two pencils both 1-D problems are solved by the same transformations. Also an efficient rank restoration method is proposed to cope with situations where multiple signals have a common carrier frequency.

Part of this material can be found in [130].

**Chapter 7. State space direction finding for wide-band emitters**

In this chapter the assumption that the signals are narrow-band is relaxed. Instead the signals are modeled by a (low order) time-invariant linear system driven by white noise. Such a model is applicable to microphone arrays for processing acoustic signals. An interesting application is the development of a hands-free mobile telephony set [14, 75]. By use of a small microphone array a car driver could make a call without having to hold a handset. The microphone array serves to enhance the speech signal in the background noise.

Existing approaches to wide-band direction finding are mainly based on reducing the wide-band problem to a set of narrow-band problems with different center frequencies [104, 145]. We do not decompose the wide-band signals into different frequency bins. Instead the novelty of our approach lies in the use of state space descriptions for the sensor outputs. We combine the conceptually new subspace algorithms for identification of linear systems [63, 121] with array signal processing to simultaneously identify the system poles and the locations of the sources.

Two algorithms are introduced. The first is a non-recursive algorithm which is related to the 2-D harmonic retrieval subspace algorithm of chapter 6. Again the estimates of the system poles and the locations are computed as the rank reducing numbers of two related matrix pencils. The second algorithm is a recursive wide-band direction finding algorithm, which is structured as a Jacobi algorithm such that the Jacobi architecture can be used for parallel implementation.

This chapter is largely based on [134]. The relation between the number of antennas and the number of signals is reported in [127]. An alternative fast algorithm based on displacement structures is reported in [126]. Early summaries of the chapter are [132, 133].

### Chapter 8. A parametric approach to direction finding in unknown ambient noise fields

The flavor of this last chapter is different from the previous ones. It is written from the viewpoint of estimation theory and the emphasis is not on recursive algorithms for real time operation on parallel machines.

We consider again communication systems in which the signals are narrow-band waves. The standard data model assumes that the sensor noise is spatially white. This assumption is realistic for noise which is generated internally in the sensors. However, it is hard to defend for ambient noise impinging on the antenna array.

It is well known that the performance of subspace algorithms for direction finding is sensitive to unknown noise correlation [141]. Therefore, good methods to model and estimate the noise correlation matrix are crucial.

The main contribution of this chapter is a simple linear model for the ambient noise. If the response of the antenna array as a function of the direction-of-arrival is known, then the noise correlation matrix is a linear combination of a set of basis matrices. The noise parameters in the model are the Fourier coefficients of the unknown noise field. In contrast to existing approaches this model is applicable to antenna arrays with an arbitrary geometry.

Secondly, we discuss the identifiability of the data model in the framework of linear matrix inequalities (LMIs) [7]. We also study maximum likelihood approaches for the estimation of the noise parameters. Unfortunately, this requires nonlinear optimization. In order to study the convergence, a concise analysis of the object function is presented. Finally, the performance of a gradient algorithm is illustrated by simulations.

This work has been initiated and elaborated during two visits to Prof. A. Paulraj at the Information Systems Laboratory of Stanford University. The noise model and the identifiability are discussed in [139].

# Chapter 2

# Concepts and Tools

In this introductory chapter we give an overview of the necessary background material from the three major areas on which the thesis is based, *i.e.*, array signal processing, linear algebra and algorithmic engineering. The first section gives a derivation of the data model for narrow-band point sources impinging on a sensor array. Specific geometric properties of this data model are crucial for the development of algorithms. The next section describes the beamforming task and introduces the QR decomposition. The generalized sidelobe canceler for recursive beamforming is discussed. This algorithm and its associated parallel architecture will form the basis for the robust LCMV algorithm in chapter 5. In the last section various direction finding (DF) algorithms are reviewed. Here, the appropriate mathematical tool is the singular value decomposition. We introduce the Jacobi algorithm for SVD updating. This algorithm and its parallel Jacobi architecture are the backbone of the recursive array processing algorithms of later chapters.

## 2.1 Narrow-band data model

Array signal processing is the branch of signal processing aiming at extracting information from propagating waves using sensor arrays. For an excellent introduction to the field, the reader is referred to [44]. Although array processing has its own peculiarities, there are close links to the more familiar theory of time series processing. Time and space are often interchangeable. A sensor array samples a wave at different (equidistant) points in space, whereas a tapped delay line samples a continuous-time signal at different points in time. Below we develop a mathematical model

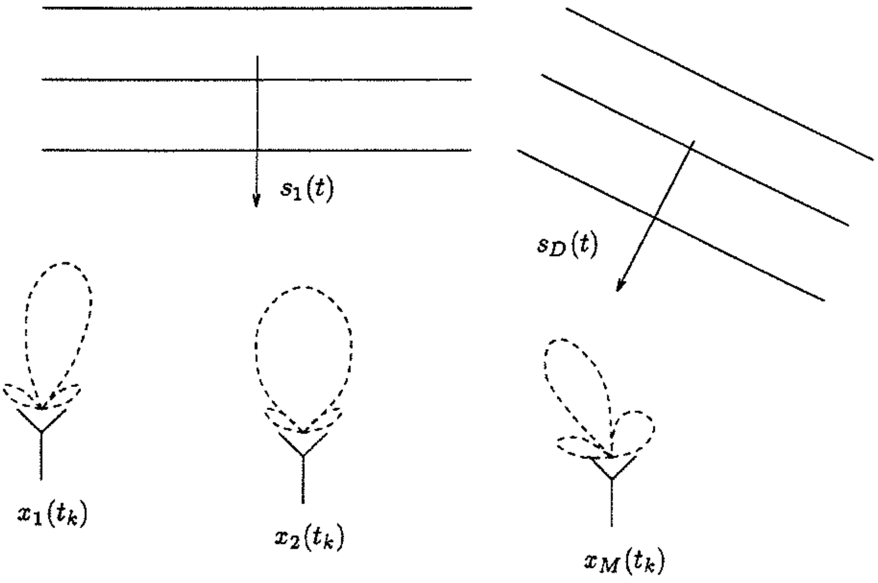


Figure 2.1: An antenna array with $M$ elements receiving $D$ incident wavefronts (full lines). The sources are in the far field of the array since the wavefronts are planar. The array elements do not have to be equal. This is indicated by the directivity patterns (dashed lines), which show the input-output gain of the element as a function of the direction-of-arrival.

for the sensor array output.

A sensor array is a group of $M$ sensors placed at different spatial locations $\{r_m\}_{m=0}^{M-1}$ ($r_m \in \mathbb{R}^3$), sampling an electro-magnetic or acoustic propagating field at a set of (regularly spaced) time instants $\{t_k\}$ (Figure 2.1). Under the assumption that the sensor is a linear device of non-negligible size (*e.g.*, a parabola antenna) with impulse response $h_m(r,t)$, the output $\tilde{x}_m(t_k)$ of sensor $m$ is related to the wave field $f(r,t)$ through a spatiotemporal filtering operation

$$\tilde{x}_m(t_k) = \iiint_\rho \int_\tau h_m(\rho,\tau) \cdot f(r_m - \rho, t_k - \tau)\, d\rho\, d\tau$$

where the vector $\rho \in \mathbb{R}^3$ is a relative coordinate with respect to the midpoint $r_m$ of the sensor. If the size of the sensor (*i.e.*, the aperture) is small by comparison with the variations in the wave field, $f(r,t)$ can be

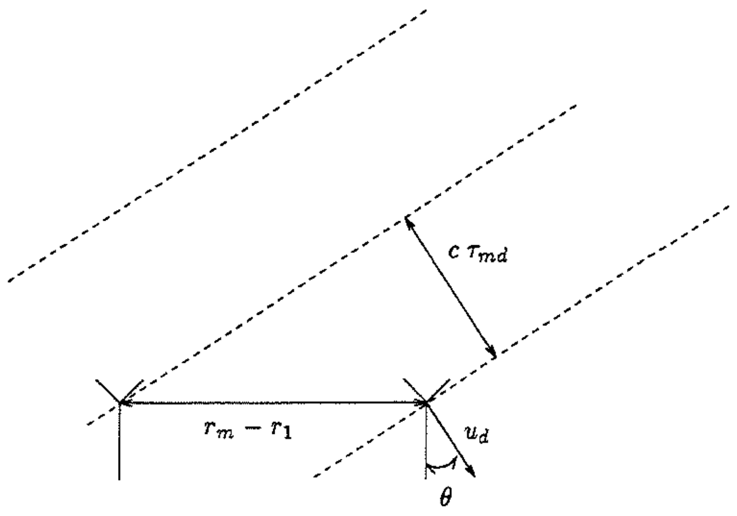


Figure 2.2: The time-difference-of-arrival $\tau_{md}$ between the reference sensor and sensor $m$ for a planar wave with propagation vector $u_d$ is given by the ratio of the projected distance between the two sensors onto $u_d$ and the propagation speed $c$.

considered constant as a function of $r$ and the 4-dimensional convolution reduces to an integral over time.

In many applications the data model can be further simplified. Consider a radio communications application. Here the amplitude-and-phase modulated information signal has a complex envelope

$$s_d(t) = \alpha_d(t)\; e^{j\phi_d(t)}$$

and the input signal to the reference sensor (sensor 1) is the information signal modulated onto a carrier wave with frequency $f_c$

$$\tilde{s}_d(t) = \alpha_d(t) \cdot \cos(2\pi f_c \cdot t + \phi_d(t)).$$

The output of sensor $m$ is a scaled and delayed sinusoidal signal of the same frequency $f_c$

$$\tilde{x}_m(t) = g_m(\theta_d) \cdot \alpha_d(t - \tau_{md}) \cdot \cos(2\pi f_c \cdot (t - \tau_{md}) + \phi_d(t - \tau_{md}))$$

where $g_m(\theta_d)$ is the gain of sensor $m$ for a signal impinging from angle-of-arrival[1] (AOA) $\theta_d$, and $\tau_{md}$ is the time-difference-of-arrival (TDOA) between the reference sensor and sensor $m$ for signal $d$. For an emitter in the far field and a homogeneous non-dispersive medium (*i.e.*, constant propagation speed $c$), this TDOA is related to the propagation direction by

$$\tau_{md} = u_d^T \cdot \frac{(r_m - r_1)}{c}$$

where the vector $u_d \in \mathbb{R}^3$ denotes the normalized propagation vector (Figure 2.2). As an example, consider a typical array for a GSM base station consisting of $M = 10$ elements spaced half a wavelength apart at 900 MHz. The maximal TDOA between two consecutive elements is then of the order of 0.5 nsec. The maximal traveling time across the array $\tau_{\max}$ is of the order of 5 nsec. A signal $s_d(t)$ is called narrow-band, if its bandwidth $B$ is smaller than the reciprocal of the maximal TDOA ($B \ll 1/\tau_{\max}$) [91]. For GSM, $B = 100$ MHz and $1/\tau_{\max} = \pm 200$ MHz such that the narrow-band condition is fulfilled.

The following approximation then holds for all $m$ and $d$

$$\begin{aligned} \alpha_d(t - \tau_{md}) &\approx \alpha_d(t) \\ \phi_d(t - \tau_{md}) &\approx \phi_d(t). \end{aligned}$$

Therefore, the complex envelope $x_m(t)$ of the sensor output $\tilde{x}_m(t)$ equals the complex input signal up to a complex scalar $a_m(\theta_d)$

$$\begin{aligned} x_m(t) &= g_m(\theta_d)\, e^{-j2\pi f_c \tau_{md}} \cdot s_d(t) \\ &= a_m(\theta_d) \cdot s_d(t). \end{aligned}$$

This simple narrow-band relation leads to an elegant fundamental matrix expression for the sensor array output. Let the vector $x_{[k]} = x(t_k) \in \mathbb{C}^M$ be

$$x_{[k]} = \begin{bmatrix} x_1(t_k) & \cdots & x_M(t_k) \end{bmatrix}^T$$

and define the data matrix $X \in \mathbb{C}^{M \times N}$ as

$$X = \begin{bmatrix} x_{[1]} & \cdots & x_{[N]} \end{bmatrix}.$$

Column $k$ of $X$ contains the array output at time $t_k$, whereas row $l$ of $X$ contains the time series observed by sensor $l$. The data matrix can now

[1]We assume that the sensor gain can be fully parameterized by the azimuth angle. In general, other parameters, *e.g.*, the elevation angle, also affect the sensor gain.

be written as the outer product $X = a(\theta_d) \cdot s_d$ of the input signal vector $s_d \in \mathbb{C}^N$

$$s_d = \begin{bmatrix} s_d(t_1) & \cdots & s_d(t_N) \end{bmatrix}$$

and the array response vector[2] $a(\theta_d) \in \mathbb{C}^M$

$$a(\theta_d) = \begin{bmatrix} a_1(\theta_d) & \cdots & a_M(\theta_d) \end{bmatrix}^T .$$

The set of array response vectors for all angles $\theta$ is called the array manifold $\mathcal{A}$

$$\mathcal{A} = \{a(\theta) \mid 0 \leq \theta < 2\pi\}.$$

If the sensor characteristics vary smoothly, then the array manifold draws a closed continuous curve in $\mathbb{C}^M$ (Figure 2.3). An array manifold $\mathcal{A}$ is called unambiguous when any set of $M$ array response vectors is linearly independent. This property is crucial for the uniqueness of the direction finding solution [91].

In case of noisy observations of multiple input signals $\{s_d(t)\}_{d=1}^D$, all sharing the same carrier frequency but impinging from distinct AOAs $\{\theta_d\}_{d=1}^D$, the data model is extended to

$$X = A \cdot S + W \tag{2.1}$$

where the input signal matrix $S \in \mathbb{C}^{D\times N}$ and the array gain matrix $A \in \mathbb{C}^{M\times D}$ are given by

$$\begin{aligned} S &= \begin{bmatrix} s_1^T & \cdots & s_D^T \end{bmatrix}^T \\ A &= \begin{bmatrix} a(\theta_1) & \cdots & a(\theta_D) \end{bmatrix}. \end{aligned}$$

The matrix $W \in \mathbb{C}^{M\times N}$ contains both noise contributions which are generated internally in the sensors and unwanted observed signals (*e.g.*, ambient noise and interference signals).

Without noise on the data, the column vectors of $X$ are contained in the column space of $A$. The span of the array gain matrix $A$ is commonly called the signal subspace S

$$\mathcal{S} = \{x \in \mathbb{C}^M | \exists y \in \mathbb{C}^D, x = A \cdot y\}.$$

[2]Note that the array response vector - also called direction-of-arrival vector or steering vector - is only dependent on the array characteristics, *i.e.*, directivity patterns and location of the sensors.

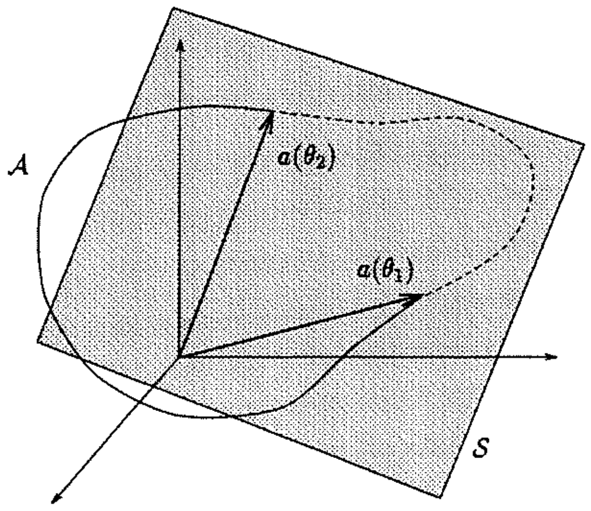


Figure 2.3: Geometry of the narrow-band data model ($M = 3, D = 2$). The array response vectors, corresponding to the AOAs are found in the intersection of the array manifold $\mathcal{A}$ and the signal subspace $\mathcal{S}$.

The orthogonal complement of the signal subspace is known as the noise subspace $\mathcal{N}$. The geometry of the narrow-band data model is shown in Figure 2.3. In the absence of noise each snapshot $x_{[k]}$ is a linear combination of the array response vectors $\{a(\theta_d)\}_{d=1}^{D}$ with varying amplitudes $\{s_d(t_k)\}_{d=1}^{D}$ and is therefore confined to the signal subspace $\mathcal{S}$. The location of the signal subspace is determined by the array response vectors and thus by the location of the sources.

## 2.2 Beamforming

A first important motivation for using sensor arrays is to enhance the signal-to-noise ratio (SNR) beyond that of a single sensor. This objective is the spatial analogue of FIR filtering in the time domain. Such a spatial linear filter is called a beamformer (Figure 2.4). Its output $y_{[k]}$ is a weighted combination of the sensor outputs

$$y_{[k]} = x_{[k]}^H \cdot w.$$

The structure of the beamformer reminds of a transversal finite im-

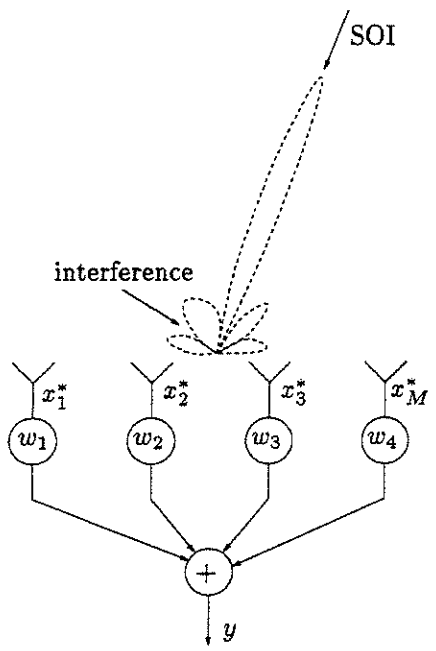


Figure 2.4: A beamformer linearly combines the antenna outputs with a weight vector $w$. The weights determine the directional gain of the array (dashed curve). The weights are such that the signal of interest (SOI) is enhanced while noise and interferences are suppressed.

pulse response (FIR) filter, in which the outputs of a tapped delay line are weighted and summed. The choice of the weight vector (*i.e.*, the filter coefficients), determines the spatial sensitivity of the sensor array. This spatial sensitivity is plotted as a directivity pattern, which is defined as

$$d(\theta) = |a(\theta)^H \cdot w|$$

for the set of angles $\theta$ under consideration. The directivity pattern shows the input-output gain as a function of the angle-of-arrival for a sinusoidal signal of a given frequency. Therefore, it is the analogue of a Bode plot of the FIR filter which shows the input-output gain as a function of frequency. Often the beamformer acts as a spatial bandpass filter, passing the signal-of-interest (SOI) in its main-lobe and rejecting signals arriving from other directions. Ideally the SNR at the output can be increased

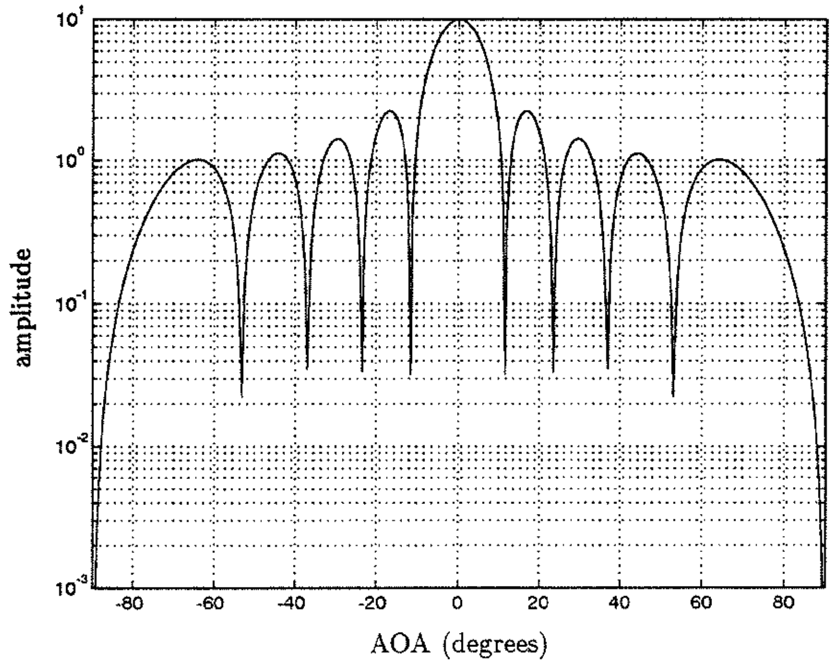


Figure 2.5: Directivity pattern (amplitude) for a 10-element ULA with spacing $\Delta = \lambda/2$ and all weights taken to be unity. The directivity pattern shows the input-output gain of the array in function of the angle-of-arrival for a signal of a given frequency.

by a factor of $M$, *i.e.*, the number of sensors. This is illustrated by the directivity pattern of a Uniform Linear Array (ULA) in Figure 2.5. Such a ULA is a common sensor array structure, composed of $M$ identical omni-directional sensors, placed regularly on a straight line.

### 2.2.1 LCMV beamforming

In many applications the beamformer needs to reconstruct an SOI coming from a known AOA $\theta_s$, maximally rejecting interferences and noise. The optimal weight vector can be determined by solving a constrained least squares problem. The total output power is the sum of the output power, due to the SOI, and the interference and noise power. Optimal signal reconstruction is obtained if the output power is minimized subject to the constraint that the signal power is kept constant. This can be accomplished by a priori imposing a non-zero gain-and-phase $\mu$ in the direction of the SOI

$$a(\theta_s)^H \cdot w = \mu. \tag{2.2}$$

Additional constraints can be added, *e.g.*, to force zero gain along known interference directions, or to shape the directivity pattern. A frequently used constraint imposes an extremum of the directivity pattern at the SOI direction

$$\frac{\partial a}{\partial \theta}(\theta_s)^H \cdot w = 0.$$

The time-averaged output power is $p = \frac{1}{N} y^H \cdot y = w^H \cdot R_x \cdot w$ where $y = X^H \cdot w$ and $R_x = \frac{1}{N} X \cdot X^H$ is the sample correlation matrix. The optimal weight is then the unique solution of the following minimization problem

$$\min_w w^H \cdot R_x \cdot w \qquad \text{subject to} \qquad C^H \cdot w = m. \tag{2.3}$$

The matrix $C \in \mathbb{C}^{M \times K}$ is the constraint matrix and $m \in \mathbb{C}^K$ is the gain vector. Hence, this approach is known as linearly constrained minimum variance (LCMV) beamforming. In order for the optimization to have a solution, the number of constraints $K$ is assumed to be smaller than the number of sensors $M$. The optimal weight vector $\bar{w}$ is readily obtained as

$$\bar{w} = R_x^{-1} \cdot C \cdot (C^H \cdot R_x^{-1} \cdot C)^{-1} \cdot m.$$

An equivalent expression for $\bar{w}$ can be derived as follows. The constraints form an underdetermined set of linear equations. All solutions are characterized by

$$\mathcal{W} = \{w | w = w_q + B \cdot w_b\}.$$

The designer is free to choose the so-called 'quiescent weight vector' $w_q \in \mathbb{C}^M$ as long as it satisfies the constraints

$$C^H \cdot w_q = m.$$

The 'blocking matrix' $B \in \mathbb{C}^{M \times (M-K)}$ is a preferably unitary matrix, whose columns span the null space of $C^H$

$$C^H \cdot B = O_{K \times (M-K)}.$$

Its name is explained by the observation that the columns of $B$ are orthogonal to $a(\theta_s)$. Therefore, they act as notch filters, blocking the SOI and passing only noise and interferences. The SOI is only present in the output of the filter $w_q$. The new unknown $w_b \in \mathbb{C}^{M-K}$ is a weight vector of reduced dimension. Minimizing the output power with respect to $w_b$ then results in

$$\bar{w}_b = -(B^H \cdot R_x \cdot B)^{-1} \cdot B^H \cdot R_x \cdot w_q. \tag{2.4}$$

This expression is implemented by the generalized sidelobe canceler (GSC) [37], whose high-level signal flow graph (SFG) is shown in Figure 2.6. We call a SFG a high-level graph when it consists of block operators transforming vectors or matrices, and the exact internal structure is not detailed. The upper rectangular operator is called the beamforming network. It contains the compound matrix $\left[\begin{array}{c|c} B & w_q \end{array}\right]$ and converts the sensor outputs from 'data space' into 'beam space' by matrix multiplication

$$\left[\begin{array}{c|c} \tilde{x}_{[k]}^H & \tilde{y}_{[k]} \end{array}\right] = x_{[k]}^H \cdot \left[\begin{array}{c|c} B & w_q \end{array}\right].$$

The lower operator represents a linear filter of size $M-K$. The expression for the optimal weight vector can be highly simplified by the use of the QR decomposition.

- *Given a matrix $Z \in \mathbb{C}^{N\times M}$, $N \geq M$, then its QR decomposition is defined by*

  $$Z = Q \cdot R$$

  *where $Q \in \mathbb{C}^{N\times M}$ is a unitary matrix $Q^H \cdot Q = I_M$ and the matrix $R \in \mathbb{C}^{M\times M}$ is upper triangular.*

The matrix $I_M \in \mathbb{R}^{M\times M}$ is the identity matrix. On condition that $Z$ has full rank $M$, this decomposition is unique up to a diagonal matrix $D$ with elements of unit-modulus. Its relevance lies in the following properties.

- The columns for $Q$ constitute an orthonormal basis of the column space of $Z$.

- The matrix $R$ is a square-root (even a Cholesky factor) of $Z^H \cdot Z$. The use of the QR decomposition avoids explicit computation of the latter matrix product. Therefore, so-called square-root algorithms have better numerical properties.

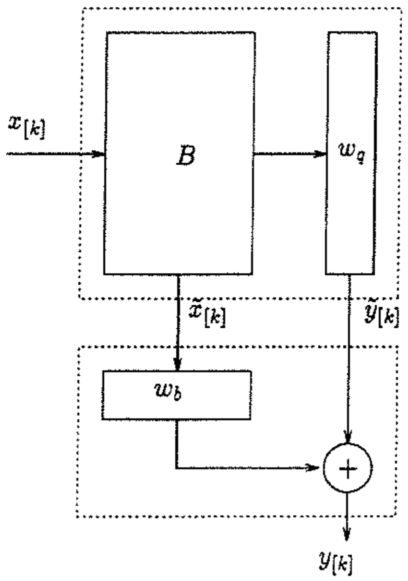


Figure 2.6: High-level signal flow graph of the generalized sidelobe canceler.

- The triangularity of $R$ allows for a cheap solution of the associated linear system by back substitution. It is also important for fast updating.

Let the transformed data matrix $\tilde{X}^H \in \mathbb{C}^{N\times(M-K)}$ and the vector $\tilde{y} \in \mathbb{C}^N$ be defined as

$$\left[\ \tilde{X}^H \mid \tilde{y}\ \right] = X^H \cdot \left[\ B \mid w_q\ \right].$$

Substituting $\tilde{X}^H$ in (2.4) for its QR decomposition $\tilde{X}^H = Q \cdot R$, we find an alternative, much simpler expression for the optimal weight,

$$\hat{w}_b = -R^{-1} \cdot u \tag{2.5}$$

where the vector $u \in \mathbb{C}^{M-K}$ is defined by $u = Q^H \cdot \tilde{y}$.

### 2.2.2 Recursive LCMV beamforming

In a time-varying environment, the optimal weight vector has to be adapted continuously. The sample correlation matrix is time-dependent and the weight optimization becomes a recursive least squares problem. Each data vector is processed as soon as it becomes available. The data matrix $X_{[k]}$ now continuously grows as new observations are appended

$$\underbrace{X_{[k]}}_{k\times M} = \left[\begin{array}{c} \alpha \cdot X_{[k-1]} \\ \hline x^H_{[k]} \end{array}\right].$$

The real scalar $\alpha < 1$ is an exponential weighting factor which is incorporated to de-emphasize older data.

One possible implementation, well-known for its numerical stability and intrinsic parallelism, is based on QRD updating. At each time $t_k$, the optimal weight is still given by Eq. (2.5)

$$\bar{w}_{b,[k]} = -R^{-1}_{[k]} \cdot u_{[k]}.$$

In order to track the weight vector, the QR decomposition of the growing matrix $\tilde{X}^H_{[k]}$ has to be updated. Assume that the QRD of $\tilde{X}^H_{[k-1]} = Q_{[k-1]} \cdot R_{[k-1]}$ is given. The updated matrix $\tilde{X}^H_{[k]}$ can be decomposed as

$$\begin{aligned}
\tilde{X}^H_{[k]} &= \left[\begin{array}{c} \alpha\, \tilde{X}^H_{[k-1]} \\ \hline \tilde{x}^H_{[k]} \end{array}\right] \\
&= \left[\begin{array}{c|c} Q_{[k-1]} & O \\ \hline O & 1 \end{array}\right] \cdot \left[\begin{array}{c} \alpha\, R_{[k-1]} \\ \hline \tilde{x}^H_{[k]} \end{array}\right] \\
&= \underbrace{\left[\begin{array}{c|c} Q_{[k-1]} & O \\ \hline O & 1 \end{array}\right] \cdot G_{[k]}}_{\left[\begin{array}{c|c} Q_{[k]} & \star \end{array}\right]} \cdot \underbrace{G^H_{[k]} \cdot \left[\begin{array}{c} \alpha\, R_{[k-1]} \\ \hline \tilde{x}^H_{[k]} \end{array}\right]}_{\left[\begin{array}{c} R_{[k]} \\ \hline 0\cdots 0 \end{array}\right]}
\end{aligned}$$

The submatrices $O$ are zero vectors of appropriate dimension and the symbol $\star$ denotes a quantity of no further interest. The transformation $G_{[k]}$ is unitary and restores the upper triangular structure of the $R$-matrix. Several choices are available for the computation of $G_{[k]}$. In view of a parallel implementation, the best choice is to construct $G_{[k]}$ as a sequence of $M-K$ Givens rotations $G^{i|M-K+1}_{[k]} \in \mathbb{C}^{(M-K+1)\times(M-K+1)}$. A Givens rotation $G^{i|j} \in \mathbb{C}^{n\times n}$ is a $2\times 2$ rotation matrix embedded in the identity matrix of size $n$. It only affects rows/columns $(i, j)$

$$G^{i|j} = \begin{bmatrix} I_{i-1} & \vdots & & \vdots & \\ \cdots & c & \cdots & s & \cdots \\ & \vdots & I_{j-i-1} & \vdots & \\ \cdots & -s^* & \cdots & c & \cdots \\ & \vdots & & \vdots & I_{N-j} \end{bmatrix}$$

where $c$ is real, and $c^2 + |s|^2 = 1$. With each Givens rotation one can associate an angle $\alpha = \tan^{-1} \frac{|s|}{c}$ and a phase factor $e^{j\phi} = \frac{s}{|s|}$. A VLSI hardware component which is especially designed to perform $2 \times 2$ rotations accurately and fast is the CORDIC (coordinate rotation digital computer) processor [35, 38, 143]. It decomposes the rotation in a sequence of so-called micro-rotations. Each micro-rotation is such that it can be realized by a single (or a few) shift-and-add operations. The CORDIC can operate in two modes. In vector rotation mode, it rotates its input vector over a given angle. In angle accumulation mode, it computes an angle such that the second component of the input is zeroed.

In the Givens QRD updating algorithm [33], the $i$th Givens rotation is computed such that it annihilates the $i$th entry of $\tilde{x}^H_{[k]}$ by combining it with the $i$th diagonal entry of $\alpha R_{[k-1]}$ (angle accumulation mode). It is then applied to row $i$ and $M - K + 1$ (vector rotation mode). A $3 \times 2$ example is given below.

$$\left[\begin{array}{cc} \times & \times \\ & \times \\ \hline \times & \times \end{array}\right] \xrightarrow{G^{1|3^H}} \left[\begin{array}{cc} \times' & \times' \\ & \times \\ \hline 0 & \times' \end{array}\right] \xrightarrow{G^{2|3^H}} \left[\begin{array}{cc} \times' & \times' \\ & \times' \\ \hline 0 & 0 \end{array}\right]$$

The explicit computation of the weight vector $\bar{w}_{b,[k]}$ requires the solution of a triangular linear system. This requires a back-substitution step, which has an $\mathcal{O}(M-K)^2$ complexity. Moreover, during back-substitution the data flow evolves in the opposite direction of the data flow during the QRD updating. This complicates efficient pipelining of both operations. In array processing one is usually not interested in knowing the weight vector explicitly[3]. The ultimate goal is knowledge of the reconstructed signal

$$y_{[k]} = \tilde{y}_{[k]} + \tilde{x}^H_{[k]} \cdot \bar{w}_{b,[k]}.$$

[3]If knowledge of the weight vector is desired, it can be obtained without back substitution by 'freezing' the adaptation and inputting an identity matrix [99].

This so-called residual signal $y_{[k]}$ can be obtained as a side product of the QRD updating. Consider the QRD update applied to the compound matrix

$$\left[\begin{array}{c|c} \alpha\, R_{[k-1]} & \alpha\, u_{[k-1]} \\ \hline \tilde{x}^H_{[k]} & \tilde{y}_{[k]} \end{array}\right] = \left[\begin{array}{c|c} G_{[k],11} & G_{[k],12} \\ \hline G_{[k],21} & G_{[k],22} \end{array}\right] \cdot \left[\begin{array}{c|c} R_{[k]} & u_{[k]} \\ \hline O & z_{[k]} \end{array}\right]$$

where the matrix $G_{[k]}$ is partitioned accordingly. It is an easy exercise to check that the following equality is satisfied

$$G_{[k],22} \cdot z_{[k]} = \tilde{y}_{[k]} + \tilde{x}^H_{[k]} \cdot \underbrace{\left(-R^{-1}_{[k]} \cdot u_{[k]}\right)}_{\bar{w}_{b,[k]}} .$$

Therefore, the optimal output of the beamformer is simply given by

$$y_{[k]} = G_{[k],22} \cdot z_{[k]}.$$

It follows from the construction of $G_{[k]}$ as a sequence of Givens rotations that the scalar $G_{[k],22}$ is the product of all cosines of the rotation angles

$$G_{[k],22} = \prod_{i=1}^{M-K} c^i_{[k]}.$$

Instead of solving a linear system, the computation of $y_{[k]}$ is reduced to a simple multiplication. This result was originally exposed by Shepherd and McWhirter [100].

The recursive generalized sidelobe canceler algorithm is summarized in Algorithm 1. The parameter $\gamma_{[k]}$ accumulates the product of the cosines of all Givens transformations.

**Algorithm 1**

```
Compute $w_q$ and $B$ such that
                 $C^H \cdot w_q = m$
                 $C^H \cdot B = O_{K\times(M-K)}$

                 $R_{[0]} \leftarrow O_{(M-K)\times(M-K)}$
                 $u_{[0]} \leftarrow O_{(M-K)\times 1}$
 for $k = 1, \ldots, \infty$
    1. Matrix-vector multiplication
                 $\left[\, \tilde{x}^H_{[k]} \mid \tilde{y}_{[k]} \,\right] \leftarrow x^H_{[k]} \cdot \left[\, B \mid w_q \,\right]$
```

2. QRD updating

$$\left[\begin{array}{c|c} R_{[k]} & u_{[k]} \\ \hline O_{1\times(M-K)} & z_{[k]} \end{array}\right] \leftarrow \cdots$$

$$\left(\prod_{i=1}^{M-K} G_{[k]}^{i|M-K+1}\right)^H \cdot \left[\begin{array}{c|c} \alpha\, R_{[k-1]} & \alpha\, u_{[k-1]} \\ \hline \tilde{x}_{[k]}^H & \tilde{y}_{[k]} \end{array}\right]$$

$$\gamma_{[k]} \leftarrow \prod_{i=1}^{M-K} G_{[k],ii}^{i|M-K+1}$$

3. beamformer output computation

$$y_{[k]} \leftarrow \gamma_{[k]} \cdot z_{[k]}$$

endfor

### 2.2.3 Parallel mapping

Algorithm 1 is very well suited for parallel implementation. For the mapping from algorithm to parallel architecture we will adhere to the canonical mapping methodology described in [50]. This methodology can be exposed in a graphical way, manipulating graphical representations of algorithms. The advantage is that the parallelism is much more apparent in a graph than in a textual description. It also allows to visualize the transformations by software tools to assist in mapping new algorithms onto application specific or general purpose parallel architectures.

The starting point is a dependence graph (DG) of the algorithm. Usually the algorithm is specified as a nested loop program in a standard computer language. This textual sequential description is converted into a DG by associating with each operation a node indexed by its loop variables. The data flow is represented by directed arcs. An output argument from one operation which is used as an input argument of another operation defines an arc $d_i$ between the corresponding nodes. The resulting DG is an acyclic directed graph showing how signals are produced and consumed by source and sink nodes.

The topology of the DG is crucial for the parallel mapping. For algorithms which are originally described as nested loops, there is a natural index space defined by the loop indices. The DG is normally represented in this index space. Important properties are *e.g.*, locality and homogeneity of the dependencies. A graph is said to have local dependencies if it exchanges signals only with its neighboring nodes in the index space. It is homogeneous if the dependency pattern is shift-invariant. All nodes then exhibit the same regular connections.

Executing the algorithm involves allocating each operation in the DG to a certain processor (assignment or placement) at a certain time (scheduling). When mapping to a given parallel machine, the processor space is fixed, whereas for an application specific parallel processor one still has the freedom to select an optimal processor configuration. When the DG is homogeneous, a linear placement and schedule is often a good approach. This corresponds to finding a placement vector $p$ and a scheduling vector $s$. All operations on straight lines parallel to the placement vector are allocated to the same processor. The execution time $t_i$ of a node with coordinate $r_i$ is given by the inner product

$$t_i = s^T \cdot r_i.$$

All operations on hyperplanes orthogonal to the scheduling vector $s$ are executed simultaneously. In order to be permissible the scheduling vector has to satisfy two conditions

$$\begin{array}{rcll} s^T \cdot d_i & \geq & 0, & \forall i \\ |s^T \cdot p| & \neq & 0. & \end{array}$$

The first condition states that an operation can only be executed if all its inputs have been computed. The second condition implies that equitemporal operations should not be allocated to the same processor.

A projected scheduled graph is called a signal flow graph (SFG). In addition to nodes and arcs, it also has delays associated with each arc. A possible SFG of the generalized sidelobe canceler is depicted in Figure 2.7. Delays are indicated by the heavy dots. The loops in the upper rectangular part store the matrix $\left[ \begin{array}{c|c} B & w_q \end{array} \right]$, whereas the loops in the lower part store the matrix $\left[ \begin{array}{c|c} R_{[k]} & u_{[k]} \end{array} \right]$. The upper rectangular array is a multiplication array. The lower triangular array performs QRD updating [31]. The processors on the diagonal compute the Givens rotations, and pass them on to the right. In addition, they accumulate the product of the cosines. The original DG is obtained by repeating this two-dimensional graph over and over again for each time iteration. This is shown in the high-level DG of Figure 2.8.

The SFG level still allows a lot of freedom. Various parallel architec-

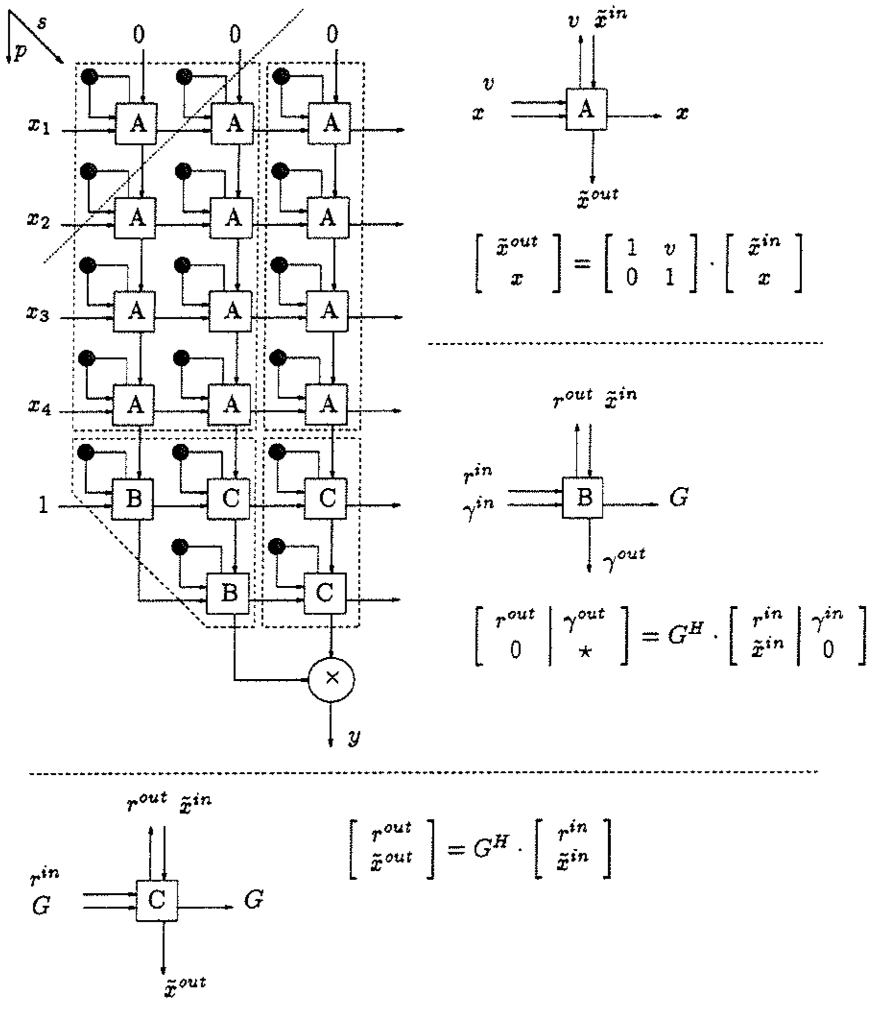

s
p
0
0
0
$x_1$
$x_2$
$x_3$
$x_4$
1
A
B
C
$y$
$v$ $\tilde{x}^{in}$
$v$
$x$
$x$
$\tilde{x}^{out}$
$\begin{bmatrix} \tilde{x}^{out} \\ x \end{bmatrix} = \begin{bmatrix} 1 & v \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \tilde{x}^{in} \\ x \end{bmatrix}$
$r^{out}$ $\tilde{x}^{in}$
$r^{in}$
$\gamma^{in}$
$G$
$\gamma^{out}$
$\left[\begin{array}{c|c} r^{out} & \gamma^{out} \\ 0 & \star \end{array}\right] = G^H \cdot \left[\begin{array}{c|c} r^{in} & \gamma^{in} \\ \tilde{x}^{in} & 0 \end{array}\right]$
$r^{out}$ $\tilde{x}^{in}$
$r^{in}$
$G$
$G$
$\tilde{x}^{out}$
$\begin{bmatrix} r^{out} \\ \tilde{x}^{out} \end{bmatrix} = G^H \cdot \begin{bmatrix} r^{in} \\ \tilde{x}^{in} \end{bmatrix}$

Figure 2.7: SFG for the generalized sidelobe canceler ($M = 4, K = 2$). The delay loops in the upper part store the matrix $\left[\begin{array}{c|c} B & w_q \end{array}\right]$, whereas the delay loops in the lower part store the matrix $\left[\begin{array}{c|c} R_{[k]} & u_{[k]} \end{array}\right]$. The vectors $p$ and $s$ indicate the placement and scheduling vector for a linear systolic architecture. The dotted line is a cut set.

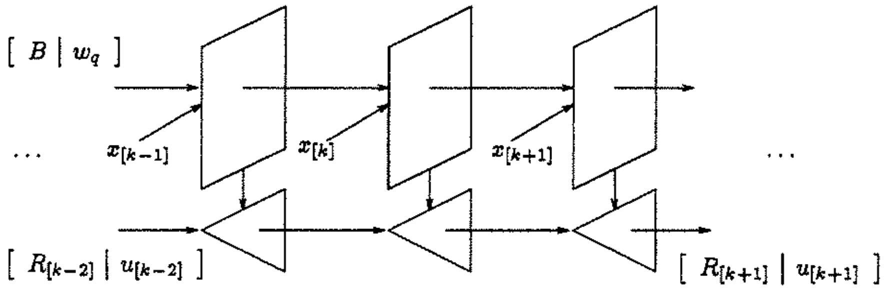

$[\ B \mid w_q\ ]$
$\dots$
$x_{[k-1]}$
$x_{[k]}$
$x_{[k+1]}$
$\dots$
$[\ R_{[k-2]} \mid u_{[k-2]}\ ]$
$[\ R_{[k+1]} \mid u_{[k+1]}\ ]$

Figure 2.8: High level DG for the recursive generalized sidelobe canceler. The graph extends infinitely over the time dimension. The SFG of the previous figure is obtained by a projection parallel to the time axis.

tures can be derived. If one is interested in a linear array, an additional[4] placement and scheduling (multi-projection) can be performed, *e.g.,* with the vectors $p$ and $s$ shown in the top-left corner of Figure 2.7. If a two-dimensional array is preferred, then only the scheduling step is performed. The schedule $s$ indicated in the figure is actually an example of a systolic schedule. A way to represent the schedule is to associate $s^T \cdot d_i$ delays with each dependency $d_i$. In this representation a systolic schedule has the property that at least one delay is present on each dependency arc. In addition to being local in space, the computation is also localized in time. The resulting systolic array has the advantage of highly pipelined operation and consequently a high throughput rate. After a very short cycle new data vectors are fed in, while the others are pushed one stage further in the pipeline.

An alternative way to convert one SFG into another SFG are the following two retiming rules [50].

- *Time-scaling*: All delays in the graph may be multiplied by a factor $n > 1$, on condition that the input and output rates are slowed down accordingly.

- *Delay-transfer*: Given any cut-set of the SFG, which partitions the graph into two components connected by a number of arcs. Delays may be added on each of the outbound arcs on condition that the same number of delays is removed on each of the inbound arcs.

[4]It is clear that the placement and scheduling vectors could have been defined directly on the DG of Figure 2.8.

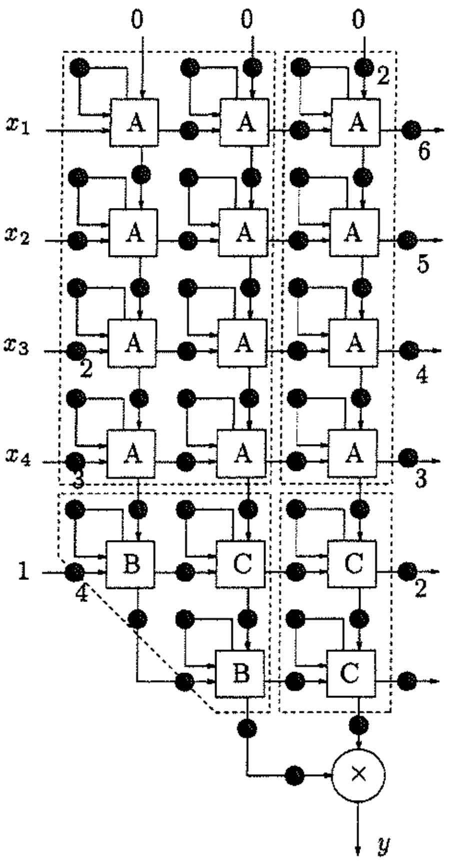


Figure 2.9: Final systolic architecture for the generalized sidelobe canceler ($M = 4, K = 2$). Multiple delays are indicated by a number.

Such a cut-set is indicated by the dotted line in Figure 2.7. It is a hyperplane orthogonal to the scheduling vector $s$. All arcs cross the cut-set in the same direction. Therefore, it is permitted to add one delay on all these arcs. By repeating this delay insertion on parallel cut-sets, the systolic array of Figure 2.9 is obtained. Due to the multiple cuts, the components of the input vector are skewed before being fed into the array. Similarly, in order to obtain all output components synchronously, the cuts introduce delays at the outputs. The pipelining period of the systolic array is only one cycle.

## 2.3 Direction finding

In addition to spatial filtering, sensor arrays are often employed for characterization of the wave field. A model for the field is postulated, *e.g.*, one assumes that the field consists of a sum of narrow-band waves, and some unknown parameters have to be estimated, *e.g.*, the number of narrow-band signals. Again one can set up a parallel with time series processing. The corresponding estimation problem is to determine the frequencies of a finite sum of sinusoidal time series drowned in noise [86].

Linear algebra is the natural tool for mathematically formulating the information extraction objective and for deriving algorithms to estimate the unknowns.

### 2.3.1 Subspace algorithms

Since 1980 high resolution methods for direction finding of narrow-band emitters have been proposed. Their resolution exceeds the Rayleigh resolution limit (*i.e.*, the limit for Fourier based techniques) by full exploitation of the narrow-band data model. An important class is formed by the so-called subspace algorithms, such as MUSIC [96], ESPRIT [92] and WSF [79, 142]. They perform the mapping from observations into a set of AOAs in two steps.

First the signal subspace $\mathcal{S}$ is determined as the column space of the data matrix $X$. It follows from the data model that in the noiseless case the rank of $X$ equals the number of impinging narrow-band (non-coherent) waves. In the case of noisy observations the data matrix will have full rank. The signal subspace then has to be estimated. For white noise on the data the appropriate tool is the singular value decomposition (SVD).

*The singular value decomposition of a matrix $Z \in \mathbb{C}^{N \times M}$, $N \geq M$ is defined by*

$$Z = U \cdot \Sigma \cdot V^H$$

*where $U \in \mathbb{C}^{N \times M}$ and $V \in \mathbb{C}^{M \times M}$ are unitary matrices and $\Sigma \in \mathbb{R}^{M \times M}$ is a diagonal matrix containing the singular values $\sigma_i \geq 0$ in non-increasing order.*

This singular value decomposition is of fundamental importance in modern signal processing, statistics, systems identification and control [93, 114, 123]. Its relevance is due to important properties, which are conceptual as well as numerical.

- From a conceptual point of view there is a relation between the SVD of $Z$, and the eigenvalue decompositions of $Z^H \cdot Z$ and $Z \cdot Z^H$. The singular values $\sigma_i$ are the square-roots of the eigenvalues. The singular vectors $u_i$ and $v_i$ are equal to the eigenvectors of the 'squared' matrices. The following approximation property provides the optimal answer for the subspace estimation problem.

  *Given a matrix $Z \in \mathbb{C}^{N \times M}$, then the matrix $Y \in \mathbb{C}^{N \times M}$, nearest in Frobenius norm or 2-norm to $Z$, and of rank $D < \min(M, N)$ is given by the truncated SVD of $Z$*

  $$Y = \sum_{i=1}^{D} u_i \cdot \sigma_i \cdot v_i^H.$$

  In the case of independent, identically distributed (i.i.d) white Gaussian noise on the data, the truncation of the SVD yields also the maximum likelihood estimator for the signal subspace. Moreover, the smaller singular values can be used for estimation of the noise level $\gamma$.

- From a numerical point of view excellent algorithms exist for computing the SVD [33]. They exploit the orthogonality of the decomposition. The theory of many applications is developed based on the eigenvalue decomposition (EVD) of the matrix $Z^H \cdot Z$. However, in actual computation the SVD is preferred over the EVD because of the potential loss of numerical accuracy associated with the explicit calculation of the matrix-matrix product.

The second mapping from estimated signal subspace to AOAs is performed via (partial) knowledge of the array manifold $\mathcal{A}$. Basically, one has to look for the $D$ array response vectors $\{a(\theta_d),\ d = 1, \cdots, D\}$[5] which are comprised in $\mathcal{S}$. Here several alternatives exist. We only mention two of them.

The MUSIC algorithm [96] was the first algorithm to fully exploit the geometry of the data model. It determines the array response vectors which are nearest to the estimated signal subspace - or most orthogonal to the estimated noise space - by a one-dimensional maximization of the object function

$$d_{MUSIC}(\theta) = \frac{1}{\|a(\theta)^H \cdot V_n\|}$$

[5]We assume an unambiguous array manifold. If this condition is not met, the mapping from subspace to AOAs may not be unique.

where the matrix $V_n \in \mathbb{C}^{M \times M-D}$ contains the singular vectors corresponding to the smallest singular values. This object function is the reciprocal of the norm of the projection of the array response vector onto the noise subspace. If the array response vector lies in the signal subspace, this projection is zero. The $D$ dominant peaks of $d_{MUSIC}(\theta)$ are selected as the AOA estimates. The performance of the MUSIC algorithm is very good [107]. However, it has a few disadvantages. First the array manifold $\mathcal{A}$ has to be known a priori. If no accurate analytic model of the array response vectors is available, then the array has to be calibrated. This is a difficult and expensive operation. Furthermore, a non-linear optimization is required. In addition to the convergence problems due to local minima, this may require a lot of computation.

The ESPRIT algorithm [92] avoids the knowledge of the array manifold and the non-linear optimization. Instead it assumes a translation structure in the array configuration (Figure 2.10). The sensor array should consist of $M$ doublets. Such a doublet is a pair of identical sensors displaced over a known constant vector. No information on the directivity patterns of the sensors is required. They may be arbitrary, as long as they are pair-wise identical in a doublet. The advantage of using this structure is that the need for non-linear optimization is circumvented. Convergence problems are avoided. Instead the AOAs can be estimated by a sequence of matrix decompositions. Below we give an outline of the method.

The array consists of two identical, but translated subarrays. Under the same assumptions as in section 2.1 the output of the first sub-array is given by

$$X = A \cdot S + W_x. \tag{2.6}$$

The output of the second sub-array obeys a similar equation

$$Y = A \cdot \Phi \cdot S + W_y \tag{2.7}$$

where the diagonal unitary matrix $\Phi \in \mathbb{C}^{D \times D}$ holds the extra phase shifts of each emitter $d$

$$\Phi_{dd} = \exp(-j2\pi\Delta \sin(\theta_d)/\lambda)$$

due to the translation of the subarrays. Now consider the noiseless matrix pencil

$$Y - \lambda \cdot X = A \cdot (\Phi - \lambda \cdot I_D) \cdot S.$$

Generically this matrix pencil has full rank $D$. However, if $\lambda = \Phi_{dd}$, the rank of the matrix pencil is reduced. Therefore, the phase shifts $\Phi_{dd}$ can

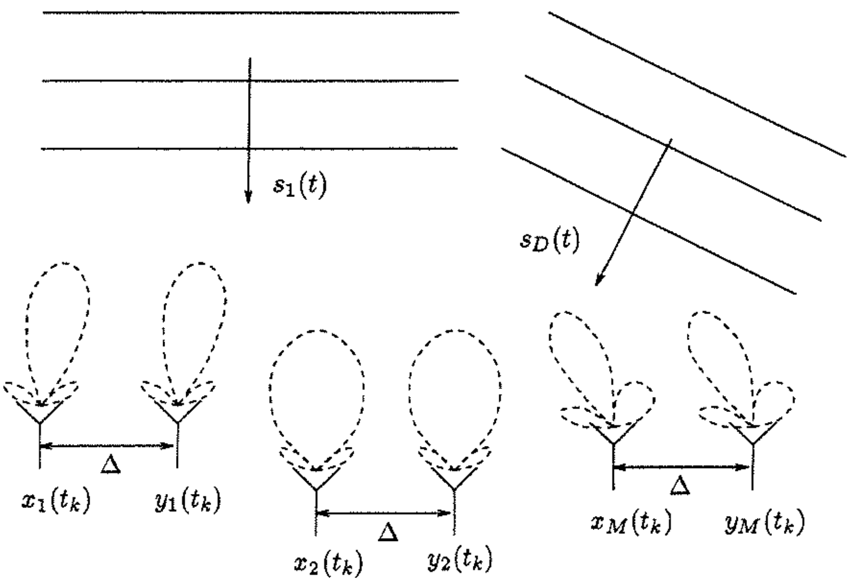


Figure 2.10: Sensor array configuration for the ESPRIT algorithm. The array consists of two identical subarrays, displaced by a vector $\Delta$.

be computed as the rank-reducing numbers of the above matrix pencil. After compression of the column and row dimensions from $N \times M$ to $D \times D$ by pre- and post-multiplying with well-chosen $D \times N$ and $M \times D$ matrices, this problem becomes a generalized eigenvalue problem. In order to compute the rank-reducing numbers with orthogonal decompositions only, the generalized Schur decomposition can be used [11]. In the noisy case, various selections of compression matrices may yield slightly different estimates. The Total Least Squares [119] version of the ESPRIT algorithm [92] is renowned for its excellent noise suppression. However, it is intrinsically sequential. For an algorithm to be amenable to easy parallel implementation, the left and right compression matrices have to be independent of each other. In [113] the compression matrices are computed based on two independent SVDs. In [71] the noise is first suppressed by an instrumental variable method. The choice of the compression matrices then does not matter anymore. A simple choice suffices.

## 2.3.2 SVD updating

The first and most time-consuming step of subspace algorithms is the computation of the SVD of the data matrix $X$. In real-time applications which require tracking of the location estimates, the signal subspace has to be updated recursively. Optimal tracking is obtained when the SVD of the data matrix is updated exactly.

Unfortunately, this is a computationally expensive operation. It requires $\mathcal{O}(M^3)$ operations per update. For many real-time applications this computational cost is a serious impediment. Therefore, approximate algorithms for SVD updating have been developed which trade off accuracy for computational complexity.

Here we concentrate on an $\mathcal{O}(M^2)$ algorithm, developed by Moonen *et al.* [66]. It is reprinted below as Algorithm 2 and computes a unitary decomposition

$$X_{[k]}^H = U_{[k]} \cdot R_{[k]} \cdot V_{[k]}^H$$

where $U_{[k]} \in \mathbb{C}^{k \times M}$ and $V_{[k]} \in \mathbb{C}^{M \times M}$ are unitary matrices, but now $R_{[k]} \in \mathbb{C}^{M \times M}$ is an upper triangular matrix which is nearly diagonal (*i.e.*, close to the true $\Sigma_{[k]}$). The algorithm only keeps track of $R_{[k]}$ and $V_{[k]}$, which is sufficient for most signal processing applications.

After appending a new observation, the augmented data matrix $X_{[k]}^H$ can be written as

$$\begin{aligned} X_{[k]}^H &= \left[\begin{array}{c} \alpha\, X_{[k-1]}^H \\ \hline x_{[k]}^H \end{array}\right] \\ &= \left[\begin{array}{c|c} U_{[k-1]} & 0 \\ \hline 0 & 1 \end{array}\right] \cdot \left[\begin{array}{c} \alpha\, R_{[k-1]} \\ \hline x_{[k]}^H \cdot V_{[k-1]} \end{array}\right] \cdot V_{[k-1]}^H. \end{aligned} \tag{2.8}$$

This decomposition has to be turned into a nearly diagonal upper triangular form again. This is accomplished by a sequence of three operations. The first operation is the matrix-vector multiplication

$$\tilde{x}_{[k]}^H = x_{[k]}^H \cdot V_{[k-1]}.$$

The second operation is a QRD updating, which works $\tilde{x}_{[k]}$ into the weighted triangular matrix $\alpha \cdot R_{[k-1]}$

$$\left[\begin{array}{c} \tilde{R}_{[k]} \\ \hline 0 \cdots 0 \end{array}\right] = \left(G_{[k]}^{1|M+1} \cdots G_{[k]}^{M|M+1}\right)^H \cdot \left[\begin{array}{c} \alpha\, R_{[k-1]} \\ \hline \tilde{x}_{[k]}^H \end{array}\right]$$

where each $G_{[k]}^{i|M+1} \in \mathbb{C}^{M+1\times M+1}$ is a Givens rotation matrix. In order to compensate for this QRD updating in the decomposition of (2.8), the Givens rotations should be applied to the columns of the augmented $U_{[k-1]}$ matrix. However, since this growing matrix is not tracked by the algorithm, this part of the computation is dropped.

The QRD updating step degrades the nearly diagonal structure of the $R$-matrix. Therefore, the third operation aims at reducing the size of the off-diagonal elements again. This is done by applying a sequence of $M-1$ Jacobi rotations, *i.e.*, double-sided (row and column) rotations. This sequence zeroes all entries in the first super-diagonal once. The $i$th Jacobi rotation follows from the SVD of the $i$th triangular $2\times 2$ block on the diagonal of $\tilde{R}_{[k]}$

$$\begin{bmatrix} \times & 0 \\ 0 & \times \end{bmatrix} = \Theta_{[k]}^{i|i+1^H} \cdot \begin{bmatrix} \tilde{R}_{[k],i,i} & \tilde{R}_{[k],i,i+1} \\ 0 & \tilde{R}_{[k],i+1,i+1} \end{bmatrix} \cdot \Phi_{[k]}^{i|i+1}.$$

For details on how to compute the rotation angles, we refer to the original paper [66]. Again, applying the row rotations $\Theta_{[k]}^{i|i+1}$ to the columns of the $U$-matrix may be omitted. On the other side, compensation for the column rotations $\Phi_{[k]}^{i|i+1}$ is mandatory. They need to be applied to the columns of $V_{[k-1]}$. The operation of a single sequence is illustrated below on a $4\times 4$ example.

$$\begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \\ & & & \times \end{bmatrix} \xrightarrow{\Theta^{1|2^H},\Phi^{1|2}} \begin{bmatrix} \times' & 0 & \times' & \times' \\ & \times' & \times' & \times' \\ & & \times & \times \\ & & & \times \end{bmatrix} \xrightarrow{\Theta^{2|3^H},\Phi^{2|3}}$$

$$\begin{bmatrix} \times' & e & \times'' & \times' \\ & \times'' & 0 & \times'' \\ & & \times' & \times' \\ & & & \times \end{bmatrix} \xrightarrow{\Theta^{3|4^H},\Phi^{3|4}} \begin{bmatrix} \times' & e & \times''' & \times'' \\ & \times'' & e & \times''' \\ & & \times'' & 0 \\ & & & \times' \end{bmatrix}$$

The zero entry created by the $i$th Jacobi rotation is filled in again by the next Jacobi rotation. These fill-in elements are denoted by the $e$s. Zeroing the complete strictly upper triangular part thus requires an iterative algorithm. Each Jacobi rotation decreases the norm of the off-diagonal elements with $|\tilde{R}_{[k],i,i+1}|^2$. However, the convergence could stall if a first super-diagonal of almost zero-entries were created, while other entries are still significant. Therefore, care has to be taken in the selection of the Jacobi rotations such that all entries are circulated towards and away from the first super-diagonal. Two solutions exist for the angles of the Jacobi rotations. Consequently two circulation strategies may be used. A first strategy always selects the rotations over the smaller angles followed by a two-sided $2 \times 2$ permutation. The second strategy always selects the larger rotation angles. Both strategies tend to move entries around such that the convergence is not stalled. More details can be found in [66]. The final description of the Jacobi SVD updating is given in Algorithm 2.

**Algorithm 2**

```
$V_{[0]} \leftarrow I_M$
$R_{[0]} \leftarrow O_{M\times M}$

for $k = 1, \cdots, \infty$
  1. Orthogonal matrix-vector multiplication
       $\tilde{x}_{[k]}^H \leftarrow x_{[k]}^H \cdot V_{[k-1]}$
  2. QRD updating
       $\left[\begin{array}{c} \tilde{R}_{[k]} \\ \hline 0 \end{array}\right] \leftarrow \prod_{i=1}^{M} G_{[k]}^{M+1-i|M+1^H} \cdot \left[\begin{array}{c} \alpha \cdot R_{[k-1]} \\ \hline \tilde{x}_{[k]}^H \end{array}\right]$
  3. Jacobi rotations
       $R_{[k]} \leftarrow \prod_{i=1}^{M-1} \Theta_{[k]}^{M-i|M+1-i^H} \cdot \tilde{R}_{[k]} \cdot \prod_{i=1}^{M-1} \Phi_{[k]}^{i|i+1}$
       $V_{[k]} \leftarrow V_{[k-1]} \cdot \prod_{i=1}^{M-1} \Phi_{[k]}^{i|i+1}$
endfor
```

The algorithm has several desirable properties. A first point is its moderate computational complexity, only $\mathcal{O}(M^2)$ per update. Moreover, the approximation error with respect to the exact SVD is in many cases acceptable. In [66] an analysis for the subspace tracking application (*e.g.*, for high-resolution parameter estimation) shows that the tracking error, defined as the distance in terms of canonical angles, between the true and the estimated signal subspace, both at time $t_k$, is bounded by the time variation of the true signal subspace in $M$ steps, provided that the signal-to-noise ratio is moderately high. A numerical example is given in

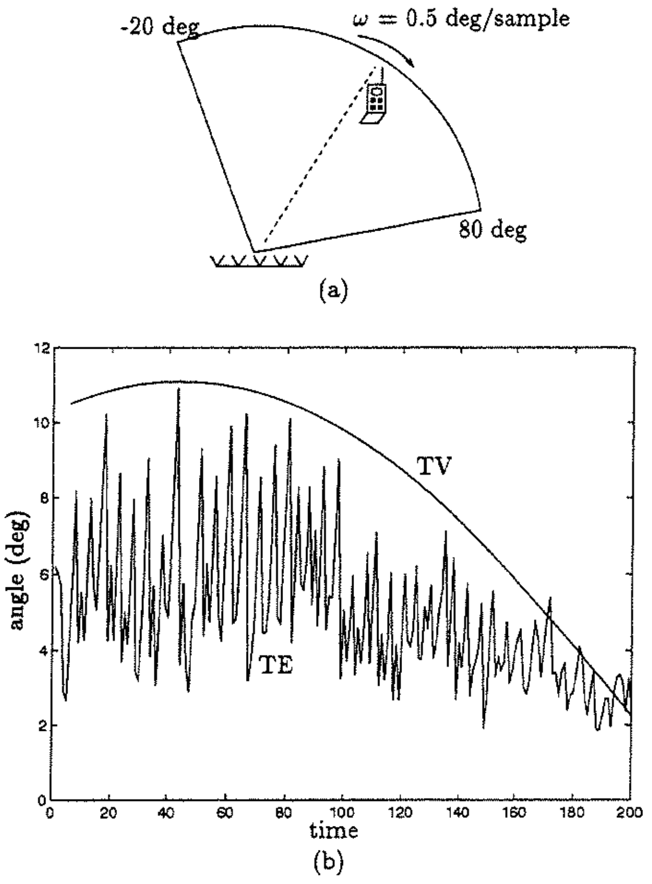


Figure 2.11: (a) A ULA with $M = 5$ antennas and $\Delta = \lambda/2$ tracks a mobile traveling with constant angular speed $\omega$ from -20 deg to 80 deg. The signal is a constant modulus signal with random phase. The SNR at the antenna outputs is 20 dB.
(b) The time variation in $M$ samples (TV) is the angle between the array manifold vectors at time $k$ and $k - M$. The tracking error of the Jacobi SVD updating algorithm is the angle between the array manifold vector and the estimated dominant singular vector both at time $k$. TV is an approximate upper bound on TE. The forgetting factor is low $\alpha = 0.8$, such that the algorithm can follow the fast variation of the system.

Figure 2.11. The relation between the non-stationarity of the data and the number of rotation sequences required for good tracking performance is studied in [54].

Secondly, the algorithm consists of orthogonal (unitary) transformations. These transformations are renowned for their good numerical behavior. There is a good reason to prefer Givens and Jacobi rotations over other types of orthogonal transformations, such as Householder transformations. Givens and Jacobi rotations are computed based on local information ($2 \times 2$ or $2 \times 1$ sub-matrices), and therefore attract much attention in the field of parallel computing.

A high-level SFG of Algorithm 2 is shown in Figure 2.12. The upper square operator takes in the vector $x_{[k]}$ and the matrix $V_{[k-1]}$. It computes the product vector $\tilde{x}_{[k]}$, which is propagated to the lower triangular operator. This operator takes care of the QRD updating and the generation of the Jacobi rotations and their application to the matrix $R_{[k-1]}$. The column transformation $\Phi_{[k]}$ is propagated upwards into the square operator, where the matrix-matrix product $V_{[k]} = V_{[k-1]} \cdot \Phi_{[k]}$ is finally computed. Details on the internal structure of the operators are given in the next signal flow graph of Figure 2.13. This SFG is an intertwining of two SFGs.

The first SFG consists of the rectangular nodes and the black arcs. The nodes execute the matrix-vector multiplication (upper square array) and QRD updating (lower triangular array). The black arcs indicate the flow of the row transformation parameters. The remarkable resemblance of the data flow of matrix-vector product and the QRD updating, is due to the fact that the matrix-vector product is considered as a row transformation

$$\begin{bmatrix} V_{[k-1]} \\ \tilde{x}_{[k]}^H \end{bmatrix} = \begin{bmatrix} I_M & O_{M\times 1} \\ x_{[k]}^H & 1 \end{bmatrix} \cdot \begin{bmatrix} V_{[k-1]} \\ O_{1\times M} \end{bmatrix}.$$

The row transformation matrix can easily be decomposed as a sequence of $M$ elementary Gauss transformations acting on 2 rows

$$G_{[k]}^{i|M+1} = \begin{bmatrix} 1 & 0 \\ x_{i,[k]}^* & 1 \end{bmatrix}.$$

The zero row $O_{1\times M}$ is put in at the top side. This first part of the SFG is actually identical to the SFG of the generalized sidelobe canceler (Figure 2.7).

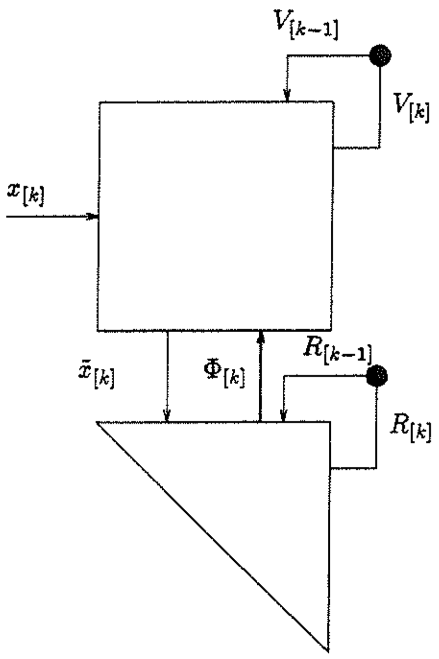


Figure 2.12: High level signal flow graph of the Jacobi-type SVD updating algorithm. The square part performs matrix-vector multiplication and column rotations. The triangular part performs QRD updating and computes the two-sided SVD rotations.

The second part of the SFG is dedicated to the Jacobi row and column rotations. They are executed in the hexagonal nodes. The flow of the rotation parameters is indicated by the grey shaded arcs. The row and column transformations are computed simultaneously on the diagonal and therefore both nodes are linked. The row rotations are then propagated to the right, whereas the column rotations are propagated upwards.

This SVD updating algorithm is essentially sequential. It contains long bidirectional dependency paths of length $\mathcal{O}(4M)$ (small arrows in Figure 2.13). Since there are no delays in one of the directions, this path cannot be pipelined using the standard retiming rules. In a synchronized system, the clock speed is upper bounded by the execution time of the longest dependency path in the algorithm. Implementing Algorithm 2 as such on a planar parallel architecture would result in a very inefficient

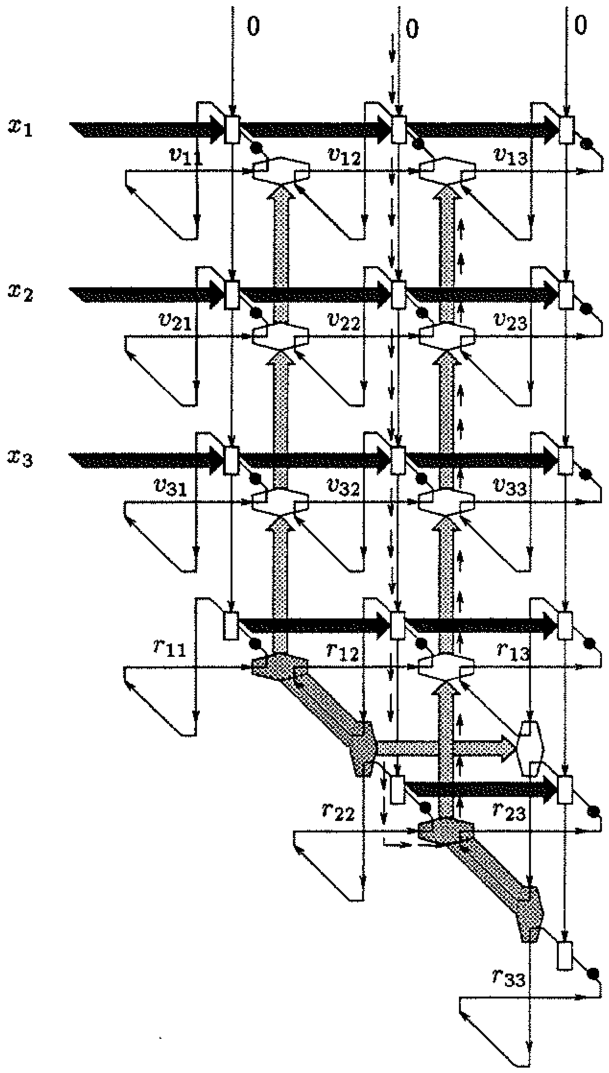


Figure 2.13: Detailed SFG of the sequential Jacobi SVD updating algorithm. The arrows indicate the long bidirectional vertical dependency path.

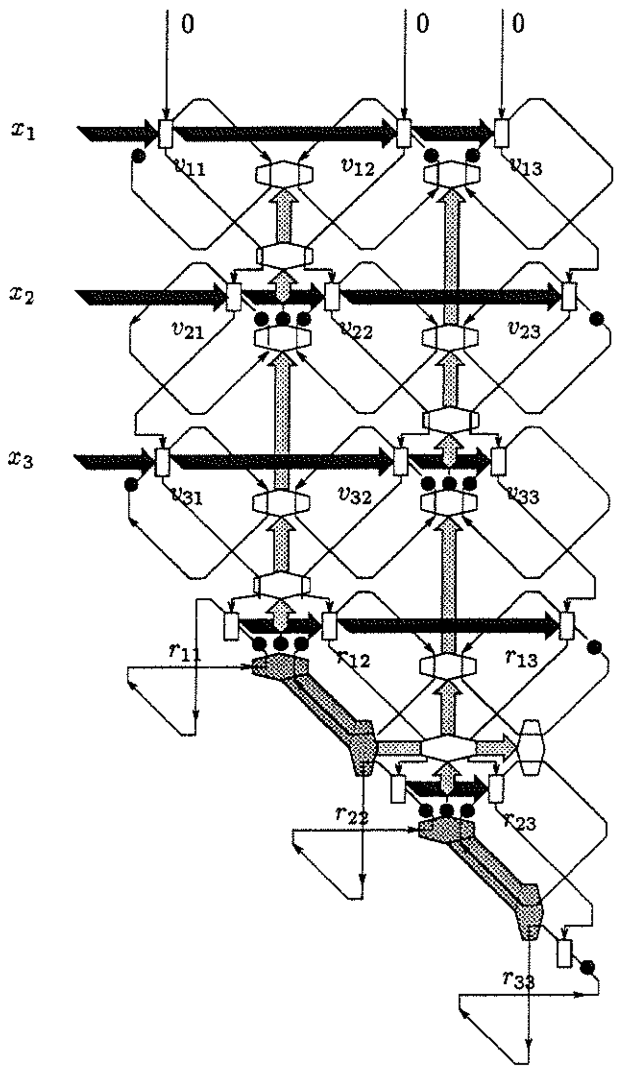


Figure 2.14: Detailed SFG of the parallel Jacobi SVD updating algorithm. The long bidirectional vertical dependency path has been removed.

array with throughput $\mathcal{O}(M^{-1})$. Therefore, the long vertical data dependency loop has to be broken. The only way is to modify the algorithm itself. The re-engineering of the Jacobi SVD algorithm in function of an efficient systolic architecture is described in [64]. We do not elaborate on this technique here. Examples of algorithmic transformations on a signal flow graph will be treated in later chapters. The final result is the SFG of Figure 2.14. This SFG corresponds to a second parallelized version of the SVD updating algorithm. Delays have been introduced on the upward vertical dependencies. Therefore this SFG can be pipelined using the time-scaling and delay-transfer retiming rules. The full size systolic architecture has $\mathcal{O}(M^2)$ nodes, and a pipelining period of 4 cycles, independent of $M$.

## 2.4 Conclusion

In this introductory chapter, we have presented the necessary background material. First, a mathematical model for the array data was derived. The concepts of signal subspaces and array manifolds were introduced. Next two common array processing problems, *i.e.,* beamforming and direction finding, were treated. The generalized sidelobe canceler algorithm for LCMV beamforming and the ESPRIT algorithm for direction finding were introduced. These algorithms are the starting point for some of the array processing algorithms in later chapters.

In practice, the signal constellation can often only be considered stationary for a limited time interval. Therefore, special attention was paid to recursive algorithms. The combination of high data rates and heavy computation of matrix decomposition algorithms motivated the study of parallel architectures. The paradigm of systolic arrays turned out to be well-matched for the type of matrix problems we address.

The generalized sidelobe canceler array presented a completed solution for adaptive and recursive parallel LCMV beamforming. The recursive direction finding problem was only partially solved here. We only focused on an efficient systolic architecture for the most burdensome part of the computation, which is the SVD updating. A recursive algorithm and systolic architecture for narrow-band direction finding, based on the Jacobi SVD updating method and the ESPRIT algorithm, is described in [71].

# Chapter 3

# Factored Jacobi SVD Updating

In this and the next chapter new algorithms and architectures are introduced for subspace tracking. This operation is an important component of adaptive direction finding algorithms. The new algorithms track the dominant subspace of the data matrix by an orthogonal matrix spanning the subspace. They are an illustration of the importance of orthogonal matrix factorizations in modern signal processing. The QR decomposition, Schur decomposition, singular value decomposition and their various generalizations have become standard components of advanced algorithms. Orthogonal decompositions are often used for their good numerical behavior. Errors due to finite precision arithmetic do not blow up if orthogonal transformations are used. However, the errors do not die out either. If no other mechanism is provided, rounding errors caused by successive orthogonal transformations accumulate. This is exactly what happens for a whole class of Jacobi-type updating algorithms, examples of which are SVD updating and narrow-band and wide-band direction finding [71, 134], ...

In this chapter, we study this problem for the prototype Jacobi algorithm, *i.e.*, SVD updating. We propose a solution which is based on a minimal factorization of an orthogonal matrix as a sequence of Givens rotations. All calculations are performed on the rotation angles, such that orthogonality is preserved by construction.

Moreover, the factorization replaces the matrix-vector multiplication in the SVD updating algorithm by a sequence of Givens rotations. This has an additional benefit on the architectural level, since now a parallel architecture can be constructed solely using rotation cells. The outcome is a new elegant systolic Jacobi array for SVD updating.

In the first section the error accumulation phenomenon is illustrated. Next, in the second section we review the factorization of an orthogonal matrix $V \in \mathbb{R}^{M \times M}$ and show how an orthogonal matrix-vector multiplication may be implemented on a triangular array of rotation cells. The third section presents the major contribution of this chapter, which is an efficient scheme to update the rotation angles without explicit computation of $V$. Finally section 4 is concerned with the new parallel architecture. Its derivation is non-trivial, due to the presence of bidirectional data flow.

## 3.1 Error accumulation

In section 2.3.2 we have studied the Jacobi SVD updating algorithm. Because this algorithm is based on orthogonal transformations, one might be tempted to consider it as fully numerically robust. Unfortunately, this is not the case. Algorithm 2 suffers from error accumulation in finite precision arithmetic. At each time instant $t_k$ the matrix[1] $V_{[k-1]}$ is multiplied by a sequence of $M-1$ Givens rotations $\Phi_{[k]}^{i|i+1}$. When executed on a standard multiply-and-accumulate processor, rounding errors in the multiplications perturb $V_{[k]}$ in a random way. These errors do not decay, but keep on accumulating. A measure for the deviation from orthogonality of $V_{[k]}$ is given by the Frobenius norm

$$\delta_{[k]} = \|V_{[k]} \cdot V_{[k]}^T - I_M\|.$$

The linear growth of $\delta_{[k]}$ in Figure 3.1 illustrates that error accumulation gradually destroys the orthogonality of $V_{[k]}$. A theoretical error analysis can be found in Gentleman [30]. This unlimited growth is clearly unacceptable. Moreover, an error analysis has shown that the orthogonality of $V_{[k]}$ is crucial for the numerical stability and accuracy of the Jacobi SVD updating algorithm [66].

The algorithm can be stabilized by including a periodic reorthogonalization scheme based on symmetric Gram-Schmidt orthogonalization [66]. The rows of $V_{[k]}$ are continuously reorthogonalized by means of $2 \times 2$

[1]We restrict ourselves to real matrices in this chapter. The generalization to complex matrices is straightforward. However, a complex $2 \times 2$ rotation has 3 real degrees of freedom, which would unnecessarily complicate the exposition.

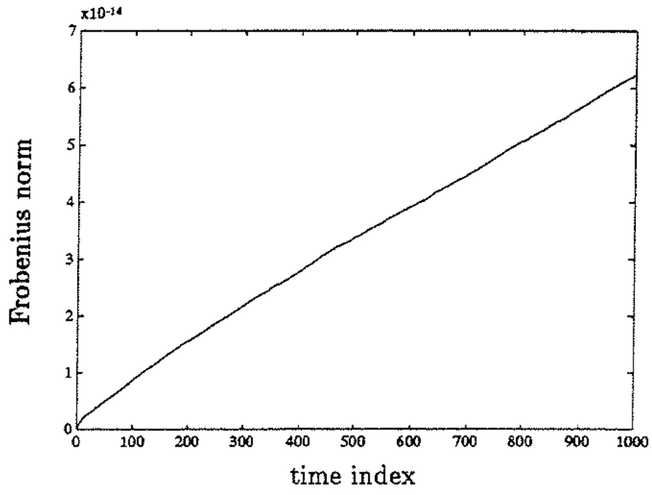


Figure 3.1: Error accumulation averaged over 10 independent simulations. In each simulation the identity matrix $I_{20}$ is multiplied by sequences of Givens rotations with uniformly distributed angles. The simulation was done in MATLAB on a DECstation 5000/120 with machine precision $\epsilon = 2.22e - 16$.

transformations. However, this scheme does not guarantee orthonormality at each iteration. In combination with exponential weighting of the data, it only keeps the deviation sufficiently low and bounded. Secondly, the resulting parallel implementation is rather tricky and elaborate.

An alternative to keep a matrix orthogonal is to parameterize the matrix in terms of a set of rotation angles. In applications where the matrix is constant, *e.g.*, orthogonal filters [90], this parameterization has been used extensively. In time-varying adaptive signal processing, updating the angles becomes an issue. In [88] a triangular array is presented for tracking the eigenvalue decomposition (EVD) of a time-varying correlation matrix. The rotation angles are updated using a steepest-descent technique. This technique is clearly not suited for the Jacobi SVD updating algorithm. In this paper we present a new technique using Givens rotations only.

## 3.2 Orthogonal matrix - vector product

In this section we study the factorization of $V_{[k]}$ as a finite chain of planar Givens rotations, each characterized by one angle. By tracking the angles instead of the matrix, $V_{[k]}$ can never leave the manifold of orthogonal matrices. Rounding errors only perturb the rotation angles, the perturbed $V_{[k]}$ still being orthogonal. First we show how an arbitrary orthogonal matrix can be factored uniquely as a chain of Givens rotations[2] $Q^{i|j}$. Secondly we use this factorization to compute a product $x_{[k]}^T \cdot V_{[k-1]}$. This results in a regular locally connected array for orthogonal matrix - vector multiplication.

**Lemma 1** *Any orthogonal matrix $V \in \mathbb{R}^{M \times M}$ can be factored uniquely into a product of $M \cdot (M-1)/2$ Givens rotations $Q^{i|j}$ and a signature matrix $S$*

$$V = \left( \prod_{i=1}^{M-1} \prod_{j=i+1}^{M} Q^{i|j} \right) \cdot S$$

*where $S$ is equal to the identity matrix of size $M$, except that the last diagonal entry is $\pm 1$.*

Example and proof

For a $4 \times 4$ orthogonal matrix, the factorization is given by

$$V = Q^{1|2} \cdot Q^{1|3} \cdot Q^{1|4} \cdot Q^{2|3} \cdot Q^{2|4} \cdot Q^{3|4} \cdot S.$$

To construct the factorization, it is sufficient to apply the well-known Givens method for QR decomposition [33].

$$\underbrace{\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}}_{V} \xrightarrow{Q^{1|2^T}} \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \xrightarrow{Q^{1|3^T}} \begin{bmatrix} \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}$$

$$\xrightarrow{Q^{1|4^T}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & \times & \times & \times \end{bmatrix} \xrightarrow{Q^{2|3^T}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \\ 0 & \times & \times & \times \end{bmatrix}$$

[2]The notation $Q^{i|j}$ is used with some flexibility. Depending on the context it denotes a $2 \times 2$ rotation matrix, or an embedding of this rotation matrix in a higher dimensional identity matrix.

$$\xrightarrow{Q^{2|4^T}} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{bmatrix} \xrightarrow{Q^{3|4^T}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \pm 1 \end{bmatrix}}_{S}$$

At each stage we need to compute an angle $\alpha^{i|j}$ such that after rotation, $v'_{ji}$ is zeroed

$$\begin{bmatrix} \cos(\alpha^{i|j}) & \sin(\alpha^{i|j}) \\ -\sin(\alpha^{i|j}) & \cos(\alpha^{i|j}) \end{bmatrix} \cdot \begin{bmatrix} v_{ii} \\ v_{ji} \end{bmatrix} = \begin{bmatrix} v'_{ii} \\ 0 \end{bmatrix}.$$

The angle $\alpha^{i|j}$ is unique if the convention is taken that $v'_{ii}$ is non-negative. This holds true if

$$\cos(\alpha^{i|j}) = \frac{v_{ii}}{\sqrt{v_{ii}^2 + v_{ji}^2}}, \qquad \sin(\alpha^{i|j}) = \frac{v_{ij}}{\sqrt{v_{ii}^2 + v_{ji}^2}}.$$

If $v_{ii} = 0 = v_{ji}$, we define $\alpha^{i|j} = 0$.

After zeroing all off-diagonal elements in a column, the diagonal entry equals 1 since the columns of an orthogonal matrix have unit-norm. The same argument holds true for the rows. Finally, the sign of the $(M, M)$-th entry of $S$ is not controlled by the algorithm. It is positive or negative depending on the sign of the determinant of $V$. ■

Comments

1. In the Jacobi-type updating algorithms only orthogonal matrices with positive determinant are encountered such that the signature matrix can be omitted.

2. Here the Givens QRD method introduces zeros column-wise by combining in column $j$ the entries on rows $i > j$ and $j$. Other choices of sequences of $\frac{M \cdot (M-1)}{2}$ $2 \times 2$ rotations are feasible. We could have zeroed each $v_{ij}$ element by rotating neighboring rows $i-1$ and $i$ instead of rows $i$ and $j$. Or we could have introduced zeros row-by-row instead of column-by-column [33]. Evidently, another choice and sequence of the rotation planes results in different values for the rotation angles. Below, we concentrate on the sequence of Lemma 1. Its dependencies are depicted in Figure 3.2. The results that will be obtained, can also be derived for the other sequences mentioned.

$$\begin{tikzcd}
Q^{1|2} \arrow[r] & Q^{1|3} \arrow[r] \arrow[dr] & Q^{1|4} \arrow[r] \arrow[dr] & Q^{1|5} \arrow[dr] & & & \\
& & Q^{2|3} \arrow[r] & Q^{2|4} \arrow[r] \arrow[dr] & Q^{2|5} \arrow[dr] & & \\
& & & & Q^{3|4} \arrow[r] & Q^{3|5} \arrow[dr] & \\
& & & & & & Q^{4|5}
\end{tikzcd}$$

Figure 3.2: The dependencies of the sequence of Lemma 1 ($M = 5$). Rotations must be sequentially processed from left to right, while rotations on a vertical line can be processed in parallel.

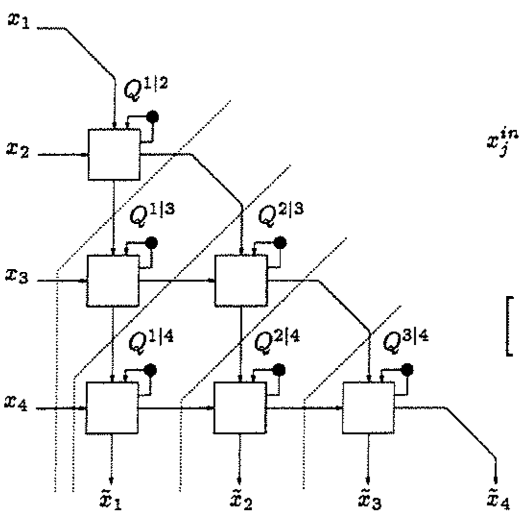
3. The ideal hardware component for fast computation of the angles, given a matrix $V$, is a CORDIC processor in angle accumulation mode [143].

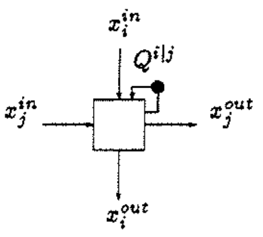
The SFG for a factored orthogonal matrix-vector product $\tilde{x}_{[k]}^T = x_{[k]}^T \cdot V_{[k-1]}$ is shown in Figure 3.3. The triangular graph consists of $M \cdot (M-1)/2$ nodes, having local and regular interconnections. The functionality of each node is the same. It stores the rotation $Q^{i|j}$ and applies it to its input pair coming in from the top and from the left. Its output data pair is propagated to the bottom and to the right respectively. Again the most efficient implementation of a node is a CORDIC processor, this time in vector rotation mode.

Explicit knowledge of the matrix $V$ can be obtained by feeding in the columns of the identity matrix, *i.e.*, the array will output the $j$-th row of $V$ in response to the unit vector $e_j$,

$$e_j = \left[ \begin{array}{ccc} O_{1\times j-1} & 1 & O_{1\times M-j} \end{array} \right]^T .$$

Adding a systolic schedule to this SFG is straightforward. The SFG already has regular local connections. To make all nodes local in time, it suffices to insert a delay cell on all arcs cut by the cut sets (dotted lines) in Figure 3.3. The throughput of the resulting systolic array is maximal, since it has a pipelining period of one cycle. All processors are 100% busy at all time.





$$\begin{bmatrix} x_i^{out} \\ x_j^{out} \end{bmatrix} = Q^{i|j^T} \cdot \begin{bmatrix} x_i^{in} \\ x_j^{in} \end{bmatrix}$$

Figure 3.3: SFG for factored orthogonal matrix-vector multiplication ($M = 4$). The nodes apply the rotation $Q^{i|j}$ to their input pair. The dotted lines represent cut sets to convert the SFG into a systolic array.

This systolic rotation array is not new. It has already been derived in the context of computing the QR decomposition of an arbitrary square matrix [1, 43]. There the array operates in two different modes. While the matrix is passed through the array, the nodes compute their rotation angles $\alpha^{i|j}$ and the triangular matrix $R$ becomes available at the output. Once the rotation angles are fixed, the array performs multiplications just as in Figure 3.3. The array also bears resemblance to the well-known Gentleman-Kung array for QRD updating [31]. However, here the rotation angles are resident in the cells, whereas in the Gentleman-Kung array they are propagated through the array.

The factored orthogonal matrix-vector multiplication has the additional benefit that the scalar multiplications in step 1 of the Jacobi SVD updating algorithm are eliminated. Therefore, the whole algorithm now consists exclusively of Givens rotations. This regularity is important in view of a possible hardware realization. It allows to construct a systolic array for SVD updating only using CORDIC processors, provided that the updating of $V_{[k-1]}$ (step 3) can be done in factored form. This is the topic of the next section.

## 3.3 Updating the angles

In step 3 of Algorithm 2, the $V_{[k-1]}$ matrix is updated with (*i.e.*, post-multiplied by) a sequence of Givens rotations $\Phi_{[k]}^{i|i+1}$. In this section we present an $\mathcal{O}(M^2)$ method to update the factors $Q_{[k-1]}^{i|j}$ directly, without explicit computation of the $V$-matrix.

The updating matrix $\Phi_{[k]}$ is defined as

$$\Phi_{[k]} = \prod_{i=1}^{M-1} \Phi_{[k]}^{i|i+1}.$$

Each transformation of the form[3] $V \leftarrow V \cdot \Phi^{i|i+1}$ will alter several rotation angles. Starting from the tail, the rotation $\Phi^{i|i+1}$ is worked backwards into the factorization and interacts with several rotations. The nature of the interaction depends on the relative position of the coordinate planes defined by the indices of the interacting rotations $Q^{k|l}$ and $\Phi^{i|i+1}$. Three types of transformations have to be considered.

1. <u>The index pairs $(k,l)$ and $(i,i+1)$ are disjoint.</u>

   In this case the rotation matrices $Q^{k|l}$ and $\Phi^{i|i+1}$ commute since they affect different rows or columns.

   $$\Phi^{i|i+1} \cdot Q^{k|l} = Q^{k|l} \cdot \Phi^{i|i+1}$$

2. <u>The index pairs $(k,l)$ and $(i,i+1)$ are equal.</u>

   Here the rotation angles of $Q^{i|i+1}$ and $\Phi^{i|i+1}$ simply add together.

   $$Q_*^{i|i+1} = Q^{i|i+1} \cdot \Phi^{i|i+1}$$

3. <u>The index pairs $(k,l)$ and $(i,i+1)$ share a common index.</u>

   This is the complicated case. Let $k = i+1$. Generically, the matrices $Q^{i+1|l}$ and $\Phi^{i|i+1}$ do not commute and it is even impossible to calculate an equivalent pair of rotations such that $\Phi_*^{i|i+1} \cdot Q_*^{i+1|l} = Q^{i+1|l} \cdot \Phi^{i|i+1}$. However, reordering the indices becomes possible if a third rotation, $Q^{i|l}$, is taken into account. The sequence of 3 Givens

[3]If no confusion can arise, the time index is omitted for notational convenience.

rotations in the $(i,l)$, $(i+1,l)$, $(i,i+1)$-planes, defines a rotation in the 3-dimensional $(i,i+1,l)$-space

$$V^{i|i+1|l} = Q^{i|l} \cdot Q^{i+1|l} \cdot \Phi^{i|i+1}.$$

This 3-dimensional rotation $V^{i|i+1|l}$ can also be represented by a different set of three Givens rotations by choosing an ordering of the coordinate planes in which the $(i,i+1)$-plane is in front.

$$V^{i|i+1|l} = \Phi_*^{i|i+1} \cdot Q_*^{i|l} \cdot Q_*^{i+1|l}$$

There is no simple trigonometric expression for the mapping from the former to the latter set of angles. A natural way is to compute $V^{i|i+1|l}$ explicitly and refactor it. The computational complexity of this $3 \times 3$ core problem is relatively low and independent of the matrix dimension $M$.

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{I_3} \xrightarrow{\Phi^{i|i+1}} \begin{bmatrix} \times & \times & 0 \\ \times & \times & 0 \\ 0 & 0 & 1 \end{bmatrix} \xrightarrow{Q^{i+1|l}} \begin{bmatrix} \times & \times & 0 \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{Q^{i|l}}$$

$$\underbrace{\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}}_{V^{i|i+1|l}} \xrightarrow{\Phi_*^{i|i+1^T}} \begin{bmatrix} \times & \times & \times \\ 0 & \times & \times \\ \times & \times & \times \end{bmatrix} \xrightarrow{Q_*^{i|l^T}} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \times & \times \\ 0 & \times & \times \end{bmatrix} \xrightarrow{Q_*^{i+1|l^T}}$$

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{I_3}$$

It is even sufficient to compute only two columns of $V^{i|i+1|l}$ for the determination of the equivalent set of rotations. By selecting the first and last column, the operation count is optimized to 7 rotations over a given angle (vector rotation) and 3 rotations in which a coordinate is zeroed (angle accumulation). On a CORDIC processor both operations have the same complexity.

Below the course of the computations in the $4 \times 4$ example is detailed for the first updating rotation $\Phi^{1|2}$. In the first line, $\Phi^{1|2}$ can be commuted with $Q^{3|4}$ (type 1). To interchange $\Phi^{1|2}$ with $Q^{2|4}$, $Q^{1|4}$ must be adjacent to $Q^{2|4}$. Therefore, $Q^{1|4}$ is commuted with $Q^{2|3}$. Then the equivalent set of rotations in the $(1,2,4)$-space is determined (type 3). The same operations are then repeated in the $(1,2,3)$-space. On the last line the angles in the $(1,2)$-plane are summed (type 2).

$$\begin{array}{rcll}
V \cdot \Phi^{1|2} & = & Q^{1|2} \cdot Q^{1|3} \cdot Q^{1|4} \cdot Q^{2|3} \cdot Q^{2|4} \cdot Q^{3|4} \cdot \Phi^{1|2} & \\
 & = & Q^{1|2} \cdot Q^{1|3} \cdot Q^{1|4} \cdot Q^{2|3} \cdot Q^{2|4} \cdot \Phi^{1|2} \cdot Q^{3|4} & (T1) \\
 & = & Q^{1|2} \cdot Q^{1|3} \cdot Q^{2|3} \cdot Q^{1|4} \cdot Q^{2|4} \cdot \Phi^{1|2} \cdot Q^{3|4} & (T1) \\
 & = & Q^{1|2} \cdot Q^{1|3} \cdot Q^{2|3} \cdot \Phi_*^{1|2} \cdot Q_*^{1|4} \cdot Q_*^{2|4} \cdot Q^{3|4} & (T3) \\
 & = & Q^{1|2} \cdot \Phi_{**}^{1|2} \cdot Q_*^{1|3} \cdot Q_*^{2|3} \cdot Q_*^{1|4} \cdot Q_*^{2|4} \cdot Q^{3|4} & (T3) \\
 & = & Q_*^{1|2} \cdot Q_*^{1|3} \cdot Q_*^{1|4} \cdot Q_*^{2|3} \cdot Q_*^{2|4} \cdot Q^{3|4} & (T2, T1)
\end{array}$$

The above computations are nicely illustrated on the SFG for orthogonal matrix-vector multiplication in Figure 3.4. Post-multiplying $V$ by $\Phi^{1|2}$ creates a new node at the (1,2)-components of the output of the graph. This node is gradually worked into the SFG. In each group of 3 encircled cells, a type 3 transformation pushes the $\Phi^{i|i+1}$-node upwards (Figure 3.4a and 3.4b). A type 2 transformation merges two nodes (Figure 3.4c). Since the $\Phi^{i|i+1}$-node propagates upwards, it is combined only with nodes which share at least one index. The commutations (type 1 transformations) follow naturally from the structure of the graph.

The SFG for updating the complete parameterization is shown in Figure 3.5. First the bottom-left node updates the rotations $Q^{1|4}$ and $Q^{2|4}$. Next, the neighboring nodes (up and to the right) perform their computation in parallel. The computation gradually evolves towards the diagonal.

The horizontal contraflow in this SFG complicates its pipelining. To introduce delay elements on all arcs, we need to apply the retiming rules. First all delays in the graph are doubled (time scaling rule), and consequently the rate is halved at which new data are fed in into the array. Then one delay is added on all outward arcs crossing the cut (dotted lines in Figure 3.5) and one delay is removed on all inward arcs (delay transfer rule). The outcome is the systolic schedule of Figure 3.6, which has a pipelining period of 2 cycles. In each cycle, only half of the processors are active. Therefore, in a hardware realization, two nodes can be clustered into one processor to obtain 100% utilization of the hardware.

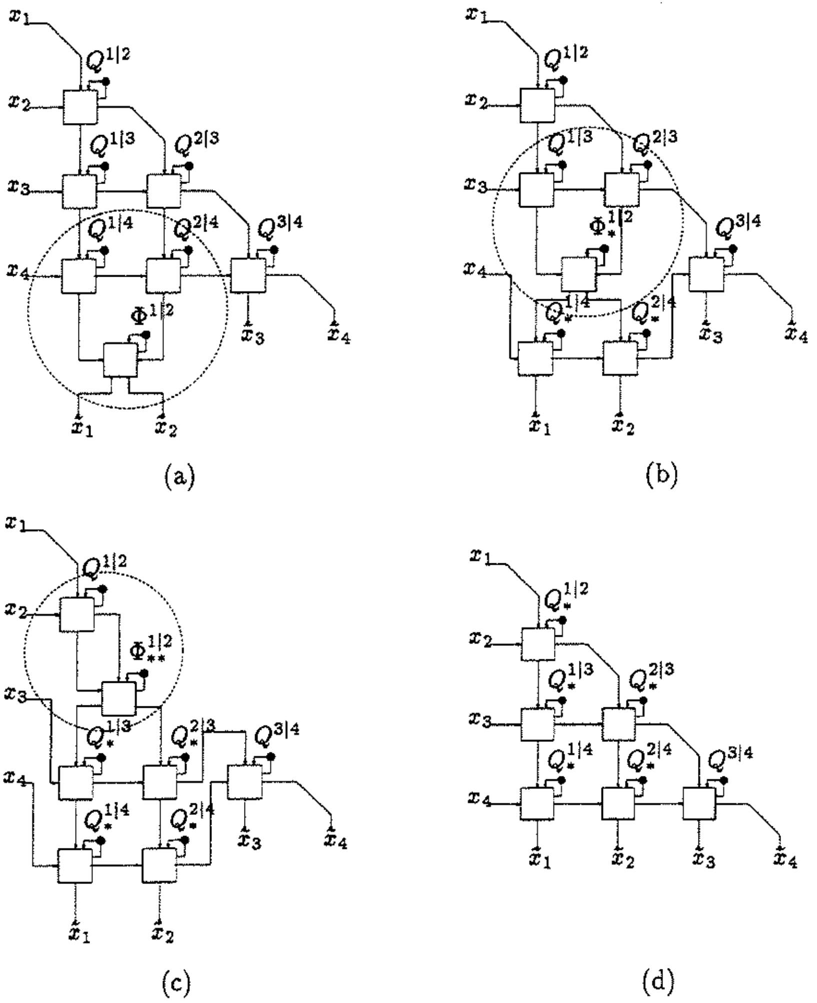
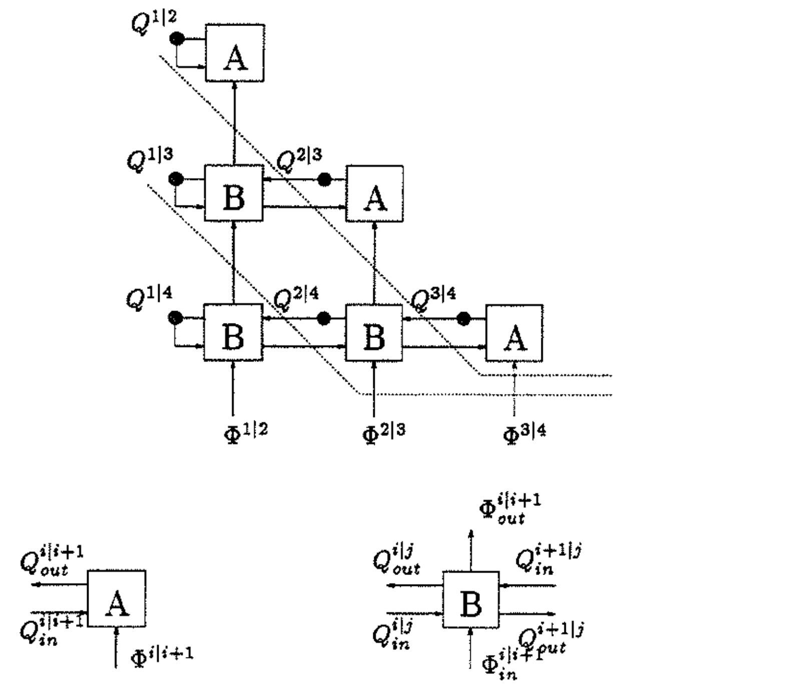


Figure 3.4: Updating the SFG for factored orthogonal matrix-vector multiplication ($M = 4$). In (a) and (b) a type 3 transformation is performed. In (c) a type 2 transformation is done.



$$Q^{i|i+1}_{out} = Q^{i|i+1}_{in} \cdot \Phi^{i|i+1} \qquad \Phi^{i|i+1}_{out} \cdot Q^{i|j}_{out} \cdot Q^{i+1|j}_{out} = Q^{i|j}_{in} \cdot Q^{i+1|j}_{in} \cdot \Phi^{i|i+1}_{in}$$

Figure 3.5: SFG for updating the angles ($M = 4$). The dotted lines are cut sets. In order to fully pipeline the SFG, first all delays have to be doubled.

## 3.4 A modified array for SVD updating

The novel factored Jacobi SVD updating algorithm is now as in Algorithm 3.

**Algorithm 3**

```
    $Q^{i|j}_{[0]} \leftarrow I_2$    for $1 \le i \le M-1; i+1 \le j \le M$
    $R_{[0]} \leftarrow O_{M\times M}$
for $k = 1, \cdots, \infty$
  1. factored orthogonal matrix-vector multiplication
        $\tilde{x}^T_{[k]} \leftarrow x^T_{[k]} \cdot \prod_{i=1}^{M-1} \prod_{j=i+1}^{M} Q^{i|j}_{[k-1]}$
```

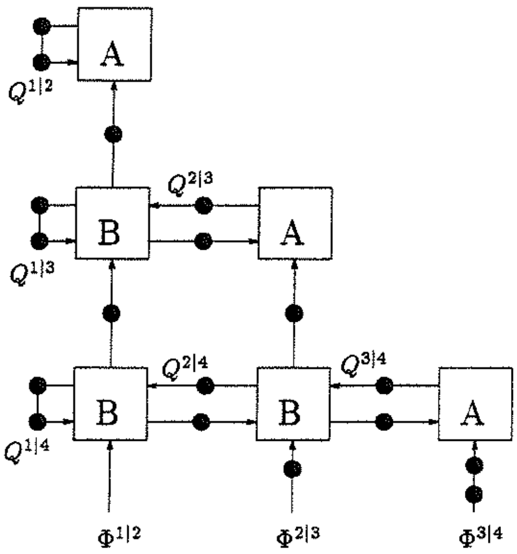


Figure 3.6: Systolic architecture for the update of the angles ($M = 4$).

2. QRD updating

$$\left[\frac{\tilde{R}_{[k]}}{0}\right] \leftarrow \prod_{i=1}^{M} G_{[k]}^{M+1-i|M+1^T} \cdot \left[\frac{\alpha \cdot R_{[k-1]}}{\tilde{x}_{[k]}^T}\right]$$

3. Jacobi rotations

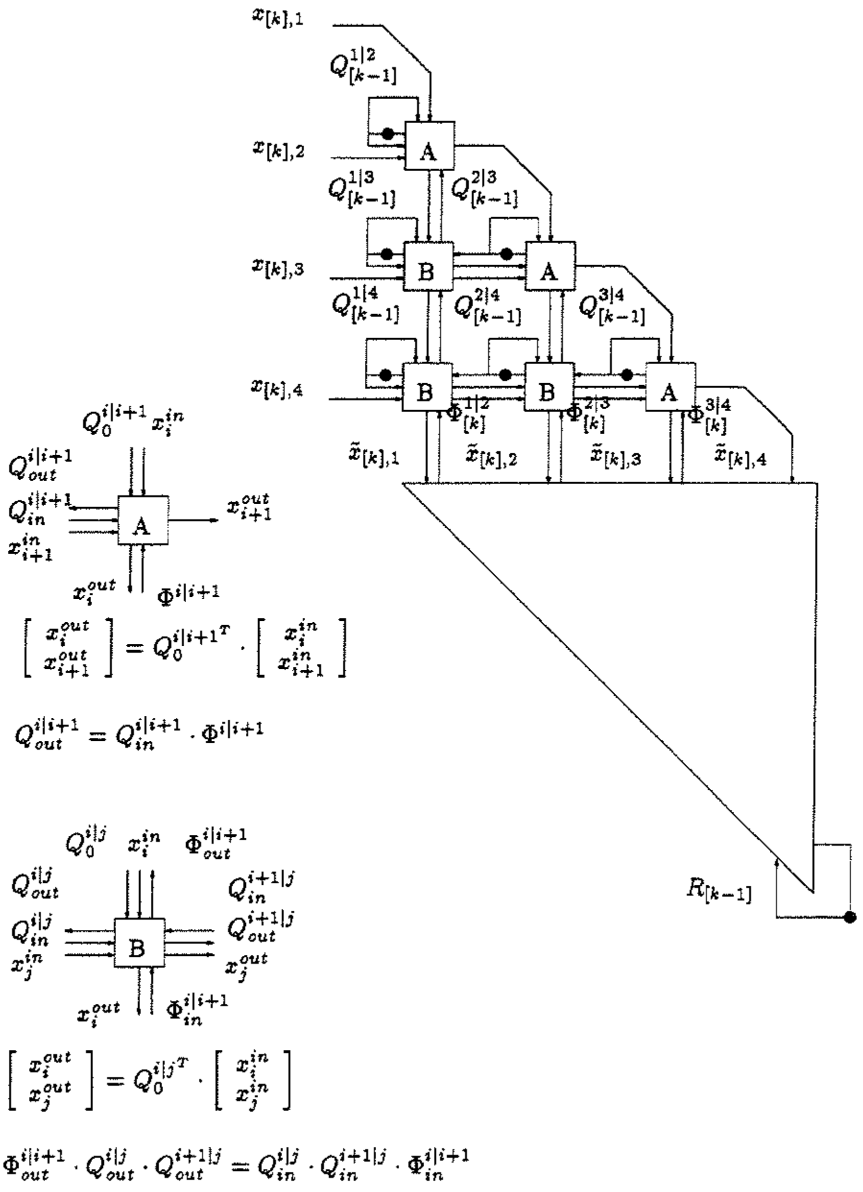
$$R_{[k]} \leftarrow \prod_{i=1}^{M-1} \Theta_{[k]}^{M-i|M+1-i^T} \cdot \tilde{R}_{[k]} \cdot \prod_{i=1}^{M-1} \Phi_{[k]}^{i|i+1}$$

```
    for $j \leftarrow 2$ to $M$
      $Q_*^{1|j} \leftarrow Q_{[k-1]}^{1|j}$
    endfor
    for $i \leftarrow 1$ to $M-1$
      $\Phi_*^{i|i+1} \leftarrow \Phi_{[k-1]}^{i|i+1}$
      for $j \leftarrow M$ downto $i+2$
        $\Phi_*^{i|i+1} \cdot Q_{[k]}^{i|j} \cdot Q_*^{i+1|j} \leftarrow Q_*^{i|j} \cdot Q_{[k-1]}^{i+1|j} \cdot \Phi_*^{i|i+1}$
      endfor
      $Q_{[k]}^{i|i+1} \leftarrow Q_*^{i|i+1} \cdot \Phi_*^{i|i+1}$
    endfor
endfor
```

The two arrays for orthogonal matrix-vector multiplication and for updating the parameterization only partially implement Algorithm 3. The complete SVD array consists of the combined triangular array of Figures 3.3 and 3.5, placed on top of a triangular array which performs QRD updating and generates the row and column transformations [67] (Figure 3.7). Here we are concerned with the factorization of $V_{[k]}$. Therefore, the lower array is represented as a black box, which takes in matrix-vector products and generates column transformations. In the upper array both functionalities of the matrix-vector product and the updating of the angles are combined in the node descriptions.

Pipelining the combined array is not straightforward. The key problem is the long vertical dependence loop. First the matrix-vector product $\tilde{x}_{[k]}$ runs downwards through the upper array storing the factorization of $V_{[k-1]}$ until it reaches the lower array. There a new column transformation $\Phi_{[k]}$ is generated based on $R_{[k-1]}$ and $\tilde{x}_{[k]}$. Finally, $\Phi_{[k]}$ runs upwards again to update $V_{[k-1]}$ to $V_{[k]}$. Only when this cycle is completed, a new observation vector $x_{[k+1]}$ can be fed in into the array. This long feedback loop causes an $\mathcal{O}(M^{-1})$ throughput. Therefore, it has to be broken to convert the SFG into a pipelined systolic array, which achieves a throughput independent of the matrix dimension $M$. Unfortunately, it is impossible to retime the SFG using the time-scaling and delay-transfer rules only. These rules can only cope with bidirectional data flow on condition that the arcs have at least one associated delay along one of the directions. This was the case with the horizontal data flow of the SFG in Figure 3.5. However, the SFG of Figure 3.7 contains vertical arcs without delays both pointing upwards and downwards.

To introduce delays on the vertical arcs, the algorithm itself has to be modified. A slightly different algorithm with a small increase in computations due to additional rotations, will result. This is the price one has to pay for the increased throughput rate of the systolic schedule. We will introduce delays on the upward $\Phi$-arcs and correct for them on all downward arcs. This technique of algorithmic transformations has already successfully been applied both in the development of a systolic array for Algorithm 2 and in the derivation of the most effective array for stable recursive least squares (RLS) updating known to date [46, 65]. Below we explain its application to the upper triarray.



$$\begin{bmatrix} x^{out}_i \\ x^{out}_{i+1} \end{bmatrix} = Q^{i|i+1^T}_0 \cdot \begin{bmatrix} x^{in}_i \\ x^{in}_{i+1} \end{bmatrix}$$

$$Q^{i|i+1}_{out} = Q^{i|i+1}_{in} \cdot \Phi^{i|i+1}$$

$$\begin{bmatrix} x^{out}_i \\ x^{out}_j \end{bmatrix} = Q^{i|j^T}_0 \cdot \begin{bmatrix} x^{in}_i \\ x^{in}_j \end{bmatrix}$$

$$\Phi^{i|i+1}_{out} \cdot Q^{i|j}_{out} \cdot Q^{i+1|j}_{out} = Q^{i|j}_{in} \cdot Q^{i+1|j}_{in} \cdot \Phi^{i|i+1}_{in}$$

Figure 3.7: SFG of the complete SVD updating problem.

Assume that in Figure 3.7 delays are added to the $\Phi$-arcs originating from the $R$-array[4]. After completion of all computations at time $t_{k-1}$, the array now stores the factorization of $V_{[k-2]}$ instead of $V_{[k-1]}$. The updates of the $V$-matrix run one time step late. Fortunately, one can easily correct for this lag. It suffices to reorder the computation at time $t_k$. If we first update $V_{[k-2]}$ to $V_{[k-1]}$ and afterwards perform the multiplication $x_{[k]}^T \cdot V_{[k-1]}$, the lower triarray receives the correct vector $\tilde{x}_{[k]}$. The modified SFG is shown in Figure 3.8. It is hardly changed. Only the functionalities in the nodes are swapped (cells A1 and B1).

The introduction of delays on the $\Phi$-arcs in the next (and subsequent) layer(s) requires much more complicated corrections. (see Figure 3.9). The updating transformations now run two time steps late, such that the $M-2$ top rows store part of the factorization of $V_{[k-3]}$ instead of $V_{[k-1]}$. Let $\bar{x}_{[k]} \in \mathbb{R}^{M-1}$ denote the partial matrix-vector multiplication

$$\bar{x}_{[k]}^T = \underline{x}_{[k]}^T \cdot \bar{V}_{[k-1]}$$

where $\underline{x}_{[k]}$ is the vector $x_{[k]}$ without its last component and the orthogonal matrix $\bar{V}_{[k-1]} \in \mathbb{R}^{(M-1)\times(M-1)}$ is the product of the factorization of $V_{[k-1]}$ where all rotations $Q_{[k-1]}^{i|M}$ involving the last row are left out. This vector $\bar{x}_{[k]}$ should be produced by the second last row of the triarray at time $t_k$ (*e.g.,* see Figure 3.8). However, in Figure 3.9 the second last row outputs the intermediate matrix-vector product

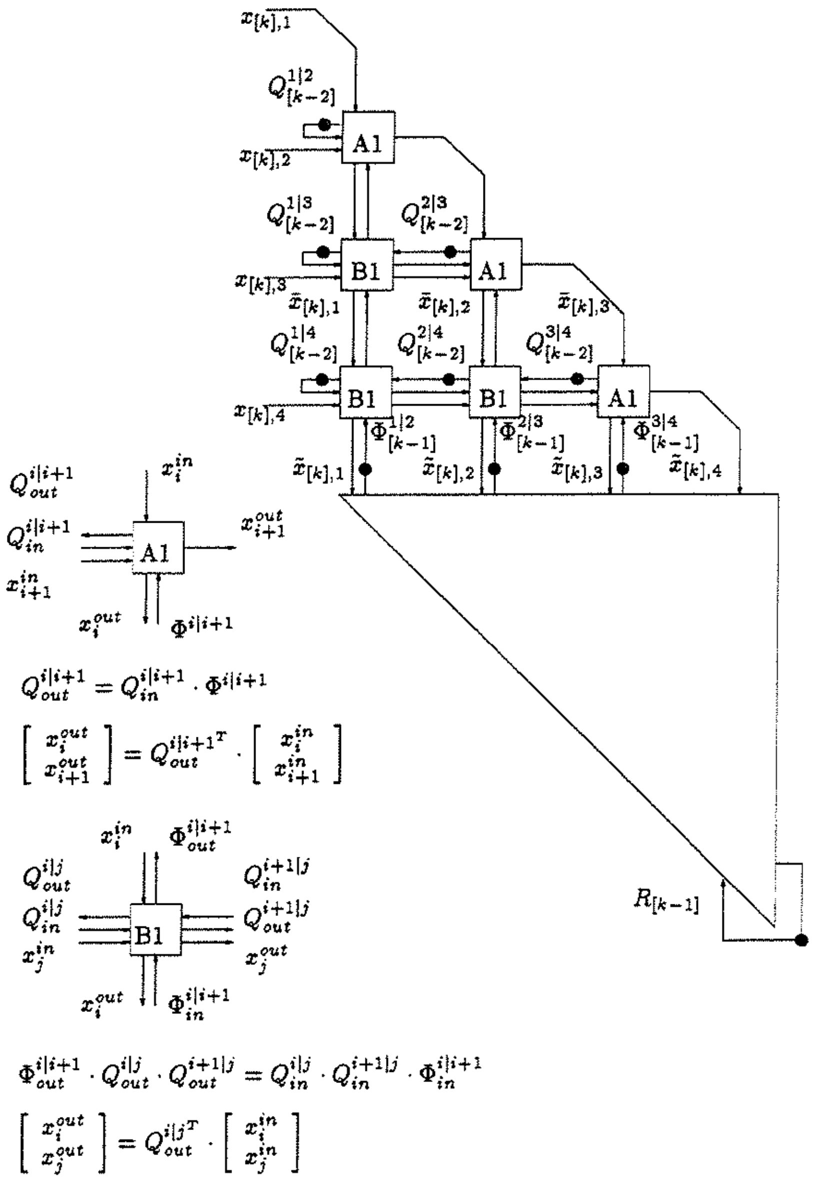
$$\eta_{[k]}^T = \underline{x}_{[k]}^T \cdot \bar{V}_{[k-2]}.$$

Since $\bar{V}_{[k-1]} = \bar{V}_{[k-2]} \cdot \bar{\Phi}_{[k-1]}$ where $\bar{\Phi}_{[k-1]}$ is the product of the $\Phi$-rotations which are propagated upwards by the last layer at time $t_k$, we can correct for the delay by an additional multiplication of $\eta_{[k]}$ and $\bar{\Phi}_{[k-1]}$

$$\bar{x}_{[k]}^T = \eta_{[k]}^T \cdot \bar{\Phi}_{[k-1]}.$$

The transformed SFG is shown in Figure 3.9. The missing rotations $\bar{\Phi}_{[k-1]}^{i|i+1}$ are applied in the additional rotation cells of type C. Nothing prevents to apply the same algorithmic transformation to the remaining $\Phi$-arcs in the graph (Figure 3.10). In this figure the C-type and B1-type cells have been merged into a single B2-type cell.

[4]We make the simplifying assumption that the $R$-array on inputting $x_{[k]}$, immediately generates the $\Phi_{[k]}$ transformations.



$$Q^{i|i+1}_{out} = Q^{i|i+1}_{in} \cdot \Phi^{i|i+1}$$

$$\begin{bmatrix} x_i^{out} \\ x_{i+1}^{out} \end{bmatrix} = Q^{i|i+1^T}_{out} \cdot \begin{bmatrix} x_i^{in} \\ x_{i+1}^{in} \end{bmatrix}$$

$$\Phi^{i|i+1}_{out} \cdot Q^{i|j}_{out} \cdot Q^{i+1|j}_{out} = Q^{i|j}_{in} \cdot Q^{i+1|j}_{in} \cdot \Phi^{i|i+1}_{in}$$

$$\begin{bmatrix} x_i^{out} \\ x_j^{out} \end{bmatrix} = Q^{i|j^T}_{out} \cdot \begin{bmatrix} x_i^{in} \\ x_j^{in} \end{bmatrix}$$

Figure 3.8: SFG after adding delays to the Φ-arcs in the last layer.

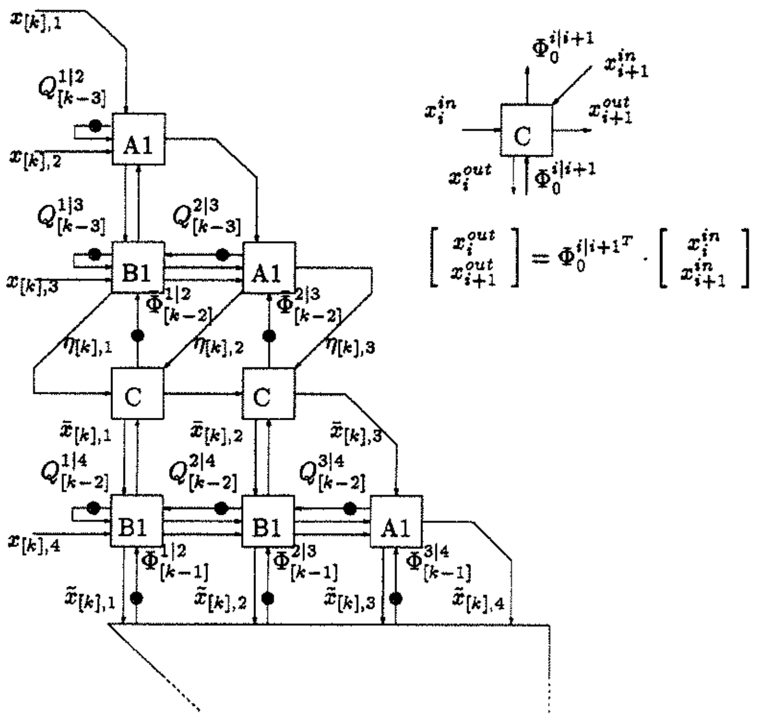
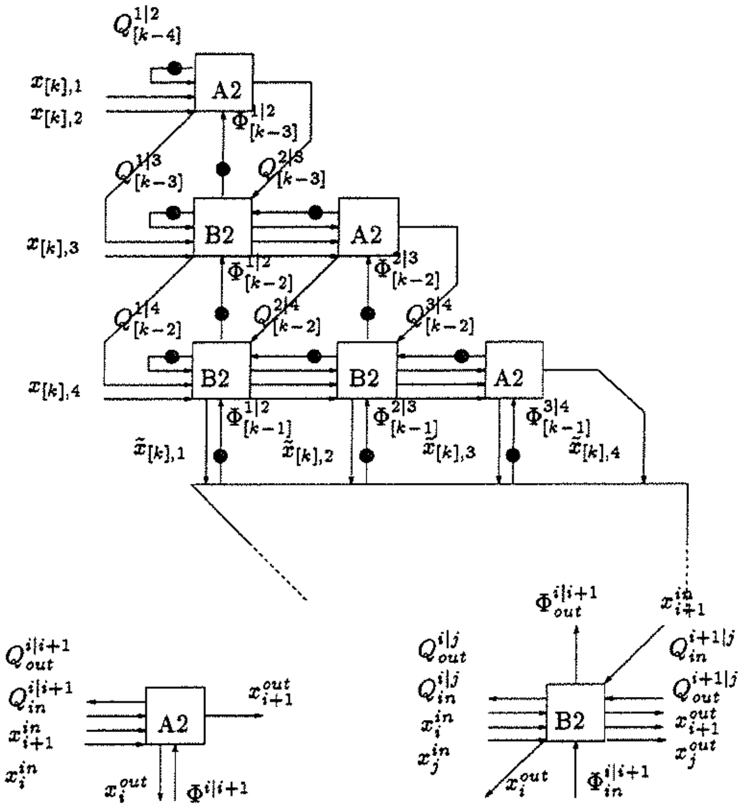


Figure 3.9: SFG with algorithmic transformations in the second last layer.

By these algorithmic transformations we have introduced a delay on all upward dependencies. Now the transformed SFG can be pipelined by making use of the time-scaling and delay-transfer rules. However, these rules cannot be applied consistently without knowledge of the internal structure of the $R$-array. Since this structure is rather complicated, we choose not to incorporate this scheduling here.

## 3.5 Conclusion

In this chapter we have presented a modified algorithm for SVD updating. Two desirable properties are achieved. First the $V$-matrix is kept orthogonal at all time by factoring $V$ as a sequence of $M \cdot (M-1)/2$ Givens rotations. This factorization prevents $V$ from drifting away from orthogonality due to linear error buildup. This is crucial for the numer-



$$Q_{out}^{i|i+1} = Q_{in}^{i|i+1} \cdot \Phi^{i|i+1}$$

$$\begin{bmatrix} x_i^{out} \\ x_{i+1}^{out} \end{bmatrix} = Q_{out}^{i|i+1^T} \cdot \begin{bmatrix} x_i^{in} \\ x_{i+1}^{in} \end{bmatrix}$$

$$\Phi_{out}^{i|i+1} \cdot Q_{out}^{i|j} \cdot Q_{out}^{i+1|j} = Q_{in}^{i|j} \cdot Q_{in}^{i+1|j} \cdot \Phi_{in}^{i|i+1}$$

$$\begin{bmatrix} x_i \\ x_{i+1}^{out} \end{bmatrix} = \Phi_{out}^{i|i+1^T} \cdot \begin{bmatrix} x_i^{in} \\ x_{i+1}^{in} \end{bmatrix}$$

$$\begin{bmatrix} x_i^{out} \\ x_j^{out} \end{bmatrix} = Q_{out}^{i|j^T} \cdot \begin{bmatrix} x_i \\ x_j^{in} \end{bmatrix}$$

Figure 3.10: Fully algorithmically transformed SFG.

ical stability of the recursive algorithm. Secondly the algorithm consists exclusively of planar rotations, which increases the regularity of the operations in its signal flow graph. The modified updating algorithm retains the $\mathcal{O}(M^2)$ operation count of the original SVD updating algorithm.

The factored SVD updating algorithm may be converted into an efficient systolic array. However, because of vertical contraflow, the algorithm has to be manipulated such that the transformed SFG can be pipelined using the time-scaling and delay-transfer retiming rules. At the cost of one extra rotation per node which is off the diagonal of the array, an efficient systolic array with $\mathcal{O}(M^0)$ throughput may be obtained. This array consists exclusively of rotation nodes. Therefore, it may be built in hardware using CORDIC-based processors only.

The parallel implementation on a systolic array makes the algorithm an ideal candidate for real-time SVD tracking applications when the application dictates a throughput unattainable with a single processor.

As a final remark, we mention recent independent work by Olszanskyj and Bojanczyk [77], in which they use the same factorization idea in an RLS algorithm. In order to improve the accuracy of a QRD downdating method, to reduce the amount of storage and to enlarge the potential for parallel implementation, they keep track of the $Q$-factor of the windowed data matrix as a Givens rotation chain. It turns out that updating and downdating this chain involves the same three types of transformations.

# Chapter 4

# Factored Spherical Subspace Tracking

One of the major applications of SVD updating is tracking a slowly time-varying subspace. In the previous chapter we have proposed a numerically stable Jacobi SVD updating algorithm. The Jacobi algorithm did not exactly track the SVD. In order to decrease the computational complexity to $\mathcal{O}(M^2)$, some tracking speed was sacrificed. Several authors have followed the same line of thought [20, 102, 149]. They further reduce the computation at the cost of larger approximation errors.

We propose a modification of the spherical subspace tracking algorithm [20]. This algorithm is a non-iterative algorithm with a very low complexity $\mathcal{O}(M \cdot D)$. Just as in the Jacobi algorithm, the orthogonal matrix which spans the subspace estimate is updated by sequences of Givens rotations. Therefore, the algorithm is also subject to error accumulation. Here we stabilize the algorithm using the factorization of the previous chapter.

In section 2 we derive a spherical SVD updating algorithm, which is a minor variation on the original spherical EVD updating algorithm. Next, in section 3 we employ the factorization of orthogonal matrices, developed in chapter 3, to parameterize the subspace tracking matrix. The novelty is that now only a dominant subspace is tracked. In section 4 we derive a linear systolic array for the factored spherical subspace tracker. This systolic array is very efficient. Finally, in section 5 we present a planar systolic array which is similar to the Jacobi architecture. Algorithmic transformations are also needed in its derivation. Due to some irregularities the final planar array is less efficient than its counterpart for Jacobi

SVD updating.

## 4.1 Spherical subspace tracking

The Jacobi SVD updating algorithm can be used to track a $D$-dimensional subspace in an $M$-dimensional ambient space at $\mathcal{O}(M^2)$ operations per update. In some applications, *e.g.,* radar systems, the number of sensors $M$ is much larger than the number of sources $D$. Although a larger number of sensors results in a better quality of the estimates, it also increases computation and memory demands quadratically. This fast increase is undesirable. Various attempts have been made to develop algorithms with a linear increase of resources as a function of $M$. An overview is given in [149]. The key observation is the fact that subspace algorithms for high-resolution direction tracking or frequency tracking do not need a full eigenvalue decomposition. Since signal and noise subspace are each other's orthogonal complement, it is sufficient to track only the subspace with the smaller dimension (Here, we assume $D < M - D$). Moreover, any (preferably orthogonal) basis spanning this subspace is acceptable. It should not necessarily be the unique basis of eigenvectors.

An attractive class of subspace tracking algorithms with $\mathcal{O}(M \cdot D)$ and $\mathcal{O}(M \cdot D^2)$ complexity, is proposed in DeGroat [20]. The sole information from the noise subspace they use is an averaged noise eigenvalue. The foundation for averaging the noise eigenvalues is the commonly adopted white noise data model. If the noise on all $M$ sensors is independent equal-power and zero-mean, then all $M - D$ noise eigenvalues are identical. In measured data, due to finite amount of data and non-stationarity, there is some spread on the noise eigenvalues. An invariant subspace of which the eigenvalues have been averaged, will be called a spherical subspace. Its eigenvectors are no longer uniquely determined. Any set of orthogonal vectors spanning that subspace will do.

The simplest spherical subspace algorithm is obtained if both signal and noise eigenvalues are averaged. As will be shown below, due to this double averaging the subspace update becomes non-iterative. It may seem a crude approximation to retain only two average eigenvalues. However, Dowling and DeGroat show that in a stationary environment this doubly spherical subspace tracker is still a consistent subspace estimator [25]. Also, the spherical subspace tracking algorithms are robust to the rank degeneracy when tracking crossing sources [20]. By redistributing eigenvalues equally over all eigenvectors, the rank of the estimated subspaces is not allowed to drop, and the parameters estimates behave quite smoothly.

There is one important restriction. The averaging method is only applicable if the eigenvalue information itself is unimportant. Although subspace algorithms perform the mapping from subspace to signal parameters, based on the subspace estimate only, they need the eigenvalues to distinguish between noise and signal subspaces. Also hypothesis tests to determine the number of signals (rank of the data matrix) rely on eigenvalue information [147]. In order to be able to detect rank incrementing or decrementing, while still discarding the exact knowledge of all singular values, a four-level spherical subspace algorithm was introduced [21]. All signal singular values except for the smallest one, and all noise singular values except for the largest one, are averaged. By monitoring the change in these four levels, an informed decision on rank changes can be made.

Here the aim is not to compare the performance of the various subspace algorithms. We only give a small numerical example in Figure 4.1. Further experimental studies can be found in [22, 149, 150]. From now on we concentrate on numerical stability and architecture design.

## 4.2 Spherical SVD updating algorithm

The spherical subspace updating algorithm we consider, is the ROSA algorithm (Rank-one update, signal averaging) of DeGroat [20]. It is a fast approximate method to perform rank-one EVD updates of the sample correlation matrix $R_{x,[k]} = \alpha^2 R_{x,[k-1]} + x_{[k]} \cdot x_{[k]}^H$. Here, we derive the algorithm from a slightly different point of view, *i.e.*, as an approximate SVD updating algorithm, with averaged singular values instead of eigenvalues. When the eigenvalue spread of the sample correlation matrix is large, the SVD version may show improved numerical accuracy over the original EVD algorithm.

The spherical SVD updating algorithm tracks a different decomposition than the Jacobi SVD updating algorithm. At each time $t_k$, it approximates $X_{[k]}^H$ by the unitary decomposition

$$X_{[k]}^H \approx \tilde{X}_{[k]}^H = U_{[k]} \cdot D_{[k]} \cdot V_{[k]}^H$$

where $U_{[k]} \in \mathbb{C}^{k \times M}$ and $V_{[k]} \in \mathbb{C}^{M \times M}$ are unitary matrices given by the

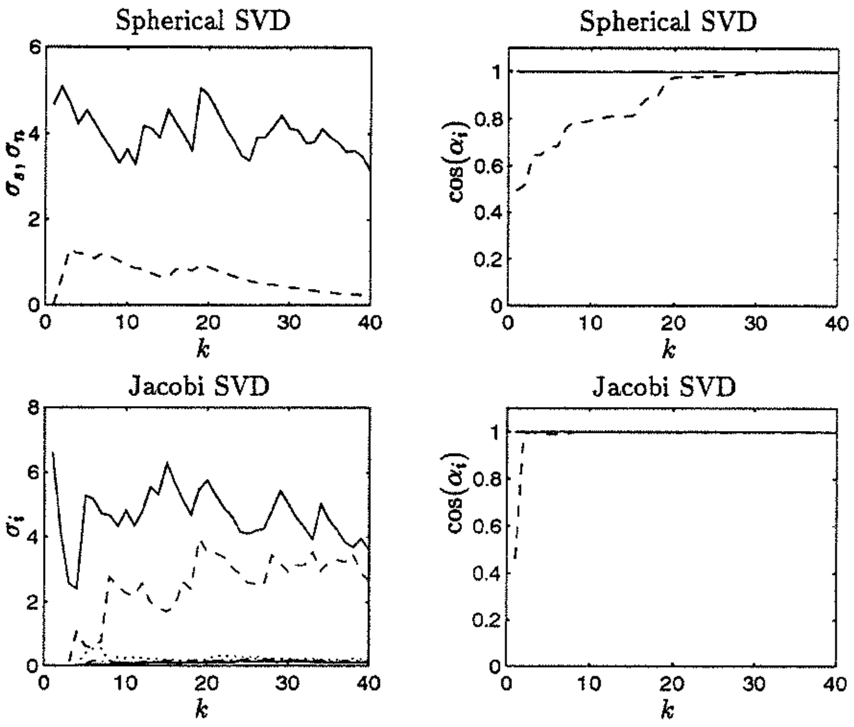


Figure 4.1: Comparison of the convergence of the spherical and Jacobi SVD updating algorithms for a stationary scenario. The example is a real matrix with $M = 5$ and $N = 40$ and singular spectrum $\{14.14, 8.77, 0.58, 0.49, 0.45\}$. Both algorithms use a forgetting factor $\alpha = 0.9$. The left part shows the evolution of the average singular levels (top) and the estimated singular values (bottom). The right part shows the cosines of the two canonical angles between the true and estimated dominant subspace. The Jacobi algorithm converges in approximately $M$ iterations, whereas the convergence of the spherical SVD algorithm is slower and depends on the decay of initial errors.

SVD of $X_{[k]}^H$ and $D_{[k]} \in \mathbb{R}^{M\times M}$ is a block-identity matrix

$$D_{[k]} = \left[\begin{array}{c|c} \sigma_{[k]}^s I_D & O \\ \hline O & \sigma_{[k]}^n I_{M-D} \end{array}\right].$$

Both the signal- and the noise subspaces are characterized by a single singular level $\sigma_{[k]}^s, \sigma_{[k]}^n$. The Frobenius norm $\|X_{[k]} - \tilde{X}_{[k]}\|$ is minimized if the singular levels $\sigma_{[k]}^s, \sigma_{[k]}^n$ are chosen as the root mean square of the two groups of singular values. The decomposition is no longer exact. This is an important difference in comparison with the Jacobi SVD updating algorithm. There, an upper triangular matrix $R_{[k]}$ is used as an approximant to the singular value matrix $\Sigma_{[k]}$. The product of the decomposition, however, equals $X_{[k]}$ at all time. The sphericalized subspace algorithm reduces the approximant to an extremely simple form, *i.e.*, a block-identity matrix with two parameters. The error in the decomposition, due to one averaging step, is small only when the spread on noise and signal singular values is low.

New data vectors are now recursively appended to the weighted spherical decomposition

$$\tilde{X}_{[k]} = \left[\begin{array}{c} \alpha\tilde{X}_{[k-1]} \\ \hline x_{[k]}^H \end{array}\right] = \left[\begin{array}{c|c} U_{[k-1]} & O \\ \hline O & 1 \end{array}\right] \cdot \left[\begin{array}{c} \alpha D_{[k-1]} \\ \hline x_{[k]}^H \cdot V_{[k-1]} \end{array}\right] \cdot V_{[k-1]}^H.$$

Just as in the Jacobi SVD updating algorithm, the first step is the matrix vector multiplication

$$\tilde{x}_{[k]}^H = x_{[k]}^H \cdot V_{[k-1]}.$$

Let $\left[\begin{array}{c|c} V_{[k-1]}^s & V_{[k-1]}^n \end{array}\right]$ denote the partitioning of $V_{[k-1]}$ into its first $D$ columns (signal subspace) and last $M-D$ columns (noise subspace), and let accordingly

$$\left[\begin{array}{c|c} \tilde{x}_{[k]}^{s^H} & \tilde{x}_{[k]}^{n^H} \end{array}\right] = x_{[k]}^H \cdot \left[\begin{array}{c|c} V_{[k-1]}^s & V_{[k-1]}^n \end{array}\right].$$

The vector $\tilde{x}_{[k]}$ has to be worked into the weighted block-identity matrix $\alpha D_{[k-1]}$. The Jacobi SVD updating algorithm used a QRD updating operation. Here, the block identity structure of $D_{[k-1]}$ can be exploited to simplify the computation. The first $D-1$ (last $M-D-1$) components of $\tilde{x}_{[k]}^s$ ($\tilde{x}_{[k]}^n$) can be zeroed by a sequence of $D-1$ ($M-D-1$) Givens rotations

$$\begin{aligned} \left[\begin{array}{cc} 0\cdots 0 & \gamma_{[k]}^s \end{array}\right] &= \tilde{x}_{[k]}^{s^H} \cdot \left(\Phi_{[k]}^{1|2} \cdots \Phi_{[k]}^{D-1|D}\right) = \tilde{x}_{[k]}^{s^H} \cdot \Phi_{[k]}^s \\ \left[\begin{array}{cc} \gamma_{[k]}^n & 0\cdots 0 \end{array}\right] &= \tilde{x}_{[k]}^{n^H} \cdot \left(\Phi_{[k]}^{M-1|M} \cdots \Phi_{[k]}^{D+1|D+2}\right) = \tilde{x}_{[k]}^{n^H} \cdot \Phi_{[k]}^n. \end{aligned}$$

Example

$$\underbrace{\begin{bmatrix} \times & \times & \times \end{bmatrix}}_{\bar{x}^{s^H}} \xrightarrow{\Phi^{1|2}} \begin{bmatrix} 0 & \times & \times \end{bmatrix} \xrightarrow{\Phi^{D-1|D}} \begin{bmatrix} 0 & 0 & \times \end{bmatrix}$$

$$\underbrace{\begin{bmatrix} \times & \times & \times \end{bmatrix}}_{\bar{x}^{n^H}} \xrightarrow{\Phi^{M-1|M}} \begin{bmatrix} \times & \times & 0 \end{bmatrix} \xrightarrow{\Phi^{D+1|D+2}} \begin{bmatrix} \times & 0 & 0 \end{bmatrix}$$

Because of its block-identity structure, $D_{[k-1]}$ commutes with $\Phi^s_{[k]}$ and $\Phi^n_{[k]}$. Therefore the update can be written as

$$\begin{aligned} \tilde{X}_{[k]} &= \left[\begin{array}{c|c} U_{[k-1]} & O \\ \hline O & 1 \end{array}\right] \cdot \left[\begin{array}{c|c} \Phi^n_{[k]} \cdot \Phi^s_{[k]} & O \\ \hline O & 1 \end{array}\right] \cdots \\ & \cdots \left[\begin{array}{cccc} \multicolumn{4}{c}{\alpha D_{[k-1]}} \\ \hline 0\cdots0 & \gamma^s_{[k]} & \gamma^n_{[k]} & 0\cdots0 \end{array}\right] \cdot \Phi^{s^H}_{[k]} \cdot \Phi^{n^H}_{[k]} \cdot V^H_{[k-1]}. \end{aligned}$$

By the commutation, we avoid to apply the column rotations $\Phi^s_{[k]}$ and $\Phi^n_{[k]}$ to $D_{[k-1]}$, which would destroy its sparse structure. Instead they act on the columns of $U_{[k-1]}$. Since the growing matrix $U_{[k]}$ is not stored, this part of the computation disappears.

As in the Jacobi SVD updating algorithm, the column rotations have to be applied to $V_{[k-1]}$. They align the last signal (first noise) singular vector along the projection of $x_{[k]}$ onto the signal (noise) subspace at time $t_{k-1}$. These alignments are made possible due to the fact that in a spherical subspace any orthogonal basis is an admissible set of singular vectors. All signal power added by $x_{[k]}$, is captured by the two aligned singular vectors and only the corresponding singular values are altered. The adjustment of the 'border' between signal- and noise subspaces takes place in this 2-dimensional space. The update in this space is a $3 \times 2$ SVD updating problem, which can be solved as a $3 \times 2$ QRD updating operation

$$\begin{bmatrix} \alpha\, \sigma^s_{[k-1]} & 0 \\ 0 & \alpha\, \sigma^n_{[k-1]} \\ \gamma^s_{[k]} & \gamma^n_{[k]} \end{bmatrix} \xrightarrow{G^{1|3^H}_{[k]}} \begin{bmatrix} \times & \times \\ 0 & \alpha\, \sigma^n_{[k-1]} \\ 0 & \times \end{bmatrix} \xrightarrow{G^{2|3^H}_{[k]}} \begin{bmatrix} \times & \times \\ 0 & \times \\ 0 & 0 \end{bmatrix}$$

followed by a triangular $2 \times 2$ SVD

$$\begin{bmatrix} \times & \times \\ 0 & \times \end{bmatrix} \xrightarrow{\Theta^{sn^H}_{[k]}, \Phi^{sn}_{[k]}} \begin{bmatrix} \bar{\sigma}^s_{[k]} & 0 \\ 0 & \bar{\sigma}^n_{[k]} \end{bmatrix}.$$

In the $2 \times 2$ SVD computation, care should be taken that the largest singular value is put first. Finally, the new singular values are obtained by reaveraging in mean square sense

$$\begin{aligned}
\sigma^{s}_{[k]} &= \sqrt{\frac{\tilde{\sigma}^{s^2}_{[k]} + (D-1)\,\alpha^2\,\sigma^{s^2}_{[k-1]}}{D}} \\
\sigma^{n}_{[k]} &= \sqrt{\frac{\tilde{\sigma}^{n^2}_{[k]} + (M-D-1)\,\alpha^2\,\sigma^{n^2}_{[k-1]}}{M-D}}.
\end{aligned}$$

Comments

1. The rotation $\Phi^{sn}$ can be computed from the following symmetric eigenvalue problem

$$\begin{bmatrix} \tilde{\sigma}^{s^2}_{[k]} & 0 \\ 0 & \tilde{\sigma}^{n^2}_{[k]} \end{bmatrix} = \Phi^{sn^H}_{[k]} \cdot \begin{bmatrix} \alpha^2\,\sigma^{s^2}_{[k-1]} + |\gamma^{s}_{[k]}|^2 & \gamma^{s^H}_{[k]}\,\gamma^{n}_{[k]} \\ \gamma^{n^H}_{[k]}\,\gamma^{s}_{[k]} & \alpha^2\,\sigma^{n^2}_{[k-1]} + |\gamma^{n}_{[k]}|^2 \end{bmatrix} \cdot \Phi^{sn}_{[k]}.$$

   This can be checked immediately by squaring the $3 \times 2$ SVD updating problem above. This formulation is part of the original ROSA EVD updating algorithm. However, to compute the entries of this symmetric EVD problem the processor must be able to perform multiplications. Therefore, on a CORDIC-based architecture the implementation as a small QRD updating and SVD problem is preferred.

2. Reaveraging the singular values requires a multiplication, a sum, a division and a square root.

$$c = \frac{\sqrt{b^2 + (K-1)\,a^2}}{\sqrt{K}}$$

   where $K \in \mathbb{N}$ and $a, b, c \in \mathbb{R}$. This computation can be replaced by (sequences of) Givens rotations. The numerator can be obtained by a reduction of the vector $v \in \mathbb{R}^K$,

$$\left[\, b \mid a \cdots a \,\right] \rightarrow \left[\, \times \mid 0 \cdots 0 \,\right].$$

   For an arbitrary vector $v$, $K-1$ rotations would be needed. Here, since almost all entries are equal, this number can be reduced to roughly $\mathcal{O}(\log_2 K)$ by using intermediate results, *e.g.*, computing $\sqrt{10}a$ can be done with 4 instead of 9 rotations.

$$\begin{array}{rcl}
\begin{bmatrix} a & a \end{bmatrix} & \rightarrow & \begin{bmatrix} \sqrt{2}a & 0 \end{bmatrix} \\
\begin{bmatrix} \sqrt{2}a & a \end{bmatrix} & \rightarrow & \begin{bmatrix} \sqrt{3}a & 0 \end{bmatrix} \\
\begin{bmatrix} \sqrt{2}a & \sqrt{3}a \end{bmatrix} & \rightarrow & \begin{bmatrix} \sqrt{5}a & 0 \end{bmatrix} \\
\begin{bmatrix} \sqrt{5}a & \sqrt{5}a \end{bmatrix} & \rightarrow & \begin{bmatrix} \sqrt{10}a & 0 \end{bmatrix}
\end{array}$$

The division by $\sqrt{K}$ requires a single rotation over an angle $\alpha = \tan^{-1}(\sqrt{K-1})$.

$$\begin{bmatrix} a & 0 \end{bmatrix} \xrightarrow{\alpha} \begin{bmatrix} \sqrt{\frac{1}{K}}a & \sqrt{\frac{K-1}{K}}a \end{bmatrix}$$

The spherical SVD updating algorithm is clearly non-iterative. Its complexity is $\mathcal{O}(M^2)$, since $M-1$ Givens rotations are applied to the columns of $V_{[k-1]}$. A basis is tracked for the signal subspace as well as for the noise subspace. In order to minimize the computation and memory, operations involving the larger subspace should be eliminated.

The sole noise-related information needed for the update of the signal subspace is the projection of $x_{[k]}$ onto the noise subspace at time $t_{k-1}$

$$x^n_{[k]} = V^n_{[k-1]} \cdot V^{n^H}_{[k-1]} \cdot x_{[k]}.$$

Because $V^s_{[k-1]}$ and $V^n_{[k-1]}$ are each other's orthogonal complement, the vector $x^n_{[k]}$ can also be computed as

$$x^n_{[k]} = (I_M - V^s_{[k-1]} \cdot V^{s^H}_{[k-1]}) \cdot x_{[k]}.$$

The aligned noise singular value $v^n_{[k]}$ is then obtained by normalization.

$$\begin{array}{rcl}
\gamma^n_{[k]} & = & \|x^n_{[k]}\| \\
v^n_{[k]} & = & \dfrac{x^n_{[k]}}{\gamma^n_{[k]}}
\end{array}$$

The update of the signal subspace now only involves $D+1$ singular vectors and simplifies to

$$\begin{bmatrix} V^s_{[k]} \mid \star \end{bmatrix} = \begin{bmatrix} V^s_{[k-1]} \mid v^n_{[k]} \end{bmatrix} \cdot \left[\begin{array}{c|c} \Phi^s_{[k]} & O \\ \hline O & 1 \end{array}\right] \cdot \left[\begin{array}{c|c} I_{D-1} & O \\ \hline O & \Phi^{sn}_{[k]} \end{array}\right]$$

where $\star$ denotes a quantity of no importance.

The final spherical SVD updating algorithm is given in Algorithm 4. Its computation per update and storage requirements are $\mathcal{O}(M \cdot D)$.

**Algorithm 4**

```
$V^s_{[0]} \leftarrow \left[\begin{array}{c} I_D \\ \hline O_{(M-D)\times D} \end{array}\right]$
$\sigma^s_{[0]}, \sigma^n_{[0]} \leftarrow 0$
for $k = 1, \cdots, \infty$
  1. orthogonal matrix-vector multiplication
       $\tilde{x}^{s^H}_{[k]} \leftarrow x^H_{[k]} \cdot V^s_{[k-1]}$
  2. noise singular vector
       $x^n_{[k]} \leftarrow x_{[k]} - V^s_{[k-1]} \cdot \tilde{x}^s_{[k]}$
       $\gamma^n_{[k]} \leftarrow \sqrt{x^{n^H}_{[k]} \cdot x^n_{[k]}}$
       $v^n_{[k]} \leftarrow \frac{x^n_{[k]}}{\gamma^n_{[k]}}$
  3. column rotations
       $\left[\begin{array}{c|c} 0 \cdots 0 & \gamma^s_{[k]} \end{array}\right] \leftarrow \tilde{x}^{s^H}_{[k]} \cdot \prod_{i=1}^{D-1} \Phi^{i|i+1}_{[k]}$
       $\begin{bmatrix} \tilde{\sigma}^s_{[k]} & 0 \\ 0 & \tilde{\sigma}^n_{[k]} \\ 0 & 0 \end{bmatrix} \leftarrow \Theta^{1|2^H}_{[k]} \cdot G^{1|2^H}_{[k]} \cdot G^{1|3^H}_{[k]} \begin{bmatrix} \alpha\sigma^s_{[k-1]} & 0 \\ 0 & \alpha\sigma^n_{[k-1]} \\ \gamma^s_{[k]} & \gamma^n_{[k]} \end{bmatrix} \cdot \Phi^{D|D+1}_{[k]}$
       $\left[\begin{array}{c|c} V^s_{[k]} & \star \end{array}\right] \leftarrow \left[\begin{array}{c|c} V^s_{[k-1]} & v^n_{[k]} \end{array}\right] \cdot \prod_{i=1}^{D} \Phi^{i|i+1}_{[k]}$
  4. updating the singular values
       $\left[\begin{array}{c|c} \hat{\sigma}^s_{[k]} & 0 \cdots 0 \end{array}\right] \leftarrow \left[\begin{array}{c|c} \tilde{\sigma}^s_{[k]} & \alpha\sigma^s_{[k-1]} \cdots \alpha\sigma^s_{[k-1]} \end{array}\right] \cdot \Psi^{D-1|D}_{s,[k]} \cdots \Psi^{1|2}_{s,[k]}$
       $\left[\begin{array}{c|c} \hat{\sigma}^n_{[k]} & 0 \cdots 0 \end{array}\right] \leftarrow \left[\begin{array}{c|c} \tilde{\sigma}^n_{[k]} & \alpha\sigma^n_{[k-1]} \cdots \alpha\sigma^n_{[k-1]} \end{array}\right] \cdot \Psi^{M-D-1|M-D}_{n,[k]} \cdots \Psi^{1|2}_{n,[k]}$
       $\left[\begin{array}{c|c} \sigma^s_{[k]} & \sqrt{D-1}\,\sigma^s_{[k]} \end{array}\right] \leftarrow \left[\begin{array}{c|c} \hat{\sigma}^s_{[k]} & 0 \end{array}\right] \cdot \Psi_{s,[k]}$
       $\left[\begin{array}{c|c} \sigma^n_{[k]} & \sqrt{M-D-1}\,\sigma^n_{[k]} \end{array}\right] \leftarrow \left[\begin{array}{c|c} \hat{\sigma}^n_{[k]} & 0 \end{array}\right] \cdot \Psi_{n,[k]}$
endfor
```

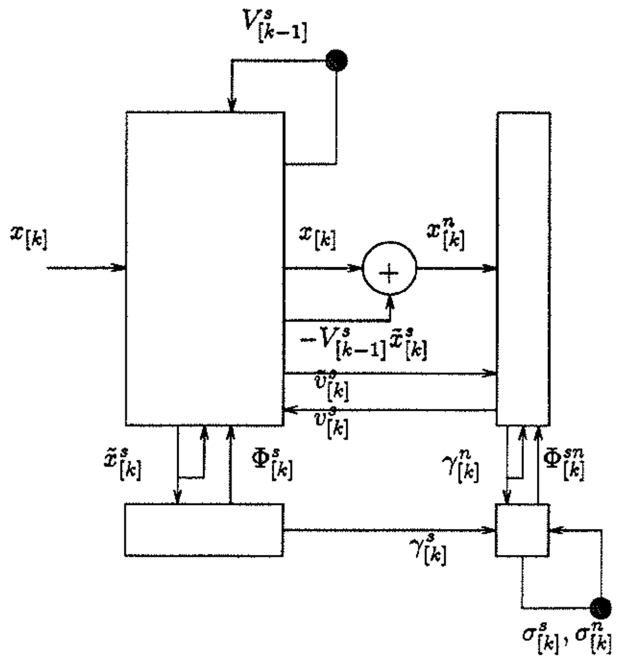


Figure 4.2: High level SFG for the spherical SVD updating algorithm. The upper part only stores a submatrix of a unitary matrix. The normalized projection on its orthogonal complement of the incoming data vector is computed in the right hand side. The updating rotations are computed in the lower part.

A high-level SFG of Algorithm 4 is shown in Figure 4.2. Similar to the SVD Jacobi array, the top-left rectangular operator performs the matrix-vector multiplication $\tilde{x}^s_{[k]}$ and the update rotations $V^s_{[k-1]} \cdot \Phi^s_{[k]}$. In addition, a second matrix-vector multiplication $V^s_{[k-1]} \cdot \tilde{x}^s_{[k]}$ has to be produced at the right-hand side. Therefore $\tilde{x}_{[k]}$ has to be entered upwards into the operator again. This creates a long dependency in the computation of length $\mathcal{O}(2M)$. The same holds for the top-right operator. First the norm $\gamma^n_{[k]}$ is computed from top to bottom. Then, in order to normalize $x^n_{[k]}$ to $v^n_{[k]}$, $\gamma^n_{[k]}$ is propagated upwards again. Also the last column of the $V^s_{[k]}$, denoted $v^s_{[k]}$, is circulated to the noise column and back such that the update $\Phi^{sn}_{[k]}$ can be performed. The inner-products of the second matrix-vector multiplication complicate the pipelining of this SFG into an efficient systolic array.

## 4.3 Factored spherical SVD updating

Algorithm 4 has two major disadvantages. The first disadvantage is its complicated data flow. The second is the deviation from orthogonality of the subspace basis $V_{[k-1]}$. Algorithm 4 uses the same principle as the Jacobi SVD updating algorithm to adapt the matrix $V_{[k]}$, *i.e.*, Givens rotations are continuously applied to its columns. Therefore, error buildup due to finite precision calculations again looms ahead.

The factorization of chapter 3 parameterizing full orthogonal matrices $V \in \mathbb{R}^{M\times M}$ as a chain of $\frac{M(M-1)}{2}$ Givens rotations, can solve both problems. Here, we are only interested in a factorization of the matrix $V^s_{[k-1]} \in \mathbb{C}^{M\times D}$. Since the factorization of Lemma 1 was constructed column wise, it suffices to truncate the Givens QRD method to column $D$

$$V^s_{[k]} = \prod_{i=1}^{D} \prod_{j=i+1}^{M} Q^{i|j}_{[k]}.$$

In the triangular SFG for the matrix-vector multiplication (Figure 3.3), only the left-most $D-1$ columns of rotation nodes are retained. Their operation does not alter.

The generation of the updating rotations $\Phi^{i|i+1}_{[k]}$, $i = 1, \cdots, D$ is easier here than in the case of the Jacobi SVD updating algorithm. No triangular QRD updating array is required. A single row of $D-1$ rotation nodes in angle accumulation mode is placed below the rotation array. These nodes take in the vector $\tilde{x}^s_{[k]}$ and gradually zero it from left to right. Meanwhile, they compute the rotations $\Phi^{i|i+1}_{[k]}$ which are propagated upwards into the rotation array. These $\Phi^{i|i+1}$-rotations are worked into the $Q^{i|j}$-rotations using the type 2 (summing angles) and type 3 (reordering 3 rotations) transformations.

The only difficulty left is the computation of the angles which characterize the vector $v^n_{[k]}$ and the norm $\gamma^n_{[k]}$. To this end, the high-level SFG of Figure 4.3 is instrumental. It shows the spherical SVD updating algorithm in factored form. The factorization of $V_{[k-1]}$ is partitioned into three parts. The left operator stores $Q^s_{[k-1]} = \prod_{i=1}^{D}\prod_{j=i+1}^{M} Q^{i|j}_{[k-1]} \in \mathbb{C}^{M\times M}$, which is the unitary operator composed of the factorization of $V^s_{[k-1]}$. The unitary operator $Q^{sn}_{[k-1]} = \prod_{j=D+2}^{M} Q^{D+1|j} \in \mathbb{C}^{(M-D)\times(M-D)}$ is constructed with the rotations which are stored in the $D+1$st column of the SFG of Figure 3.3. The right lower triangular operator stores the operator, consisting of the remaining rotations in the factorization of $V_{[k-1]}$,

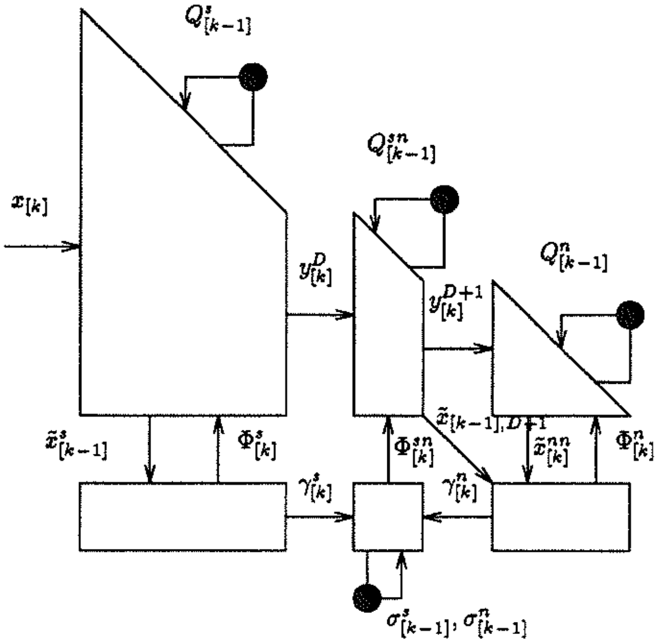


Figure 4.3: High level SFG for the factored spherical SVD updating algorithm. The factorization of the unitary tracking matrix is partitioned in three parts.

*i.e.,* $Q^n_{[k-1]} = \prod_{i=D+2}^{M-1} \prod_{j=i+1}^{M} Q^{i|j}_{[k-1]} \in \mathbb{C}^{(M-D-1)\times(M-D-1)}$. Here, both the signal- and noise subspaces are tracked.

In the non-factored spherical subspace algorithm, computation and memory needs are cut by avoiding storage of $V^n_{[k]}$ in the algorithm. Here, $Q^n_{[k]}$ must be eliminated. Consider this right operator in detail. First, it takes in the vector $y^{D+1}_{[k]} \in \mathbb{C}^{M-D-1}$ from the left, and produces the orthogonal matrix-vector product $\tilde{x}^{nn^H}_{[k]} = y^{D+1^H}_{[k]} \cdot Q^n_{[k-1]}$ at its bottom-side. The row of rotation nodes below then computes the updating rotations $\Phi^n_{[k]}$ such that the vector $\tilde{x}^{nn}_{[k]}$ comprising the last $M-D+1$ components of $\tilde{x}_{[k]}$, is completely zeroed. This means that after updating the complete factorization, the rotation array produces a vector with only two non-zero components at entries $D$ and $D+1$. This can only be true if the updated array $Q_{[k]}$ generates a zero-vector $y^{D+1}_{[k]}$. This follows from the fact that $Q^n_{[k]}$ is unitary, and thus preserves the norm of $y^{D+1}_{[k]}$.

The right lower triangular operator $Q^n_{[k]}$ can therefore be eliminated by forcing the nodes in column $D$ to generate zero components at their right outputs. This is easily accomplished by operating these rotation nodes in angle accumulation mode, instead of vector rotation mode.

Algorithm 4 describes the final spherical SVD updating algorithm. It is oriented towards a CORDIC processor since it is comprised of rotations only (except for scalings by the weighting $\alpha$).

**Algorithm 5**

$$\begin{array}{rcll} Q^{i|j}_{[0]} & \leftarrow & I_2 & \text{for } 1 \le i \le D; i+1 \le j \le M \\ \sigma^s_{[0]}, \sigma^n_{[0]} & \leftarrow & 0 & \end{array}$$

```
for $k = 1, \cdots, \infty$
 1. orthogonal matrix-vector multiplication
```

$$\left[\ \tilde{x}^{s^H}_{[k]} \,\middle|\, y^{D^H}_{[k]}\ \right] \leftarrow x^H_{[k]} \cdot \left( \prod_{i=1}^{D} \prod_{j=i+1}^{M} Q^{i|j}_{[k-1]} \right)$$

```
 2. noise singular vector rotations
```

$$\left[\ \gamma^n_{[k]} \,\middle|\, 0 \cdots 0\ \right] \leftarrow y^{D^H}_{[k]} \cdot Q^{D+1|D+2}_{[k]} \cdots Q^{D+1|M}_{[k]}$$

```
 3. column rotations
```

$$\left[\ 0 \cdots 0 \,\middle|\, \gamma^s_{[k]}\ \right] \leftarrow \tilde{x}^{s^H}_{[k]} \cdot \prod_{i=1}^{D-1} \Phi^{i|i+1}_{[k]}$$

$$\begin{bmatrix} \tilde{\sigma}^s_{[k]} & 0 \\ 0 & \tilde{\sigma}^n_{[k]} \\ 0 & 0 \end{bmatrix} \leftarrow \Theta^{1|2^H}_{[k]} \cdot G^{1|2^H}_{[k]} \cdot G^{1|3^H}_{[k]} \begin{bmatrix} \alpha\sigma^s_{[k-1]} & 0 \\ 0 & \alpha\sigma^n_{[k-1]} \\ \gamma^s_{[k]} & \gamma^n_{[k]} \end{bmatrix} \cdot \Phi^{D|D+1}_{[k]}$$

```
    for $j \leftarrow 2$ to $M$
      $Q^{1|j}_* \leftarrow Q^{1|j}_{[k-1]}$
    endfor
    for $i \leftarrow 1$ to $D$
      $\Phi^{i|i+1}_* \leftarrow \Phi^{i|i+1}_{[k-1]}$
      for $j \leftarrow M$ downto $i+2$
        $\Phi^{i|i+1}_* \cdot Q^{i|j}_{[k]} \cdot Q^{i+1|j}_* \leftarrow Q^{i|j}_* \cdot Q^{i+1|j}_{[k-1]} \cdot \Phi^{i|i+1}_*$
      endfor
      $Q^{i|i+1}_{[k]} \leftarrow Q^{i|i+1}_* \cdot \Phi^{i|i+1}_*$
```

```
        endfor
    4. updating the singular values
$\left[\ \hat{\sigma}^s_{[k]}\ \middle|\ 0\cdots 0\ \right] \leftarrow \left[\ \tilde{\sigma}^s_{[k]}\ \middle|\ \alpha\sigma^s_{[k-1]}\cdots\alpha\sigma^s_{[k-1]}\ \right]\cdot\Psi^{D-1|D}_{s,[k]}\cdots\Psi^{1|2}_{s,[k]}$
$\left[\ \hat{\sigma}^n_{[k]}\ \middle|\ 0\cdots 0\ \right] \leftarrow \left[\ \tilde{\sigma}^n_{[k]}\ \middle|\ \alpha\sigma^n_{[k-1]}\cdots\alpha\sigma^n_{[k-1]}\ \right]\cdot\Psi^{M-D-1|M-D}_{n,[k]}\cdots\Psi^{1|2}_{n,[k]}$

            $\left[\ \sigma^s_{[k]}\ \middle|\ \sqrt{D-1}\,\sigma^s_{[k]}\ \right] \leftarrow \left[\ \hat{\sigma}^s_{[k]}\ \middle|\ 0\ \right]\cdot\Psi_{s,[k]}$
        $\left[\ \sigma^n_{[k]}\ \middle|\ \sqrt{M-D-1}\,\sigma^n_{[k]}\ \right] \leftarrow \left[\ \hat{\sigma}^n_{[k]}\ \middle|\ 0\ \right]\cdot\Psi_{n,[k]}$
  endfor
```

## 4.4 A linear array

The SFG of Algorithm 5 is shown in Figure 4.4. The node descriptions are given in Figure 4.5. This SFG looks very much the same as the SFG of Jacobi SVD updating. This is the natural consequence of the fact that the structure and functionality of both algorithms are very similar. Therefore, when pursuing a systolic array implementation, one is confronted with the same problem of a long vertical dependency path. However, in some applications the throughput specifications may already be met by mapping the algorithm onto a linear array. We first discuss this simple case. The more complicated derivation of a planar systolic array is postponed to the next section.

The mapping is derived starting from the SFG of Figure 4.4. A natural way to assign computation to processors, is to map all nodes in one column to a single processor (placement vector $p$ in Figure 4.4). The linear array then consists of $D+1$ processors.

Because of the bidirectional vertical data flow, the scheduling (*i.e.*, allocation of the computation over time) can not be done with a single scheduling vector. Instead, two scheduling vectors are necessary, resulting in a piece-wise linear schedule.

The orthogonal matrix-vector product and the annihilation of the components has dependencies from top to bottom and from left to right. All dependencies are satisfied if the first scheduling vector ($s_1$ in Figure 4.4) points from the top-left to the bottom-right corner of the SFG. The rotation updating process has dependencies pointing upwards and to the right. Therefore the second scheduling vector ($s_2$ in Figure 4.4) should point from the bottom-left corner to the diagonal.

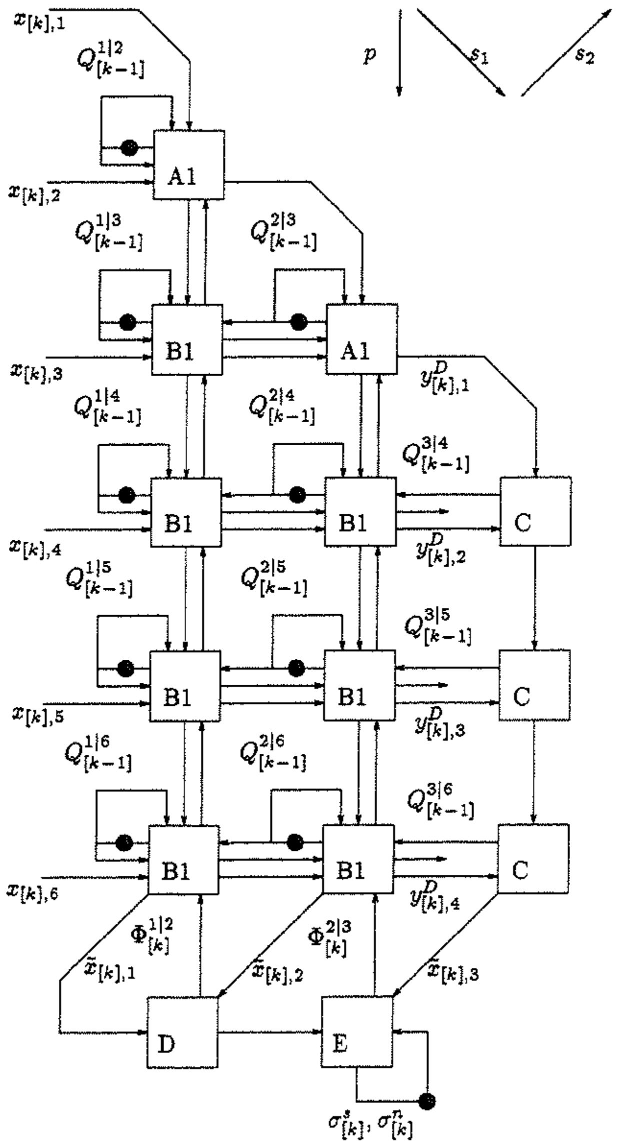
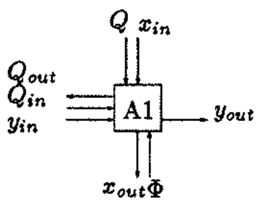
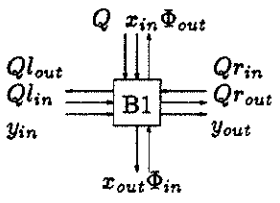


Figure 4.4: Detailed SFG for the spherical SVD updating algorithm ($M = 6, D = 2$). The description of the rotation nodes is given in the next figure. The vectors $p$ and $s_1, s_2$ are the placement and scheduling vectors for a linear systolic array.



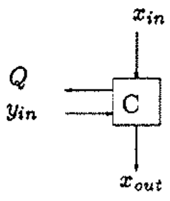
$$\begin{bmatrix} x_{out} \\ y_{out} \end{bmatrix} = Q^H \cdot \begin{bmatrix} x_{in} \\ y_{in} \end{bmatrix}$$

$$Q_{out} = Q_{in} \cdot \Phi$$

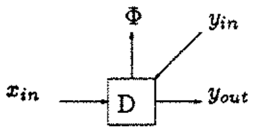


$$\begin{bmatrix} x_{out} \\ y_{out} \end{bmatrix} = Q^H \cdot \begin{bmatrix} x_{in} \\ y_{in} \end{bmatrix}$$

$$\Phi_{out} \cdot Ql_{out} \cdot Qr_{out} = Ql_{in} \cdot Qr_{in} \cdot \Phi_{in}$$

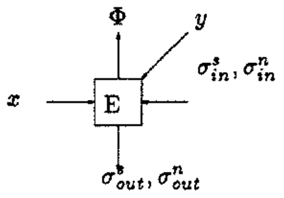


$$\begin{bmatrix} x_{out} \\ 0 \end{bmatrix} = Q^H \cdot \begin{bmatrix} x_{in} \\ y_{in} \end{bmatrix}$$



$$\begin{bmatrix} 0 \\ y_{out} \end{bmatrix} = \Phi^H \cdot \begin{bmatrix} x_{in} \\ y_{in} \end{bmatrix}$$

$$\Phi \cdot \begin{bmatrix} \sigma^{s^2} & 0 \\ 0 & \sigma^{n^2} \end{bmatrix} \cdot \Phi^H = \begin{bmatrix} \alpha^2 \sigma^{s^2}_{in} + |x|^2 & xy \\ xy & \alpha^2 \sigma^{n^2}_{in} + |y|^2 \end{bmatrix}$$


$\Phi$
$y$
$\sigma^s_{in}, \sigma^n_{in}$
$x$
E
$\sigma^s_{out}, \sigma^n_{out}$

$$\sigma^s_{out} = \sqrt{\frac{(D-1)\alpha^2\sigma^{s^2}_{in} + \sigma^{s^2}}{D}}$$

$$\sigma^n_{out} = \sqrt{\frac{(M-D-1)\alpha^2\sigma^{n^2}_{in} + \sigma^{n^2}}{M-D}}$$

Figure 4.5: Node descriptions for the spherical SVD updating algorithm.

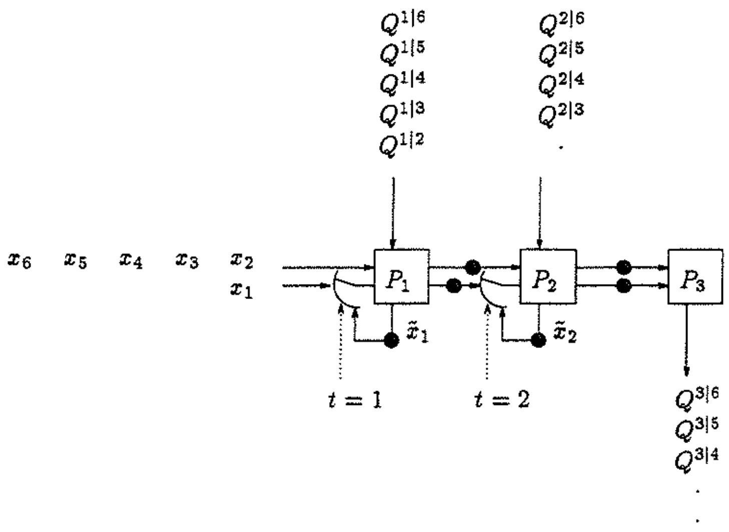


Figure 4.6: Phase 1 of the linear systolic array ($M = 6, D = 2$): orthogonal matrix-vector multiplication. The $t = i$ expressions control the switches.

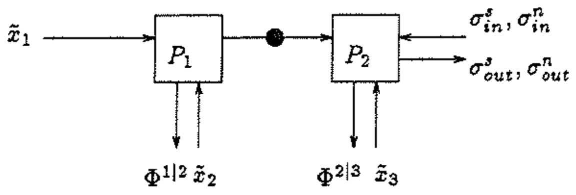

$\tilde{x}_1$
$P_1$
$P_2$
$\sigma^s_{in}, \sigma^n_{in}$
$\sigma^s_{out}, \sigma^n_{out}$
$\Phi^{1|2}\ \tilde{x}_2$
$\Phi^{2|3}\ \tilde{x}_3$

Figure 4.7: Phase 2 of the linear systolic array: generation of the column rotations.

In a first phase the linear array computes the orthogonal matrix-vector product and the rotations in column $D + 1$. The timing is indicated in Figure 4.6. The first component $\tilde{x}_1$ becomes available after cycle $M - 1$. The initialization of the $\tilde{x}_i$s requires some control of the inputs. During the second phase the column rotations $\Phi^{i|i+1}$ are computed (Figure 4.7). Finally, during the last phase the factorization is updated (Figure 4.8). The pipelining period of the linear array is $(M - 1) + 1 + (M - 1) = 2M - 1 = \mathcal{O}(M^{-1})$ cycles.

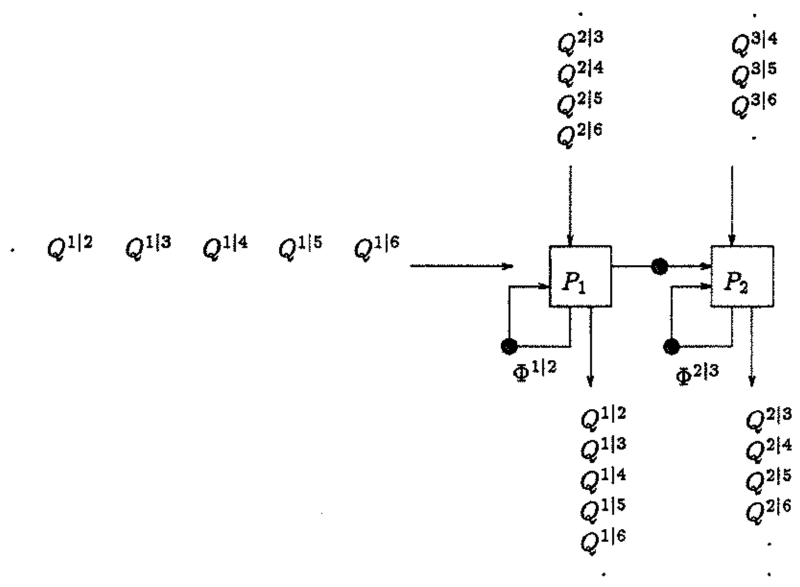


Figure 4.8: Phase 3 of the linear systolic array: application of the column rotations.

In [28] a similar linear array for the original spherical subspace ROSA algorithm is derived starting from the textual algorithmic description. The derivation is tough to follow, and it is difficult to see that all steps are correct. This example gives a flavor of the power and the clarity of the graphical methodology to map and schedule an algorithm onto various parallel architectures.

## 4.5 A planar array

In this section we consider the conversion of the SFG of Figure 4.4 into a more parallel SFG in which the long dependency paths have disappeared. This will again involve the use of algorithmic transformations similar to the Jacobi SVD updating algorithm (Algorithm 3). However, it turns out that additional corrections are needed at the right-hand border due to the truncation of the array. Therefore, the derivation is a bit harder and the resulting planar array is more complicated and less efficient than the Jacobi SVD updating array.

The SFG of Figure 4.4 specifies two processes. The first is the matrix-vector multiplication $\left[\, \tilde{x}^{s^H}_{[k]} \mid y^{D^H}_{[k]} \,\right] = x^H_{[k]} \cdot \left(\prod_{i=1}^{D} \prod_{j=i+1}^{M} Q^{i|j}_{[k-1]}\right)$. The second process is the updating of the rotations $Q^{i|j}_{[k-1]}$. Both processes are tightly linked, since the $\Phi^{i|i+1}$- and $Q^{D+1|j}$-rotations which are produced by annihilation of the matrix-vector multiplication, are instantaneously consumed by the updating process. Consider the upper-left block in Figure 4.9 bordered by the dashed line. Its input-output behavior should not change with respect to Figure 4.4. However, to break the long vertical data dependency loop, it is necessary to insert delays on the incoming $\Phi$- and $Q$-arcs. This causes the state of the upper-left block (*i.e.*, the set of stored rotation angles) to lag one cycle behind. One can easily correct for this delay by first updating the rotations from $Q^{i|j}_{[k-2]}$ to $Q^{i|j}_{[k-1]}$, such that the new vector $x_{[k]}$ is rotated correctly. The new node descriptions are given in Figure 4.10.

The transformations in the subsequent layers are again more involved. Although the array does not store the matrix $\hat{V}_{[k]} = \left[\, V^s_{[k]} \mid v^n_{[k]} \,\right] \in \mathbb{C}^{M\times(D+1)}$ but its factorization, the notation of the latter is elaborate and complicates the exposition. Therefore, we derive the algorithmic transformations using the explicit matrix notation $\hat{V}_{[k]}$. Delaying the $\Phi$-arcs pointing upwards and $Q$-arcs pointing to the right, increases the lag of the state of the upper-left block to 2 cycles (Figure 4.11). Without the additional delays, the $M-2$nd row of rotation cells outputs the matrix-vector multiplication

$$\tilde{x}^H_{[k]} = \underline{x}^H_{[k]} \cdot \bar{V}_{[k-1]}$$

where $\bar{V}_{[k]}$ is the product of the factorization of $\hat{V}_{[k]}$, without the rotations involving row $M$. Due to the inserted delays, there is a lag in the matrix-vector product

$$\eta^H_{[k]} = \underline{x}^H_{[k]} \cdot \bar{V}_{[k-2]}.$$

At the next cycle, $\bar{V}_{[k-2]}$ is updated by the rotations $\bar{\Phi}_{[k-1]}$ coming from the last layer

$$\bar{V}_{[k-1]} = \left[\, \bar{V}^s_{[k-1]} \mid \star \,\right] = \left[\, \bar{V}^s_{[k-2]} \mid \underline{v}^n_{[k-2,k-1]} \,\right] \cdot \bar{\Phi}_{[k-1]}$$

where the vector $v^n_{[k-1,k]}$ is the unit vector proportional to the projection of $x_{[k]}$ onto the noise subspace $V^n_{[k-1]}$.

$$v^n_{[k-1,k]} \propto V^n_{[k-1]} \cdot V^{n^H}_{[k-1]} \cdot x_{[k]}$$

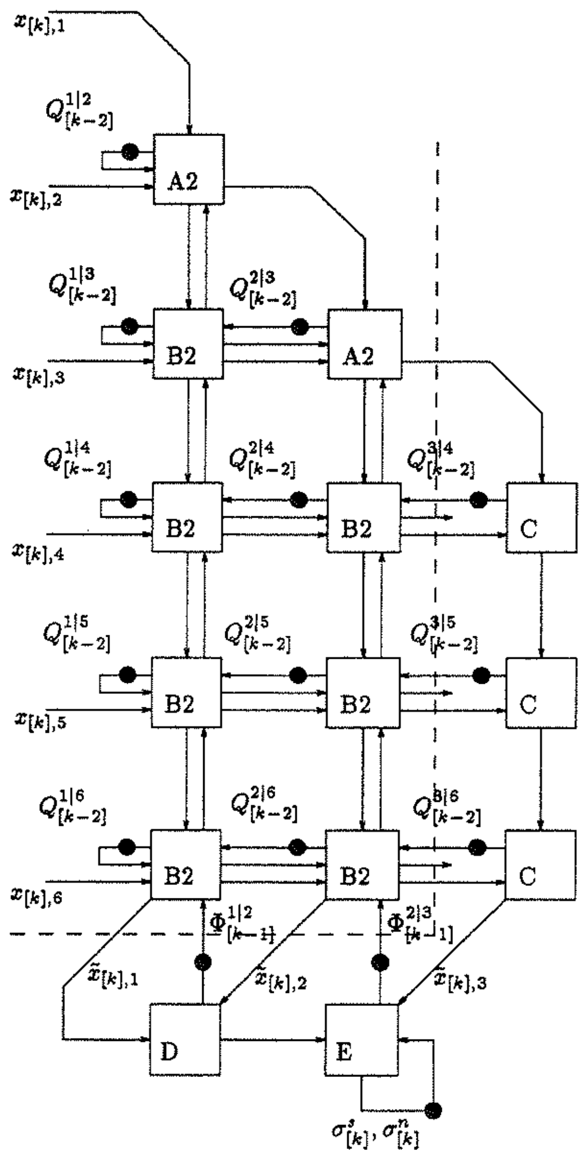
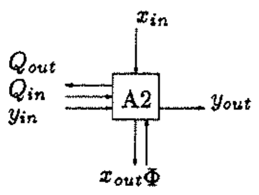
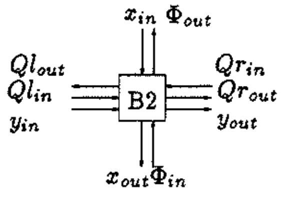


Figure 4.9: First phase of the mapping from SFG to planar array: insertion of a first set of delays. The new node descriptions are given in the next figure.



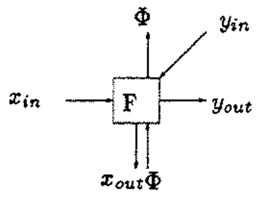
$$Q_{out} = Q_{in} \cdot \Phi$$

$$\begin{bmatrix} x_{out} \\ y_{out} \end{bmatrix} = Q_{out}^H \cdot \begin{bmatrix} x_{in} \\ y_{in} \end{bmatrix}$$

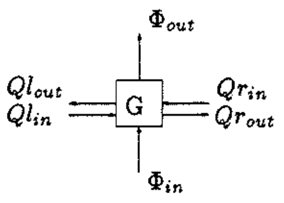


$$\Phi_{out} \cdot Ql_{out} \cdot Qr_{out} = Ql_{in} \cdot Qr_{in} \cdot \Phi_{in}$$

$$\begin{bmatrix} x_{out} \\ y_{out} \end{bmatrix} = Ql_{out}^H \cdot \begin{bmatrix} x_{in} \\ y_{in} \end{bmatrix}$$



$$\begin{bmatrix} x_{out} \\ y_{out} \end{bmatrix} = \Phi^H \cdot \begin{bmatrix} x_{in} \\ y_{in} \end{bmatrix}$$



$$\Phi_{out} \cdot Ql_{out} \cdot Qr_{out} = Ql_{in} \cdot Qr_{in} \cdot \Phi_{in}$$

Figure 4.10: Node descriptions of transformed nodes and correction nodes.

It suffices to apply the 'missing' rotations $\bar{\Phi}_{[k-1]}$ to $\eta_{[k]}^H$ to correct for the delay

$$\bar{x}_{[k]}^H = \eta_{[k]}^H \cdot \bar{\Phi}_{[k-1]}$$

such that the last layer receives the input as specified in Algorithm 5. In Figure 4.11 this operation is done in the extra nodes labeled $F$.

Corrections are also needed on the left-pointing $Q$-arcs. The input to the $C$-nodes are also components of $\eta_{[k]}$, such that the nodes compute (the factorization) of a vector $v_{[k-2,k]}^n$ which is proportional to the projection of $x_{[k]}$ onto the noise subspace $V_{[k-2]}^n$ at time $t_{k-2}$. This is not the desired

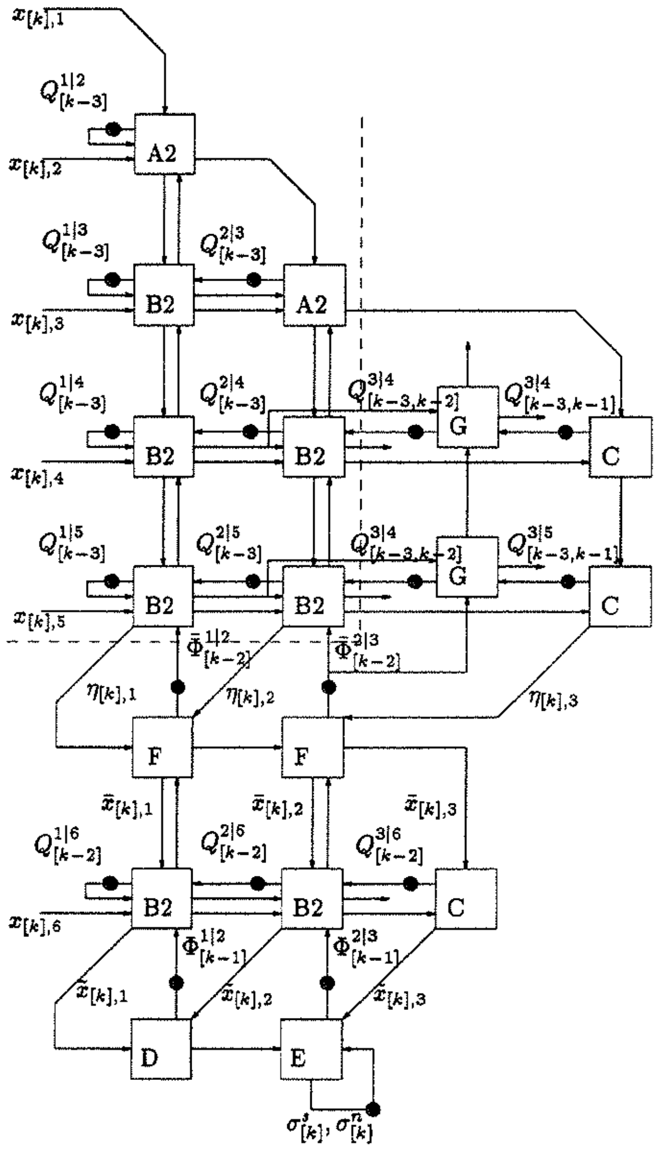


Figure 4.11: SFG after insertion of the second set of delays. Additional rotations have to be inserted to algorithmically correct for the lags.

behavior prescribed by the algorithm. Instead, the projection onto the noise subspace at time $t_{k-1}$ is needed. Therefore one should not overlook to update (the factorization of) $v^n_{[k-2,k]}$ to $v^n_{[k]} = v^n_{[k-1,k]}$. The nature of the update can be derived as follows. Without delays the SFG computes vectors $\tilde{x}_{[k-2,k]} \in \mathbb{C}^{D+1}$, $v^n_{[k-2,k]} \in \mathbb{C}^M$ such that

$$x_{[k]} = \left[ \begin{array}{cc|c} V^s_{[k-2]} & v^n_{[k-2,k]} & \star \end{array} \right] \cdot \left[ \begin{array}{c} \tilde{x}_{[k-2,k]} \\ \hline O_{(M-D-1)\times 1} \end{array} \right].$$

The signal subspace estimate $V^s_{[k-2]}$ is updated from time $t_{k-2}$ to $t_{k-1}$ by the column rotation matrix $\Phi_{[k-1]}$. This column operation affects the first $D+1$ columns of the $V$ matrix and leaves all remaining ones unaltered. The new decomposition is

$$\begin{aligned} x_{[k]} &= \left[ \begin{array}{cc|c} V^s_{[k-2]} & v^n_{[k-2,k]} & \star \end{array} \right] \cdot \Phi_{[k-1]} \cdot \Phi^H_{[k-1]} \cdot \left[ \begin{array}{c} \tilde{x}_{[k-2,k]} \\ \hline O_{(M-D-1)\times 1} \end{array} \right] \\ &= \left[ \begin{array}{cc|c} V^s_{[k-1]} & g_{[k]} & \star \end{array} \right] \cdot \left[ \begin{array}{c} \Phi^H_{[k-1]}\tilde{x}_{[k-2,k]} \\ \hline O_{(M-D-1)\times 1} \end{array} \right]. \end{aligned}$$

From this last equation it follows that the vector $g_{[k]}$ is aligned according to the projection of $x_{[k]}$ onto the noise subspace at time $t_{k-1}$, such that

$$v^n_{[k]} = v^n_{[k-1,k]} = g_{[k]}.$$

Above we already discussed the updating of the components of $\tilde{x}_{[k-2,k]}$ in the $F$-nodes. Similarly the updating of the factorization of $v^n_{[k-2,k]}$ is taken care of in the additional $G$-nodes. For clarity, these $G$-nodes have been drawn at the right of the $B2$-nodes in Figure 4.11. The drawback of this placement is that the regularity of the SFG is lost. It is better to think of the SFG in 3 dimensions (Figure 4.15). The $G$-nodes are then lying in a second parallel plane behind the $B2$-nodes with which they share inputs. When mapping these $G$- and $B2$-nodes onto each other and merging the $F$- and the underlying $B2$-nodes, Figure 4.12 is obtained.

Repeating the algorithmic transformations in the remaining layers finally results in the SFG of Figure 4.13. The long data dependency loops have disappeared from the SFG. All dependencies are local and regular. If bidirectional data flow occurs, a delay is present in one of the directions. Therefore the algorithmically transformed SFG can easily be turned into a systolic array. First the delays have to be tripled (time-scaling rule) and then the delay-transfer rule is applied to cut sets parallel

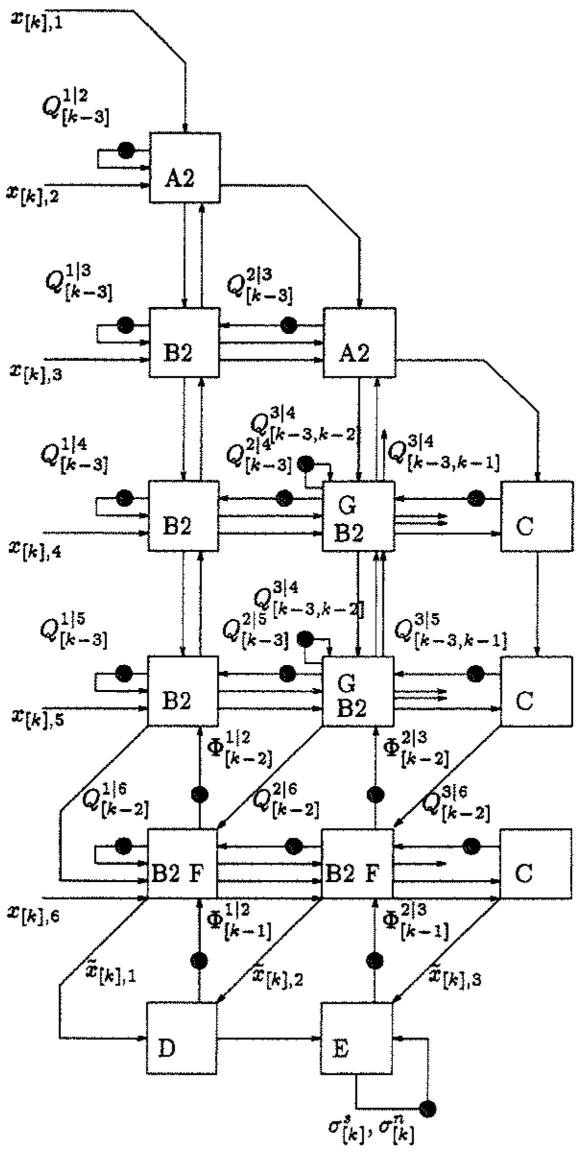


Figure 4.12: SFG after insertion of the second set of delays. Regular and correction nodes are merged.

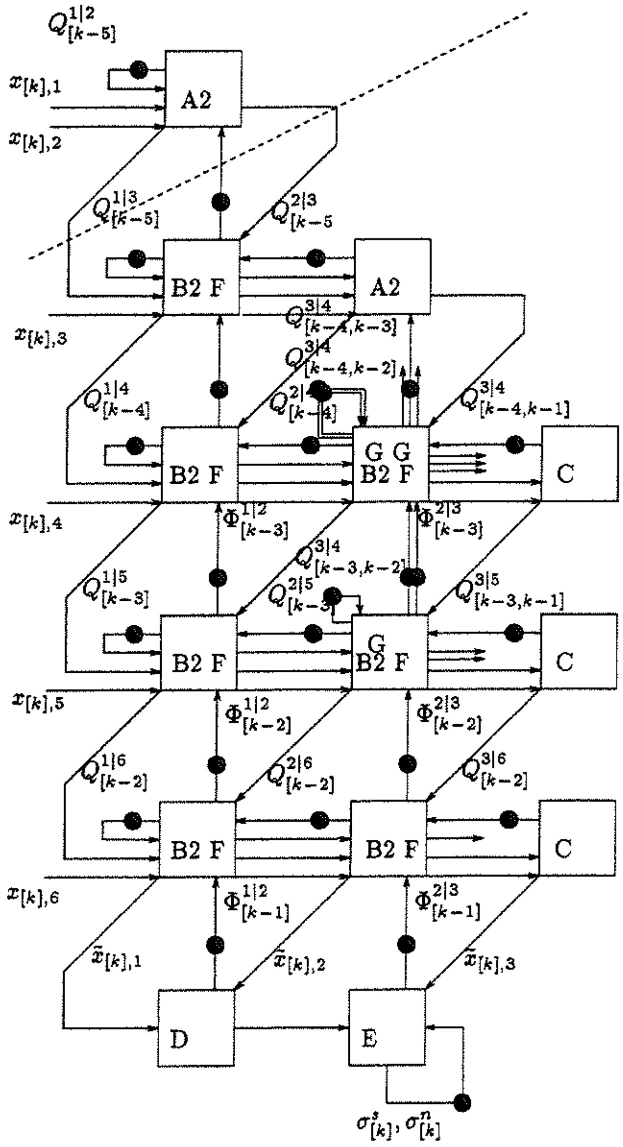


Figure 4.13: SFG after insertion of the third set of delays.

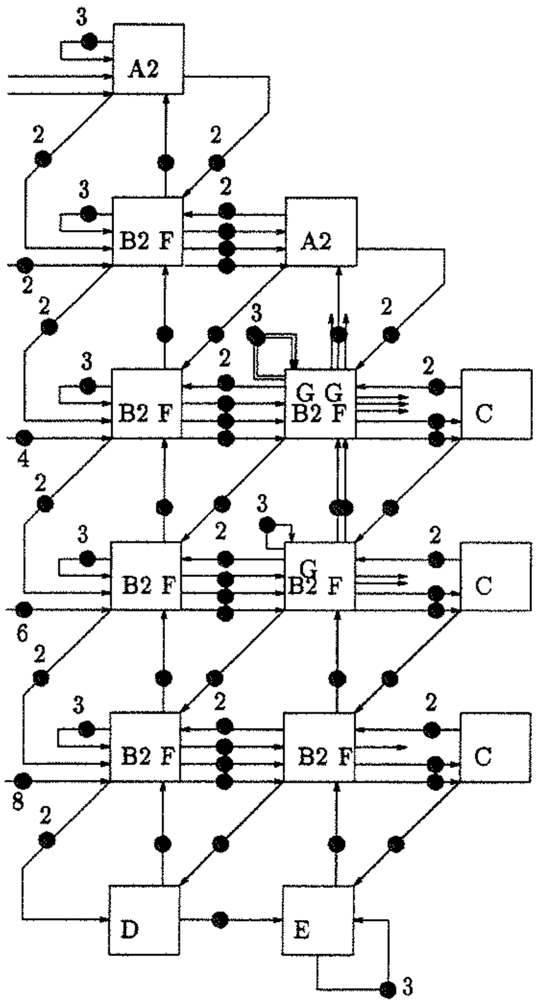


Figure 4.14: Planar systolic architecture for spherical SVD updating

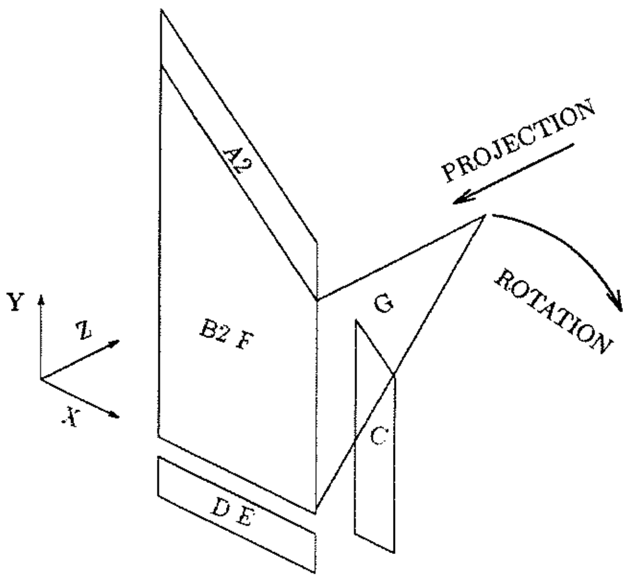


Figure 4.15: Relative position of the nodes in a three-dimensional SFG. To obtain a two-dimensional array, the triangle of $G$-nodes can be projected orthogonally onto the $XY$-plane (regular connections but load imbalance), or rotated in between the $B2F$-polygon and the $C$-polygon (load balance but broadcast).

to the dashed line in Figure 4.13. The final systolic array is depicted in Figure 4.14.

Of course, the regularity in the interconnections is due to the orthogonal projection of all $G$-nodes onto one hierarchical node. The trade-off of this mapping is the linear increase of the load of the nodes in column $D$. This increase is caused by the growing gap between generation and consumption of the $Q^{D+1|j}$-rotations. Since a systolic array is synchronized by a central clock, the operation speed of the full array is governed by the slowest node. If the problem size $M$ is large, the throughput of the array can be slowed down to $\mathcal{O}(1/M)$. The alternative is not to project all $G$-nodes onto the same column. Instead, the triangle of $G$-nodes can be rotated (oblique projection) from the $YZ$-plane into the $XY$-plane (Figure 4.15). The $G$-nodes are then placed in between the nodes of column $D$ and $D+1$. The load balancing between the nodes of this array is better, and the clock speed is no longer dependent on $M$. The drawback of this second mapping is the irregularity and non-locality in the interconnection pattern. It requires broadcast links of some data from one node to several others. The length of the broadcast is proportional to $M$. An optimal choice will depend on technological constraints.

## 4.6 Conclusion

In this chapter we have studied the spherical subspace tracking algorithm in which both the signal as well as the noise singular values are averaged. First a spherical SVD updating algorithm was derived. It differs mainly from the original EVD updating algorithm, in that it can be implemented using rotations only.

To cure the build-up of numerical errors, which could destabilize the recursive algorithm, we proposed to track the factorization of the unitary subspace tracking matrix. The spherical subspace tracking algorithm is a second example of the usefulness of the factorization, which was developed in the previous chapter. Other examples will follow. We conjecture that a large class of adaptive algorithms which update orthogonal (unitary) matrices by sequences of Givens rotations might benefit from the factorization.

Next, a linear systolic array was derived. It has the advantage that no extra computation is involved. It is also an efficient array, since with $D + 1$ processors its pipelining period is $2M - 1$ cycles.

Finally, we developed a 3-dimensional systolic architecture in which one triangle of nodes is orthogonal to the plane in which all other nodes are contained. This array is impractical to construct. Two planar arrays were obtained by linear projections. The orthogonal projection leads to a systolic array with local connections but suffering from a linearly increasing load imbalance. The oblique projection yields an array with a good load balance, but non-local interconnections. Several intermediate solutions may be derived as well. With $\mathcal{O}(M \cdot D)$ processors, the systolic array attains a $\mathcal{O}(1)$ throughput.

Chapter 5

# Robust Adaptive LCMV Beamforming

In section 2.2 it was shown that the beamforming problem can be formulated as a linearly constrained recursive least-squares problem. Typically one or more constraints correspond to array response vectors, fixing the gain in certain look directions. It is common to impose a unity gain in the direction of the signal of interest (SOI). As a second example one may force a zero gain in the direction of a known interference source. In practice, it is hard to perfectly know these array response vectors. Inaccuracies in the positions of the sensors, random gain-and-phase errors or variations in the direction of the SOI are almost unavoidable. The effect of these errors should not be overlooked. Even small errors in the nominal array response vector used in the constraints, can cause already a severe decrease in the signal-to-interference-plus-noise ratio (SINR) at the output of the beamformer [41, 144].

We propose to solve the sensitivity problem by estimating the array response vector directly from the data. A necessary condition for the applicability of the method is that a reference signal is available. An example is the GSM system in which each normal burst contains in the middle a training sequence of 26 bits known to the receiver. The advantages of this approach are clear. Pointing and calibration errors can be eliminated, and the beamformer is able to track the signal-of-interest. A complication is that now a recursive least squares (RLS) problem with a time-varying constraint has to be solved. Several parallel architectures are known for adaptive beamforming [59, 69, 146, 152]. Our method of choice is the generalized sidelobe canceler (GSC) discussed in chapter 2, because it is best suited to adaptation of the constraint. Moreover, the algorithm can be formulated as a Jacobi algorithm.

In section 5.1 the sensitivity of the output SINR is explained. Next, section 5.2 proposes a cure based on estimation of the array response vector from the data. Section 5.3 demonstrates the increased robustness by means of simulations. Finally, in section 5.4 the mapping from algorithm onto architecture is discussed. Throughout the chapter the notation of section 2.2 is used.

## 5.1 Signal leakage

To examine the effects of a small mismatch in the array response vector, let $a(\theta_s) \in \mathbb{C}^M$ denote the actual (but unknown) array response vector and assume $c = a(\theta_s) + \delta a(\theta_s) \in \mathbb{C}^M$ is the nominal array response vector used in the constraint matrix. Thus, the vector $\delta a(\theta_s)$ contains the (small) deviations between actual and nominal array response vector.

In the ideal case where $c = a(\theta_s)$ is correct, no SOI components are contained in the so-called reference inputs $\tilde{x}^H_{[k]} = x^H_{[k]} \cdot B$. The SOI only appears in the primary signal $\tilde{y}_{[k]} = x^H_{[k]} \cdot w_q$. This desired behavior is achieved by the orthogonality conditions $B^H \cdot c = B^H \cdot a(\theta_s) = O_{(M-K)\times 1}$ imposed on the choice of a suitable blocking matrix $B$. However, in case of a small mismatch, some signal leakage into the reference inputs will occur. Let $s_{[k]}$ denote the SOI sample at time $t_k$. The signal leakage

$$B^H \cdot a(\theta_s) \cdot s_{[k]} = -B^H \cdot \delta a(\theta_s) \cdot s_{[k]}$$

causes a decrease of the output SINR. In the example of Figure 5.1 the choice of $c = a(\theta_s) + \delta a(\theta_s)$, $B$ and $w_q$ is such that there is only a leakage $\epsilon$ into the first reference input

$$\begin{aligned} a(\theta_s)^H \cdot B &= \epsilon \cdot \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \\ w_q &= \frac{1}{\|c\|^2} \cdot c. \end{aligned}$$

The fixed quiescent filter $w_q$ results in a SINR of 17.1 dB in the primary channel $\tilde{y}$. For each value of the leakage coefficient $\epsilon$ the SINR at the output is computed using the optimal weight vector. The decrease of the output SINR is pronounced. A small leakage of about 1.5% ($\epsilon = 0.015$) in one reference channel, already lowers the SINR to the level of the

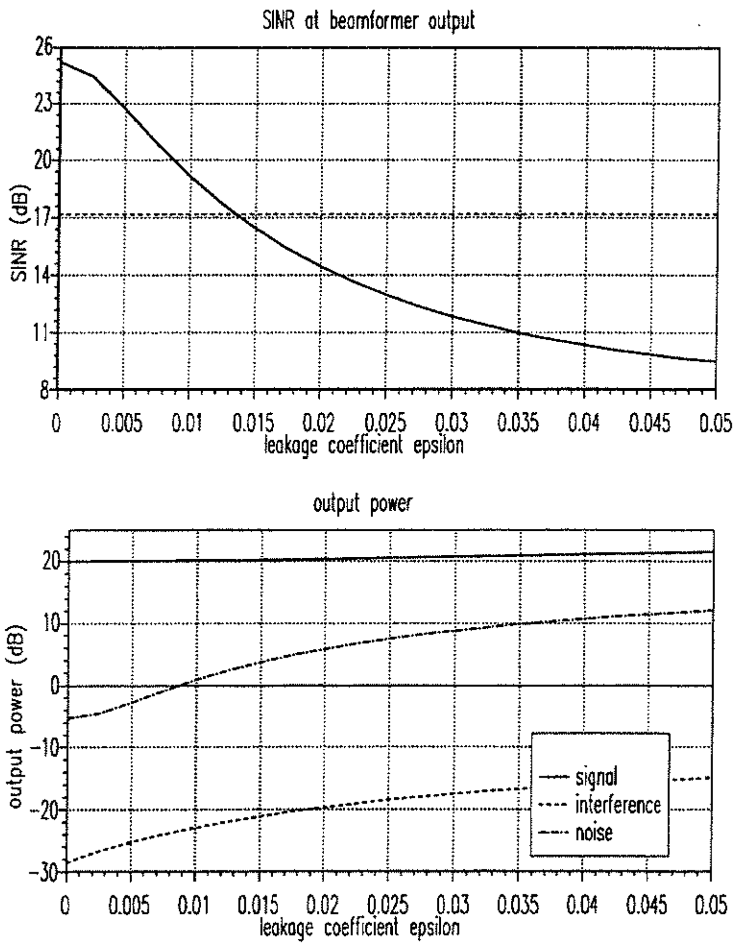


Figure 5.1: The effects of signal leakage: a 4-element half-wavelength ULA receives a signal (0 deg, 20 dB) and an interference (20 deg, 10 dB). A signal leakage of $\epsilon$ in one reference input causes a rapid decrease in the output SINR (upper part). The lower part shows that this is due to an increase of the output noise level. The interference level remains 25 dB below the noise. Simulations were performed in Xmath.

fixed filtered primary channel $\tilde{y}$. When even more leakage is present, the adaptive filter deteriorates the SINR instead of improving it.

The lower half of the figure shows that this decrease in SINR is due to a same increase in the white noise level at the output. The interference signal is still suppressed to an insignificant level. In this example the signal power in the output remains almost constant. In other cases substantial signal cancelation may occur.

It should be noted that the decrease of the output SINR is depending on all parameters of the scene. Especially for large SINRs at the individual sensors, the decrease is pronounced. This counterintuitive observation is best explained in Figure 5.2. We assume a virtual array with two elements, in the sense that all vectors involved are real. The array response vector of the signal is $s^T = \begin{bmatrix} 0 & 1 \end{bmatrix}$. The SINR at the input is 50. The output power (Eq. (2.3)) is given by

$$p(w) = w^T \cdot \begin{bmatrix} 1 & 0 \\ 0 & 51 \end{bmatrix} \cdot w,$$

a few contour lines of which are plotted in the figure. For a given weight vector the output noise power is proportional to $||w||^2$, whereas the signal power is proportional to $|s^T \cdot w|^2$. Therefore the SINR at input and output are related by a factor $\cos^2(\alpha)$ where

$$\cos(\alpha) = \frac{|s^T \cdot w|}{||w||}.$$

Vector $w_A$ gives the optimal weight vector when the exact constraint is used. It is proportional to the array response vector $s$. The vector $w_B$ denotes the optimal weight vector when the constraint vector is erroneously taken to be $c^T = \begin{bmatrix} \cos(85^o) & \sin(85^o) \end{bmatrix}$. The misalignment between $s$ and $w_B$ increases as the SINR increases and the contour ellipses stretch even further. In this simple example and for large input SINRs $\cos(\alpha)$ is inversely proportional to the SINR. In [41] similar conclusions are derived for the general case.

## 5.2 An adjustable constraint

The underlying cause of the signal leakage is the unavoidable imperfect a priori knowledge of the array response vector. Similarly to the approach of Castedo *et al.* [10], we propose to alleviate this problem by estimating $a(\theta_s)$ directly from the data.

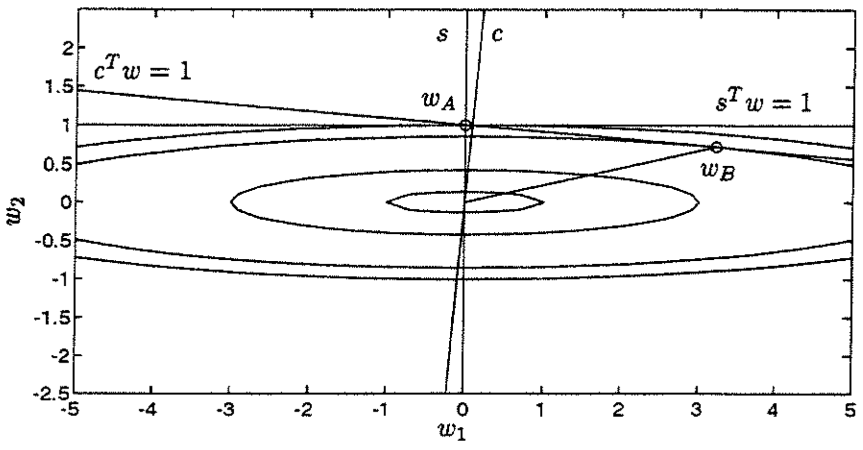


Figure 5.2: Virtual two element array, exact array response vector $s^T = \begin{bmatrix} 0 & 1 \end{bmatrix}$, perturbed constraint vector $c^T = \begin{bmatrix} \cos(85^o) & \sin(85^o) \end{bmatrix}$, signal power = 50, noise power = 1. The weight vectors $w_A$ and $w_B$ are the optimal weight vectors for the unperturbed and perturbed constraints respectively.

In many communications systems the source periodically sends a reference signal $r_{[k]}$ which is known to the receiver. In other applications the reconstructed waveform can act as a reference signal (*e.g.*, decision feedback operation of blind equalizers [82]). In both cases the actual array response vector $a(\theta_s)$ can be estimated by forming the cross-correlation vector between the sensor outputs and the reference signal

$$p = E\{x_{[k]} \cdot r_{[k]}^*\} = E\{a(\theta_s) \cdot s_{[k]} \cdot r_{[k]}^*\} + E\{n_{[k]} \cdot r_{[k]}^*\}.$$

If $r_{[k]}$ is correlated with $s_{[k]}$, but uncorrelated with all interferences and noise, then $p$ is proportional to $a(\theta_s)$, *i.e.*,

$$p = \rho_{sr} \cdot a(\theta_s).$$

The signal leakage can thus be counteracted by replacing the time invariant constraint

$$c^H \cdot w = \mu$$

where $c \approx a(\theta)$ is the steering constraint vector by the time variant constraint

$$p_{[k]}^H \cdot w = \mu \cdot \frac{p_{[k]}^H \cdot w_{q,fix}}{c^H \cdot w_{q,fix}} = \mu'_{[k]}$$

where the recursive estimate $p_{[k]}$ is defined as

$$p_{[k]} = \beta \cdot p_{[k-1]} + (1-\beta) \cdot x_{[k]} \cdot r^*_{[k]}$$

and $\beta$ is again an exponential forgetting factor. The right-hand side $\mu$ is rescaled to $\mu'_{[k]}$ by means of some linear function of the elements of $p_{[k]}$ and $c$ (both approximately parallel to $a(\theta_s)$). The vector $w_{q,fix}$ is yet to be defined. In order to develop a robust LCMV beamformer, below we combine the recursive estimation of the cross-correlation vector with the GSC algorithm.

The time-varying nature of the constraint matrix $C_{[k]}$ has two effects. First the quiescent weight pattern $w_{q,[k]}$ becomes time-varying, since it has to satisfy

$$\begin{bmatrix} C_{fix}^H \\ p_{[k]}^H \end{bmatrix} \cdot w_{q,[k]} = \begin{bmatrix} m_{fix} \\ \mu'_{[k]} \end{bmatrix}.$$

Here $C_{fix} \in \mathbb{C}^{M \times (K-1)}$ and $m_{fix} \in \mathbb{C}^{K-1}$ correspond to the fixed, non time-depending constraints. Secondly the blocking matrix $B_{[k]}$ should be kept orthogonal to $C_{[k]}$ and thus becomes time-varying as well.

First we focus on the time-dependence of the quiescent weight $w_{q,[k]}$. In the derivation below, it may be useful to keep already Figure 5.4 in mind.

An equivalent set of constraints in which the last constraint is orthogonalized with respect to the space spanned by the fixed constraints, is given by

$$\begin{bmatrix} C_{fix}^H \\ p_{[k]}^{\perp H} \end{bmatrix} \cdot w_{q,[k]} = \begin{bmatrix} m_{fix} \\ \mu_{[k]}^{\perp} \end{bmatrix}$$

where

$$\begin{aligned} p_{[k]}^{\perp} &= (I - C_{fix} \cdot (C_{fix}^H \cdot C_{fix})^{-1} \cdot C_{fix}^H) \cdot p_{[k]} \\ \mu_{[k]}^{\perp} &= \mu'_{[k]} - p_{[k]}^H \cdot (C_{fix} \cdot (C_{fix}^H \cdot C_{fix})^{-1} \cdot m_{fix}). \end{aligned}$$

For the quiescent weight vector we choose the minimum norm solution

$$w_{q,[k]} = \underbrace{C_{fix} \cdot (C_{fix}^H \cdot C_{fix})^{-1} \cdot m_{fix}}_{w_{q,fix}} + p_{[k]}^{\perp} \cdot (p_{[k]}^{\perp H} \cdot p_{[k]}^{\perp})^{-1} \cdot \mu_{[k]}^{\perp}.$$

Define the unit vector $q_{[k]}$ as

$$q_{[k]} = \frac{p^{\perp}_{[k]}}{\|p^{\perp}_{[k]}\|}.$$

The primary channel can then be written as

$$\begin{aligned}
\tilde{y}_{[k]} &= x^H_{[k]} \cdot w_{q,[k]} \\
&= \underbrace{x^H_{[k]} \cdot w_{q,fix}}_{\tilde{y}_{2,[k]}} + \underbrace{x^H_{[k]} \cdot q_{[k]}}_{\tilde{y}_{1,[k]}} \cdot \frac{\mu^{\perp}_{[k]}}{\|p^{\perp}_{[k]}\|} \\
&= \tilde{y}_{2,[k]} + \tilde{y}_{1,[k]} \cdot \frac{(\mu'_{[k]} - p^H_{[k]} \cdot w_{q,fix})}{\|p^{\perp}_{[k]}\|}.
\end{aligned}$$

The primary channel is seen to be a 'time-varying' linear combination of the 'time-varying' channel $\tilde{y}_1$ and the fixed channel $\tilde{y}_2$. The output of the beamformer is also given by the same linear combination of the outputs $e_1$ and $e_2$ of two separate beamformers, with $\tilde{y}_1$ and $\tilde{y}_2$ as primary inputs respectively

$$\begin{aligned}
y_{[k]} &= e_{2,[k]} + \frac{(\mu'_{[k]} - p^H_{[k]} \cdot w_{q,fix})}{\|p^{\perp}_{[k]}\|} \cdot e_{1,[k]} \\
&= e_{2,[k]} + \underbrace{(\frac{\mu}{c^H \cdot w_{q,fix}} - 1)}_{\mu''} \cdot \frac{p^H_{[k]} \cdot w_{q,fix}}{\|p^{\perp}_{[k]}\|} \cdot e_{1,[k]}.
\end{aligned}$$

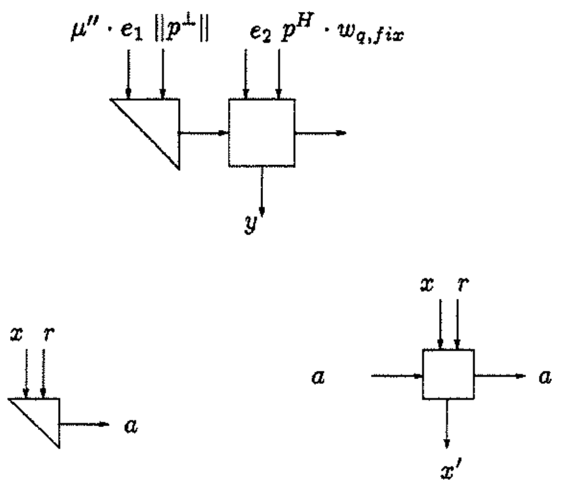
The output $y_{[k]}$ is the Schur complement of $\|p^{\perp}_{[k]}\|$ in the matrix $S_{[k]} \in \mathbb{C}^{2\times 2}$

$$S_{[k]} = \begin{bmatrix} \|p^{\perp}_{[k]}\| & p^H_{[k]} \cdot w_{q,fix} \\ -\mu'' e_{1,[k]} & e_{2,[k]} \end{bmatrix}.$$

Such a Schur complement can easily be computed by reducing the matrix $S_{[k]}$ to upper triangular form [151]. One possibility is to use Gauss transformations

$$\begin{bmatrix} \|p^{\perp}_{[k]}\| & p^H_{[k]} \cdot w_{q,fix} \\ 0 & y_{[k]} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -a_{[k]} & 1 \end{bmatrix} \cdot \begin{bmatrix} \|p^{\perp}_{[k]}\| & p^H_{[k]} \cdot w_{q,fix} \\ -\mu'' e_{1,[k]} & e_{2,[k]} \end{bmatrix}$$

where $a_{[k]} = \frac{-\mu'' e_{1,[k]}}{\|p^{\perp}_{[k]}\|}$. A parallel implementation of matrix triangulariza-



$$\begin{bmatrix} r \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -a & 1 \end{bmatrix} \cdot \begin{bmatrix} r \\ x \end{bmatrix} \qquad \begin{bmatrix} r \\ x' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -a & 1 \end{bmatrix} \cdot \begin{bmatrix} r \\ x \end{bmatrix}$$

Figure 5.3: A parallel $1 \times 2$ Schur complement array performing Gauss transformations. This array performs the linear combination of the two channels.

tion is the Schur complement array of Figure 5.3, which performs Gauss transformations instead of rotations. Alternatively, one may prefer rotations to perform the triangularization on condition that the output of the array is normalized by multiplication with the cosines of the rotation angles (see section 2.2).

A global view on the modified GSC signal flow graph (so far) is presented in Figure 5.4. The major modifications with respect to the SFG of the standard GSC (Figure 2.7) are the presence of two primary channels and the small additional post-processing section in the lower right corner.

The second challenge is updating $B_{[k]}$, $q_{[k]}$ and $S_{[k]}$ such that the orthogonality conditions

$$\begin{aligned} q_{[k]} &\propto (I - C_{fix} \cdot (C_{fix}^H \cdot C_{fix})^{-1} \cdot C_{fix}^H) \cdot p_{[k]} \\ B_{[k]}^H \cdot \begin{bmatrix} C_{fix} & q_{[k]} \end{bmatrix} &= O_{(M-K) \times K} \end{aligned}$$

are satisfied at each sampling time $t_k$. These equations imply that $\tilde{p}_{[k]}$,

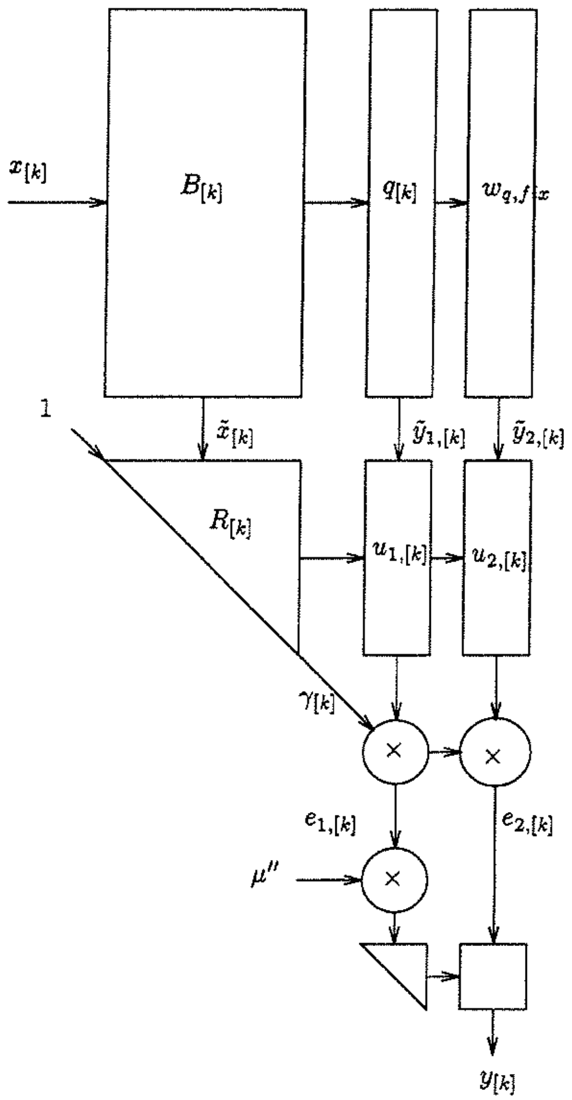


Figure 5.4: SFG of the generalized sidelobe canceler array with the time-varying primary channel included. In comparison with the fixed GSC of Figure 2.7, two weight vectors $w_{q,fix}$ and $q_{[k]}$ are present and a small output section is added.

the 'a posteriori' cross-correlation vector in 'beam space', should equal

$$\tilde{p}_{[k]}^H \overset{def}{=} p_{[k]}^H \cdot \left[ \begin{array}{c|c} B_{[k]} & q_{[k]} \end{array} \right] = \left[ \begin{array}{ccc|c} 0 & \cdots & 0 & * \end{array} \right].$$

The 'a priori' cross-correlation vector in 'beam space' $\bar{p}_{[k]}$ is computed as

$$\bar{p}_{[k]}^H = p_{[k]}^H \cdot \left[ \begin{array}{c|c} B_{[k-1]} & q_{[k-1]} \end{array} \right].$$

If its first $M-K$ components are not zero, they can be zeroed by applying a sequence of $M-K$ Givens rotations. The $i$th Givens rotation $V_{[k]}^{i|i+1}$ is computed to annihilate the $i$th component of $\bar{p}_{[k]}$ by combining it with the neighboring $i+1$st component. A 3-element example is given below.

Example:

$$\begin{bmatrix} \bar{p} \\ \bar{p} \\ \bar{p} \end{bmatrix} \overset{V^{1|2^H}}{\longrightarrow} \begin{bmatrix} 0 \\ \bar{p} \\ \bar{p} \end{bmatrix} \overset{V^{2|3^H}}{\longrightarrow} \begin{bmatrix} 0 \\ 0 \\ \tilde{p} \end{bmatrix}$$

The transformations are such that $\|p_{[k]}^{\perp}\| = \|\tilde{p}_{[k]}\|$ is given by the magnitude of the last component of $\tilde{p}_{[k]}$. Let $V_{[k]} = \prod_{i=1}^{M-K} V_{[k]}^{i|i+1}$ represent the sequence of $M-K$ column transformations, applied to $\bar{p}_{[k]}^H$. This leads to the following updating equation

$$\left[ \begin{array}{c|c} B_{[k]} & q_{[k]} \end{array} \right] = \left[ \begin{array}{c|c} B_{[k-1]} & q_{[k-1]} \end{array} \right] \cdot V_{[k]}.$$

The matrix $B_{[k-1]}$ and the vector $q_{[k-1]}$ are realigned in their own span. This implies that if they are properly initialized, they will always remain orthogonal to $C_{fix}$. Obviously the same column transformations have to be applied to the $R$-matrix. Let

$$X_{[k]} = \left[ \begin{array}{c|c|c|c} \alpha^{k-1} x_{[1]} & \alpha^{k-2} x_{[2]} & \cdots & x_{[k]} \end{array} \right]$$

denote the exponentially weighted data matrix. One can write the following QR decompositions

$$\begin{aligned} \text{a priori: } \bar{X}'_{[k]} &= X_{[k]}^H \cdot \left[ \begin{array}{c|c} B_{[k-1]} & q_{[k-1]} \end{array} \right] \\ &= \bar{Q}'_{[k]} \cdot \bar{R}'_{[k]} \\ \text{a posteriori: } \tilde{X}'_{[k]} &= X_{[k]}^H \cdot \left[ \begin{array}{c|c} B_{[k]} & q_{[k]} \end{array} \right] = \bar{X}'_{[k]} \cdot V_{[k]} \\ &= \bar{Q}'_{[k]} \cdot \bar{R}'_{[k]} \cdot V_{[k]}. \end{aligned}$$

Unfortunately a column transformation $V_{[k]}^{i|i+1}$ on $\bar{R}'_{[k]}$ creates a fill-in under the diagonal of $\bar{R}'_{[k]}$. An additional row rotation $U_{[k]}^{i|i+1}$ is required to restore the upper triangular structure of $\bar{R}'_{[k]}$. In the example below a column rotation $V^{1|2}$ is applied to the first 2 columns of the triangular matrix. This results in a fill-in $e$ at entry (2,1). The row rotation $U_{[k]}^{1|2^H}$ is then computed such that the fill-in $e$ is zeroed. This computation is repeated for subsequent fill-ins.

<u>Example: creation and annihilation of fill-ins</u>

$$\begin{bmatrix} r & r & r \\ & r & r \\ & & r \end{bmatrix} \overset{V^{1|2}}{\longrightarrow} \begin{bmatrix} r' & r' & r \\ e & r' & r \\ & & r \end{bmatrix} \overset{U^{1|2^H}}{\longrightarrow} \begin{bmatrix} r'' & r'' & r' \\ 0 & r'' & r' \\ & & r \end{bmatrix} \overset{V^{2|3}}{\longrightarrow}$$

$$\begin{bmatrix} r'' & r''' & r'' \\ 0 & r''' & r'' \\ & e & r' \end{bmatrix} \overset{U^{2|3^H}}{\longrightarrow} \begin{bmatrix} r'' & r'' & r' \\ 0 & r'''' & r''' \\ & 0 & r'' \end{bmatrix}$$

After the complete update the a posteriori QR decomposition is given as

$$\begin{aligned}
\tilde{X}_{[k]}^H &= X_{[k]}^H \cdot \left[ \begin{array}{cc|c} B_{[k]} & q_{[k]} & w_{q,fix} \end{array} \right] \\
&= \left[ \begin{array}{c|c} \bar{Q}'_{[k]} & \star \end{array} \right] \cdot \left[ \begin{array}{c|c} \bar{R}'_{[k]} & \bar{u}'_{f,[k]} \\ \hline O & \star \end{array} \right] \cdot \left[ \begin{array}{c|c} V_{[k]} & O \\ \hline O & 1 \end{array} \right] \\
&= \left[ \begin{array}{c|c} \bar{Q}'_{[k]} & \star \end{array} \right] \cdot \left[ \begin{array}{c|c} U_{[k]} & O \\ \hline O & 1 \end{array} \right] \cdot \left[ \begin{array}{c|c} U_{[k]}^H & O \\ \hline O & 1 \end{array} \right] \cdot \left[ \begin{array}{c|c} \bar{R}'_{[k]} \cdot V_{[k]} & \bar{u}'_{f,[k]} \\ \hline O & \star \end{array} \right] \\
&= \left[ \begin{array}{c|c} \bar{Q}'_{[k]} \cdot U_{[k]} & \star \end{array} \right] \cdot \left[ \begin{array}{c|c} U_{[k]}^H \cdot \bar{R}'_{[k]} \cdot V_{[k]} & U_{[k]}^H \cdot \bar{u}'_{f,[k]} \\ \hline O & \star \end{array} \right]
\end{aligned}$$

where the $O$s denote zero matrices of appropriate dimension. From these equations it is clear that for consistency, the row transformation $U_{[k]}^H$ should be applied to the vector $\bar{u}'_{f,[k]}$ too. The signal flow graph of the complete generalized sidelobe canceler with an adjustable constraint is shown in Figure 5.5.

The textual description is given in algorithm 6. The operator null$(\cdot)$ computes an orthogonal matrix spanning the kernel of its input argument. Due to the fact that $\tilde{p}_{[k-1]}^H = \left[ \begin{array}{ccc|c} 0 & \cdots & 0 & \star \end{array} \right]$, the sequence of column rotations $V_{[k]}^{i|i+1}$ computed from $\tilde{p}_{[k]} = \beta \cdot \tilde{p}_{[k-1]}^H + (1-\beta) \cdot r_{[k]} \cdot \tilde{x}_{[k]}^H$, already annihilates $\tilde{x}_{[k]}$ except for the last entry. Therefore the QRD updating reduces to a $1 \times 1$ problem, and the beamformer outputs $e_{1,[k]}$ and $e_{2,[k]}$ are readily available.

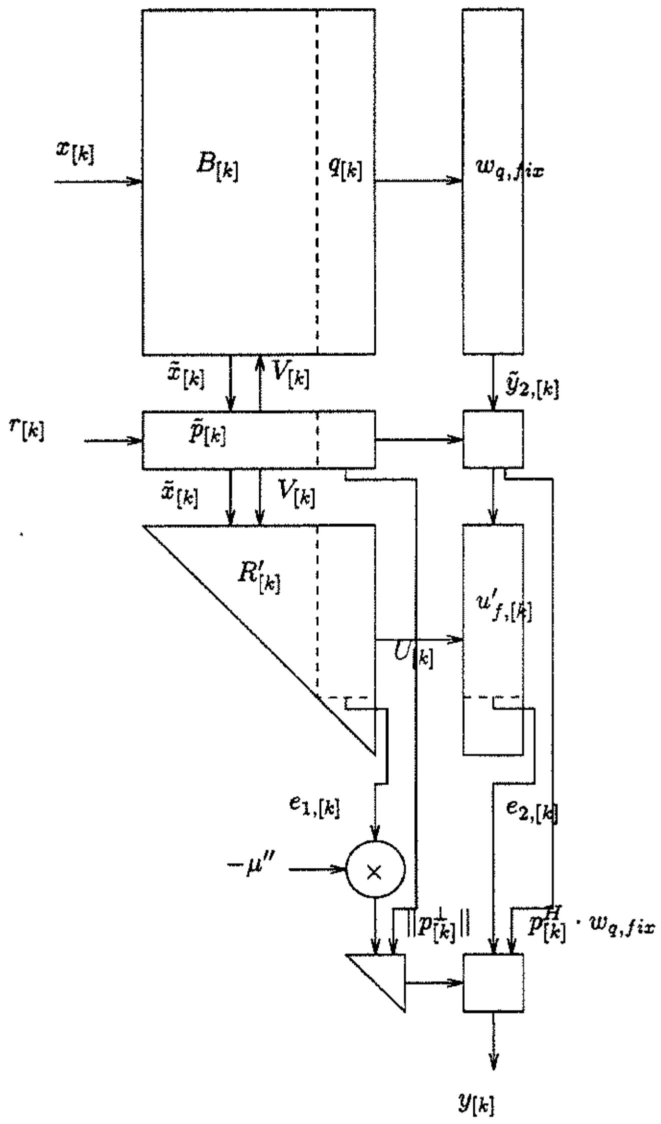


Figure 5.5: Final SFG for robust LCMV beamforming with an adjustable constraint.

**Algorithm 6**

$$\begin{aligned}
B_{[0]} &\leftarrow \mathrm{null}(\left[\begin{array}{cc} C_{fix} & p_{[0]} \end{array}\right]^H) \\
w_{q,fix} &\leftarrow C_{fix}\cdot(C_{fix}^H\cdot C_{fix})^{-1}\cdot m_{fix} \\
p_{[0]}^{\perp} &\leftarrow (I_M - C_{fix}\cdot(C_{fix}^H\cdot C_{fix})^{-1}\cdot C_{fix}^H)\cdot p_{[0]} \\
q_{[0]} &\leftarrow \frac{p_{[0]}^{\perp}}{\|p_{[0]}^{\perp}\|} \\
R'_{[0]} &\leftarrow O_{(M-K+1)\times(M-K+1)} \\
u'_{f,0} &\leftarrow O_{(M-K+1)\times 1} \\
\tilde{p}_{[0]}^H &\leftarrow p_{[0]}^H\cdot\left[\begin{array}{c|c} B_{[0]} & q_{[0]} \end{array}\right]
\end{aligned}$$

for $k = 1, \ldots, \infty$

1. Matrix-vector multiplications

$$\left[\begin{array}{c|c|c} \tilde{x}_{[k]}^H & \tilde{y}_{1,[k]} & \tilde{y}_{2,[k]} \end{array}\right] \leftarrow x_{[k]}^H\cdot\left[\begin{array}{c|c|c} B_{[k-1]} & q_{[k-1]} & w_{q,fix} \end{array}\right]$$

2. Beamforming network update

$$\begin{aligned}
\left[\begin{array}{c|c} \bar{p}_{[k]}^H & p_{[k]}^H\cdot w_{q,fix} \end{array}\right] &\leftarrow \beta\cdot\left[\begin{array}{c|c} \tilde{p}_{[k-1]}^H & p_{[k-1]}^H\cdot w_{q,fix} \end{array}\right]+\cdots \\
&\qquad\cdots+(1-\beta)\cdot r_{[k]}\cdot\left[\begin{array}{cc|c} \tilde{x}_{[k]}^H & \tilde{y}_{1,[k]} & \tilde{y}_{2,[k]} \end{array}\right]
\end{aligned}$$

$$\underbrace{\left[\begin{array}{ccc|c} 0 & \cdots & 0 & \|p_{[k]}^{\perp}\| \end{array}\right]}_{\tilde{p}_{[k]}^H} \leftarrow \bar{p}_{[k]}^H\cdot\prod_{i=1}^{M-K} V_{[k]}^{i|i+1}$$

$$\left[\begin{array}{cc|c} B_{[k]} & q_{[k]} & w_{q,fix} \\ \hline \multicolumn{2}{c|}{R'_{[k]}} & u'_{f,[k]} \\ \hline 0\cdots 0 & e_{1,[k]} & e_{2,[k]} \end{array}\right] \leftarrow \left[\begin{array}{c|c|c} I_M & O & O \\ \hline O & \left(\prod_{i=1}^{M-K} U_{[k]}^{i|i+1}\right)^H & O \\ \hline O & O & 1 \end{array}\right]\cdots$$

$$\cdots\left[\begin{array}{cc|c} B_{[k-1]} & q_{[k-1]} & w_{q,fix} \\ \hline \multicolumn{2}{c|}{R'_{[k-1]}} & u'_{f,[k-1]} \\ \hline \tilde{x}_{[k]}^H & \tilde{y}_{1,[k]} & \tilde{y}_{2,[k]} \end{array}\right]\cdot\left[\begin{array}{c|c} \prod_{i=1}^{M-K} V_{[k]}^{i|i+1} & O \\ \hline O & 1 \end{array}\right]$$

3. QRD update

$$\left[\begin{array}{c|c} R'_{[k]} & u'_{f,[k]} \\ \hline 0\cdots 0 & \star \end{array}\right] \leftarrow G_{[k]}^{M-K+1|M-K+2}\cdot\left[\begin{array}{c|c} R'_{[k]} & u'_{f,[k]} \\ \hline 0\cdots 0\ e_{1,[k]} & e_{2,[k]} \end{array}\right]$$

4. beamformer output computation

$$y_{[k]} \leftarrow e_{2,[k]} + \mu'' e_{1,[k]} \cdot \frac{p_{[k]}^H \cdot w_{q,fix}}{\|p_{[k]}^{\perp}\|}$$

endfor

## 5.3 Simulation results

In the simulation of Figure 5.6 the following scene is considered. A ten-element uniform linear array with inter-element spacing of a half-wavelength, receives a 20 dB SOI at 0 deg, a 10 dB main beam interference at 5 deg and a 30 dB interference at 17 deg, which is approximately the maximum of the first sidelobe (see Figure 2.5). The white noise level is 0 dB. The snapshots of the SOI, interferences and noise are complex independent identically distributed (i.i.d.) zero-mean Gaussian random variables. Furthermore the SOI is assumed to be a training sequence known to the receiver, and is used as the reference signal. The weighting factors $\alpha$ and $\beta$ are set to 0.999. The following set of two constraints is chosen

$$\begin{bmatrix} a(\theta_s) & \frac{\partial}{\partial\theta}a(\theta_s) \end{bmatrix}^H \cdot w = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

The first constraint is the mandatory steering constraint, fixing a unity gain at $\theta_s$. The second constraint imposes an extremum in the directivity pattern at $\theta_s$, hereby already increasing the robustness of the beamformer [3]. Only the steering constraint is recursively estimated. Errors in the second constraint are not compensated for.

The SINR curves and the directivity patterns after 1000 snapshots are averaged over 50 runs. In each run inaccurate knowledge of the array response vector is simulated by independent random perturbations on the gain and phase of each antenna element such that the gain is uniformly distributed in the interval $[0.99, 1.01]$ and the phase in the interval $[-5.4, 5.4]$ deg, corresponding to position inaccuracies of 3%. The same signals are used in all runs.

As seen in the upper half of Figure 5.6, the effect of the perturbations is severe. Without perturbations the algorithm attains an average SINR level of $\pm 13$ dB. With the perturbations added the SINR drops to $\pm 4$ dB. However, if the array response vector is recursively estimated, the algorithm very quickly reaches a level of $\pm 18$ dB, which is even higher

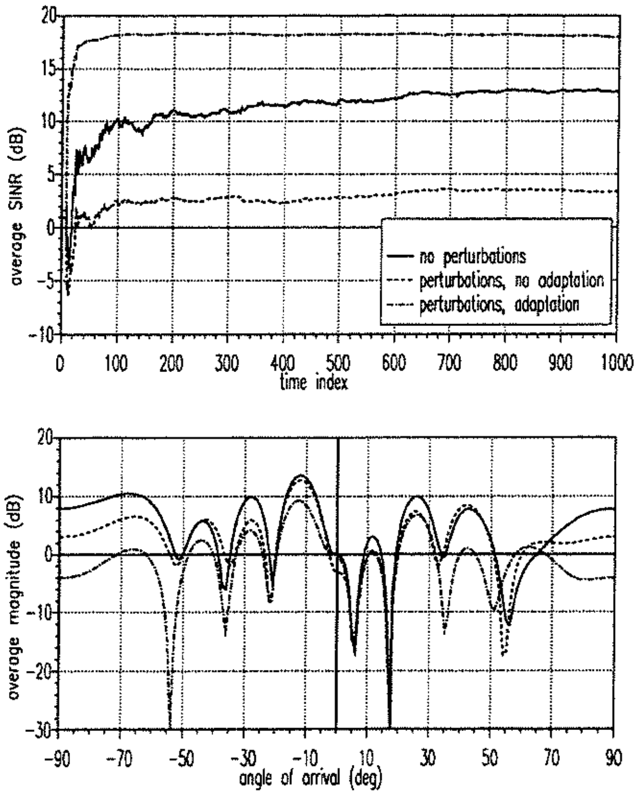


Figure 5.6: Output SINR (upper figure) and the final directivity pattern (lower figure) averaged over 50 runs. Sensor: ULA with $M = 10$. Signals: SOI (0 deg, 20 dB) and 2 interferences at (5 deg, 10 dB) and (17 deg, 30 dB). Weights: $\alpha = \beta = 0.99$.
Case 1: (full line) no perturbations and no adaptive estimation.
Case 2: (dashed line) perturbations: gain errors are uniformly distributed in [-1,1]%, phase errors in [-5.4, 5.4] deg. No adaptive estimation.
Case 3: (dash-dot line) same perturbations. Array response vector is recursively estimated.

than in the perturbation free case. The reason for this surprising result is that the algorithm not only compensates for perturbations in the array response vector, but also for perturbations caused by the finite sample effect. In [27] an analysis is presented which shows that estimation errors due to small data records have the same effect as if a perturbed array response vector were used.

The lower half of Figure 5.6 shows the corresponding average directivity patterns. In all three cases the steep notches at the interference angles are approximately the same. The sidelobe levels however differ, influencing the white noise power in the output. In the case where the perturbations are not compensated for, signal cancelation occurs since the gain at 0 deg is only $-4$ dB. The extremum at 0 deg imposed by the derivative constraint, is also clearly noticeable.

## 5.4 Mapping onto the Jacobi architecture

The SFG of the robust adaptive adaptive LCMV beamformer (Figure 5.5) can be mapped efficiently onto the Jacobi architecture for SVD updating. The structure of both algorithms is identical. Both algorithms consist of a sequence of orthogonal matrix-vector multiplication, QRD updating and two-sided rotations.

The main difference is the computation of the column and row rotations. In the Jacobi SVD updating algorithm, they are the result of an SVD on consecutive $2 \times 2$ diagonal submatrices. In the robust LCMV algorithm, the zeroing of the cross-correlation vector and the annihilation of fill-in entries determines the rotation values. Therefore, we can organize the computation of the robust LCMV beamforming algorithm on the same Jacobi array without loss of performance on condition that the cross-correlation vector $\tilde{p}_{[k]}$ is stored in the diagonal nodes.

The functionality of the nodes in the $R$-array is actually simpler than for SVD updating. Except for the bottom node, the QRD updating can be omitted. Also the row and column rotations can now be computed sequentially, whereas in the SVD updating algorithm they have to be computed simultaneously. A last remark concerns the implementation of the output section combining the outputs $e_{1,[k]}$ and $e_{2,[k]}$. Since this involves only a scalar multiplication and a $1 \times 2$ Schur complement array, this computation can easily be added to the nodes in the last but one row of the $R$-array. The resulting irregularity in the node descriptions is very small.

Again numerical instability may occur due to error accumulation in the beamforming matrix $\left[\begin{array}{cc|c} B_{[k]} & q_{[k]} & w_{q,fix} \end{array}\right]$. In order to avoid numerical error-buildup, one can work with a factorization of the beamforming matrix as described in chapter 3, *i.e.*, the matrix is stored as a sequence of Givens rotations. The matrix has orthonormal columns, except for the quiescent weight vector which may not have a unit norm. This is not a big problem. The inner product with the fixed weight vector can always be performed as a multiplication with the normalized unit-norm vector followed by a scalar multiplication with its norm.

## 5.5 Conclusion

In this chapter an algorithm for robust LCMV beamforming was derived. The sensitivity of the SINR at the beamformer output with respect to gain and phase errors in the array response vector is circumvented by estimating the array response vector directly from the data. A necessary condition is that a reference signal is available. The algorithm is based on the well-known generalized sidelobe canceler with fixed constraints. Simulations indicated that this approach works well and also corrects for finite sampling effects.

The SFG of the algorithm is very similar to the SFG of the Jacobi SVD updating algorithm. Its topology is hardly different. Only the computation of the rotations, which is an internal node program, differs. Therefore the SFG can be mapped onto the Jacobi architecture of section 2.3.2 without loss of performance. The fact that the Jacobi array is able to execute different recursive algorithms by slightly reprogramming the nodes, makes it a fairly general architecture.

The mapping onto the Jacobi array is not the only possibility. A second architecture, specifically tuned towards this beamforming algorithm is described in [129]. It does not store the cross-correlation vector in the diagonal nodes, but in the first row of the $R$-array. Although its throughput is slightly higher, the fact that it is not so general makes it less attractive for hardware realization.

# Chapter 6

# 2-D Harmonic Retrieval

In the last three chapters we turn our attention to the direction finding problem. Each chapter makes different assumptions on the data model. In the standard data model for DOA estimation of multiple narrow-band sources discussed in section 2.1, all sources share the same known carrier frequency. Here we look at a generalized problem. As shown in Figure 6.1 the sources are still assumed to be narrow-band point sources located in the far field of the sensor array. However, their carrier frequencies are no longer known and may differ. For each source both the angular position and the carrier frequency has to be estimated simultaneously. This model applies for instance to the surveillance radar system discussed in chapter 1.

Since the carrier frequencies may differ, it is not longer possible to work with a base-band representation for the signals. Instead, the model for the data matrix is a sum of two-dimensional (2-D) complex sinusoids. The components of the frequency couples are related to the angle-of-arrival, respectively carrier frequency, of the wavefronts. Estimating these frequency couples is known as the 2-D harmonic retrieval problem [87].

In section 6.1 an overview of the existing methods is given. Section 6.2 states the data model. Next, in section 6.3 we develop our algebraically coupled matrix pencils (ACMP) algorithm. It is more computationally efficient than existing methods in the literature. It splits the 2-D problem into two related 1-D estimation problems. In each direction the frequencies are estimated using a computationally efficient shift-invariant subspace algorithm. A further increase in efficiency is due to the algebraic pairing of the horizontal and vertical estimates. Finally, in section 6.4 we compare the estimation performance of the new algorithm with various

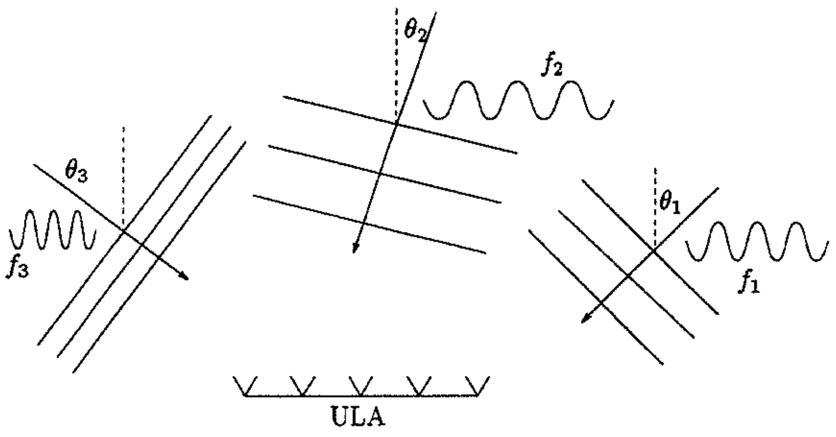


Figure 6.1: A Uniform Linear Array receives multiple narrow-band signals with unknown frequencies $f_i$ and angles of arrival $\theta_i$ from the far-field of the array.

other algorithms by simulations.

## 6.1 Introduction

Two-dimensional spectral estimation is a problem with many applications, *e.g.*, in image processing and array processing. Starting from a 2-D data set, one has to determine its 2-D spectral content. Many 1-D high resolution methods such as the minimum variance method and maximum entropy method, have already been extended to the multidimensional case [56]. However, they require searching over the entire 2-D frequency space or finding the roots of a 2-D polynomial. Often their computational requirements are prohibitive.

From one-dimensional spectral estimation, it is well-known that for harmonic retrieval the costly optimization or polynomial root-finding steps can be avoided. If the time-series consists of a linear combination of a number of sinusoidal signals with unknown amplitude, phase and frequency, then the frequencies can be estimated by means of matrix decomposition techniques only. An example is the ESPRIT algorithm which cleverly exploits the shift-invariant structure of the low-dimensional signal subspace.

For a matrix of 2-D sinusoids a similar low-rank structure holds. This fact was first recognized by Rao and Kung [87]. Their state space method is the first to fully exploit the eigenstructure of the data for spectral estimation. Since the data model is separable along the vertical and horizontal axis, one can model the data by two independent 1-D state space systems. The poles of these systems are related to the frequency components. They are estimated by a state space realization algorithm based on the shift-invariance of the two observability matrices. The decomposition of the 2-D problem into two independent 1-D estimation problems gives rise to a new question. Which horizontal estimate corresponds to which vertical estimate? In [87] this pairing problem is solved by estimating the amplitudes corresponding to each possible pairing. The combinations with the largest amplitudes are finally selected. However, the algorithm of [87] breaks down when multiple 2-D sinusoids share a same frequency component due to rank degeneracy.

This problem is addressed by the matrix enhancement and matrix pencil (MEMP) method of Hua [39]. An enhanced matrix is constructed in which the data are repeated in a double Hankel structure. In the enhanced matrix the original rows and columns are treated differently. This approach effectively restores the full rank of the matrices spanning the signal subspaces in a way which is similar to spatial smoothing schemes for direction-of-arrival estimation of fully coherent narrow-band signals [98, 111]. The cost is additional computation. This is due to the linear increase of the dimensions of the enhanced matrix with the number of 2-D sinusoids in the data. The second problem of pairing the estimates is again solved by the use of a trial-and-error scheme. Basically all possible pairings are checked for the orthogonality of their corresponding vectors to the estimated noise subspace.

Here we present a more efficient 2-D subspace algorithm. This algebraically coupled matrix pencils (ACMP) algorithm is capable of estimating multiple frequency pairs sharing a common component. It uses a different enhancement technique, such that the matrix dimensions do not increase with the total number of signals. Instead they grow proportional to the maximum multiplicity of a frequency component, *i.e.*, the maximum number of times that a frequency component is shared. As opposed to the MEMP algorithm, no enhancing is necessary if the data do not contain shared frequencies. Moreover, no additional operations to pair the frequency components are required. The pairing is obtained by simultaneously solving two algebraically related generalized eigenvalue problems.

## 6.2 Data model

Let the signal consist of a sum of $D$ complex two-dimensional sinusoids with unknown frequency pairs $(u_i, v_i)$ and unknown complex amplitude-and-phase factors $a_i$. Examples where this data model applies, are observations of narrow-band signals taken by a uniform linear sensor array and images of periodically structured patterns.

The data matrix $Z \in \mathbb{C}^{N\times M}$ can then be written as a sum of $D$ rank-one matrices due to each of the spectral components

$$Z = \sum_{i=1}^{D} a_i \cdot x_i \cdot y_i^T$$

where $a_i = r_i \cdot \exp(j\rho_i) \in \mathbb{C}$ with $r_i \in \mathbb{R}$ the amplitude and $\rho_i \in [0, 2\pi[$ the phase of the $i$th sinusoid and $x_i \in \mathbb{C}^N$ and $y_i \in \mathbb{C}^M$ are Vandermonde vectors

$$\begin{aligned} x_i &= \begin{bmatrix} 1 & \phi_i & \phi_i^2 & \cdots & \phi_i^{N-1} \end{bmatrix}^T \\ y_i &= \begin{bmatrix} 1 & \psi_i & \psi_i^2 & \cdots & \psi_i^{M-1} \end{bmatrix}^T. \end{aligned}$$

In image processing the poles $\phi_i$ and $\psi_i$ are given by $\phi_i = \exp(j2\pi u_i \Delta_x)$ and $\psi_i = \exp(j2\pi v_i \Delta_y)$ where $(u_i, v_i)$ and $(\Delta_x, \Delta_y)$ are the spatial frequencies and spatial sampling intervals along the horizontal and vertical axes respectively. In the array processing example $\psi_i$ and $\phi_i$ are defined as $\psi_i = \exp(j2\pi f_i \Delta \sin(\theta_i)/c)$ and $\phi_i = \exp(j2\pi f_i T_s)$ with $(f_i, \theta_i)$ the center frequency and DOA of the $i$th emitter and $\Delta, c$ and $T_s$ the distance between the antenna elements, the speed of propagation of the wave and the sampling interval respectively. More generally, the data may contain transient signals, the poles of which lie inside the unit circle.

The data matrix can also be written as follows

$$Z = X_N \cdot A \cdot Y_M^T + W$$

where the $i$th columns of the Vandermonde matrices $X_N \in \mathbb{C}^{N\times D}$ and $Y_M \in \mathbb{C}^{M\times D}$ hold the vectors $x_i$ and $y_i$ and $A \in \mathbb{C}^{D\times D}$ is a diagonal matrix containing the complex scalars $a_i$. In the sequel the subindex of the Vandermonde matrices, indicating the number of rows, will often be omitted for conciseness. The matrix $W$ represents additive measurement noise on the signals.

The data are linearly dependent on the complex amplitude factors $a_i$ but non-linearly on the poles $(\phi_i, \psi_i)$. Given estimates of the poles, the estimation of the amplitudes reduces to a least squares problem. This is shown in appendix 6.A. Below we solely address the estimation of the pole pairs $(\phi_i, \psi_i)$. For this objective the Vandermonde structure of $X$ and $Y$ will be exploited.

## 6.3 The Algebraically Coupled Matrix Pencil algorithm

The ACMP algorithm is a computationally efficient subspace algorithm for estimating the pole pairs $(\phi_i, \psi_i)$. For the estimation of the $\phi_i$s ($\psi_i$s) a 'vertical' ('horizontal') matrix pencil is constructed. By careful selection of the matrices used ('= algebraic coupling of the matrix pencils') the pairing of the $\phi_i$s and $\psi_i$s is obtained without additional operations.

The ACMP algorithm is based on two properties of the data matrix $Z$. The first property is the fact that in the theoretical case of noiseless data $Z$ has rank $D$ (except for cases with coinciding frequency components). This subspace property allows to reduce the noise level by approximating the original noisy data matrix by an optimal rank $D$ matrix.

The second useful property is the Vandermonde structure of the $X$ and $Y$ matrices. Vandermonde matrices are a special class of matrices obeying a shift-invariant subspace property,

$$\overline{X} = \underline{X} \cdot \Phi \qquad \text{and} \qquad \overline{Y} = \underline{Y} \cdot \Psi$$

where $\Phi$ and $\Psi$ are diagonal matrices containing all $\phi_i$s and $\psi_i$s. The matrices $\overline{V}$ and $\underline{V}$ denote a matrix $V$ after omission of its first and last row respectively. Similarly $|V$ and $V|$ denote the matrix $V$ now with the first respectively last column omitted.

Below we first develop a subspace algorithm for noiseless data in which no frequency components are shared by multiple sinusoids. Then we extend the algorithm with a smoothing scheme such that it is also applicable to the case in which shared frequency components occur. Finally we comment on the refinements in the case of noisy data.

### 6.3.1 Noiseless data

In the noiseless case the following equations hold for the top-left, top-right, bottom-left and bottom-right submatrices obtained by omitting the first or last column and row.

$$Z_{tl} \;=\; \underline{Z|} = \underline{X} \cdot A \cdot \underline{Y}^T$$

$$\begin{aligned}
Z_{tr} &= \underline{|Z} = \underline{X} \cdot A \cdot \overline{Y}^T = \underline{X} \cdot (A \cdot \Psi) \cdot \underline{Y}^T \\
Z_{bl} &= \overline{Z}| = \overline{X} \cdot A \cdot \underline{Y}^T = \underline{X} \cdot (\Phi \cdot A) \cdot \underline{Y}^T \\
Z_{br} &= \overline{|Z} = \overline{X} \cdot A \cdot \overline{Y}^T = \underline{X} \cdot (\Phi \cdot A \cdot \Psi) \cdot \underline{Y}^T
\end{aligned}$$

These submatrices can all be written as the product of the same Vandermonde matrices $\underline{X}$ and $\underline{Y}^T$, but with a different diagonal matrix in between them. Consider now the following two sets of matrix pencils

$$\begin{aligned}
Z_{tr} - \mu \cdot Z_{tl} &= \underline{X} \cdot A \cdot (\Psi - \mu \cdot I) \cdot \underline{Y}^T \\
Z_{br} - \mu \cdot Z_{bl} &= \overline{X} \cdot A \cdot (\Psi - \mu \cdot I) \cdot \underline{Y}^T
\end{aligned}$$

and

$$\begin{aligned}
Z_{br} - \lambda \cdot Z_{tr} &= \underline{X} \cdot A \cdot (\Phi - \lambda \cdot I) \cdot \overline{Y}^T \\
Z_{bl} - \lambda \cdot Z_{tl} &= \underline{X} \cdot A \cdot (\Phi - \lambda \cdot I) \cdot \underline{Y}^T.
\end{aligned}$$

In the first set of 'horizontal' matrix pencils a submatrix at the left side is subtracted from a submatrix at the right side. In the second set of 'vertical' matrix pencils a submatrix at the top side is subtracted from a submatrix at the bottom side.

In the common case that the dimensions of the data matrix are bigger than the number of sinusoidal components (*i.e.* $M > D$, $N > D$) and that there are no shared poles (*i.e.* $\phi_i \neq \phi_j, \psi_i \neq \psi_j$ for all $i \neq j$), each of the above matrix pencils has a generic rank $D$. Only if $\lambda = \psi_i$ and $\mu = \phi_i$, their rank drops to $D - 1$. These pairs of rank-reducing numbers $(\lambda, \mu)$ are thus equal to the pairs of poles $(\phi_i, \psi_i)$ we want to estimate. Since both horizontal (vertical) matrix pencils provide estimates of $\Psi$ $(\Phi)$, it is sufficient to select only one matrix pencil from each set. It now remains to be shown how the rank reducing numbers can be calculated. Below a simple algorithm is given which uses matrix decompositions only. It is inspired by the ESPRIT algorithm for 1-D DOA estimation.

## ACMP algorithm

1. Compute the singular value decomposition of $Z_{tl} \in \mathbb{C}^{(N-1)\times(M-1)}$

$$Z_{tl} = U \cdot \Sigma \cdot V^H$$

where $U \in \mathbb{C}^{(N-1)\times D}$ and $V \in \mathbb{C}^{(M-1)\times D}$ have unitary columns and $\Sigma \in \mathbb{R}^{D\times D}$ is a diagonal matrix.

2. Multiply the horizontal and vertical matrix pencil containing $Z_{tl}$ on the left by $U^H$ and on the right by $V$ to compress all matrix dimensions from $(N-1) \times (M-1)$ to $D \times D$.

$$\begin{aligned} U^H \cdot (Z_{tr} - \mu \cdot Z_{tl}) \cdot V &= F \cdot (\Psi - \mu \cdot I) \cdot G \\ &\overset{def}{=} C_{tr} - \mu \cdot C_{tl} \\ U^H \cdot (Z_{bl} - \lambda \cdot Z_{tl}) \cdot V &= F \cdot (\Phi - \lambda \cdot I) \cdot G \\ &\overset{def}{=} C_{bl} - \lambda \cdot C_{tl} \end{aligned}$$

where $F = U^H \cdot \underline{X} \cdot A$ and $G = \underline{Y}^T \cdot V$ are full rank $D \times D$ matrices. The matrix $C_{tl} = \Sigma$ is already given by the SVD of $Z_{tl}$.

3. Compute the eigenvalue decomposition (EVD) of

$$C_{tl}^{-1} \cdot C_{tr} = G^{-1} \cdot \Psi \cdot G.$$

This is equivalent to computing the generalized eigenvalue decomposition of the matrix pair $(C_{tr}, C_{tl})$. Since $C_{tl} = \Sigma$, inverting $C_{tl}$ is always possible and very cheap. This EVD provides the set of all 'horizontal' poles $\psi_i$.

4. Apply the eigentransformations $G$ to the other pencil

$$G \cdot (C_{tl}^{-1} \cdot C_{bl}) \cdot G^{-1} = \Phi.$$

The same transformation $G$ which is calculated based on the first pencil, can be used to calculate the $\phi_i$s. Only one eigenvalue problem has to be solved. The second one is diagonalized by the same transformations. This remarkable property guarantees an identical ordering of the $\lambda$s and the $\mu$s. Each $\mu_i$ and $\lambda_i$ correspond to the same ($i$th) sinusoidal component. By this 'algebraic coupling' of the matrix pencils, estimates of frequency pairs are obtained instead of two unordered sets of frequency estimates. The pairing of the 1-D estimates is a by-product of the computation, and does not require any additional operations.

The necessary condition for this property to hold is that all matrices involved can be written as the product of the same left and right matrices with possibly a different diagonal matrix in the middle.

This method to pair 1-D estimates in an algebraic manner was proposed by van der Veen *et al.* [115], where they address the problem of simultaneous azimuth and elevation angle estimation of narrowband emitters.

The computational complexity of the algorithm above is very low. It requires only one SVD of a $(M-1) \times (N-1)$ matrix, one EVD of a small $D \times D$ matrix and a few matrix-matrix multiplications.

### 6.3.2 Shared frequencies

The above algorithm fails if multiple frequency pairs have one component in common. The obvious reason is the rank deficiency of the $X$ or $Y$ matrices in that case[1]. To recover all sinusoidal components the full rank $D$ property of the submatrices has to be restored.

In [39] Hua constructs an so-called enhanced block Hankel matrix by partitioning and stacking the original data matrix. A large block Hankel matrix is formed, the blocks of which are Hankel matrices constructed from one row of data. The block Hankel matrix is of rank $D$ on condition that the number of block rows $K$ and the number of rows per block $L$ is larger than the number of sinusoids $D$. This method of enhancing has a few drawbacks. First, as shown in Figure 6.8 in appendix 6.C, rows and columns are treated differently. Secondly if there are no shared frequency components, then in the previous section it was argued that all frequencies can be computed from the original data matrix. However, the MEMP algorithm requires that also in this case the large matrix is constructed. In addition to increasing the computational burden, the growth in matrix dimensions also limits the applicability of the MEMP algorithm to data sets in which the number of sinusoids is smaller than half of the smallest dimension of the data matrix. A third drawback is that the construction of the matrix pencils in the MEMP algorithm is not compatible with the algebraic coupling technique. Therefore, an additional step must be included to pair the 1-D estimates. In [39] the orthogonality with respect to the estimated noise subspace of the signal vectors corresponding to all possible pairings is used as selection criterion. In appendix 6.C we show how the MEMP algorithm may be modified such that the algebraic pairing technique is applicable.

Below we present an alternative rank-restoration technique. It treats rows and columns in a symmetric way. The increase in the dimensions of the matrix is only proportional to the multiplicities of the poles. And the algebraic pairing of the matrix pencils is retained.

[1]In narrow-band direction finding all emitters share the same carrier frequency too. The rank degeneracy is circumvented by using a base-band representation of the signals.

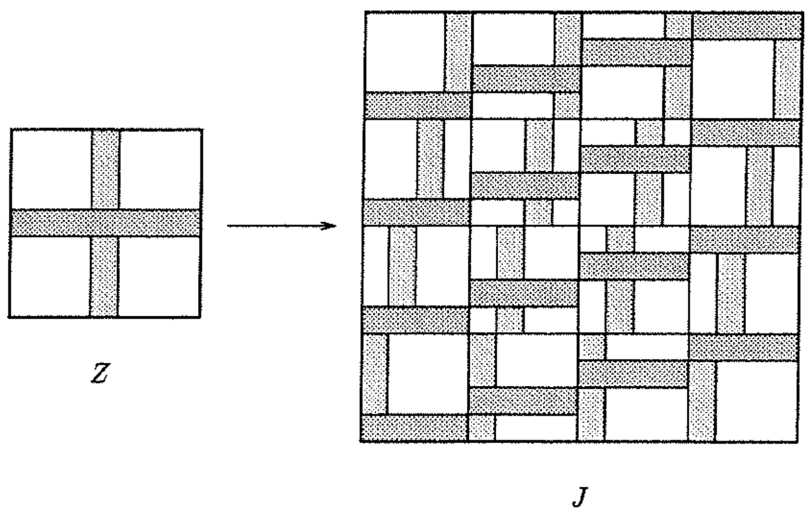


Figure 6.2: Illustration of the ACMP enhancing method. $Z \in \mathbb{C}^{7\times 7}, K = L = 4$.

Let the matrix $J \in \mathbb{C}^{K(N-L+1)\times L(M-K+1)}$ be defined as

$$J = \begin{bmatrix} Z^{1,1} & Z^{2,1} & \dots & Z^{L,1} \\ Z^{1,2} & Z^{2,2} & \dots & Z^{L,2} \\ \vdots & \vdots & & \vdots \\ Z^{1,K} & Z^{2,K} & \dots & Z^{L,K} \end{bmatrix}$$

where the $(k,l)$th block component $J_{kl} \in \mathbb{C}^{(N-L+1)\times(M-K+1)}$ is

$$J_{kl} = Z^{l,k} = Z_{l|N-L+l,k|M-K+k}$$

The index notation $_{i|j}$ denotes all rows (columns) from the $i$th to the $j$th row (column). The enhancing method is illustrated in Figure 6.2. The blocks of the enhanced matrix $J$ are submatrices taken from the data matrix $Z$. Consecutive blocks in a block row, respectively block column, of $J$ are vertically, respectively horizontally, shifted overlapping submatrices of $Z$. It is easy to check that $J$ satisfies a decomposition

$$J = X_e \cdot A \cdot Y_e^T$$

where $X_e$ and $Y_e$ are the Khatri-Rao product [8] of two Vandermonde matrices, *i.e.*, the $i$th column of the product is the Kronecker product of the $i$th columns of the two multiplicands.

$$\begin{aligned} X_e &= Y_K \odot X_{N-L+1} \\ &= \begin{bmatrix} X_{N-L+1}^T & \Psi \cdot X_{N-L+1}^T & \cdots & \Psi^{K-1} \cdot X_{N-L+1}^T \end{bmatrix}^T \\ Y_e &= X_L \odot Y_{M-K+1} \\ &= \begin{bmatrix} Y_{M-K+1}^T & \Phi \cdot Y_{M-K+1}^T & \cdots & \Phi^{L-1} \cdot Y_{M-K+1}^T \end{bmatrix}^T \end{aligned}$$

The rank of the enhanced matrix $J$ will be $D$, if both $X_e$ and $Y_e$ have rank $D$. The Vandermonde matrix $X$ can only be rank-deficient if several $\phi_i$s coincide. However, the corresponding $\psi_i$s are necessarily different. Consider the submatrix of $X_e$ by stacking every $N-L+1$st row. This is a Vandermonde matrix in the $\psi_i$s. Therefore, the matrices $X_e$ and $Y_e$ have full rank $D$ under the conditions that

$$\begin{aligned} N-L+1 &\geq D \\ M-K+1 &\geq D \\ K &\geq \max_i\{\#\phi_i\}, \quad i=1,\cdots,D \\ L &\geq \max_i\{\#\psi_i\}, \quad i=1,\cdots,D \end{aligned}$$

where $\#\phi_i$ is the multiplicity of $\phi_i$, *i.e.* the number of sinusoids having $\phi_i$ in common. In contrast to the MEMP enhancing method, the rank is already restored when a dimension of the matrix is multiplied with the maximal multiplicity of a pole along that direction.

The ACMP algorithm can now be applied to submatrices of the matrix $J$. However, the process of omitting the leading and trailing rows and columns has to be performed on the block components of $J$, *e.g.*, the top-left matrix $J_{tl}$ has block components

$$J_{tl,kl} = \underline{Z}|^{l,k} = Z_{l|N-L-1+l,k|M-K-1+k}.$$

After defining in a similar way the top-right and bottom-left matrices $J_{tr}$ and $J_{bl}$, the ACMP algorithm can operate on the matrix pencils $J_{tr} - \mu \cdot J_{tl}$ and $J_{bl} - \lambda \cdot J_{tl}$. However, a small modification is required for numerical reasons. The eigenvectors (and thus part of the matrix $G$) of an eigenvalue with multiplicity $> 1$ are not uniquely determined. Therefore it is safer to compute the eigentransformations $G$ based on a linear combination of the two matrix pencils

$$C_{tl}^{-1} \cdot (\beta \cdot C_{tr} + (1-\beta) \cdot C_{bl}) = G^{-1} \cdot \Lambda \cdot G$$

where $\beta$ is a scalar such that $\Lambda$ does not contain repeated eigenvalues. The eigentransformations $G$ are then welldefined. The pole estimates $\Psi$ and $\Phi$ are finally computed by applying $G$ to $C_{tl}^{-1} \cdot C_{tr}$ and $C_{tl}^{-1} \cdot C_{bl}$.

### 6.3.3 Noisy data

Additive noise on the data destroys the low-rank property of the data matrix $Z$. Its rank will now be equal to $\min(M, N)$. However, it is well-known that low-rank matrices perturbed by noise, can be well recovered by use of the SVD. Therefore in step 1 of the ACMP algorithm the SVD of $Z_{tl}$ is truncated explicitly to rank $D$.

In the case of i.i.d zero-mean noise it can be shown that after truncation of the SVD of the data matrix $Z$, the estimation of the frequencies is still asymptotically unbiased and consistent. If a dimension of the matrix grows to infinity, the estimates of the frequencies along the other direction converge with probability 1 to their exact values [17].

However, truncating the SVD of the enhanced matrix $J$ no longer results in a consistent estimate, due to the coherence of the noise on the elements which are repeated. An optimal approximant has low rank $D$ and satisfies the same linear structure as the original enhanced matrix $J$, *i.e.*, it belongs to the same subspace of dimension $M \times N$ in an ambient matrix space of dimension $K(N - L + 1) \times L(M - K + 1)$. It is possible to impose this structure on the low-rank approximant, *e.g.*, by an alternating projections algorithm. In each iteration the structured matrix is approximated by an unstructured rank $D$ matrix by truncating its SVD. Then this unstructured matrix is projected onto the subspace of structured matrices by averaging all elements which should be equal. This preprocessing step is computationally expensive, since each iteration involves the SVD of a large matrix.

A cheaper alternative for increased noise suppression is proposed in [115]. In the noiseless case, the eigentransformations calculated from one matrix pencil perfectly diagonalize the other matrix pencil. Therefore, the $D \times D$ matrices $C_{tl}^{-1} \cdot C_{tr}$ and $C_{tl}^{-1} \cdot C_{bl}$ commute. This is no longer true for noisy observations. Further noise suppression and improved accuracy is obtained by replacing the matrices above by the 'closest' set of commuting matrices. Although this computation requires non-linear optimization, an important advantage is that now only small $D \times D$ matrices are involved. Moreover, approximate algebraic methods are developed in [115]. One of the methods computes additive corrections $\Delta A, \Delta B \in \mathbb{C}^{D \times D}$ to the given non-commuting matrices $A, B \in \mathbb{C}^{D \times D}$ such that they approximately

| $i$ | $r_i$ | $\rho_i$ | $u_i$ | $v_i$ |
|---|---|---|---|---|
| 1 | 1 | 0 | 0.24 | 0.26 |
| 2 | 1 | 0 | 0.24 | 0.24 |
| 3 | 1 | 0 | 0.26 | 0.24 |

Table 6.1: Parameters of the simulation example

commute. The matrices $\Delta A, \Delta B$ are the minimum norm solution of the equation

$$(A + \Delta A) \cdot (B + \Delta B) = (B + \Delta B) \cdot (A + \Delta A)$$

after omission of the second-order terms. Under the assumption that the corrections are small, this first order correction is reported to give good results. The computation involved is low in comparison to the main SVD computation, since only a linear system in $2D^2$ unknowns has to be solved.

## 6.4 Simulations

Many variations and many refinements on the basic 2-D ACMP algorithm may be devised. They will all behave differently in terms of noise rejection. An analytical asymptotic analysis up to first order of the performance of SVD based algorithms is hard, although not impossible [53]. However, this is beyond the scope of this chapter. Here we make some observations on the robustness of the algorithms with respect to noise based on Monte Carlo simulations.

More specifically we repeat the simulations of [39]. A $20 \times 20$ matrix is generated with $D = 3$ sinusoids whose parameters are given in Table 6.1. The noise is additive independent normally distributed and zero-mean. The spacing of the frequencies is less than the Rayleigh resolution limit which is $\Delta u = \Delta v = 0.05$ for a data sequence of length 20. Therefore, resolving and accurately estimating the three peaks is impossible with a Fourier based algorithm. Moreover, two pairs of sinusoids share a frequency component. Enhancement by at least a factor of $K = L = 2$ is necessary.

We compare the performance of four algorithms on this simulation. The first algorithm is the original MEMP algorithm [39]. The second algorithm is the modified MEMP algorithm of appendix 6.C, which is

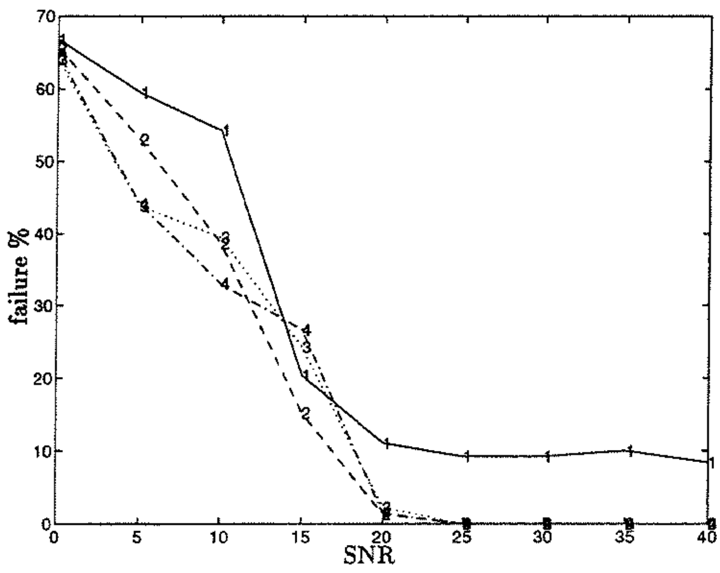


Figure 6.3: Comparison of the estimation accuracy of 4 four algorithms (here and 3 subsequent figures). Here the failure rate is plotted versus the SNR. Algorithms: (1) MEMP (2) AC-MEMP (3) ACMP (4) ACMP 1st. Enhancement factors are set to $K = L = 6$. The simulation results are averaged for 250 independent runs for each SNR. A failure is reported if an estimate lies outside the square region $(\pm 0.1, \pm 0.1)$ about each exact frequency pair. Due to wrong pairing of the 1-D estimates, the MEMP algorithm has a high failure rate, even at high SNRs.

also based on algebraic coupling, but uses the original MEMP enhancing method. It is denoted by the 'AC MEMP' algorithm. The third algorithm is our ACMP algorithm. Finally, the fourth algorithm is the ACMP algorithm to which we added the first order commutation method [115]. It is called the 'ACMP 1st' algorithm.

In the Monte Carlo simulation the four algorithms are tested on 250 independent data sets for each signal-to-noise ratio. The enhancement factors are set to $K = L = 6$. Although their meaning in the MEMP and ACMP enhancing method is different, the size of the enhanced data matrix is roughly $K \cdot L$ times the size of the original data matrix. Therefore, it is fair to compare all algorithms with the same factors $K$ and $L$.

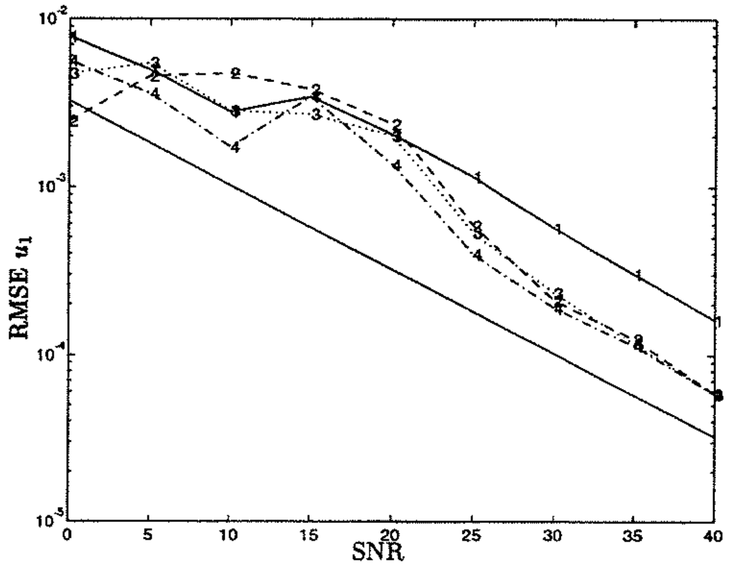


Figure 6.4: Same scenario as Figure 6.3. Plot of the RMSE on $u_1$ versus SNR. The full line is the Cramér-Rao Bound on $u_1$. Algorithms 2 to 4 have a comparable performance, better than algorithm 1.

Some simulation results are reported in Figures 6.3 to 6.6. Figure 6.3 shows the average failure rate versus SNR. The failure rate is more or less the same for all algorithms, except for the MEMP algorithm at higher SNRs. As can be seen from the scatter plot of Figure 6.6, the 10 % failures of the MEMP algorithm are due to a wrong pairing of the horizontal and vertical estimates.

Figures 6.4 and 6.5 show the evolution of the root mean square error (RMSE) on the $u$-components of the first and second sinusoid. Because of the short data length the CRB is not attained by these subspace algorithms. From Figure 6.4 one is tempted to conclude that the algebraically coupled algorithms outperform the MEMP algorithm. However, this MEMP algorithm has a higher accuracy at estimating the $u_2$ component.

The noise suppression scheme of the ACMP 1st algorithm decorrelates the estimates of the outer sinusoids. The scatter plot of the ACMP al-

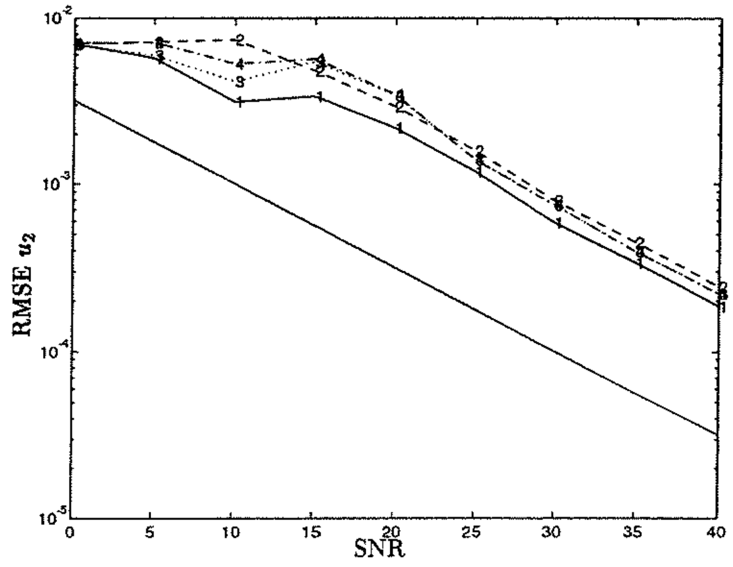


Figure 6.5: Same scenario as Figure 6.3. Plot of the RMSE on $u_2$ versus SNR. The full line is the Cramér-Rao Bound on $u_2$. Algorithms 2 to 4 have a comparable performance, worse than algorithm 1.

gorithm in Figure 6.6 exhibits correlation between these estimates, since the estimate clouds of the sinusoids are aligned towards each other. By approximately looking for nearby commuting matrices, these estimates are decorrelated (ACMP 1st). This refinement also roughly halves the RMSE on $u_1$ at lower SNRs. The RMSE on the $u_2$ component does not alter significantly.

More simulations did not bring clear evidence to rank the four subspace algorithms according to their robustness with respect to noise. At higher enhancement factors $K = L = 10$ the failure rate for lower SNRs drops somewhat, but the accuracy of the estimates does not increase, presumably because of the noise coherence in the enhanced matrix.

The second Monte Carlo simulation illustrates the consistency of the subspace algorithms for growing data matrices. The same data matrix as in the previous simulation is considered at SNR=30 dB. Now the number of columns $M$ is varied from 10 to 60 in steps of 10. For each dimension 25

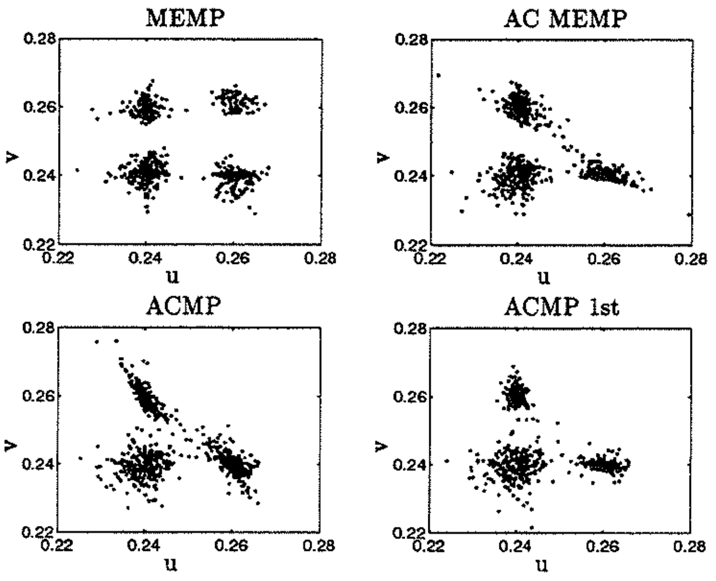


Figure 6.6: Same scenario as Figure 6.3. Scatter plots for SNR = 20 dB. The failures of MEMP are clearly seen to be due to wrong pairing of the $u$ and $v$ estimates. The estimate clouds of the outer sinusoids are less aligned towards each other by imposing approximate commutativity (ACMP 1st).

independent runs are performed using the ACMP algorithm. Figure 6.7 shows the experimental RMS error on $u_1$ and the Cramér-Rao bound. They approach each other as $M$ increases.

## 6.5 Conclusion

In this chapter we have presented the ACMP algorithm for estimating the parameters of a 2-D line spectrum in noise. Although we only considered undamped sinusoidal signals, the algorithm can equally well be applied to the general case in which the signals consist of 2-D damped sinusoids. Further generalizations to higher dimensional problems are straightforward. Because of the separability of the data model, the multidimensional estimation problem is split up in a set of algebraically coupled 1-D estimation problems. These are then solved by an efficient ESPRIT-like algorithm

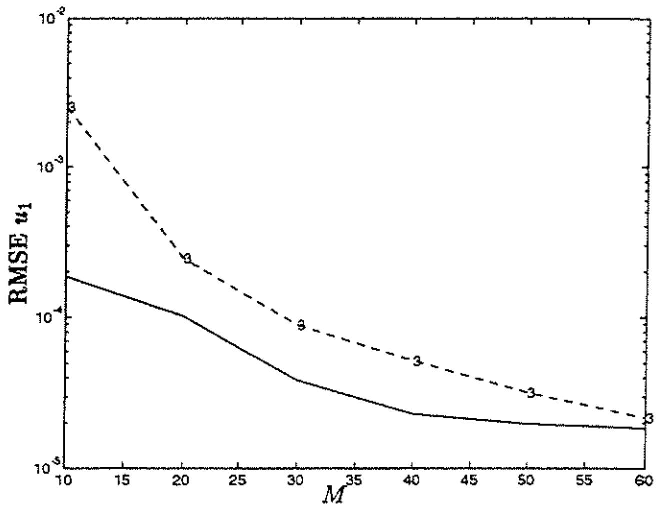


Figure 6.7: Dashed line: RMSE on $u_1$. Full line: Cramér-Rao bound. 25 independent runs, $K = L = 6$, SNR = 30 dB, $N = 20$ and $M$ varies.

which does not require optimization. Instead the shift-invariant property of the column- and row-subspaces is exploited to derive an algorithm which consists exclusively of a sequence of matrix transformations.

Furthermore a rank-restoration method has been presented to deal with situations in which a same frequency component is shared by multiple sinusoids. The method requires less repetition of data than earlier enhancing methods and is compatible with the algebraic pairing method.

Finally, Monte Carlo simulations indicate that the performance of the different algorithms for noisy finite data records does not differ significantly, except for one feature. Due to the algebraic coupling of the horizontal and vertical matrix pencils, wrong pairings of the estimates are eliminated.

No adaptive and parallel version of the algorithm was presented here. However, the algorithm can be considered as a special case of the wideband direction finding algorithm discussed in the next chapter.

# Appendices

## 6.A Estimation of the amplitudes

Let $q = \text{vec}(Q)$ denote the operation of stacking the columns of a matrix $Q$ into a single column vector $q$. The data model can be rewritten as an overdetermined linear system

$$\text{vec}(Z) = S(\phi, \psi) \cdot a(r, \rho) + \text{vec}(W)$$

where $a \in \mathbb{C}^D$ is a vector containing the complex mode weights, and the matrix $S \in \mathbb{C}^{MN \times D}$ is the Khatri-Rao product of $Y$ and $X$, *i.e.*, column $i$ of $S$ is the Kronecker product $y_i \otimes x_i$.

Given estimates of the set $\{(\phi_i, \psi_i), i = 1, \cdots, D\}$, the matrix $S$ can be constructed. In the case of white noise, the optimal least-squares estimate is then given by

$$a = (S^H \cdot S)^{-1} \cdot S^H \cdot \text{vec}(Z).$$

## 6.B Cramér-Rao Bound

There are 4 unknowns associated with each sinusoid, *i.e.*, the amplitude $r_i$, the phase $\rho_i$ and the horizontal and vertical normalized frequencies $u_i$ and $v_i$. These unknowns can be stacked in a parameter vector $\theta \in \mathbb{R}^{4D}$. The variance on any unbiased estimator $\hat{\theta}_i$ is fundamentally lower bounded by the Cramér-Rao bound (CRB) [81, 94]

$$E\left((\hat{\theta}_i - \theta_i)^2\right) \geq (F_\theta^{-1})_{ii}$$

where the Fisher information matrix $F_\theta \in \mathbb{R}^{4D \times 4D}$ is the covariance matrix of the score function $s_\theta(z) \in \mathbb{R}^{4D}$

$$F_\theta = E\left(s_\theta(z) \cdot s_\theta^T(z)\right).$$

This score function $s_\theta(z)$ is the gradient vector of the log-likelihood function

$$s_\theta(z) = \frac{\partial}{\partial\theta} \ln p_\theta(z)$$

where $p_\theta(z)$ is the probability density function of the data. In the derivation below we assume that the matrix $Z$ has been stacked in a vector $z = \text{vec}(Z)$ as in appendix 6.A.

For Gaussian i.i.d. zero-mean noise, the probability density function is

$$p_\theta(z) = \frac{1}{(2\pi\gamma)^{MN}} \exp\left(-\frac{1}{\gamma}(z-m)^H \cdot (z-m)\right)$$

where $\gamma$ is the noise power and the mean $m \in \mathbb{C}^{MN}$ is defined as $m = S \cdot a$ with $S$ and $a$ given in appendix 6.A. This mean is a function of all unknown parameters. The log-likelihood function $l_\theta(z) = \ln p_\theta(z)$ is easily derived as

$$l_\theta(z) = -MN\ln(2\pi\gamma) - \frac{1}{\gamma}\left((z-m)^H(z-m)\right).$$

The next stochastic variable to compute is the score function $s_\theta(z)$. Calculating the derivatives of $l_\theta(z)$ gives a gradient vector

$$s_\theta(z) = \frac{2}{\gamma}\left(\frac{\partial m_r^T}{\partial\theta}\cdot(z_r - m_r) + \frac{\partial m_i^T}{\partial\theta}\cdot(z_i - m_i)\right)$$

where column $i$ of the Jacobian matrices $\frac{\partial m_r}{\partial\theta}, \frac{\partial m_i}{\partial\theta} \in \mathbb{R}^{MN\times 4D}$ contains the derivatives with respect to $\theta_i$ of the real and imaginary part of the vector $m$.

The Fisher information matrix is finally obtained as

$$\begin{aligned}
F_\theta &= E\left(s_\theta(z)\cdot s_\theta^T(z)\right) \\
&= \frac{4}{\gamma^2}\{\frac{\partial m_r^T}{\partial\theta}\underbrace{E\left((z_r-m_r)^T(z_r-m_r)\right)}_{\frac{\gamma}{2}I_{4D}}\frac{\partial m_r}{\partial\theta} + \cdots \\
&\quad \frac{\partial m_i^T}{\partial\theta}\underbrace{E\left((z_i-m_i)^T(z_i-m_i)\right)}_{\frac{\gamma}{2}I_{4D}}\frac{\partial m_i}{\partial\theta} + \cdots \\
&\quad \frac{\partial m_r^T}{\partial\theta}\underbrace{E\left((z_r-m_r)^T(z_i-m_i)\right)}_{O_{4D\times 4D}}\frac{\partial m_i}{\partial\theta} + \cdots \\
&\quad \frac{\partial m_i^T}{\partial\theta}\underbrace{E\left((z_i-m_i)^T(z_r-m_r)\right)}_{O_{4D\times 4D}}\frac{\partial m_r}{\partial\theta}\} \\
&= \frac{2}{\gamma}\mathrm{Re}\left(\frac{\partial m^H}{\partial\theta}\cdot\frac{\partial m}{\partial\theta}\right)
\end{aligned}$$

To conclude the derivation of the CRB, we need the derivatives of the mean $m$

$$\begin{aligned}
\frac{\partial m}{\partial r_i} &= \exp(j\rho_i)(y_i \otimes x_i) \\
\frac{\partial m}{\partial \rho_i} &= ja_i(y_i \otimes x_i) \\
\frac{\partial m}{\partial u_i} &= j2\pi a_i(y_i \otimes (W_N \cdot x_i)) \\
\frac{\partial m}{\partial v_i} &= j2\pi a_i((W_M \cdot y_i) \otimes x_i)
\end{aligned}$$

where the weighting matrix $W_k \in \mathbb{R}^{k\times k}$ is a diagonal matrix with entries $\begin{bmatrix} 0 & 1 & \cdots & k-1 \end{bmatrix}$. An alternative component-wise derivation of this Cramér-Rao bound is given in [39].

## 6.C A modified MEMP algorithm

In this appendix we show how the MEMP algorithm of [39] can be adapted such that the pairing of the estimates is obtained automatically.

The enhanced matrix $J \in \mathbb{C}^{(K\cdot L)\times(M-L+1)(N-K+1)}$ in the MEMP method is a block Hankel matrix, in which each block is again a Hankel matrix constructed with the entries from one row of the original data matrix

$$J = \begin{bmatrix} Z_1 & Z_2 & \cdots & Z_{N-K+1} \\ Z_2 & Z_3 & \cdots & Z_{M-K+2} \\ \vdots & & & \vdots \\ Z_K & Z_{K+1} & \cdots & Z_N \end{bmatrix}$$

where a block $Z_i \in \mathbb{C}^{L\times M-L+1}$ is defined as

$$Z_i = \begin{bmatrix} z_{i,1} & z_{i,2} & \cdots & z_{i,M-L+1} \\ z_{i,2} & z_{i,3} & \cdots & z_{i,N-L+2} \\ \vdots & & & \vdots \\ z_{i,L} & z_{i,L+1} & \cdots & z_{i,M} \end{bmatrix}.$$

The composition of the enhanced matrix is illustrated in Figure 6.8. Rows and columns are treated differently. Entries from a single row are found in an anti-diagonal of blocks, whereas entries from a single column are located on a particular anti-diagonal of each block.

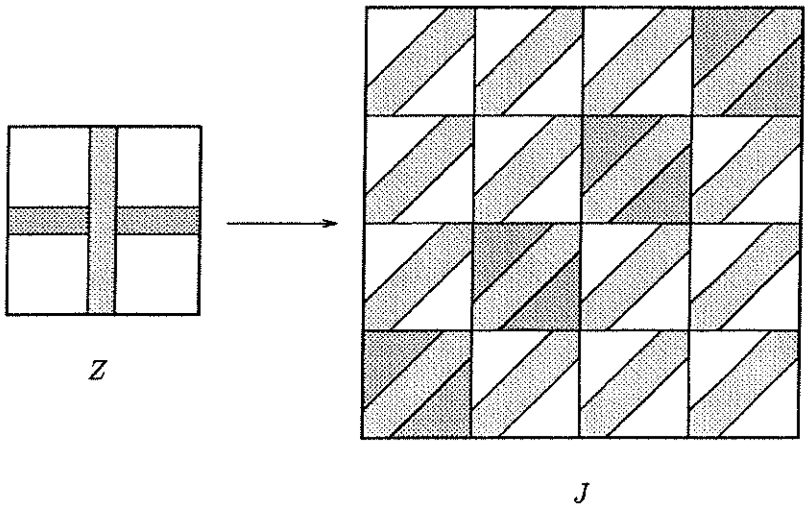


Figure 6.8: Illustration of the MEMP enhancing method. $Z \in \mathbb{C}^{7\times 7}, K = L = 4$.

For noiseless data this double Hankel matrix has a decomposition

$$J = X_e \cdot A \cdot Y_e^T$$

where the definitions of $X_e$ and $Y_e$ are

$$\begin{aligned} X_e &= X_K \odot Y_L \\ &= \begin{bmatrix} Y_L^T & \Phi \cdot Y_L^T & \cdots & \Phi^{K-1} \cdot Y_L^T \end{bmatrix}^T \\ Y_e &= X_{N-K+1} \odot Y_{M-L+1} \\ &= \begin{bmatrix} Y_{M-L+1}^T & \Phi \cdot Y_{M-L+1}^T & \cdots & \Phi^{N-K} \cdot Y_{M-L+1}^T \end{bmatrix}^T \end{aligned}$$

In the original MEMP algorithm the poles are estimated from two independent matrix pencils. The pairing of the frequency components is obtained by looking for those manifold vectors which lie closest to the estimated left singular space.

Here we want to avoid this pairing problem. Instead we construct two matrix pencils consisting of matrices which have the same eigen-transformations. As in the original MEMP algorithm we only use the shift invariances of the left factor $X_e$. Three matrices $J_{tt}, J_{tb}, J_{bt} \in$

$\mathbb{C}^{(L-1)(K-1)\times(M-L+1)(N-K+1)}$ are defined as

$$\begin{aligned}
J_{tt} &= X_{tt}\cdot A\cdot Y_e^T \\
&= \left[\begin{array}{cccc} \underline{Y_L}^T & \Phi\cdot\underline{Y_L}^T & \cdots & \Phi^{K-2}\cdot\underline{Y_L}^T \end{array}\right]^T\cdot A\cdot Y_e^T \\
J_{tb} &= X_{tb}\cdot A\cdot Y_e^T \\
&= \left[\begin{array}{cccc} \overline{Y_L}^T & \Phi\cdot\overline{Y_L}^T & \cdots & \Phi^{K-2}\cdot\overline{Y_L}^T \end{array}\right]^T\cdot A\cdot Y_e^T \\
J_{bt} &= X_{bt}\cdot A\cdot Y_e^T \\
&= \left[\begin{array}{cccc} \Phi\cdot\underline{Y_L}^T & \Phi^2\cdot\underline{Y_L}^T & \cdots & \Phi^{K-1}\cdot\underline{Y_L}^T \end{array}\right]^T\cdot A\cdot Y_e^T
\end{aligned}$$

The matrix $J_{tt}$ consists of the submatrix of $J$ in which the last block row and the last row of each block are omitted. The matrix $J_{tb}$ consists of the submatrix of $J$ in which the last block row and the first row of each block are omitted. The matrix $J_{bt}$ consists of the submatrix of $J$ in which the first block row and the last row of each block are omitted. These matrices have a rank equal to the number of complex sinusoids $D$ on condition that

$$\begin{aligned}
K &> D \\
L &> D \\
(M-L+1)(N-K+1) &\geq D.
\end{aligned}$$

The poles can now be estimated from the two matrix pencils

$$\begin{aligned}
J_{tb}-\lambda\cdot J_{tt} &= X_{tt}\cdot(\Psi-\lambda\cdot I_D)\cdot A\cdot Y_e^T \\
J_{bt}-\mu\cdot J_{tt} &= X_{tt}\cdot(\Phi-\mu\cdot I_D)\cdot A\cdot Y_e^T.
\end{aligned}$$

Algebraic pairing of the estimates is again obtained by computing the eigentransformations from one of the pencils and applying them to the other.

# Chapter 7

# State Space Direction Finding for Wide-Band Emitters

In the previous chapter we have studied the extension of the ESPRIT algorithm for DOA estimation of narrow-band sources to the case in which the carrier frequencies of the emitters are unknown. Figure 7.1 shows another extension in which the radiating sources emit wide-band signals. An example where this model applies are acoustic arrays for adaptive acoustic noise reduction, *e.g.,* in hands-free mobile telephony [75] and hearing aids [12]. The wide-band signals are modeled by a (low-order) time-invariant linear system driven by white noise. The novelty in our approach lies in the use of state space descriptions for the sensor outputs. The algorithms to estimate the system poles and the location of the sources are inspired by the recent subspace algorithms for system identification [63, 120].

In section 7.2 the data model for the wide-band signals is developed. This is done in a state space description framework. Next, section 7.3 is devoted to the novel state space direction finding method. The properties of the data model which are instrumental in the estimation of TDOAs are discussed. Also the relation between the number of sensors and the number of emitters is clarified. The section is concluded with a simple non-recursive algorithm. In section 7.4 this algorithm is used as the backbone of a recursive wide-band direction finding algorithm. Again care is taken that this algorithm is amenable to efficient implementation on the parallel Jacobi architecture. Finally, section 7.5 gives an evaluation of the algorithms by means of simulations.

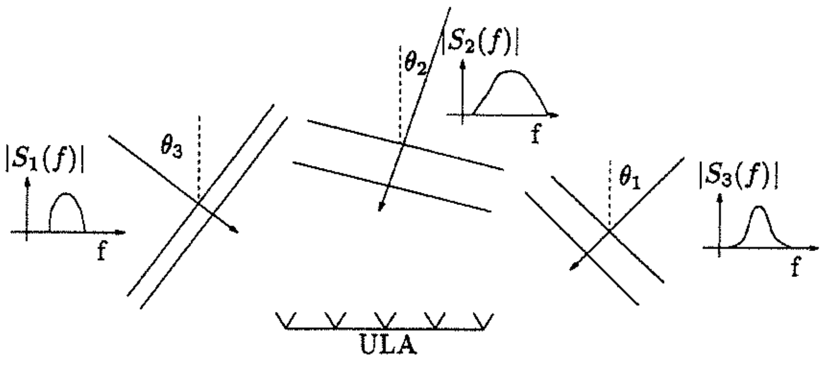


Figure 7.1: A Uniform Linear Array receives multiple wide-band signals with unknown spectral power $|S_i(f)|^2$ and angles of arrival $\theta_i$ from the far-field of the array.

## 7.1 Introduction

A natural approach to the wide-band estimation problem is to decompose it into a series of related narrow-band estimation problems. First a finite set of frequencies is selected and after narrow-band filtering, the signal subspaces are estimated in each of these frequency bins. Then the mapping from signal subspaces to directions-of-arrivals is done by optimizing a generalized narrow-band criterion. Examples of such methods are the unit circle eigendecomposition rational signal subspace (UCERSS) algorithm of Su and Morf [104], and the coherent signal subspace (CSS) method of Wang and Kaveh [40, 145]. A disadvantage of these methods is that they utilize only the signal power within individual narrow frequency bins. Therefore, a large number of frequency points may be needed for good estimation accuracy. Furthermore, it is unclear how to select these frequencies if no a priori information of the spectral content of the signals is available.

An attractive alternative is to model the sensor signals as the output of a multiple input multiple output (MIMO) linear system driven by white noise. The wide-band signals are then decomposed into a finite number of modes and a narrow-band-like technique is applied to each of the modes. In this way all signal power can be captured by only a limited number of modes. This type of modeling is especially attractive when the required model order is low. It was introduced by Su and Morf

[74, 104] where they extend the concept of signal subspaces from narrow-band to wide-band signals using vectors of rational functions in the delay operator $z^{-1}$. In fact, their modal decomposition signal subspace (MDSS) algorithm [105] applies the MUSIC criterion [96] to each of the modes. Therefore, it also inherits the basic disadvantages of this method, namely a priori knowledge and storage of the array manifold over the entire z-domain is required and a costly nonlinear optimization procedure has to be performed. Regalia and Loubaton propose an alternative search method to estimate the rational subspaces by adapting lossless filters according to a power splitting criterion [89]. This method is well suited for recursive operation.

On the other hand, the ESPRIT algorithm [92] discussed in section 2.3.1 does not require searching at all. The prerequisite is that the sensor array consists of two identical subarrays displaced by a known constant vector, but no further knowledge of the array characteristics is required. Due to this special structure the global optimization step to map signal subspaces onto DOAs, can be eliminated since the DOAs are now computable by using matrix decomposition techniques only.

A generalization of the ESPRIT algorithm to ARMA modeled wide-band signals has been presented by Ottersten and Kailath [78]. They develop their algorithm starting again from the z-domain description. The system poles and corresponding residue matrices are estimated by means of an overdetermined Yule-Walker method based on a finite number of estimated output correlation matrices for increasing time lags. Then the directions-of-arrival of each of the modes are determined by applying the ESPRIT algorithm to the residue matrices. The algorithm has a number of drawbacks. From the numerical point of view it is bad to explicitly compute correlation matrices. Also determining the system poles as the roots of a differential equation may be ill-conditioned. From the implementation point of view their method turns out to be intrinsically sequential. Therefore, it is unsuited for real-time operation.

Here a novel numerically reliable algorithm is presented, which can be viewed as an alternative generalization of the ESPRIT algorithm from narrow-band to wide-band emitters. The algorithm works directly on the data, *i.e.*, it is no longer based on z-domain descriptions. Instead it models the wide-band signals by state space descriptions in the time domain. This idea is apparently new in the field of wide-band DOA estimation. The algorithm shares some ideas with the recently developed subspace algorithms for system identification [63, 120, 121].

A basic advantage of using time domain descriptions is the fact that recursive and parallel algorithms are feasible. We will show how our wide-band DOA estimation algorithm can be formulated as a Jacobi algorithm, which is highly parallel and can be mapped efficiently on the systolic Jacobi architecture discussed in section 2.3.2.

## 7.2 Data model

We again consider the configuration of Figure 2.10, in which the sensor array consists of doublets. The propagation medium is assumed to be non-dispersive and homogeneous, with constant propagation speed $c$. The time-difference-of-arrival (TDOA) between the X- and the Y-sensor of a doublet for a signal impinging from the far field under an angle $\theta_l$, is given by

$$\tau_l = \frac{\Delta \sin(\theta_l)}{c}. \tag{7.1}$$

The directions-of-arrival $\theta_l$ are measured from broadside in counterclockwise direction. Once the $\tau_l$s are estimated, the DOAs are easily retrieved from Eq. (7.1). The sensor set-up is identical to the narrow-band case. Only the incident signals now have a wide-band spectrum.

The block diagram of Figure 7.2 presents the wide-band signal model in more detail. The vector $v_l(t) \in \mathbb{R}^M$ collects the array observations due to the $l$th emitter. These observations are modeled as the output of a continuous-time SIMO (single input multiple output) linear system $S_l$ of finite order $n_l$ driven by an independent scalar stochastic process $u_l(t) \in \mathbb{R}$,

$$\begin{aligned} \dot{z}_l(t) &= A_l \cdot z_l(t) + B_l \cdot u_l(t) \\ v_l(t) &= C_l \cdot z_l(t) + D_l \cdot u_l(t) \end{aligned}$$

where $A_l \in \mathbb{R}^{n_l \times n_l}, B_l \in \mathbb{R}^{n_l \times 1}, C_l \in \mathbb{R}^{M \times n_l}$ and $D_l \in \mathbb{R}^{M \times 1}$ are the system matrices of the $l$th emitter and $z_l(t) \in \mathbb{R}^{n_l}$ is the state vector of $S_l$. The assumptions on the independent stochastic processes $u_l(t)$ are that they are zero-mean and almost white, in the sense that they have a finite correlation time which is smaller than the sampling period $T_s$

$$E\left(u_l(t) \cdot u_l(t+\tau)\right) = 0, \quad \forall \tau \geq T_s. \tag{7.2}$$

The system $S_l$ is a concatenation of a linear system modeling the $l$th emitter and a linear system modeling the array response, which depends

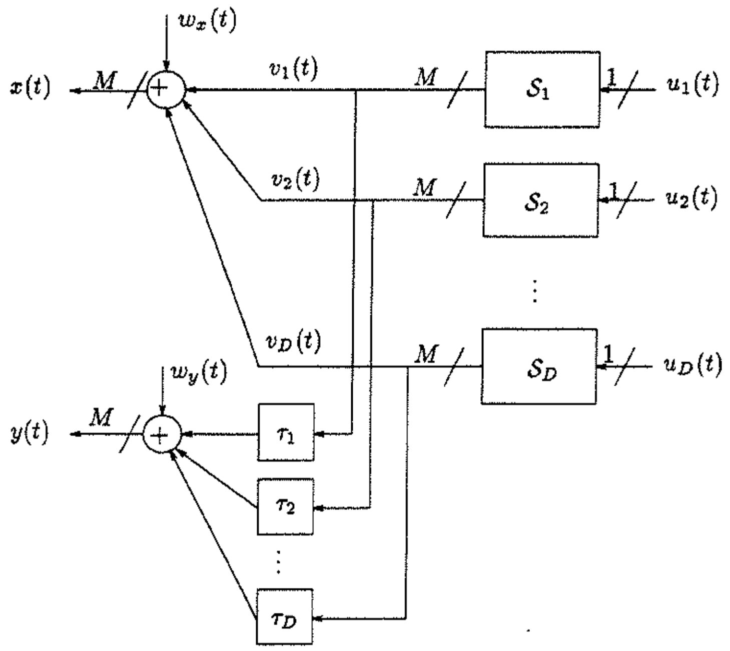


Figure 7.2: Block diagram of the wide-band signal model. Each emitter has an associated linear system $\mathcal{S}_l$ which models the transfer from its (almost) white noise input $u_l(t)$ to the X-array output. The summation at the Y-array output is preceded by the propagation delay blocks $\tau_l$.

on the angle $\theta_l$. Generically the bandwidth of a sensor is much larger than the bandwidth of the emitter signals. Therefore, the sensor dynamics are in most cases negligible or at least clearly separable from the dynamics of the emitter signals. However, this assumption is not crucial for the algorithm below.

The X-array outputs are collected in $x(t) \in \mathbb{R}^M$

$$x(t) = \sum_{l=1}^{D} v_l(t) + w_x(t)$$

where $D$ is the total number of emitters and the vector $w_x(t)$ adds independent white measurement noise, uncorrelated with the emitter signals.

The full state space model for $x(t)$ is as follows

$$\begin{aligned}
\dot{z}(t) &= \underbrace{\begin{bmatrix} A_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & A_D \end{bmatrix}}_{A} \cdot z(t) + \underbrace{\begin{bmatrix} B_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & B_D \end{bmatrix}}_{B} \cdot u(t) \\
x(t) &= \underbrace{\begin{bmatrix} C_1 & \dots & C_D \end{bmatrix}}_{C} \cdot z(t) + \underbrace{\begin{bmatrix} D_1 & \dots & D_D \end{bmatrix}}_{D} \cdot u(t) + w_x(t)
\end{aligned}$$

with obvious definitions for $z(t) \in \mathbb{R}^n$ and $u(t) \in \mathbb{R}^D$

$$\begin{aligned}
z(t) &= \begin{bmatrix} z_1(t)^T & \cdots & z_D(t)^T \end{bmatrix}^T \\
u(t) &= \begin{bmatrix} u_1(t) & \cdots & u_D(t) \end{bmatrix}^T.
\end{aligned}$$

The global system matrix $A \in \mathbb{R}^{n \times n}$ is thus a block diagonal matrix of dimension $n = \sum_{l=1}^{D} n_l$. In most cases $A$ can be reduced to diagonal form by a similarity transformation. Only when there are eigenvalues with multiplicity $> 1$, this diagonalization may not be possible.

The observations made by the Y-array are given by

$$y(t) = \sum_{l=1}^{D} v_l(t - \tau_l) + w_y(t).$$

Here $\tau_l$ can be any real number in the interval

$$-qT_s \le -\frac{\Delta}{c} \le \tau_l \le \frac{\Delta}{c} \le qT_s$$

where $q \in \mathrm{N}$ is the smallest positive integer such that the last inequality holds and $T_s$ is the sampling period. Now define the time instant $t_0 = t - qT_s$ such that $t_0 \le t - \tau_l$ for $1 \le l \le D$. Then the state $z_l(t - \tau_l)$ can be written as the sum of a term due to the state $z_l(t_0)$ and a term due to the inputs from time $t_0$ on

$$z_l(t - \tau_l) = e^{A_l(t - t_0 - \tau_l)} \cdot z_l(t_0) + \int_{t_0}^{t - \tau_l} e^{A_l(t - \tau_l - \sigma)} \cdot B_l \cdot u_l(\sigma) \cdot d\sigma.$$

This expansion leads to the following equation for $y(t)$

$$y(t) = C \cdot e^{A(t - t_0)} \cdot \Phi \cdot z(t_0) + f_y(t_0, t) + w_y(t) \tag{7.3}$$

where the matrices $e^{A(t-t_0)}, \Phi \in \mathbb{R}^{n\times n}$ are defined as

$$\begin{aligned} e^{A(t-t_0)} &= \begin{bmatrix} e^{A_1(t-t_0)} & O & O \\ O & \ddots & O \\ O & O & e^{A_D(t-t_0)} \end{bmatrix} \\ \Phi &= \begin{bmatrix} e^{-A_1\tau_1} & O & O \\ O & \ddots & O \\ O & O & e^{-A_D\tau_D} \end{bmatrix}, \end{aligned}$$

$f_y(t_0, t)$ is a term summing the input contributions

$$f_y(t_0, t) = \sum_{l=1}^{D} \int_{t_0}^{t-\tau_l} h_l(t - \tau_l - \sigma) \cdot u_l(\sigma) \cdot d\sigma$$

and $h_l(t)$ is the impulse response of the $l$th system

$$\begin{aligned} h_l(t) &= C_l \cdot e^{A_l t} \cdot B_l, & t > 0 \\ &= D_l & t = 0. \end{aligned}$$

A similar expression for $x(t)$ is easily obtained

$$x(t) = C \cdot e^{A(t-t_0)} \cdot z(t_0) + f_x(t_0, t) + w_x(t) \tag{7.4}$$

where

$$f_x(t_0, t) = \sum_{l=1}^{D} \int_{t_0}^{t} h_l(t - \sigma) \cdot u_l(\sigma) \cdot d\sigma.$$

Comparing Eqs. (7.3) and (7.4) we see that the terms due to the state at time $t_0$, only differ in the occurrence of a matrix $\Phi$. This property is due to the doublet structure of the sensor array. Since the X- and the Y-array are identical, they share the same system matrices. The TDOAs become apparent in the exponent of the block components of $\Phi$, reflecting the exponential decay of the state $z(t_0)$. This observation will be utilized in the algorithm of the next section.

## 7.3 State space direction finding

In this section we show how the TDOAs can be estimated as a function of the corresponding generalized eigenvalues of two related matrix pencils. Each matrix pencil can be viewed as a generalization of a narrow-band

ESPRIT pencil. The first pencil exploits a shift-in-time to estimate the modes present in the X-sensor outputs. Its generalized eigenvalues are $\lambda_k = e^{p_l T_s}$ where the $p_l$s are the poles of the continuous-time system. The second pencil exploits the shift-in-space between the X- and Y-arrays. Here the generalized eigenvalues are $\mu_k = e^{-p_l \tau_l}$.

## 7.3.1 Input-output equations

In the sequel we make extensive use of block Hankel matrices $X_{k|k+i-1} \in \mathbb{R}^{Mi \times j}$ consisting of sampled data vectors $x_{[k]} = x(k.T_s)$

$$X_{k|k+i-1} = \begin{bmatrix} x_{[k]} & x_{[k+1]} & \cdots & x_{[k+j-1]} \\ x_{[k+1]} & x_{[k+2]} & \cdots & x_{[k+j]} \\ \vdots & & & \vdots \\ x_{[k+i-1]} & x_{[k+i]} & \cdots & x_{[k+i+j-2]} \end{bmatrix}. \tag{7.5}$$

The subscripts in the above definition indicate the discrete-time indices of the first and last snapshot in the first column. It will turn out that the parameters $i$ and $j$ must be chosen such that $Mi > n$ and $j \gg Mi$. In practice this choice is not trivial since in general the total number of modes $n$ is unknown. Here the assumption is made that $i$ can be properly determined using standard rank determination tests [147, 153].

Block Hankel matrices, formed with the sensor outputs, can be written as a linear combination of a state matrix and an input matrix (plus noise terms). This is a generalization of the input-output equations (7.3,7.4). The following important expressions are obtained by straightforward element-wise substitution of (7.3,7.4) [63]

$$\begin{aligned} X_{q+i|q+2i-1} &= \Gamma_i \cdot A^q \cdot Z_i + F_{x,(i,q)} + W_{x,q+i|q+2i-1} && (7.6) \\ X_{q+i+1|q+2i} &= \Gamma_i \cdot A^{q+1} \cdot Z_i + F_{x,(i,q+1)} + W_{x,q+i+1|q+2i} && (7.7) \\ Y_{q+i|q+2i-1} &= \Gamma_i \cdot \Phi \cdot A^q \cdot Z_i + F_{y,(i,q)} + W_{y,q+i|q+2i-1} && (7.8) \end{aligned}$$

where $\Gamma_i \in \mathbb{R}^{Mi \times n}$ is the extended observability matrix

$$\Gamma_i = \begin{bmatrix} C^T & (CA)^T & \cdots & (CA^{i-1})^T \end{bmatrix}^T,$$

the block diagonal matrix $A \in \mathbb{R}^{n \times n}$ is defined as $A = e^{AT_s}$, $Z_i \in \mathbb{R}^{n \times j}$ contains a state sequence

$$Z_i = \begin{bmatrix} z_{[i]} & z_{[i+1]} & \cdots & z_{[i+j-1]} \end{bmatrix},$$

$F_{\star,(i,q)} \in \mathbb{R}^{Mi \times j}$ has as $(k,l)$th entry

$$F_{\star,(i,q)}(k,l) = f_\star((i+l-1)T_s, (i+q+k+l-2)T_s)$$

and the block Hankel matrices $W_{\star,k|k+i-1}$ are defined as in Eq. (7.5).

### 7.3.2 An instrumental variable method

It is interesting to compare the first terms, due to the states $Z_i$, in Eqs. (7.6-7.8) with the narrow-band ESPRIT data model of Eqs. (2.6,2.7). In the case of wide-band emitters the extended observability matrix $\Gamma_i$ plays the role of the narrow-band array gain matrix $A$, the state sequence $Z_i$ replaces the signal matrix $S$ and the block diagonal matrix $\Phi$ is the equivalent of the narrow-band diagonal matrix $\Phi$. The number of narrow-band emitters $D$ is substituted by the total number of modes $n$.

In order to isolate the state term, we use an instrumental variable approach. We look for an additional matrix such that its row space is orthogonal to the row spaces of the input and noise matrices in Eqs. (7.6-7.8), but has a non-zero projection onto the row space of the state term. A natural choice is the matrix of past outputs

$$X_{0|i-1} = \Gamma_i \cdot Z_0 + F_{x,(0,0)} + W_{x,0|i-1}.$$

We now define the projected matrices

$$\begin{aligned} X^p_{q+i|q+2i-1} &= X_{q+i|q+2i-1}/X_{0|i-1} \\ X^p_{q+i+1|q+2i} &= X_{q+i+1|q+2i}/X_{0|i-1} \\ Y^p_{q+i|q+2i-1} &= Y_{q+i|q+2i-1}/X_{0|i-1} \end{aligned}$$

where $A/B$ is a shorthand notation [121] for the orthogonal projection of the row space of $A$ onto the row space of $B$

$$A/B = \frac{1}{j} AB^T(BB^T)^{-1}B.$$

Under the assumption that the inputs and the measurement noise are independent zero-mean (almost) white stochastic processes, and that current states are uncorrelated with current and later inputs, the following relations hold asymptotically

$$\lim_{j\to\infty}\frac{1}{j}\begin{bmatrix} Z_i \\ F_{x,(i,q)} \\ F_{x,(i,q+1)} \\ F_{y,(i,q)} \\ W_{x,q+i|q+2i} \\ W_{y,q+i|q+2i-1}\end{bmatrix}\cdot\begin{bmatrix} Z_0^T & F_{x,(0,0)}^T & W_{x,0|i-1}^T\end{bmatrix}=\begin{bmatrix} \star & \star & O \\ O & O & O \\ O & O & O \\ O & O & O \\ O & O & O \\ O & O & O\end{bmatrix}$$

where the $\star$s denote non-zero matrices and the $O$s denote zero matrices of appropriate dimension.

**Proof**

Assuming ergodicity, the operator $\lim_{j\to\infty}\frac{1}{j}\sum_1^j(\cdot)$ converges with probability 1 to the expectation operator $E(\cdot)$. Many of the orthogonality relations are straightforward to proof. Below we proof the orthogonality of the input matrices.

$$\begin{aligned}
&\lim_{j\to\infty}\frac{1}{j}(F_{y,(i,q)}\cdot F_{x,(0,0)}^T)(k,l) \quad = \cdots \\
= \; & \lim_{j\to\infty}\frac{1}{j}\sum_{m=1}^{j}(f_y((i+m-1)T_s,(i+q+k+m-2)T_s)\cdot \\
& f_x^T((m-1)T_s,(m+l-2)T_s)) \\
= \; & E(f_y(iT_s,(i+q+k-1)T_s)\cdot f_x^T(0,(l-1)T_s)) \\
= \; & E\left(\left[\sum_{m=1}^{D}\int_{iT_s}^{(i+q+k-1)T_s-\tau_m} h_m((i+q+k-1)T_s-\tau_m-\sigma)\cdot u_m(\sigma)d\sigma\right]\cdot\right. \\
& \left.\left[\sum_{n=1}^{D}\int_0^{(l-1)T_s} h_n((l-1)T_s-\xi)\cdot u_n(\xi)d\xi\right]^T\right) \\
= \; & \sum_{m=1}^{D}\sum_{n=1}^{D}\int_{iT_s}^{(i+q+k-1)T_s-\tau_m} d\sigma\int_0^{(l-1)T_s} d\xi\left(h_m(\cdot)\underbrace{E(u_m(\sigma)\cdot u_n^T(\xi))}_{0}h_n(\cdot)^T\right)
\end{aligned}$$

The fact that $E(u_m(\sigma)\cdot u_n^T(\xi))=0$ follows from the independence of the driving stochastic processes if $m\neq n$, and from the following inequality chain if $m=n$

$$0\le\xi\le(l-1)T_s\le(i-1)T_s<iT_s\le\sigma$$

and the assumption that the maximal correlation time of all driving noises $u_m(t)$ is smaller than the sampling period (Eq. (7.2)). ■

After projection we can write the following low-rank decomposition

$$\begin{bmatrix} X^p_{q+i|q+2i-1} \\ X^p_{q+i+1|q+2i} \\ Y^p_{q+i|q+2i-1} \end{bmatrix} = \begin{bmatrix} \Gamma_i \\ \Gamma_i \cdot A \\ \Gamma_i \cdot \Phi \end{bmatrix} \cdot G$$

where the matrix $G \in \mathbb{R}^{n \times j}$ is $G = A^q \cdot Z_i / X_{0|i-1}$.

These asymptotic equations clearly resemble the ESPRIT equations in the narrow-band case, except that now $A$ and $\Phi$ may be block diagonal instead of diagonal matrices. The three matrices share a common column space of dimension $n$ - the signal subspace - Span$(\Gamma_i)$, and a common row space Span$(G^T)$, where Span$(Q)$ denotes the range of a matrix $Q$, *i.e.*, the space spanned by the columns of $Q$. To reconstruct the signal subspace, it is clearly necessary that $Mi \geq n$.

### 7.3.3 Estimating the TDOAs

The final stage in the estimation of the TDOAs resembles the 2-D harmonic retrieval algorithm of the previous chapter. Two low-rank matrix pencils are constructed

$$\begin{aligned} X^p_{q+i+1|q+2i} - \lambda \cdot X^p_{q+i|q+2i-1} &= \Gamma_i \cdot (A - \lambda \cdot I_n) \cdot G & (7.9) \\ Y^p_{q+i|q+2i-1} - \mu \cdot X^p_{q+i|q+2i-1} &= \Gamma_i \cdot (\Phi - \mu \cdot I_n) \cdot G & (7.10) \end{aligned}$$

The rank-reducing numbers $\lambda = e^{pT_s}$ of the first matrix pencil give information on the spectral content of the signals. The rank-reducing numbers $\mu = e^{-p\tau_l}$ of the second pencil give information on the location of the sources. Given the numbers associated with the same pole $p$ of $A$, the TDOA $\tau_l$ of source $l$ can finally be computed as

$$\tau_l = -T_s \cdot \frac{\ln \mu}{\ln \lambda}.$$

The only problem left is how to match the generalized eigenvalues of the two matrix pencils. Again the transformations which diagonalize matrix pencil (7.9) also diagonalize (7.10). This is due to the fact that the $l$th block entries of $\Phi$ and $A$ are related by

$$\Phi_l = e^{-A_l \tau_l} = (e^{A_l T_s})^{-\tau_l/T_s} = (e^{AT_s})_l^{-\tau_l/T_s} = A_l^{-\tau_l/T_s}.$$

and their eigenvectors are the same

$$\begin{aligned} A_l &= Q_l \cdot \Lambda_l \cdot Q_l^{-1} \\ \Phi_l &= Q_l \cdot \underbrace{\Lambda_l^{-\tau_l/T_s}}_{M_l} \cdot Q_l^{-1}. \end{aligned}$$

Therefore if the eigentransformations computed from one of the pencils, are applied to the other, the $k$th eigenvalues $\lambda_k$ and $\mu_k$ both correspond to the same pole $p$ of the continuous-time system.

In the practical case where the projection is not done with infinite horizon ($j$ is a finite number), the measurement noise and inputs are not annihilated perfectly. All three matrices $X^p_{q+i+1|q+2i}$, $X^p_{q+i|q+2i-1}$ and $Y^p_{q+i|q+2i-1}$ are then noisy and (generically) have full rank $Mi$. The low-rank property can be imposed by solving the following optimization problem.

Find matrices $F \in \mathbb{R}^{Mi\times n}$, $G \in \mathbb{R}^{n\times j}$ and diagonal matrices $D_1, D_2 \in \mathbb{R}^{n\times n}$ which minimize the Frobenius norm of

$$\begin{bmatrix} X^p_{q+i|q+2i-1} \\ X^p_{q+i+1|q+2i} \\ Y^p_{q+i|q+2i-1} \end{bmatrix} - \begin{bmatrix} F \\ F\cdot D_1 \\ F\cdot D_2 \end{bmatrix} \cdot G.$$

Because of the structure in the low-rank matrices this minimization problem is not solvable in terms of a singular value decomposition, but requires a non-linear optimization procedure. Algorithms that solve a similar so-called 'Multiple Invariance ESPRIT' narrow-band problem, have been proposed [108]. However, as in the case of noisy 2-D harmonic retrieval, approximate refinements can be computed which do not require a lot of computation [115].

## 7.3.4 Relation doublets versus emitters

In order to estimate all modes and their corresponding TDOAs the rank of the pencils in Eqs. (7.9,7.10) must be $n$. The diagonal matrices $A$ and $\Phi$ always have rank $n$. Since the matrix $G \in \mathbb{R}^{n\times j}$ is a projected stochastic state sequence, its generic rank is $n$ as well. The rank of the matrix pencils thus equals the rank of the observability matrix $\Gamma_i$. The Popov-Belevitch-Hautus (PBH) eigenvector test [45] states that a matrix pair $\{C, A\}$ is observable if and only if there exists no eigenvector of $A$ lying in the null space of $C$,

$$\neg(\exists x \neq 0) \text{ such that } A \cdot x = \lambda x \text{ and } C \cdot x = O.$$

Below we examine this condition in detail. Let $K$ be the maximum geometric multiplicity of an eigenvalue $p_i$ of $A$ and let $\mathcal{S}_1$ denote the corresponding eigenspace. Let $\mathcal{S}_2$ denote the null space of $C$. The dimension of $\mathcal{S}_2$ is $n - M$ if $C$ has full rank $M$ (we assume $n > M$). This property of unambiguity can be imposed by proper sensor array design. According to the dimension theorem of Grassmann

$$\begin{aligned} \underbrace{\dim(\mathcal{S}_1)}_{K} + \underbrace{\dim(\mathcal{S}_2)}_{n-M} &= \underbrace{\dim(\mathcal{S}_1 + \mathcal{S}_2)}_{\leq n} + \underbrace{\dim(\mathcal{S}_1 \cap \mathcal{S}_2)}_{=0} \text{ (PBH)} \\ \iff \quad M &\geq K. \end{aligned}$$

Therefore, the number of antenna doublets $M$ must at least be equal to the maximum geometric multiplicity of an eigenvalue $p$ of $A$. If no wide-band emitters share the same pole, then one sensor doublet suffices to estimate all DOAs. In the special case of narrow-band direction finding, all emitters share the same pole $p = j2\pi f_c$ where $f_c$ is the common carrier frequency. This implies that in this case there must be at least as many doublets as narrow-band emitters.

The same relation doublets versus emitters also holds for the wide-band algorithm of Ottersten and Kailath [78]. There it is derived based on the rank of certain residue matrices.

### 7.3.5 Basic algorithm

Our aim is not to develop the best possible matrix pencil algorithm in terms of robustness and noise rejection, but to develop an algorithm which can also be implemented recursively and which is amenable to parallel implementation.

A first concern is to avoid the squaring of data when computing the projection of the block Hankel data matrices, because of the potential loss of numerical accuracy. The appropriate mathematical tool is the LQ decomposition (LQD, transpose of the QRD) of $X_{0|i-1}$

$$X_{0|i-1} = L \cdot Q$$

where $L \in \mathbb{R}^{Mi \times Mi}$ is a lower triangular matrix and $Q \in \mathbb{R}^{Mi \times j}$ has orthogonal rows. The projection onto the row space of $X_{0|i-1}$ can then be done in a numerically reliable manner by means of the matrix $Q$

$$A/X_{0|i-1} = A \cdot Q^T \cdot Q.$$

The common factor $Q$ may be left out, leading to a first row compression

$$\begin{aligned} X^r_{q+i|q+2i-1} &= X_{q+i|q+2i-1} \cdot Q^T \\ X^r_{q+i+1|q+2i} &= X_{q+i+1|q+2i} \cdot Q^T \\ Y^r_{q+i|q+2i-1} &= Y_{q+i|q+2i-1} \cdot Q^T \end{aligned}$$

where $X^r_{q+i|q+2i-1}, X^r_{q+i+1|q+2i}, Y^r_{q+i|q+2i-1} \in \mathbb{R}^{Mi \times Mi}$ are now square matrices.

The second step in the algorithm is the estimation of the signal subspace Span$(\Gamma_i)$. Under the assumption of zero-mean independent perturbations on the matrix entries, this signal subspace can be estimated as the space spanned by the $n$ left singular vectors, corresponding to the $n$ largest singular values of the matrix $X^r_{q+i|q+2i-1}$. Alternatively, the estimate of the signal subspace may be refined by truncating the SVD of the $Mi \times 3Mi$ matrix

$$\left[\; X^r_{q+i|q+2i-1} \;\middle|\; X^r_{q+i+1|q+2i} \;\middle|\; Y^r_{q+i|q+2i-1} \;\right]$$

at the cost of additional computation. The accuracy improvement will depend on the noise level. The estimated signal subspaces are then used to compress the dimensions of the matrices from $Mi \times Mi$ to $n \times n$

$$X^c_{q+i|q+2i-1} = U_s^T \cdot X^r_{q+i|q+2i-1} \cdot V_s.$$

The compressed matrices $X^c_{q+i+1|q+2i}$ and $Y^c_{q+i|q+2i-1}$ are defined in a similar fashion.

The third step is the computation of the generalized eigenvalues of the matrix pencils Eqs. (7.9,7.10). They can also be calculated using orthogonal transformations only by means of the generalized Schur decompositions (GSD) [11] of the $n \times n$ compressed matrices

$$\begin{aligned} X^c_{q+i+1|q+2i} - \lambda \cdot X^c_{q+i|q+2i-1} &= Z \cdot (T_{q+i+1|q+2i} - \lambda \cdot T_{q+i|q+2i-1}) \cdot W^T \\ Y^c_{q+i|q+2i-1} - \mu \cdot X^c_{q+i|q+2i-1} &= Z \cdot (S_{q+i|q+2i-1} - \mu \cdot T_{q+i|q+2i-1}) \cdot W^T \end{aligned}$$

where again $Z^T \cdot Z = I_n$ and $W^T \cdot W = I_n$ but now $T_{q+i+1|q+2i}, T_{q+i|q+2i-1}, S_{q+i|q+2i-1} \in \mathbb{R}^{n \times n}$ are lower triangular matrices. The generalized eigenvalues are then found as the ratios of the diagonal entries of $T_{q+i+1|q+2i}$ and $T_{q+i|q+2i-1}$ and of $S_{q+i|q+2i-1}$ and $T_{q+i|q+2i-1}$ respectively. The final non-recursive algorithm is given below.

**Algorithm 7**

1. LQ decomposition
$$X_{0|i-1} = L \cdot Q$$

2. Projection
$$\begin{aligned} X^r_{q+i|q+2i-1} &= X_{q+i|q+2i-1} \cdot Q^T \\ X^r_{q+i+1|q+2i} &= X_{q+i+1|q+2i} \cdot Q^T \\ Y^r_{q+i|q+2i-1} &= Y_{q+i|q+2i-1} \cdot Q^T \end{aligned}$$

3. Signal subspace estimation
$$X^r_{q+i|q+2i-1} = \left[\begin{array}{c|c} U_s & U_n \end{array}\right] \cdot \left[\begin{array}{c|c} \Sigma_s & \\ \hline & \Sigma_n \end{array}\right] \cdot \left[\begin{array}{c} V_s^T \\ \hline V_n^T \end{array}\right]$$

4. Row and column compression
$$\begin{aligned} X^c_{q+i|q+2i-1} &= U_s^T \cdot X^r_{q+i|q+2i-1} \cdot V_s (= \Sigma_s) \\ X^c_{q+i+1|q+2i} &= U_s^T \cdot X^r_{q+i+1|q+2i} \cdot V_s \\ Y^c_{q+i|q+2i-1} &= U_s^T \cdot Y^r_{q+i|q+2i-1} \cdot V_s \end{aligned}$$

5. Generalized Schur decomposition
$$\begin{aligned} T_{q+i|q+2i-1} &= Z^T \cdot X^c_{q+i|q+2i-1} \cdot W \\ T_{q+i+1|q+2i} &= Z^T \cdot X^c_{q+i+1|q+2i} \cdot W \\ S_{q+i|q+2i-1} &= Z^T \cdot Y^c_{q+i|q+2i-1} \cdot W \end{aligned}$$

6. Estimation of $\lambda, \mu$ and $\tau$

```
      for $l = 1, \ldots, n$
            $\lambda_l = \frac{T_{q+i+1|q+2i}(l,l)}{T_{q+i|q+2i-1}(l,l)}$
            $\mu_l = \frac{S_{q+i|q+2i-1}(l,l)}{T_{q+i|q+2i-1}(l,l)}$
            $\tau_l = -T_s \frac{\ln \mu_l}{\ln \lambda_l}$
      endfor
```

## 7.4 Parallel and adaptive wide-band direction finding

In real-time applications new columns are continuously appended to the data matrices $X_{0|i-1}$, $X_{q+i|q+2i-1}$, $X_{q+i+1|q+2i}$ and $Y_{q+i|q+2i-1}$, *e.g.*,

$$X_{0|i-1,[k]} = \left[ \begin{array}{c|c} \alpha \cdot X_{0|i-1,[k-1]} & x_{k|k+i-1} \end{array} \right].$$

In this recursive definition $\alpha \leq 1$ is a exponential weighting factor. The aim is to track the estimates of the TDOAs at each sampling instant.

In the development of Algorithm 7 care has been taken to use only orthogonal operations, such as the LQD (one-sided orthogonal transformations) and the SVD and GSD (two-sided orthogonal transformations). A recursive algorithm can then be derived using LQD updating and SVD/GSD updating as its building blocks. For these operations parallel solutions have already been presented in chapter 2.

The recursive algorithm tracks a slightly different orthogonal decomposition of the data matrices than Algorithm 7.

$$\begin{aligned} X_{0|i-1} &= U \cdot L \cdot Q^T \\ X^p_{q+i|q+2i-1} &= Z \cdot T^0 \cdot Q^T \\ X^p_{q+i+1|q+2i} &= Z \cdot T^1 \cdot Q^T \\ Y^p_{q+i|q+2i-1} &= Z \cdot S \cdot Q^T \end{aligned}$$

where $U, Z \in \mathbb{R}^{Mi \times Mi}$ are orthogonal matrices, $Q \in \mathbb{R}^{j \times Mi}$ has orthonormal columns, $L \in \mathbb{R}^{Mi \times Mi}$ is a lower triangular matrix, and $T^0, T^1, S$ are block diagonal matrices. Each of the last three decompositions is of the form

$$\left[ \begin{array}{c|c} Z_s & Z_n \end{array} \right] \cdot \left[ \begin{array}{c|c} T_s & \\ \hline & T_n \end{array} \right] \cdot \left[ \begin{array}{c} Q_s^T \\ \hline Q_n^T \end{array} \right].$$

The signal block $T_s$ is lower triangular corresponding to the GSD, and the noise block $T_n$ may be a full matrix. Perfect separation between signal and noise subspace implies zero off-diagonal blocks. This separation is pursued by the SVD computation. However, inside the signal subspace, we do not need to compute a basis of singular vectors. Instead we are looking for the orthogonal basis associated with the generalized Schur decomposition of a matrix pair. The SVD and GSD are thus operating on different submatrices and do not interfere with each other. If the row-

and column spaces of the three matrices are not perfectly identical due to noise, small non-zero entries may occur.

It is easily checked that this orthogonal SVD/GSD decomposition results from the combination of steps 3,4 and 5 of Algorithm 7. As an example, the following parts of the decomposition of $X_{q+i|q+2i-1}$ can be identified: $Z_s = U_s \cdot Z$, $Z_n = U_n$, $Q_s = Q \cdot V_s \cdot W$, $Q_n = Q \cdot V_n$, $T_s = T_{q+i|q+2i-1}$ and $T_n = \Sigma_n$.

However, the LQ decomposition of $X_{0|i-1}$ is modified into a three-factor decomposition. Because the growing matrix $Q^T$ accumulating all column rotations is not tracked, it is necessary to apply the SVD/GSD column rotations to the $L$ matrix too. However, this destroys its triangularity. Therefore, an additional orthogonal row operator $U$ is introduced to keep $L$ triangular. It differs from the row operator $Z$ associated with the SVD and GSD.

Updating these two-sided orthogonal decompositions can again be performed as a Jacobi algorithm, specified in Algorithm 8. Its high-level SFG is shown in Figure 7.3. In step 1 the incoming data are multiplied with the matrices $U$ and $Z$ respectively. Step 2 updates the LQD of $X_{0|i-1}$. An orthogonal operator $G_{[k]}$ is computed such that $\tilde{x}_{k-q-i|k-q-1}$ is 'rotated' into the matrix $\alpha R_{[k-1]}$. In order to update the projection this transformation is also applied onto the remaining data. In step 3 sequences of rotations are computed based on $T^0_{[k]}$ to update the SVD/GSD decompositions. Details on how to compute the rotation parameters are given in [131]. The number of applied rotations (here $Mi-1$) is a compromise between accuracy and computational complexity [67]. Finally in step 4 the TDOAs are estimated based on the current diagonal entries.

**Algorithm 8**

$$\begin{aligned}
L_{[q+i-1]} &\longleftarrow O_{Mi \times Mi} \\
T^0_{[q+i-1]} &\longleftarrow O_{Mi \times Mi} \\
T^1_{[q+i-1]} &\longleftarrow O_{Mi \times Mi} \\
S_{[q+i-1]} &\longleftarrow O_{Mi \times Mi} \\
U_{[q+i-1]} &\longleftarrow I_{Mi} \\
Z_{[q+i-1]} &\longleftarrow I_{Mi}
\end{aligned}$$

for $k = q+i, \cdots, \infty$

```
  1. Matrix-vector multiplications

         $\tilde{x}^T_{k-q-i|k-q-1} \longleftarrow x^T_{k-q-i|k-q-1} \cdot U_{[k-1]}$
         $\tilde{x}^T_{k|k+i-1} \longleftarrow x^T_{k|k+i-1} \cdot Z_{[k-1]}$
         $\tilde{x}^T_{k+1|k+i} \longleftarrow x^T_{k+1|k+i} \cdot Z_{[k-1]}$
         $\tilde{y}^T_{k|k+i-1} \longleftarrow y^T_{k|k+i-1} \cdot Z_{[k-1]}$

  2. LQD updating and projection

         $\left[\begin{array}{c|c|c|c} L^T_{[k]} & T^{0^T}_{[k]} & T^{1^T}_{[k]} & S^T_{[k]} \\ \hline O & \star & \star & \star \end{array}\right] \longleftarrow \cdots$

         $\cdots \quad G^T_{[k]} \cdot \left[\begin{array}{c|c|c|c} \alpha L^T_{[k-1]} & \alpha T^{0^T}_{[k-1]} & \alpha T^{1^T}_{[k-1]} & \alpha S^T_{[k-1]} \\ \hline \tilde{x}^T_{k-q-i|k-q-1} & \tilde{x}^T_{k|k+i-1} & \tilde{x}^T_{k+1|k+i} & \tilde{y}^T_{k|k+i-1} \end{array}\right]$

  3. SVD/GSD steps
     for $l = 1, \cdots, Mi-1$

         $L_{[k]} \longleftarrow \Phi^{l|l+1^T}_{[k]} \cdot L_{[k]} \cdot \Theta^{l|l+1}_{[k]}$
         $T^0_{[k]} \longleftarrow \Psi^{l|l+1^T}_{[k]} \cdot T^0_{[k]} \cdot \Theta^{l|l+1}_{[k]}$
         $T^1_{[k]} \longleftarrow \Psi^{l|l+1^T}_{[k]} \cdot T^1_{[k]} \cdot \Theta^{l|l+1}_{[k]}$
         $S_{[k]} \longleftarrow \Psi^{l|l+1^T}_{[k]} \cdot S_{[k]} \cdot \Theta^{l|l+1}_{[k]}$
         $U_{[k]} \longleftarrow U_{[k]} \cdot \Phi^{l|l+1}_{[k]}$
         $Z_{[k]} \longleftarrow Z_{[k]} \cdot \Psi^{l|l+1}_{[k]}$

     endfor

  4. Compute estimates
     for $l = 1, \cdots, n$

         $\tau_l \longleftarrow -T_s \dfrac{\ln T^1_{[k],ll} - \ln T^0_{[k],ll}}{\ln S_{[k],ll} - \ln T^0_{[k],ll}}$

     endfor

endfor
```

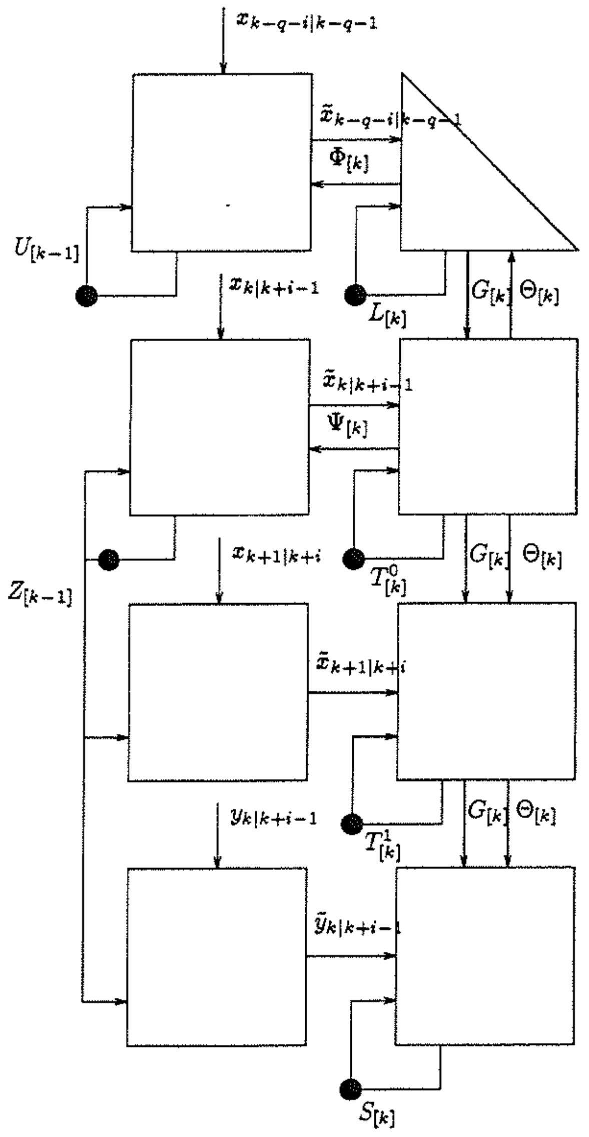


Figure 7.3: High-level SFG of the adaptive wide-band direction finding algorithm. A two-sided decomposition of four matrices is tracked. The upper part tracks a modified LQD decomposition of $X_{0|i-1}$. The lower three parts track a combined SVD/GSD decomposition of $X^p_{q+i|q+2i-1}$,$X^p_{q+i+1|q+2i}$ and $Y^p_{q+i|q+2i-1}$.

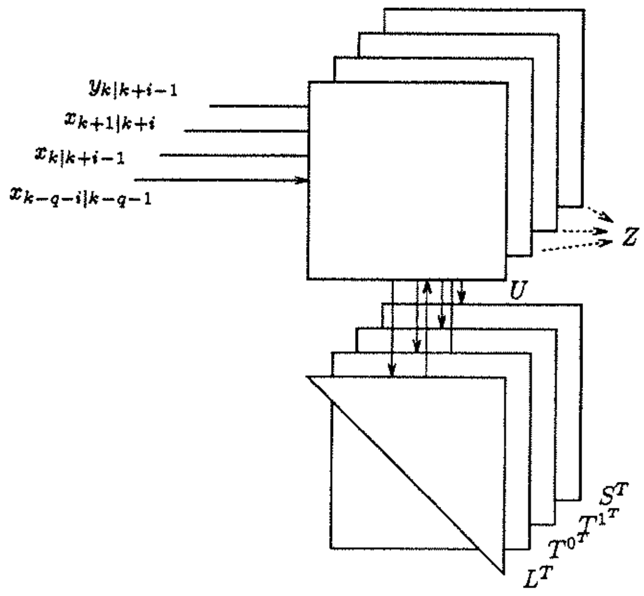


Figure 7.4: Stacked high-level SFG of the adaptive wide-band direction finding algorithm, foreshadowing the implementation on the Jacobi array.

This adaptive wide-band algorithm is again structurally equivalent to the SVD updating algorithm. A 'layered' and transposed SFG is shown in Figure 7.4. Except for the computation of the TDOA estimates, the algorithm involves only matrix-vector multiplications and one- and two-sided elementary rotations. The data dependencies are unaltered. Therefore this adaptive wide-band direction finding algorithm can be mapped efficiently onto the systolic Jacobi array of chapter 2. Again, a limited flexibility of the node hardware is needed. The Jacobi algorithm now simultaneously operates on 4 matrices such that the memory needs are higher. And the way in which the rotation parameters are computed differs.

Of course, in order to avoid numerical error build-up and to implement the algorithm on a rotation array, the $U$ and $Z$ matrices can again be factored as sequences of rotations as proposed in chapter 3.

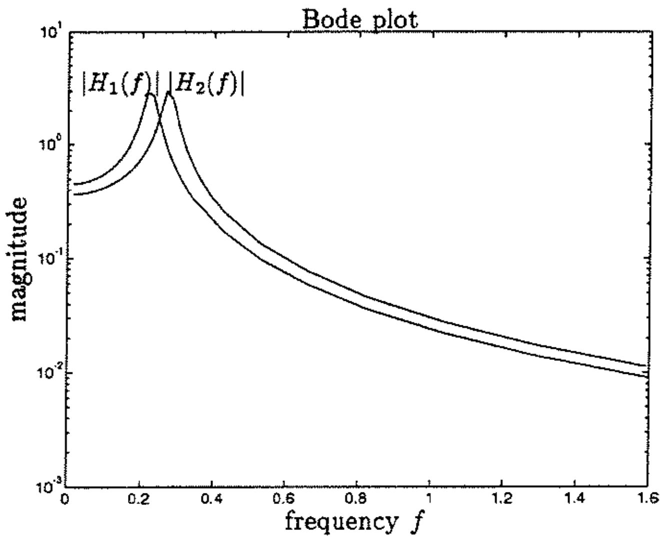


Figure 7.5: The spectra of the two wide-band emitters overlap. The poles are $p_1 = -0.1 \pm j1.4107$ and $p_2 = -0.1 \pm j1.7292$.

## 7.5 Simulations

In the simulations below a uniform linear array (ULA) with $M$ omnidirectional elements is used. Their frequency response is flat (equal to 1) over the whole frequency band of interest. The subarrays are maximally overlapping, *i.e.*, the X-array consists of elements 1 to $M-1$, and the Y-array of elements 2 to $M$. The interelement distance is taken to be $\Delta = c.T_s$, such that all TDOAs are smaller than one sampling period.

We assume two wide-band emitters are present. Their spectra are plotted in Figure 7.5. There is a considerable spectral overlap. The wide-band signals are generated by steering uncorrelated zero-mean Gaussian (almost) white noise with equal power through second-order systems with transfer functions

$$\begin{aligned} H_1(s) &= \frac{0.894}{s^2+0.2s+2} \\ H_2(s) &= \frac{1.095}{s^2+0.2s+3}. \end{aligned}$$

The numerator of both systems is chosen such that $\int_{-\infty}^{\infty} |H(f)|^2 \cdot df = 1$.

The simulation data are computed by sampling the wide-band signals, where per sampling period 10 points are calculated by continuous-time simulation of the two systems. In between these simulated points, linear interpolation is used.

The estimates of the TDOAs are computed as

$$\hat{\tau} = -T_s \cdot \mathrm{Re} \frac{\ln(\hat{\mu})}{\ln(\hat{\lambda})}.$$

Although the TDOAs could be estimated based on the moduli or on the phase of $\hat{\lambda}$ and $\hat{\mu}$, the TDOA estimates using this combined modulus-and-phase estimator turned out to be more accurate.

### 7.5.1 Non-recursive direction finding

In the first set of simulations, various non-recursive algorithms are compared. The first algorithm is the non-recursive state space ESPRIT algorithm of section 7.3.5. The second algorithm is a refined version of this algorithm. Just as for the 2-D harmonic retrieval algorithm of chapter 7, the eigentransformations of both pencils are not exactly the same if noise is present. Therefore, algorithm 2 incorporates the Schur method of [115] which is compatible with the Jacobi architecture. After applying the eigentransformations of the first pencil to the second, this second matrix is not yet fully triangularized. The Schur method further triangularizes the matrix by a few more two-sided rotations with small rotation angle. This last condition is needed in order to keep the same ordering of the eigenvalues in both pencils. The third algorithm is the wide-band Z-domain ESPRIT (Z-ESPRIT) algorithm of Ottersten *et al.* [78], which is discussed in the introduction. The number of time lags used in this algorithm corresponds to the number of block rows in the block Hankel matrix. Both parameters are set to $i = 8$.

The sensor array has $M = 4$ sensors. The number of samples is $n = 500$. In the first simulation the sources have normalized TDOAs $\tau_1 = 0.2, \tau_2 = 0.6$. The sample statistics are given in Figure 7.6. The Z-ESPRIT algorithm is unable to resolve both sources, even at large SNRs. Therefore, a large bias is present. On the contrary, the state space algorithms resolve both sources equally well. No improvement is seen by including the Schur refinement.

In the second simulation both emitters are at the same position, but their spectra are taken identical to $H_1(s)$. They fully overlap. This is a

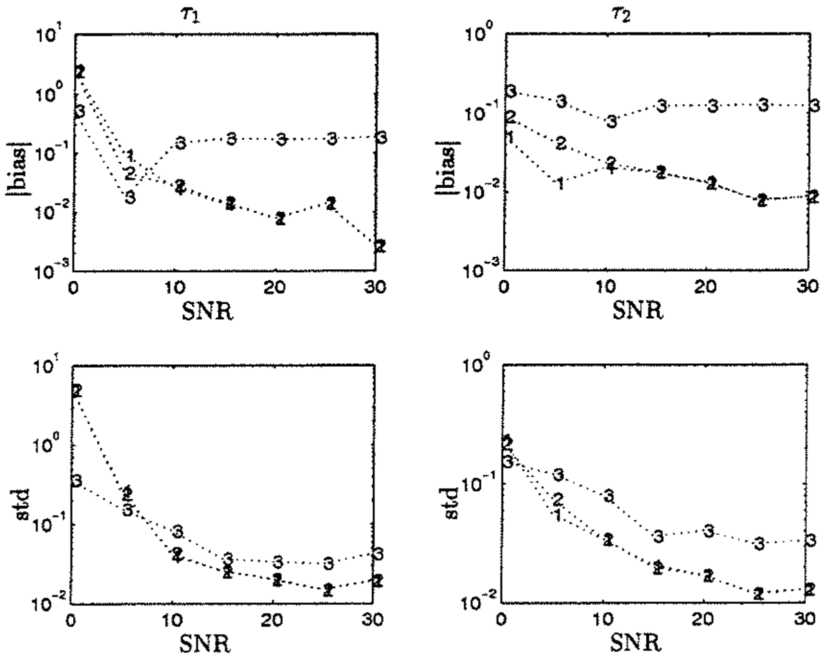


Figure 7.6: Comparison of the estimation accuracy of 3 algorithms. Algorithm 1: state space ESPRIT, Algorithm 2: state space ESPRIT with Schur refinement, Algorithm 3: Z-ESPRIT. Two wide-band sources with the overlapping spectra of Figure 7.5 and TDOAs $\tau = \begin{bmatrix} 0.2 & 0.6 \end{bmatrix}$. Other parameters: $n = 500, i = 8, q = 1$. Averaged over 25 independent runs. The upper (lower) figures show the bias (standard deviation) on $\tau_1$ ($\tau_2$). The Z-ESPRIT method performs worse than the state space algorithms.

wide-band extension of the narrow-band model where multiple narrow-band sources have the same center frequency. Figure 7.7 shows that now the Schur refinement improves the estimates considerably. The reason is that the poles of both sources coincide. Therefore, their eigentransformations are not unique within the associated eigenspace. The Schur algorithm further aligns the eigenvectors such that also the second pencil is triangularized.

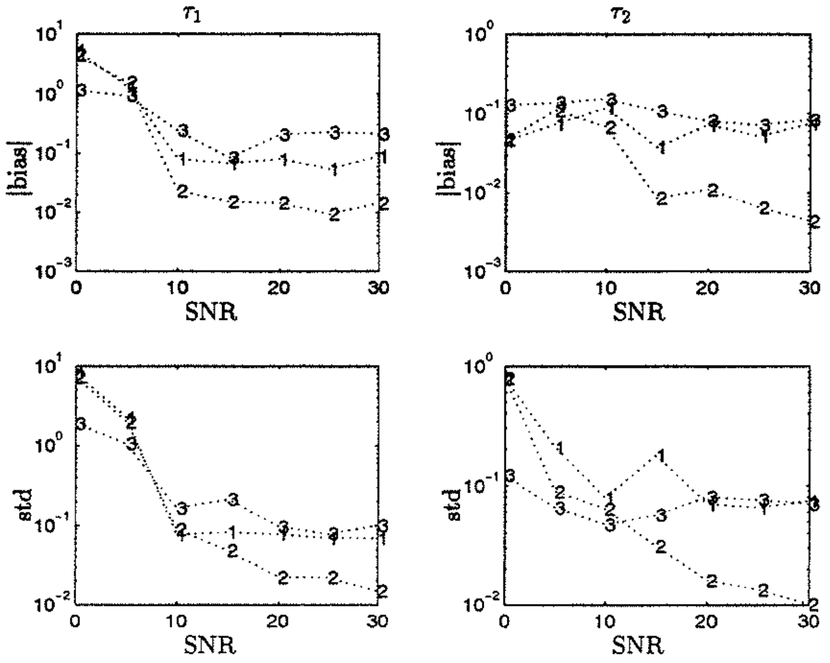


Figure 7.7: Same comparison as Figure 7.6, except that the spectra of the two wide-band sources are identical: $H(s) = \frac{0.894}{s^2+0.2s+3}$. With coinciding poles, the Schur refinement (Algorithm 3) improves substantially upon the state space method (Algorithm 2).

### 7.5.2 Recursive wide-band direction finding

Here we evaluate the adaptive wide-band algorithm of the previous section in a time-varying environment. The wide-band emitters have again the overlapping, not identical spectra. The true evolution of the TDOAs of the two wide-band emitters is shown in Figure 7.8 (dash-dot line). At first the TDOAs are 0.4 and 0.6. However, from time $k = 300$ to $k = 500$ the TDOAs linearly evolve towards their final values 0.8 and 0.2 respectively, corresponding to angles-of-arrival of $\theta_1 = 53.1$ deg and $\theta_2 = 11.5$ deg. The SNR of both wide-band signals is 20 dB.

The parameters of the adaptive algorithm are selected as follows. The number of sensors is taken to be $M = 6$. The number of block rows in the Hankel matrices is $i = 6$ and the delay parameter is $q = 1$. The exponential weighting factor $\alpha$ is 0.992 as a compromise between tracking speed and smoothness of the estimates.

The estimated TDOAs, averaged over 20 independent runs, are given by the dotted lines. In the beginning the algorithm needs some snapshots to converge. From time $k = \pm 80$ on, the separation between signal and noise subspace has been properly accomplished and the estimates converge to a steady state. However a small bias remains visible.

From $k = 300$ on, the algorithm tries to track the change in its environment. It lags behind and evolves exponentially towards the new TDOAs. A smaller weighting factor $\alpha$ would result in faster tracking at the cost of more variance in the estimates during time-invariant periods.

When the two emitters cross, the algorithm does not break down since the system poles are different. The small gap where the curves should cross, is due to the fact that the ordered estimates are averaged. In each individual experiment this gap was absent.

## 7.6 Conclusion

The contribution of this chapter is threefold. First a state-space model for a class of wide-band direction finding problems was presented. The model relies on two assumptions: the sensor array consists of two identical but translated subarrays, and the wide-band signals are generated by a linear system driven by white noise signals.

Secondly it was shown that the data model exhibits a multiple invariance structure. Solving such a structured estimation problem requires in general non-linear optimization techniques. A simple algorithm was presented for estimating the time-differences-of-arrival. The algorithm estimates the generalized eigenvalues of two related matrix pencils by means of numerically robust orthogonal transformations.

Finally an adaptive algorithm was derived. The computational complexity of the algorithm is moderate, namely $\mathcal{O}(M^2 i^2)$. This algorithm is also amenable to parallel implementation because of its fine-grain regular structure. A simulation showed that the algorithm is able to track scenes in which the locations of the emitters are slowly time-varying.

In [126] a related non-recursive subspace algorithm is presented. It treats the outputs of the X- and Y-subarrays symmetrically. The computational complexity of the LQ decomposition is decreased by exploiting the low displacement rank of the block Hankel data matrices [13].

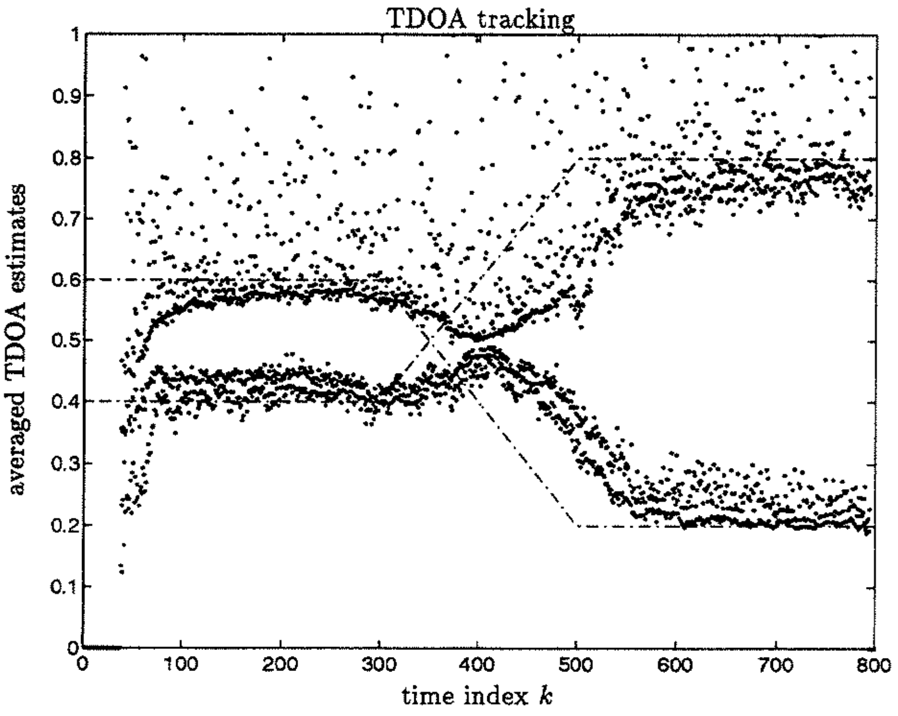


Figure 7.8: Recursive wide-band direction finding algorithm. Two second-order wide-band sources with the overlapping spectra of Figure 7.5. ULA with $M = 6$ sensors. $i = 6, q = 1, \alpha = 0.992$. The dash-dot line shows the exact evolution of the TDOAs. At each time the dots represent the ordered 4 (2 second-order sources) TDOA estimates, averaged over 25 independent runs. The algorithm needs $\pm$ 80 iterations to converge. Due to the exponential weighting the TDOAs estimates lag behind in the time-varying part.

Chapter 8

# A Parametric Approach to Direction Finding in Unknown Ambient Noise Fields

In this last chapter we return to the standard direction finding problem of multiple narrow-band sources with a common carrier frequency. However, we relax the modeling assumption that the measurement noise is spatially white. An accurate noise model is a crucial practical issue. It is well known that the performance of subspace based high resolution direction finding algorithms can be highly sensitive to errors in the noise correlation matrix. In our approach the noise correlation matrix will belong to a certain model set. This model set is specifically tuned to the antenna array under consideration.

The flavor of this chapter is different from the previous ones. We do not look for a recursive algorithm suited for real-time execution on a parallel machine. Instead, we concentrate on modeling the noise and optimally estimating the noise parameters. Algorithms related to maximum likelihood estimation are studied. They will require non-linear optimization.

In section 8.1 various approaches to direction finding in colored noise environments are discussed. Our approach is model based. We develop in section 8.2 a new physical model for the ambient noise impinging on a sensor array and show that the elements of the noise correlation matrix are linear in these parameters. Next, in section 8.3 we concentrate on the identifiability of the noise parameters. In section 8.4 we propose a two-stage subspace algorithm. Unlike true maximum likelihood algorithms, this algorithm has the nice feature that it separates the optimization of signal and noise parameters. First the noise parameters are obtained by a non-linear minimization of a pseudo maximum likelihood criterion. Secondly, the signal parameters are estimated by one of the standard high resolution DF algorithms after prewhitening with the optimal noise correlation matrix. In section 8.5 the performance of this algorithm is studied by simulations.

## 8.1 Introduction

All high resolution direction finding (DF) algorithms, such as MUSIC and ESPRIT, assume that the noise at each sensor is zero-mean, has equal power and is uncorrelated from sensor to sensor. Such noise is called spatially white. Its correlation matrix is the identity matrix multiplied by the (unknown) noise power. The algorithms can be extended to spatially colored noise fields, on condition that the noise correlation matrix is known up to a scalar multiplier. By filtering the data ('prewhitening') with the inverse of the noise correlation matrix, the noise is decorrelated and the high-resolution algorithms can be applied without modification.

The performance of high-resolution direction finding algorithms is sensitive to deviations in the white noise assumption, especially when the signal powers are low. The accuracy of the signal parameters relies on the accuracy of the signal subspace estimate. If there is considerable noise correlation, the estimated subspace may be severely biased. A theoretical analysis of the effect of unmodeled coloring on high-resolution algorithms and the derivation of worst case noise scenarios can be found in [141].

In applications one often faces the problem that the noise field is non-white but unknown. The noise correlation matrix has to be estimated along with the signal parameters. In a limited number of applications, *e.g.*, speech or data communications, the noise correlation matrix can be estimated when the sources are absent. If this is not feasible, one has to estimate signal parameters and noise correlation simultaneously.

A first class of estimation methods is based on higher order statistics (HOS), *e.g.*, [110]. They rely on the assumption that the probability density function of the noise is Gaussian, whereas the statistics of the signals (*e.g.*, communication signals) are non-Gaussian. Since cumulants of order $> 2$ are blind for Gaussian statistics, the estimation of signal parameters based on cumulants is ideally not deteriorated by the noise.

A second class of methods consists of instrumental variable methods [103]. In scenarios where the correlation in time of the signals is much longer than the noise correlation, one can eliminate the noise by projecting new data onto older data. This property has already been used in the wide-band direction finding algorithm in chapter 7.

A third class restricts the noise correlation matrix to a prescribed model set [6, 51]. Indeed, if the noise correlation matrix can be any arbitrary positive Hermitian matrix, then the estimation problem is not uniquely solvable. One could always assume that no signals are present and set the noise correlation matrix equal to the observed correlation matrix. By limiting the number of degrees of freedom of the noise correlation matrix, the direction finding task becomes well-posed again.

In [6, 29, 34] maximum likelihood and least squares algorithms are studied for noise correlation matrices which are linear combinations of a set of known basis matrices. The noise parameters are the unknown coefficients. As an example of a linear parameterization, one can approximate the noise power in function of the angle-of-arrival as a piece-wise constant function. If the array manifold is known, then each basis matrix can be computed corresponding to a particular angular segment of the ambient noise field. This approximation method poses the problem of how to select the angular intervals. An accurate model may require a lot of parameters. On the other hand, this parameterization guarantees each basis matrix to be positive definite, which eases the design of an algorithm.

In [51] the sensor correlation is modeled by a spatial ARMA process. The noise covariance can then be expressed as $R_n = N_{MA} \cdot N_{AR}^{-1} \cdot N_{MA}^H$, where the elements of $N_{MA}$ are linear in the MA coefficients and the elements of $N_{AR}$ are quadratic in the AR coefficients. Although a small number of parameters may suffice to model the noise field, this parameterization has the disadvantage that it is only applicable to regular array geometries.

Our method also belongs to the class of parametric noise models. It is based on the Fourier expansion of the ambient noise field. Each basis matrix then corresponds to a harmonic mode. It has the advantage that it is applicable to arbitrary arrays. The noise parameters have a clear physical meaning and the noise correlation matrix is linear in these parameters.

## 8.2 Modeling the noise field

We consider the data model for $D$ far-field narrow-band sources sharing the same center frequency $f_0$, received by a sensor array of $M$ elements

$$x_{[k]} = A_\theta \cdot s_{[k]} + n_{[k]}.$$

The notation $A_\theta$ is used to stress the dependence of the array gain matrix on the DOAs. The noise vector is denoted as $n_{[k]}$ to stress that the noise is not necessarily white. We adopt a stochastic model for the base-band signals, *i.e.*, they are assumed to be temporally white, zero-mean, Gaussian signals with covariance matrix $R_s \in \mathbb{C}^{D\times D}$

$$\begin{aligned} E\{s_{[k]} \cdot s_{[l]}^H\} &= R_s \cdot \delta(t_k - t_l) \\ E\{s_{[k]} \cdot s_{[l]}^T\} &= O_{D\times D}. \end{aligned}$$

The measurement noise is also assumed to be a zero-mean temporally-white complex Gaussian process with covariance matrix $\gamma R_n \in \mathbb{C}^{M\times M}$ where $R_n$ is a normalized noise covariance matrix (*e.g.*, $\mathrm{tr}(R_n) = M$)

$$\begin{aligned} E\{n_{[k]} \cdot n_{[l]}^H\} &= \gamma R_n \cdot \delta(t_k - t_l) \\ E\{n_{[k]} \cdot n_{[l]}^T\} &= O_{M\times M}. \end{aligned}$$

The output covariance matrix for zero-lag is then given by

$$R_x = E\{x_{[k]} \cdot x_{[k]}^H\} = A_\theta \cdot R_s \cdot A_\theta^H + \gamma R_n. \tag{8.1}$$

We now detail this expression by including a physical model for the noise. There are two different noise phenomena. A first kind of noise is generated internally in the sensors, *e.g.*, thermal noise in the receivers. Since the physical noise processes in the sensors are independent, it is reasonable to model the noise correlation matrix for this type of noise and identical sensors as

$$R_n^{(1)} = \gamma I_M$$

where $\gamma$ is the noise power.

A second source is the noise which is received by the sensor array. In all environments, background noise in the frequency band of interest is unavoidable. The assumption that this ambient noise is uncorrelated from sensor to sensor is hard to defend. An example is a base station in a mobile communications system, placed at a city border. It is evident that the urban district generates more noise than the rural area. This spatial dependence causes noise correlation from sensor to sensor. Below we derive a model for this ambient noise.

Assume that one can associate with each angle-of-incidence an infinitesimal noise source in the far field of the array with angle-dependent noise power $p(\theta)$. The ambient noise correlation matrix is then given by the integral

$$R_n^{(2)} = \int_0^{2\pi} a(\theta) \cdot p(\theta) \cdot a(\theta)^H \cdot d\theta. \tag{8.2}$$

In general $p(\theta)$ will not be constant, but will smoothly depend on $\theta$. This assumption of smoothness of the noise field is not restrictive. Sharp changes in the noise field are due to strongly localized noise sources, for which the distinction with a point source (a signal or interference) becomes vague.

Since $p(\theta)$ is a real periodic function, a real Fourier expansion exists and is given by

$$p(\theta) = u_0 + \sum_{l=1}^{\infty}(u_l \cos(l\theta) + v_l \sin(l\theta)).$$

Depending on how smooth $p(\theta)$ behaves, the coefficients of the higher harmonic terms will rapidly vanish. By truncating the Fourier expansion to the first $Q$ terms, $Q$ real Fourier coefficients characterize the noise field.

When substituting this Fourier expansion for $p(\theta)$ in Eq. (8.2), we obtain a linear parameterization of the ambient noise correlation matrix

$$R_n = \sum_{l=0}^{Q-1} \alpha_l R_n^l$$

where $\alpha_{2l} = u_l, \alpha_{2l-1} = v_l$ and each basis matrix $R_n^l$ represents the noise covariance matrix generated by a pure harmonic noise field

$$\begin{aligned} R_n^{2l} &= \int_0^{2\pi} a(\theta) \cdot \cos(l\theta) \cdot a(\theta)^H \cdot d\theta \\ R_n^{2l-1} &= \int_0^{2\pi} a(\theta) \cdot \sin(l\theta) \cdot a(\theta)^H \cdot d\theta. \end{aligned}$$

### Example: Uniform Linear Array

The $l$th basis matrix $R_n^l$ is function of the array manifold only. If an analytic expression for the array manifold is available, then entry $(p, q)$ can be found by integration over the range of angles. The expression for a ULA is derived below. Define $x = \frac{2\pi\Delta}{\lambda}$ where $\Delta$ is the distance between adjacent sensors and $\lambda$ is the wavelength of the carrier wave.

$$\begin{aligned}
& R_n^{2l}(p,q) + jR_n^{2l-1}(p,q) = \cdots \\
= & \int_0^{2\pi} \underbrace{\exp(-j(p-1)x\sin(\theta))}_{a(\theta)_p} \cdot \underbrace{\exp(jl\theta)}_{l\text{th mode}} \cdot \underbrace{\exp(j(q-1)x\sin(\theta))}_{a^*(\theta)_q} d\theta \\
= & \int_0^{2\pi} \exp(j(l\theta - (p-q)x\sin(\theta)))d\theta \\
= & 2\int_0^{\pi} \cos(l\theta - (p-q)x\sin(\theta))d\theta + j0 \\
= & 2\pi(\operatorname{sign}(p-q))^l J_l(|p-q|x) \qquad (8.3)
\end{aligned}$$

where $J_l(x)$ denotes the Bessel function of the first kind and order $l$. Here we used the identity $J_{-l}(x) = (-1)^l J_l(x)$ and the Bessel integral formula for integer $l$ and positive $x$

$$J_l(x) = \frac{1}{\pi}\int_0^{\pi} \cos(l\theta - x\sin(\theta))d\theta.$$

As an example, consider a 3-element ULA with $\Delta = \lambda/2$. The first two non-zero basis matrices are given by

$$\begin{aligned}
R_n^0 &= \begin{bmatrix} J_0(0) & J_0(\pi) & J_0(2\pi) \\ J_0(\pi) & J_0(0) & J_0(\pi) \\ J_0(2\pi) & J_0(\pi) & J_0(0) \end{bmatrix} = \begin{bmatrix} 1.00 & -0.30 & 0.22 \\ -0.30 & 1.00 & -0.30 \\ 0.22 & -0.30 & 1.00 \end{bmatrix} \\
R_n^1 &= j\begin{bmatrix} J_1(0) & J_1(\pi) & J_1(2\pi) \\ -J_1(\pi) & J_1(0) & J_1(\pi) \\ -J_1(2\pi) & -J_1(\pi) & J_1(0) \end{bmatrix} = j\begin{bmatrix} 0.00 & 0.28 & -0.21 \\ -0.28 & 0.00 & 0.28 \\ 0.21 & -0.28 & 0.00 \end{bmatrix}
\end{aligned}$$

Here we have left out the scaling factor $2\pi$ since it can be incorporated into the Fourier coefficients. Because of the shift invariance in a ULA the correlation matrices are Toeplitz. If $l$ is even, $R_n^{2l}$ is real symmetric and $R_n^{2l-1} = O_{M\times M}$. If $l$ is odd, $R_n^{2l} = O_{M\times M}$ and $R_n^{2l-1}$ is imaginary anti-symmetric. Only $R_n^0$ corresponding to the average noise power, is a positive definite matrix. All harmonic basis matrices have trace zero and are therefore indefinite. The $R_n^0$ matrix is not a scalar multiple of the identity matrix. Its associated eigenspectrum is plotted in Figure 8.1 for $M = 8$. The eigenvalue spread is $\lambda_{\max}/\lambda_{\min} = 3.83$. Therefore even a signal observed in a constant cylindrical ambient noise field must be prewhitened.

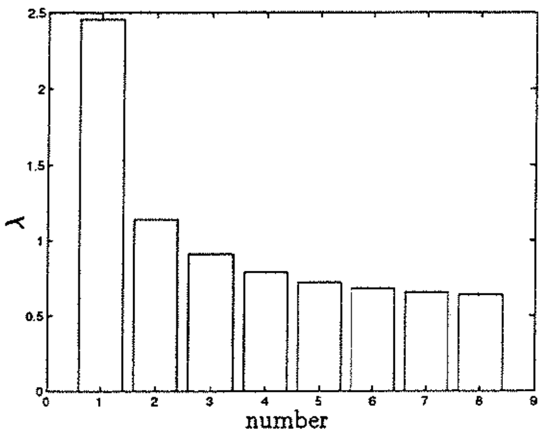


Figure 8.1: Eigenspectrum for the noise correlation matrix $R_n^0$ induced by a cylindrically isotropic ambient noise field on an 8-element ULA with $\Delta = \lambda/2$.

## 8.3 Identifiability

The class of models we consider for the covariance matrix $R_x$ is given by Eq. (8.1). The output covariance matrix depends on the DOAs $\{\theta_i, i = 1, \cdots, D\}$, the source covariance matrix $R_s$ ($D^2$ real unknowns) and the Fourier coefficients $\{\alpha_l, 0 \leq l \leq Q-1\}$. An important question is the uniqueness of the map from parameters to output covariance matrix. In other words, is there more than a single set of parameters which gives the same $R_x$? If the answer is affirmative, it is impossible to find unique estimates for the noise parameters and the DOAs for this $R_x$. In the case of white noise, it is well-known that the DOA parameters and the signal correlation matrix are unique on condition that the array manifold is unambiguous [80]. Now, we also establish some conditions for the number of noise parameters $Q$.

A first natural bound follows from the linear independence of the noise correlation basis matrices. They are Hermitian matrices living in $\mathbb{C}^{M\times M}$. The dimensionality of this matrix space is $M^2$. It never makes sense to include basis matrices $R_n^l$ which are linear combinations of other basis matrices. Therefore, although the Fourier expansion of the ambient noise power is an infinite series, a first upper bound on $Q$ is the number of degrees of freedom of the matrix space. This upper bound may be smaller than $M^2$. For example, due to the regularity in a uniform linear array, all matrices are Hermitian Toeplitz matrices. Their number of degrees of freedom is only $2M-1$.

A second natural bound follows from counting the number of equations and unknowns. For a given $R_x$, Eq. (8.1) is a set of $M^2$ non-linear continuous scalar equations in $D(D+1)+Q$ unknowns. If the number of unknowns surpasses the number of equations, then generically there is an infinity of solutions defined by an (implicit) function. If this is the case, there is not even local identifiability. A local neighborhood of a solution contains other solutions. Therefore, the Fisher information matrix at a solution point becomes singular. In order to guarantee local identifiability, we have to impose a second condition

$$Q \leq M^2 - D(D+1).$$

These two conditions are necessary, but not sufficient. Even if $Q$ is (much) smaller than the previous two bounds, a finite number of isolated solutions may exist. To clarify this point, Eq. (8.1) is rewritten as

$$R_x = S(\alpha) + R_n(\alpha)$$

where $S(\alpha)$ is the signal covariance matrix induced at the antenna array outputs and $R_n(\alpha)$ is the noise correlation matrix.

$$\begin{aligned} S(\alpha) &= R_x - \sum_{l=0}^{Q-1} \alpha_l R_n^l \geq 0 \\ R_n(\alpha) &= \sum_{l=0}^{Q-1} \alpha_l R_n^l \geq 0 \end{aligned}$$

Both matrix expressions are affine in the noise parameters and have to be positive semi-definite to be physically meaningful. Such expressions are called linear matrix inequalities (LMIs) [7, 122]. The parameter vectors $\alpha \in \mathbb{R}^Q$ satisfying such an LMI, form a convex set. This is illustrated in the small example $(M=3, D=1, Q=2)$ of Figure 8.2. The full curves in the figure indicate the points $\alpha$ for which the signal covariance matrix $S(\alpha)$ is rank deficient. They are the solutions to

$$\det(S(\alpha)) = \det\left(\begin{bmatrix} 3-\alpha_1 & 2-\alpha_2 & 1 \\ 2-\alpha_2 & 3-\alpha_1 & 2-\alpha_2 \\ 1 & 2-\alpha_2 & 3-\alpha_1 \end{bmatrix}\right) = 0$$

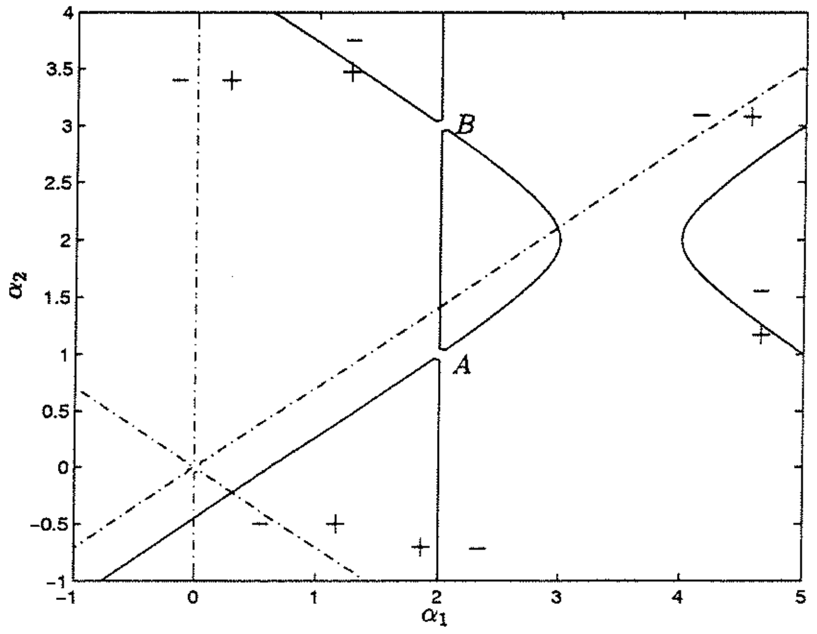


Figure 8.2: Contour lines of zero determinant for $S(\alpha) = R_x - \alpha_1 R_n^1 - \alpha_2 R_n^2$ (full line) and $R_n(\alpha) = \alpha_1 R_n^1 + \alpha_2 R_n^2$ (dash-dot line) where $R_n^1 = I_3, R_n^2 =$ Toeplitz([0, 1, 0]). There are two rank-1 matrices $S(\alpha)$ at points A and B. Solution A: $\theta = 0$ deg, $\alpha_1 = 2, \alpha_2 = 1, R_s = 1$. Solution B: $\theta = 90$ deg, $\alpha_0 = 2, \alpha_1 = 3, R_s = 1$. However, solution $B$ violates the positivity constraint on $R_n(\alpha)$ (signature {++-}).

and are given by the equations

$$\begin{aligned} \alpha_1 &= 2 \\ (\alpha_1 - 3.5)^2 - 2(\alpha_2 - 2)^2 &= 0.25 \end{aligned}$$

Similarly, the dash-dot curves represent the solutions to $\det(N(\alpha)) = 0$.

$$\begin{aligned} \alpha_1 &= 0 \\ \alpha_1 &= \pm\sqrt{2}\alpha_2 \end{aligned}$$

All points on the boundary of the convex set of positive semi-definite matrices $S(\alpha)$ represent a decomposition of $R_x$ in a sum of a rank-2 matrix

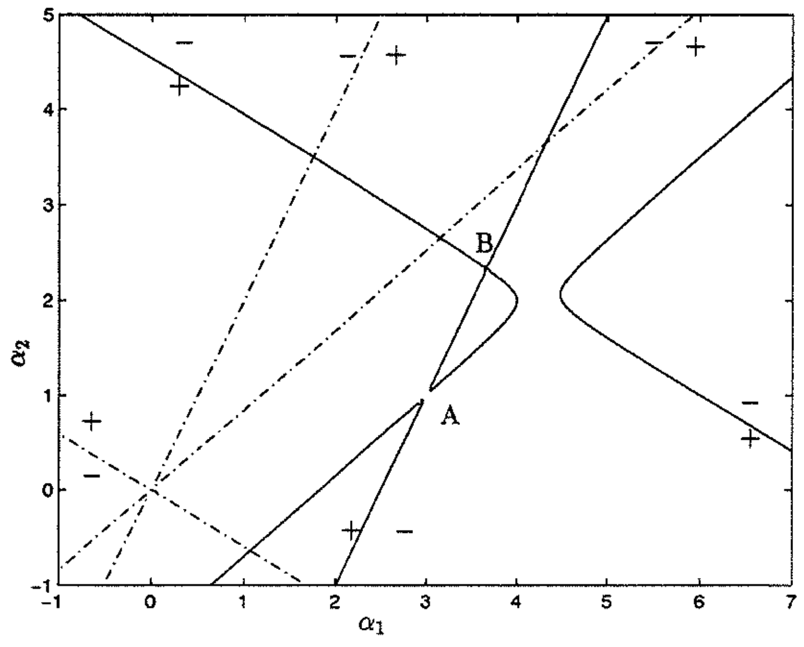


Figure 8.3: Contour lines of zero determinant for $S(\alpha)$ (full line) and $R_n(\alpha)$ (dash-dot line). ULA with $M = 3, \Delta = \lambda/2$. $R_n^1 = I_3, R_n^2 =$ Toeplitz([0, 1, 0.5]). There are two valid parameterizations for $R_x$. Solution A: $\theta = 0$ deg, $\alpha_1 = 3, \alpha_2 = 1, R_s = 1$. Solution B has parameters $\theta = 90$ deg, $\alpha_1 = 11/3, \alpha_2 = 7/3, R_s = 1/3$.

and a full-rank noise matrix. When two curves intersect, two eigenvalues are zero. In general, the multiplicity of the zero eigenvalue is given by the number of curves intersecting at that point. In this example, there are only two points ($A$ and $B$) which yield a decomposition of $R_x$ as a rank-1 matrix and a noise matrix. Point $A$ corresponds to the correct decomposition. The vector corresponding to point $B$ happens to be a valid array manifold vector for $\theta = 90$ deg. In this example point $B$ can be eliminated since it violates the condition $N(\alpha) \geq 0$. However, as shown in Figure 8.3 this is not necessarily the case.

In general, finding a sufficient bound for $Q$, given $M, D, \theta$, such that there exists only one decomposition for $R_x$ is a hard problem. David [16] proofs the following theorem.

*Given two real symmetric $M \times M$ LMIs $S(\alpha) \geq 0$ and $N(\alpha) \geq 0$ and an integer $D < M$. Checking if there exists an $\alpha$ such that rank $S(\alpha) \leq D$ and $S(\alpha) \geq 0, N(\alpha) \geq 0$, is an NP-complete problem.*

Our problem is more general, since the LMIs contain Hermitian matrices. Because real symmetric LMIs are a subset of Hermitian LMIs, the corresponding problem for complex LMIs is also NP-complete. However, this theorem only proofs that there exist choices of $S(\alpha), N(\alpha)$ for which checking the existence of a solution is an NP-complete problem. It does not proof that this holds true for each specific choice of the basis matrices. In our case, we also know by construction that there exists one solution. We are interested in the existence of a solution which differs from the known one.

In any case, finding a sufficient condition for $Q$ such that the model is globally identifiable, seems to be out of reach. In practice, in the many examples studied during simulations with moderate values for $Q$, no identifiability problems were observed.

## 8.4 Maximum likelihood and subspace direction finding

In this section we investigate two approaches for estimating the signal- and noise parameters. The first approach is optimal maximum likelihood estimation. The second one is a subspace approach which is suboptimal but simplifies the non-linear optimization.

The maximum likelihood approach for direction finding in the case of white noise has been studied extensively [80, 94, 140]. It is well known that the optimization of the linear parameters $R_s$ and $\gamma$ in Eq. (8.1) can be separated from the optimization of the DOA parameters. For a given $\theta$ the maximum likelihood values of $\gamma$ and $R_s$ are given by

$$\tilde{\gamma}_\theta = \frac{1}{M-D}\mathrm{tr}(P_{A_\theta}^{\perp} \cdot \hat{R}_x) \tag{8.4}$$

$$\tilde{R}_{s,\theta} = A_\theta^{\dagger} \cdot (\hat{R}_x - \tilde{\gamma}_\theta I_M) \cdot A_\theta^{\dagger H} \tag{8.5}$$

where the projector matrix is $P_{A_\theta}^{\perp} = I_M - A_\theta \cdot \left(A_\theta^H \cdot A_\theta\right)^{-1} \cdot A_\theta^H$. A multi-dimensional non-linear search over a concentrated object function remains

$$\min_\theta V(\theta) = \ln\det(A_\theta \cdot \tilde{R}_{s,\theta} \cdot A_\theta^H + \tilde{\gamma}_\theta I_M).$$

The minimization of this likelihood function is difficult and has to be done numerically. However, the resulting estimates may be expected to be excellent. Asymptotically the maximum likelihood estimator is known to achieve the Cramér-Rao bound [80].

Maximum likelihood estimation in the colored noise case grows even more complicated. For a given noise correlation matrix $R_n = R_n^{H/2} \cdot R_n^{1/2}$ known up to a scalar factor $\gamma$, Eqs. (8.4,8.5) still hold after replacing $R_x$ and $\hat{R}_x$ by their prewhitened counterparts

$$\begin{aligned} R_w &= R_n^{-H/2} \cdot R_x \cdot R_n^{-1/2} \\ \hat{R}_w &= R_n^{-H/2} \cdot \hat{R}_x \cdot R_n^{-1/2}. \end{aligned}$$

The non-linear optimization now extends over the signal parameters $\theta$ as well as the noise parameters $\alpha$. The cost of the multi-dimensional optimization rapidly becomes prohibitive.

Therefore, we look for a subspace based algorithm which first optimizes the noise parameters and secondly estimates the signal parameters. The underlying idea is as follows. Suppose that the noise parameters were known, then for large sample amounts the dominant $D$-dimensional eigenspace of the prewhitened sample covariance matrix is a consistent estimator for the true signal subspace. This subspace estimate is the optimal solution to a slightly different maximum likelihood problem (called 'pseudo ML' in [153]). The model class now considered is

$$R_x = F \cdot F^H + \gamma(R_n^0 + \sum_{l=1}^{Q-1} \alpha_l R_n^l)$$

where the matrix $F \in \mathbb{C}^{M \times D}$ is an arbitrary rank $D$ matrix. Knowledge of the array manifold is at first not included. The estimated subspace converges to the exact subspace for large amounts of snapshots. Therefore, the signal parameters can be derived by looking for the array manifold vectors which are closest to the estimated subspace. This subspace approach has the important advantage that the optimization can now be separated. First an optimal estimate for the noise correlation matrix is determined. Then after prewhitening the signal parameters are estimated by a standard high resolution algorithm such as MUSIC, ESPRIT or WSF.

The same approach has been taken by Le Cadre [51] where an ARMA parameterization for the noise is used. Below we first rederive the optimization criterion which is only a function of the noise parameters. We discuss its physical meaning and analyze its shape. Finally we propose a gradient algorithm.

The conditional probability density of the output data matrix $X$ is given by

$$p(X|R_x) = \pi^{-MN} \det(R_x)^{-N} \exp(-N\text{tr}(R_x^{-1}\hat{R}_x))$$

where $N$ is the number of snapshots and $\hat{R}_x = \frac{1}{N}\sum_{k=1}^{N} x_{[k]}x_{[k]}^H$ is the sample covariance matrix. The maximum likelihood estimator maximizes this conditional probability function with respect to the unknown parameters of $R_x$. Equivalently, after omission of terms independent of $R_x$ and scaling factors, one can minimize the negative log-likelihood function

$$l(R_x) = \ln\det(R_x) + \text{tr}(R_x^{-1}\hat{R}_x).$$

Using the properties that

$$\begin{aligned} \det R_x &= \det(R_n^{H/2} \cdot R_w \cdot R_n^{1/2}) = \det(R_w) \cdot \det(R_n) \\ \text{tr}(R_x^{-1} \cdot \hat{R}_x) &= \text{tr}(R_w^{-1} \cdot \hat{R}_w), \end{aligned}$$

we can express $l(R_x)$ in function of the whitened observations

$$l(R_x) = \ln\det(R_w) + \text{tr}(R_w^{-1}\hat{R}_w) + \ln\det(R_n). \qquad (8.6)$$

Only the first two terms depend on the unknowns $F$ and $\gamma$. If we denote the partitioned eigendecomposition of the prewhitened sample correlation matrix $\hat{R}_w$ by

$$\hat{R}_w = \left[\begin{array}{c|c} \hat{Q}_s & \hat{Q}_n \end{array}\right] \cdot \left[\begin{array}{c|c} \hat{\Lambda}_s & \\ \hline & \hat{\Lambda}_n \end{array}\right] \cdot \left[\begin{array}{c|c} \hat{Q}_s & \hat{Q}_n \end{array}\right]^H$$

where the eigenvalues are sorted in descending order, then the optimal expressions for $F_w = R_n^{-H/2}F$ and $\gamma$ are

$$\begin{aligned} \tilde{F}_w \cdot \tilde{F}_w^H &= \hat{Q}_s \cdot \left(\hat{\Lambda}_s - \tilde{\gamma} I_D\right) \cdot \hat{Q}_s^H \\ \tilde{\gamma} &= m_\alpha = \frac{1}{M-D} \sum_{i=D+1}^{M} \hat{\lambda}_i. \end{aligned}$$

The optimal estimator for $\gamma$ is the arithmetic mean of the $M-D$ smallest eigenvalues of the prewhitened sample correlation matrix. Substituting these optimal expressions in Eq. (8.6) yields

$$\begin{aligned}
\ln\det R_w &= (M-D)\ln m_\alpha + \ln\prod_{i=1}^{D}\hat{\lambda}_i \\
&= (M-D)(\ln m_\alpha - \ln g_\alpha) + \ln\det\hat{R}_w \\
\operatorname{tr}(\hat{R}_w\cdot R_w^{-1}) &= M
\end{aligned}$$

where $g_\alpha$ is the geometric mean of the $M-D$ smallest eigenvalues of the prewhitened sample correlation matrix

$$g_\alpha = \left(\prod_{i=D+1}^{M}\hat{\lambda}_i\right)^{\frac{1}{M-D}}.$$

Therefore the optimal value of $l(R_x)$ is

$$\begin{aligned}
\min_{F,\gamma} l(R_x) &= (M-D)\ln\frac{m_\alpha}{g_\alpha} + \ln\det\hat{R}_w + \ln\det R_n + M \\
&= (M-D)\ln\frac{m_\alpha}{g_\alpha} + \ln\det\hat{R}_x + M.
\end{aligned}$$

The last terms are clearly independent of the noise parameters $\alpha$, such that our final object function for the noise parameters is

$$V(\alpha) = (M-D)\ln\frac{m_\alpha}{g_\alpha}.$$

**Property 1** *The function $V(\alpha)$ is only defined over the set of positive definite matrices $R_n$.*

The $\hat{\lambda}_i$s are the eigenvalues of the prewhitened matrix $\hat{R}_w$ or equivalently $\hat{R}_x \cdot R_n^{-1}$. Since $R_n$ is the correlation matrix of the ambient noise field, it should be positive definite. The function $V(\alpha)$ acts itself as a barrier function for the convex set $R_n(\alpha) \geq 0$. It becomes infinite as an eigenvalue of $R_n$ approaches zero.

**Property 2** *For positive eigenvalues $V(\alpha)$ is always positive and equal to 0 if and only if all eigenvalues are identical.*

This property establishes a nice physical meaning of the ratio of arithmetic and geometric mean. It is a measure of the equality of the smallest $M-D$ eigenvalues and thus of the quality of the noise correlation matrix estimate. Therefore, it is sometimes used as a test statistic to determine the number of sources in a white noise environment [153]. This so-called sphericity test is also part of the more popular AIC and MDL criteria.

**Proof**

The extrema of $V(\hat{\lambda}_{D+1}, \cdots, \hat{\lambda}_M)$ are given by calculating the zeros of its gradient.

$$\begin{aligned}(\nabla_{\hat{\lambda}} V)_i = \frac{\partial V}{\partial \hat{\lambda}_i} &= (M-D)\left(\frac{1}{m_\alpha}\frac{\partial m_\alpha}{\partial \hat{\lambda}_i} - \frac{1}{g_\alpha}\frac{\partial g_\alpha}{\partial \hat{\lambda}_i}\right) \\ &= \frac{1}{m_\alpha} - \frac{1}{\hat{\lambda}_i}\end{aligned}$$

Therefore, the unique extremum is attained if all eigenvalues are equal

$$\hat{\lambda}_i = m_\alpha, \qquad i = D+1, \cdots, M.$$

The Hessian at the extremum is

$$H = \frac{1}{m_\alpha^2}\left(I_{M-D} - \frac{1}{M-D} 1_{(M-D)\times(M-D)}\right)$$

where $1_{k\times l}$ is a matrix consisting of ones. The Hessian has $M-D-1$ eigenvalues 1, and one eigenvalue 0. The extremum is therefore a minimum. The zero eigenvalue is due to the fact that $V$ is invariant to a scaling of all eigenvalues. This scale-invariance is eliminated by fixing one coefficient (*e.g.*, the first) of the noise correlation matrix, *i.e.*, $R_n = R_n^0 + \sum_{l=1}^{Q-1} \alpha_l R_n^l$. ■

The above property shows that the minimum of the object function in function of the eigenvalues is unique. Unfortunately, we cannot directly control the eigenvalues. They are functions of the Fourier coefficients of the ambient noise field. The behavior of the object function in terms of these parameters is much more complex.

**Property 3** *The function $V(\alpha)$ is the minimum of a set of branch functions which are analytic in the $\alpha_l$s.*

A well known result from matrix perturbation theory states that the eigenvalues of a matrix which evolves as an analytic function of some parameter are analytic functions on condition that the eigenvalues are unordered [9, 19, 47]. The object function $V(\alpha)$ only uses a subset of these analytic eigenvalue functions, *i.e.*, the $M-D$ smallest ones. However, these analytic eigenvalue functions may cross each other. In general we can express $V(\alpha)$ as the lower hull of a set of $\binom{M}{M-D}$ analytic functions $V(\hat{\lambda}_{i_1}^a, \cdots, \hat{\lambda}_{i_{M-D}}^a)$ where $\hat{\lambda}_i^a$ is the $i$th analytic eigenvalue and

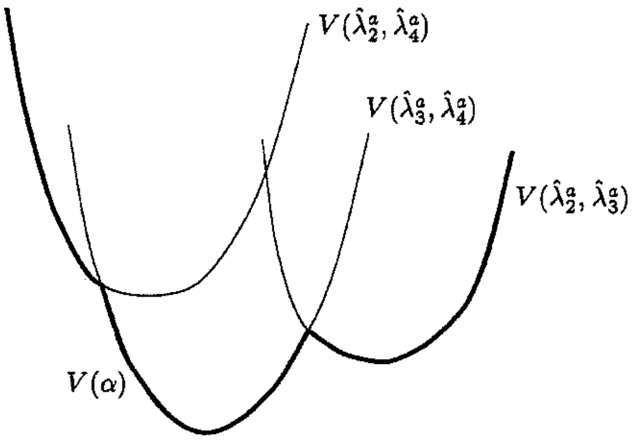
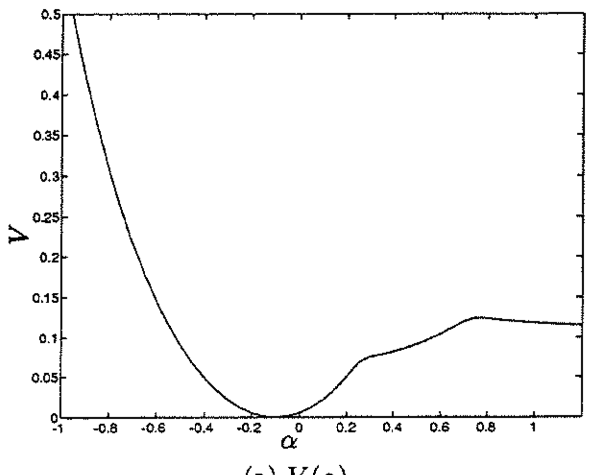

$V(\hat\lambda_2^a, \hat\lambda_4^a)$
$V(\hat\lambda_3^a, \hat\lambda_4^a)$
$V(\hat\lambda_2^a, \hat\lambda_3^a)$
$V(\alpha)$

Figure 8.4: Evolution of $V(\alpha)$ (thick line) when the smallest analytic signal eigenvalue and the largest analytic noise eigenvalue cross ($M = 4, D = 2$). Three out of the six possible branches are shown (thin lines). Local minima in $V(\alpha)$ may arise when the derivatives of the crossing branches have an opposite sign at an intersection.

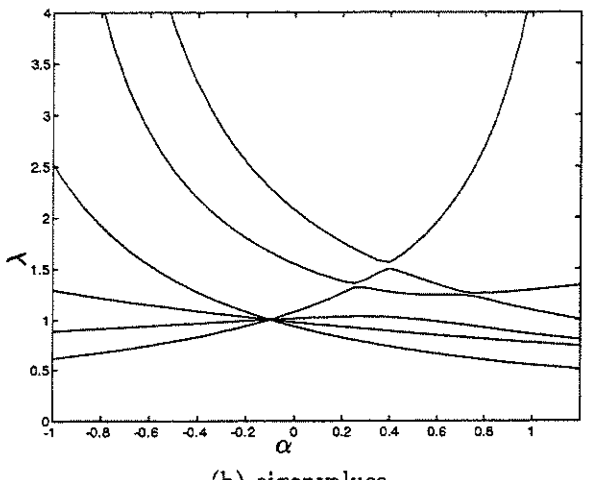
$\{i_1, \cdots, i_{M-D}\}$ is a set of indices taken from $\{1, \cdots, M\}$. This is illustrated in Figure 8.4 for $M = 4, D = 2$. Although each branch can behave nicely, *e.g.*, in the figure the branch functions are drawn as convex functions, the lower hull $V(\alpha)$ may exhibit local minima. In practice the behavior is still more complicated. In Figure 8.5 a small example for a single noise parameter is presented. The upper part shows the evolution of $V(\alpha)$. The two bends in the curve remind of the non-differentiable points caused by crossing eigenvalues in Figure 8.4. Here they are smoothened. This can be explained by the evolution of the eigenvalues shown in the lower part of Figure 8.5. The two bends are exactly located where the smallest signal and the largest noise eigenvalue tend to intersect. However, the approaching eigenvalues repel each other and smoothly diverge again. Instead of intersecting, the ordered eigenvalue curves exhibit local extrema.

**Property 4** *The gradient $\nabla_\alpha V$ is given by*

$$(\nabla_\alpha V)_l = \mathrm{tr}\left\{\left(I_{M-D} - \frac{1}{m_\alpha}\hat\Lambda_n\right)\cdot \hat Q_n^H \cdot \left(R_n^{-H/2}\cdot R_n^l \cdot R_n^{-1/2}\right)\cdot \hat Q_n\right\}.$$



(a) $V(\alpha)$



(b) eigenvalues

Figure 8.5: Evolution of $V(\alpha)$ and the eigenvalues $\hat{\lambda}$ for a ULA with $M = 6, \Delta = \lambda/2$ and $D = 2$ uncorrelated sources with DOAs $(0, 15)$ deg and normalized power $p = 0.1$. The noise correlation matrix is $R_n = R_n^0 + \alpha R_n^2$ with $R_n^l$ given by Eq. 8.3 and $\alpha = -0.1$.

**Proof**

The gradient of $V(\alpha)$ with respect to the parameters is given by

$$(\nabla_\alpha V)_l = \frac{\partial V}{\partial \alpha_l} = \sum_{i=D+1}^{M} \frac{\partial V}{\partial \hat{\lambda}_i} \cdot \frac{\partial \hat{\lambda}_i}{\partial \alpha_l}.$$

or

$$\nabla_\alpha V = J_\alpha^{\hat{\lambda}} \cdot \nabla_{\hat{\lambda}} V$$

where the Jacobian matrix $J_\alpha^{\hat{\lambda}} \in \mathbb{R}^{(Q-1)\times(M-D)}$ is defined as

$$(J_\alpha^{\hat{\lambda}})_{li} = \frac{\partial \hat{\lambda}_{D+i}}{\partial \alpha_l}.$$

The $\hat{\lambda}_i$s are the smaller eigenvalues of the prewhitened sample correlation matrix $\hat{R}_w$. From perturbation theory for Hermitian matrices it is known that for simple eigenvalues

$$\frac{\partial \hat{\lambda}_i}{\partial \alpha_l} = \hat{q}_i^H \cdot \frac{\partial \hat{R}_w}{\partial \alpha_l} \cdot \hat{q}_i.$$

Starting from the identity $R_n^{-H/2} \cdot R_n \cdot R_n^{-1/2} = I_M$ and the fact the square root $R_n^{1/2}$ is an upper triangular Cholesky factor, it is readily derived that

$$\frac{\partial R_n^{-1/2}}{\partial \alpha_l} = -R_n^{-1/2} \cdot \text{upph}(R_n^{-H/2} \cdot \frac{\partial R_n}{\partial \alpha_l} \cdot R_n^{-1/2})$$

where the operator upph$(A)$ takes the upper triangular part of the matrix $A$ and halves its diagonal elements. The final expression for the elements of the Jacobian $J_\alpha^{\hat{\lambda}}$ is

$$\frac{\partial \hat{\lambda}_i}{\partial \alpha_l} = -\hat{\lambda}_i \left(\hat{q}_i^H \cdot R_{nw}^l \cdot \hat{q}_i\right)$$

where $R_{nw}^l \in \mathbb{R}^{M\times M}$ is the prewhitened noise basis matrix $R_{nw}^l = R_n^{-H/2} \cdot R_n^l \cdot R_n^{-1/2}$. ∎

Due to the occurrence of local minima, a simple descent method is not sufficient to converge to the global minimum. However, a local minimum can be detected since the associated value of $V(\alpha)$ is not zero. Moreover, experiments suggest that local minima are associated with (nearly) crossing noise and signal eigenvalues. When the SNRs are high, a distinct separation between noise and signal eigenvalues is present for all $\alpha$, lowering the probability of local minima. Finally, a global minimum exists only when the actual noise correlation matrix truly belongs to the model set. In practice, it is unlikely that perfect prewhitening is attained due to undermodeling and finite sample effects. The algorithm can then still provide an important improvement of the location estimates.

## 8.5 Simulations

In this section we study the performance and robustness of the parameterized noise correlation method by means of simulations.

All simulations are performed with the BFGS quasi-Newton algorithm with mixed quadratic and cubic line search [55]. This is the default routine for unconstrained non-linear programming in MATLAB's optimization toolbox [36]. A constrained optimization, ensuring the positive definiteness of the $R_n$-estimate, could have been implemented. However, the function $V(\alpha)$ is itself already a barrier function, tending to infinity as an eigenvalue tends to zero. Also, the optimal matrix $R_n$ is always full-rank and usually far off the boundary of the convex set of positive definite matrices. Therefore, an unconstrained optimization algorithm is preferred. Because of the occurrence of local minima, the selection of a good initial point is crucial. In the absence of any information on the coloring, we always chose the matrix $R_n^0$ as the starting point. This matrix corresponds to a cylindrically uniform constant ambient noise field.

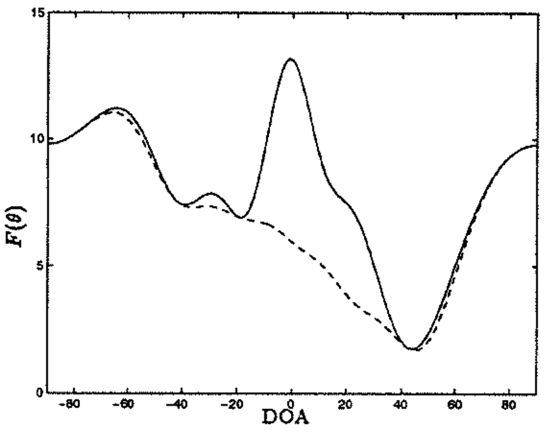
The simulated data are generated by the model equation

$$X = A_\theta \cdot S + R_n^{1/2} \cdot W$$

where $R_n^{1/2}$ is the Cholesky factor of the given noise correlation matrix and $S \in \mathbb{C}^{D\times N}$ and $W \in \mathbb{C}^{M\times N}$ are matrices containing complex i.i.d. zero mean Gaussian noise. As $R_x$ is now the sample correlation matrix of this finite data record, Eq. (8.1) is only approximately satisfied. In all simulations the noise field is heavily colored

$$p(\theta) = 1 - 0.7\sin(\theta) + 0.4\cos(\theta) - 0.2\sin(2\theta) + 0.2\cos(2\theta).$$

The number of coefficients is $Q = 5$ which is larger than the dimension of the noise subspaces considered. The signal constellation consists of $D = 2$ incoherent sources at $(0, 20)$ deg. Their power will be varied.



ULA

$$\begin{aligned} M &= 6 \\ D &= 2 \\ \Delta &= \lambda/2 \\ \theta_s &= (0, 20) \text{ deg} \\ p_s &= (-7, -10) \text{ dB} \\ \alpha &= \begin{bmatrix} 1.0 \\ -0.7 \\ 0.4 \\ -0.2 \\ 0.2 \end{bmatrix} \end{aligned}$$

Figure 8.6: Spatial power spectrum. Full line: output spectrum, Dashed line: noise spectrum.

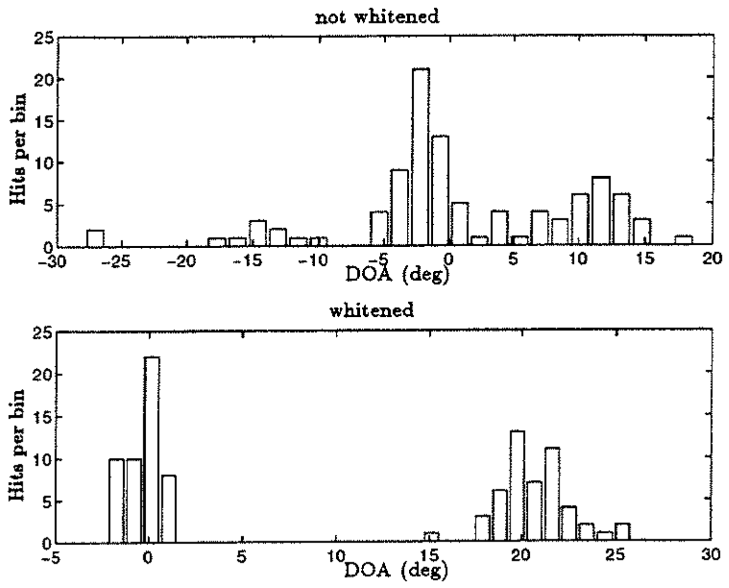
As the noise parameters themselves are of minor importance, the performance of the estimation algorithm is assessed by the quality of the DOA estimates after prewhitening. In the first two simulations the sensor array is a $M = 6$ element half-wavelength ULA. The signal powers are weak $p_s = (-7, -10)$ dB respectively. The 0 dB level is defined as the average noise power over all sensors, *i.e.*, $\text{tr}(R_n)/M$. The spatial output power spectrum

$$F(\theta) = a(\theta)^H \cdot R_x \cdot a(\theta)$$

for this scenario is shown in Figure 8.6 (full line). Similarly the noise power spectrum is shown by the dashed line. The first source at 0 deg is still discernible (smeared out by the low-resolution spatial Fourier transform) above the average noise level. However, the second source at 20 deg is hardly noticeable.

Figure 8.7 shows the histograms and the sample statistics of the DOA estimates obtained with the TLS-ESPRIT algorithm without and with estimation of the noise parameters. Without prewhitening the algorithm is unable to detect the weaker source, and the DOA estimates for the stronger source are heavily biased. After prewhitening both sources are resolved.

At higher SNR the influence of colored noise is less dramatic. Figure 8.8 shows the histograms and sample statistics when the respec-

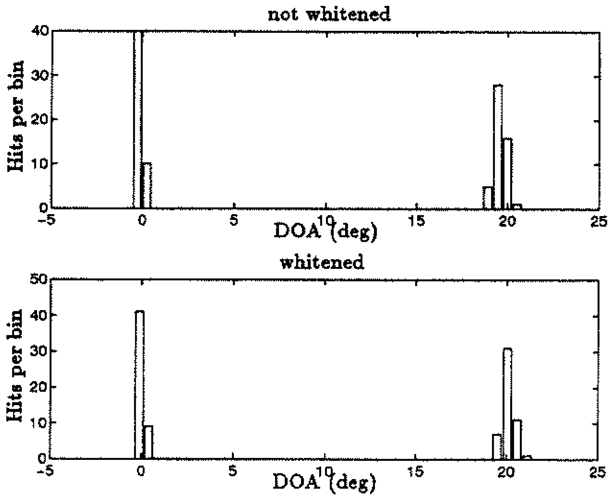


| | not whitened | | whitened | |
|---|---|---|---|---|
| deg | 0 | 20 | 0 | 20 |
| bias | 5.6827 | 12.6193 | 0.2694 | -0.7035 |
| std | 6.4275 | 5.5088 | 0.9549 | 1.9193 |

Figure 8.7: Histogram and sample statistics of the DOA estimates for the scenario of Figure 8.6. 50 independent runs. $N = 250$ samples. $M = 6$ antennas. $p_s = (-7, -10)$ dB. In this case of low signal power the algorithm is unable to resolve the two sources without prewhitening.

tive signal powers are $p_s = (3, 0)$ dB. Now both sources are well estimated without even correcting for the colored noise field. However, the prewhitening still substantially reduces the small bias induced by the undermodeling. Its influence on the variance of the DOA estimates is less pronounced. In fact, because of the estimation errors on the additional noise parameters, the variance of the DOA estimates may increase. For a detailed discussion of these issues, we refer to [141].

In the last simulation we study the effects of undermodeling. The number of antennas is decreased to $M = 4$ and only $Q = 3$ of the 5 noise parameters are estimated. The power of the signal sources is set to



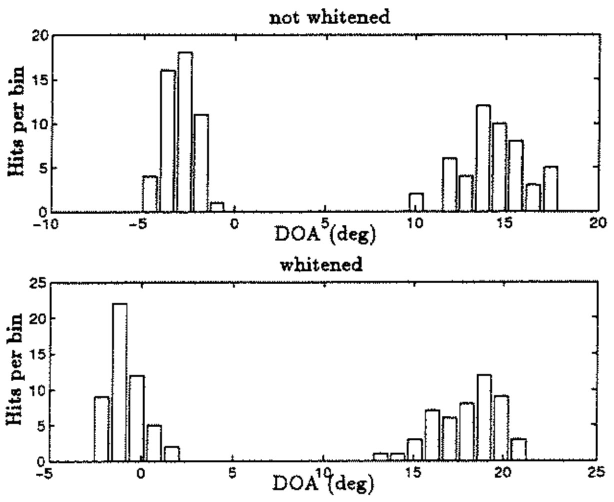
| | not whitened | | whitened | |
|---|---|---|---|---|
| deg | 0 | 20 | 0 | 20 |
| bias | 0.1773 | 0.4318 | 0.0066 | -0.0852 |
| std | 0.1842 | 0.3280 | 0.1848 | 0.3553 |

Figure 8.8: Histogram and sample statistics for the scenario of Figure 8.6. 50 independent runs. $N = 250$ samples. $M = 6$ antennas. $p_s = (3, 0)$ dB. In this case of medium signal power both sources are resolved also without prewhitening. But prewhitening reduces the bias on the DOA estimates.

$p_s = (0, -3)$ dB. As shown in Figure 8.9 the algorithm is still capable of decreasing the bias considerably.

## 8.6 Conclusion

We have proposed a parametric model-based algorithm for DOA estimation of multiple narrow-band sources in unknown ambient noise fields. First, we derived a physical model for the ambient noise impinging on an array with arbitrary but known geometry. The resulting noise correlation matrix is an affine combination of noise basis matrices which only depend on the array geometry. The linear parameters are the Fourier coefficients of the ambient noise field and have to be estimated along with the signal parameters. We also presented two upper bounds on the number of



| | not whitened | | whitened | |
|---|---|---|---|---|
| deg | 0 | 20 | 0 | 20 |
| bias | 3.0284 | 5.6723 | 0.8013 | 2.0275 |
| std | 0.9287 | 1.7481 | 0.9678 | 1.8715 |

Figure 8.9: Histogram and sample statistics for the scenario of Figure 8.6. Undermodeling: only 3 out of 5 noise coefficients are estimated. Only 50 independent runs. $N = 250$ samples. $M = 4$ antennas. $p_s = (0, -3)$ dB.

identifiable parameters.

Finally we introduced a subspace based algorithm which has the property that the estimation of the noise parameters is separated from the estimation of the signal parameters. The object function is the ratio of the arithmetic to the geometric mean of the noise eigenvalues of the prewhitened sample correlation matrix. Minimizing this function is not trivial, since local minima may occur. The estimation accuracy and the robustness obtained with a general non-linear optimization algorithm were illustrated by simulations.

# Chapter 9

# Conclusion and Open Problems

## 9.1 Overview of the contributions

In this thesis we have proposed novel algorithms mainly for adaptive antenna arrays. They have great promise for wireless communication systems such as cellular mobile telephony and paging systems. The acoustic arrays are employed in hands-free telephony *e.g.,* in a car cabin and underwater exploration [112]. The algorithms have to process high data rates in real-time. Therefore, a major concern was the ease with which these algorithms can be implemented on parallel computers. Hereby, we did not design an application-specific architecture for each new algorithm. Instead, we formulated the recursive algorithms as Jacobi algorithms. This approach is in line with previous research on an efficient parallel (systolic) Jacobi architecture. This architecture emerges as a fairly general computational architecture on which an ever expanding set of recursive signal processing algorithms can be mapped efficiently.

Three signal processing tasks have been studied. The first task was *subspace tracking.* The locations of the signal sources determine the dominant subspace of the data matrix. As the sources move, this subspace has to be estimated recursively. Two algorithms were proposed: a Jacobi SVD updating algorithm and a spherical subspace tracking algorithm. The original versions of the two algorithms loose their numerical stability by error accumulation in finite precision arithmetic. This problem was overcome by a factorization of the orthogonal subspace tracking matrix as a sequence of Givens rotations, each parameterized by a rotation angle.

This parameterization had already been in use for non-adaptive signal processing applications, such as orthogonal filters. We showed that it is also applicable to recursive algorithms by presenting an efficient scheme which updates the rotation angles as each new snapshot is fed in. Moreover, the factored algorithms can be implemented in an elegant way on parallel architectures. The factored Jacobi SVD updating algorithm leads to a modified Jacobi architecture which consists solely of rotation nodes. For the spherical subspace tracking algorithm, two application specific architectures – a linear and a planar – were derived starting from the signal flow graph.

Next, a new *robust adaptive beamforming* algorithm was introduced. The beamformer features an adjustable constraint, which is recursively estimated. It alleviates the decrease of the SINR when errors are present in the designed constraint matrix. It also enables the tracking of the signal-of-interest. Starting from the known generalized sidelobe canceler algorithm for minimum variance beamforming with fixed constraints, an adaptive algorithm was derived. This algorithm could be formulated as a Jacobi algorithm, such that it is amenable to parallel implementation on the Jacobi architecture.

The last group of algorithms were *direction finding* algorithms. First the computationally attractive ESPRIT algorithm for direction-of-arrival estimation of multiple narrow-band sources sharing a known carrier frequency was extended to the case in which the carrier frequencies are unknown. The carrier frequency and the position of the narrow-band emitters must be estimated simultaneously. This problem is called the 2-D harmonic retrieval problem. Due to a novel rank restoration method, our algebraically coupled matrix pencil algorithm can cope with multiple sources sharing a same spatial frequency component. Moreover, the algorithm is more efficient than current alternatives.

The second extension of the ESPRIT algorithm concerns the case of wide-band signals captured by a sensor array. Here, we developed a subspace algorithm using state space descriptions of the signals. We combined subspace algorithms for system identification with subspace algorithms for direction finding. The former estimate the spectrum of the signal sources, the latter their position. The combined subspace algorithm only makes use of matrix decompositions. Therefore, a recursive version of the algorithm could be developed. It is again a Jacobi algorithm, now operating on multiple data matrices.

Finally, a new parametric approach was presented for direction finding in the case that the sensor noise has an unknown correlation. Methods to provide a good estimate of the noise correlation are extremely important from a practical viewpoint. We proposed a physical linear model for the ambient noise correlation matrix. It is based on a Fourier expansion of the ambient noise power and incorporates knowledge of the antenna array characteristics. Two upper bounds on the number of identifiable noise parameters were derived. In order to estimate the noise parameters a pseudo maximum likelihood algorithm was developed. It has the nice property that the estimation of the noise parameters is decoupled from the estimation of the signal parameters.

## 9.2 Suggestions for further research

The algorithms we have proposed, were developed from a more or less theoretical viewpoint. They are based on previous fundamental work on simultaneous design of parallel algorithms and architectures and on the theory of array signal processing. As an example the direction finding algorithms make use of concepts from linear algebra and system theory. The assumptions on the data model are in many applications only partially fulfilled.

A first deviation is caused by *errors in the array manifold model.* Inaccuracies in the sensor positions, small differences in the transceiver hardware, electro-magnetic coupling of the antennas,... are responsible for deviations between the mathematical model and the actual array response. Part of them may be taken into account by calibration of the array.

A second concern is *the influence of the propagation environment.* An example is the standard data model for direction finding of narrowband sources. Due to reflection and diffraction a signal from a mobile reaches the base station via multiple paths with different intensities and delays. In addition to the time delay spread caused by the different path lengths, and the frequency spread caused by the Doppler effect induced by the movement of the mobile, each signal also has an angular spread due to the different locations of the main reflectors. In urban areas the dominant ray which normally comes from the exact angular position of the car may even be blocked. The coherent multipath creates an array response vector which is the sum of multiple array manifold vectors, but is not part of the array manifold itself. Therefore, it becomes difficult to associate an angular position with such an unstructured array response vector. Moreover, the instantaneous array response vector is highly time-varying, due to the time-varying intensities and phases of each of the signal paths.

Finally, *the transmitting problem* is much harder than the receiving problem. A straightforward approach is to measure the receive channel vector and use it to compute the transmit weight vectors, eventually after correcting for the frequency translation between the receive and the transmit band (45 MHz in GSM). The difference in frequency causes also a difference in the propagation environment. Therefore, the above straightforward approach is likely to perform poorly. The base station has no idea how accurately it is transmitting, because it has no knowledge of the SINRs at the mobiles. More advanced transmitting schemes involving periodic feedback from the mobile to the base station have recently been proposed [32].

From this discussion it is clear that robustness with respect to model deviations and stochastic characteristics of the mobile channel are at least as important as high accuracy of the location estimate. Only recently, researchers start to pay full attention to the application side. Two areas emerge where more research is required.

On the application side there is a lack of *a realistic mathematically tractable point-to-array propagation model.* In this thesis the algorithms have been simulated on mathematical channel models which are extremely simple. Unfortunately, very few field measurements with antenna arrays have been reported to date [76]. However, more realistic point-to-array models can be obtained by generalization of existing point-to-point models [5, 15]. The major problem here is the choice of an appropriate distribution for the positions of scatters.

On the algorithmic side two areas for further research can be defined. First, we only evaluated the estimation accuracy and the robustness of the algorithms by means of simulations. Although simulations are widely accepted as a test vehicle for algorithms, the approach remains unsatisfactory. Often, it is hard to predict how simulation results are to be generalized to other configurations. A better way is to develop *a statistical analysis* giving closed-form expressions for the estimation errors. The asymptotic analysis of several subspace algorithms for direction finding of narrow-band emitters has already been successfully pursued by a number of authors [49, 79]. However, the extension to highly structured matrices or wide-band emitters is not trivial. Moreover, it is necessary to investigate the sensitivity of the algorithms with respect to different kinds of errors in the model.

Finally, there is ample room for refined algorithms. In wireless communication systems the exact location of the mobiles is only an intermediate result of the computation. The ultimate goal is a good signal reconstruction. Due to the multipath environment, there is uncertainty on the angular position of a mobile. Every mobile may have different angles associated with it. Therefore, the structure on the column space of the data matrix (the spatial structure) is not well defined anymore. To circumvent this difficulty, many new ideas have been proposed. One of the more promising ones is to develop algorithms which make use of signal structure to reconstruct the messages. Communication signals are highly structured. Often they are cyclostationary [95, 101, 148]. Frequency or phase modulated signals have a constant modulus [116] and digital signals can only have a small number of constellations [73, 109, 117]. This signal structure imposes strong conditions on the row space of the data matrix. Moreover it is often better retained by the mobile channel than the spatial structure. The references cited above, make use of this addition knowledge in order to improve the signal reconstruction and the location estimates. But this area still offers many challenges to the signal processing community. Recently, algorithms have been formulated to optimally approximate low-rank structured matrices which are affine in their parameters [18]. In array signal processing the structure of the data matrix is more complex. The column space and the row space of the low-rank data matrix depend non-linearly on the array manifold and the signal set. More research is required to formulate good answers to these *structured matrix problems.*

# Bibliography

[1] H.M. Ahmed, J.-M. Delosme, and M. Morf, "Highly concurrent computing structures for matrix arithmetic and signal processing," *Computer*, pp. 65–82, Jan. 1982.

[2] S. Anderson, M. Millnert, M. Viberg, and B. Wahlberg, "An adaptive array for mobile communication systems," *IEEE Trans. on VT*, vol. 40, no. 1, pp. 230–236, Feb. 1991.

[3] S.P. Applebaum and D.J. Chapman, "Adaptive arrays with main beam constraints," *IEEE Trans. on AP*, vol. 24, no. 5, pp. 650–662, Sept. 1976.

[4] J. Arnbak, "The European (r)evolution of wireless digital networks," *IEEE Communications Magazine*, pp. 74–82, Sept. 1993.

[5] H. Bertoni, W. Honcharenko, L. Rocha Maciel, and H. Xia, "UHF propagation prediction for wireless personal communications," *IEEE Proceedings*, vol. 82, no. 9, pp. 1333–1359, Sept. 1994.

[6] J.F. Böhme and D. Kraus, "On least squares methods for direction of arrival estimation in the presence of unknown noise fields," in *Proc. ICASSP*, pp. 2833–2836, 1988.

[7] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, vol. 15 of *Studies in Applied Mathematics*. SIAM, 1994.

[8] J. Brewer, "Kronecker products and matrix calculus in system theory," *IEEE Trans. on CAS*, vol. 25, pp. 772–780, Sept. 1978.

[9] A. Bunse-Gerstner, R. Byers, V. Mehrmann, and N. Nichols, "Numerical computation of an analytic singular value decomposition of a matrix valued function," *Numerische Mathematik*, vol. 60, pp. 1–39, Nov. 1991.

[10] L. Castedo, C.-Y. Tseng, and L.J. Griffiths, "An adjustable constraint approach for robust adaptive beamforming," in *Signal Processing VI, Proc. EUSIPCO*, pp. 1121–1124, 1992.

[11] J.-P. Charlier and P. Van Dooren, "A Jacobi-like algorithm for computing the generalized Schur form of a regular pencil," *Journal Comp. Appl. Math.*, vol. 27, pp. 17–36, 1989.

[12] D. Chazan, Y. Medan, and U. Shvadron, "Noise cancellation for hearing aids," *IEEE Trans. on ASSP*, vol. 36, pp. 1697–1905, 1988.

[13] J. Chun, T. Kailath, and H. Lev-Ari, "Fast parallel algorithms for QR and triangular factorization," *SIAM J. Sci. Stat. Comput.*, vol. 8, no. 6, pp. 899–913, Nov. 1989.

[14] I. Claesson, S. Nordholm, B. Bengtsson, and P. Eriksson, "A multi-DSP implementation of a broad-band adaptive beamformer for use in a hands-free mobile radio telephone," *IEEE Trans. on VT*, vol. 40, no. 1, pp. 194–202, Feb. 1991.

[15] D. Cox, "910 MHz Urban mobile radio propagation: multipath characteristics in New York City," *IEEE Trans. on Comm.*, vol. 21, no. 11, pp. 1188–1194, Nov. 1973.

[16] J. David, *Algorithms for analysis and design of robust controllers.* PhD thesis, Katholieke Universiteit Leuven, Dept. E.E., Mar. 1994.

[17] B. De Moor, "The singular value decomposition and long and short spaces of noisy matrices," *IEEE Trans. on SP*, vol. 41, no. 9, pp. 2826–2838, Sept. 1993.

[18] B. De Moor, "Structured total least squares and $l_2$ approximation problems," *Linear Algebra and its Applications*, vol. 188-189, pp. 163–207, July 1993.

[19] B. De Moor and S. Boyd, "Analytic properties of singular values and vectors," Tech. Rep. 89-28, Katholieke Universiteit Leuven, Dept. E.E., 1989.

[20] R.D. DeGroat, "Noniterative subspace tracking," *IEEE Trans. on SP*, vol. 40, no. 3, pp. 571–577, Mar. 1992.

[21] R.D. DeGroat and E.M. Dowling, "Sphericalized subspace tracking: convergence and detection schemes," in *Proc. 26th annual Asilomar conference*, pp. 561–565, 1992.

[22] R.D. DeGroat and R.A. Roberts, "A family of rank-one subspace updating methods," in *SVD and Signal Processing: Algorithms, Applications and Architectures* (E. Deprettere, ed.), pp. 277–300, North-Holland, 1988.

[23] E. Deprettere and A.J. van der Veen, eds., *Algorithms and parallel VLSI architectures*, vol. 1-2. Elsevier, 1991.

[24] E.F. Deprettere, H.W. van Dijk, and G.J. Hekstra, "A Jacobi signal processing unit for time-adaptive SVD," in *Proc. ICASSP*, vol. 2, pp. 493–496, Apr. 1994.

[25] E.M. Dowling and R.D. DeGroat, "Adaptation dynamics of the spherical subspace tracker," *IEEE Trans. on SP*, vol. 40, no. 10, pp. 2599–2602, Oct. 1992.

[26] P. Feautrier, "Compiling for massively parallel architectures: a perspective," in *Algorithms and parallel VLSI architectures III, (To appear)* (M. Moonen and F. Catthoor, eds.), Leuven, p. 12, Elsevier, 1994.

[27] D.D. Feldman and L.J. Griffiths, "A constraint projection approach for robust adaptive beamforming," *IEEE Trans. on SP*, vol. 42, no. 4, pp. 867–876, Apr. 1994.

[28] Z. Fu, E.M. Dowling, and R.D. DeGroat, "Spherical subspace tracking on systolic arrays," tech. rep., The university of Texas at Dallas, 1993.

[29] J.-J. Fuchs, "Estimation of the number of signals in the presence of unknown correlated sensor noise," *IEEE Trans. on SP*, vol. 40, no. 5, pp. 1053–1061, May 1992.

[30] W.M. Gentleman, "Error analysis of QR decomposition by Givens transformations," *Linear Algebra and its Applications*, vol. 10, pp. 189–195, 1975.

[31] W.M. Gentleman and H.T. Kung, "Matrix triangularization by systolic arrays," in *Real-Time Signal Processing IV, Proc. SPIE*, vol. 298, pp. 19–26, 1982.

[32] D. Gerlach and A. Paulraj, "Spectrum reuse using transmitting antenna arrays with feedback," in *Proc. ICASSP*, vol. 4, pp. 97–100, 1994.

[33] G.H. Golub and C.F. Van Loan, *Matrix Computations*. John Hopkins, 2nd ed., 1989.

[34] B. Göransson, "Direction finding in the presence of spatially correlated noise fields," in *Signal Processing VII: Theory and Appl., Proc. EUSIPCO*, pp. 1321–1324, 1994.

[35] J. Götze and G.J. Hekstra, "Adaptive approximate rotations for computing the EVD," in *SVD and Signal Processing III* (B. De Moor and M. Moonen, eds.), Leuven, p. 8, Elsevier, Aug. 1994.

[36] A. Grace, *MATLAB optimization Toolbox*. The MathWorks Inc., 1990.

[37] L.J. Griffiths and C.W. Jim, "An alternative approach to linearly constrained adaptive beamforming," *IEEE Trans. on AP*, vol. 30, no. 1, pp. 27–34, Jan. 1982.

[38] Yu Hen Hu, "CORDIC-based VLSI architectures for digital signal processing," *IEEE Signal processing magazine*, pp. 16–33, July 1992.

[39] Y. Hua, "Estimating two-dimensional frequencies by matrix enhancement and matrix pencil," *IEEE Trans. on SP*, vol. 40, no. 9, pp. 2267–2280, Sept. 1992.

[40] H. Hung and M. Kaveh, "Coherent wide-band ESPRIT method for directions of arrival estimation of multiple wide-band sources," *IEEE Trans. on SP*, vol. 38, no. 2, pp. 354–356, Feb. 1990.

[41] N.K. Jablon, "Adaptive beamforming with the generalized sidelobe canceller in the presence of array imperfections," *IEEE Trans. on Ant. Prop.*, vol. 34, no. 8, pp. 996–1012, Aug. 1986.

[42] G. Jacobi, "Über ein leichtes Verfahren die in der Theorie der säcularstörungen vorkommenden gleichungen numerisch aufzulösen," *J. Reine u. Angew. Math.*, vol. 30, pp. 51–94, 1846.

[43] K. Jainandunsing and E.F. Deprettere, "A new class of parallel algorithms for solving systems of linear equations," *SIAM J. Sci. Stat. Comput.*, vol. 10, no. 5, pp. 880–912, Sept. 1989.

[44] D.H. Johnson and D.E. Dudgeon, *Array signal Processing.* Prentice-Hall, 1993.

[45] T. Kailath, *Linear Systems.* Prentice-Hall, 1980.

[46] P. Kapteijn, H. van Dijk, and E.F. Deprettere, "Controlling the critical path in time adaptive QR$^{-1}$ recursions," *VLSI Signal Processing*, vol. VII, pp. 326–335, 1994.

[47] T. Kato, *Perturbation theory for linear operators.* Springer Verlag, 1976.

[48] E. Kogbetliantz, "Diagonalization of general complex matrices as a new method for solution of linear equations," in *Proc. Int. Congres Math.*, vol. 2, Amsterdam, The Netherlands, pp. 356–357, 1954.

[49] H. Krim, P. Forster, and J. Proakis, "Operator approach to performance analysis of root-MUSIC and root-min-norm," *IEEE Trans. on SP*, vol. 40, no. 7, pp. 1687–1696, July 1992.

[50] S.Y. Kung, *VLSI array processors.* Prentice-Hall, 1988.

[51] J.P. Le Cadre, "Parametric methods for spatial signal processing in the presence of unknown colored noise fields," *IEEE Trans. on ASSP*, vol. 37, no. 7, pp. 965–983, July 1989.

[52] W. Lee, *Mobile communications engineering.* McGraw-Hill, 1982.

[53] F. Li and R.J. Vaccaro, "Analytical performance prediction of subspace-based algorithms for DOA estimation," in *SVD and signal processing, II* (R. Vaccaro, ed.), pp. 243–260, Elsevier, 1991.

[54] F. Lorenzelli and K. Yao, "On updating rate of Jacobi SVD arrays and data nonstationarity," in *Advanced Signal Processing Algorithms, Architectures and Implementations V, Proc. of SPIE* (F.T. Luk, ed.), vol. 2296, San Diego, USA, pp. 414–425, July 1994.

[55] D.G. Luenberger, *Linear and nonlinear progamming.* Addison-Wesley, 2nd ed., 1984.

[56] J. McClellan, "Multidimensional spectral estimation," *Proc. IEEE*, vol. 70, no. 9, pp. 1029–1039, Sept. 1982.

[57] J.G. McWhirter, "Algorithmic engineering in adaptive signal processing," *IEE Proceedings-F*, vol. 139, no. 3, pp. 226–232, June 1992.

[58] J.G. McWhirter and I.K. Proudler, "Algorithmic engineering: a worked example," in *Signal Processing VI, Proc. EUSIPCO*, pp. 5–12, 1992.

[59] J.G. McWhirter and T.Shepherd, "Systolic array processor for MVDR beamforming," *IEE Proceedings-F*, vol. 136, pp. 75–80, Apr. 1989.

[60] M. Moonen, *Jacobi-type updating algorithms for signal processing, systems identification and control.* PhD thesis, Katholieke Universiteit Leuven, Nov. 1990.

[61] M. Moonen, "Mapping directionally weighted recursive least squares computations on an SVD updating array," in *Signal processing VII, Proc. EUSIPCO*, vol. 3, pp. 1863–1866, Sept. 1994.

[62] M. Moonen and F. Catthoor, eds., *Algorithms and parallel VLSI architectures III.* Elsevier, 1994.

[63] M. Moonen, B. De Moor, L. Vandenberghe, and J. Vandewalle, "On- and off-line identification of linear state-space models," *Int. Journal of Control*, vol. 49, no. 1, pp. 219–232, 1989.

[64] M. Moonen and E.Deprettere, "On the derivation of parallel filter structures for adaptive eigenvalue and singular value decomposition," tech. rep., Katholieke Universiteit Leuven, Dept. E.E., 1995.

[65] M. Moonen, I.K. Proudler, J.G. McWhirter, and G. Hekstra, "On the formal derivation of a systolic array for recursive least squares estimation," in *Proc. ICASSP*, vol. 2, pp. 477–480, Apr. 1994.

[66] M. Moonen, P. Van Dooren, and J. Vandewalle, "An SVD updating algorithm for subspace tracking," *SIAM J. Matrix Anal. Appl.*, vol. 13, no. 4, pp. 1015–1038, Oct. 1992.

[67] M. Moonen, P. Van Dooren, and J. Vandewalle, "A systolic array for SVD updating," *SIAM J. Matrix Anal. Appl.*, vol. 14, no. 2, pp. 353–371, Apr. 1993.

[68] M. Moonen, P. Van Dooren, and F. Vanpoucke, "On the QR algorithm and updating the SVD and URV decomposition in parallel," *Linear Algebra and its Applications*, vol. 188-189, pp. 549–568, 1993.

[69] M. Moonen and J. Vandewalle, "A systolic array for recursive least squares computation," *IEEE Trans. on SP*, vol. 41, no. 2, pp. 906–911, Feb. 1993.

[70] M. Moonen and F. Vanpoucke, "A fully parallel SVD-regularized adaptive LCMV beamformer," in *Adaptive systems, intelligent approaches, massively parallel computing and emergent techniques in signal processing and communications. Proc. of COST 229 Workshop* (D. Docampo and A. Figueiras, eds.), Bayona, Spain, pp. 109–113, Oct. 1994.

[71] M. Moonen, F. Vanpoucke, and E. Deprettere, "Parallel and adaptive high resolution direction finding," *IEEE Trans. on SP*, vol. 42, no. 9, pp. 2439–2448, Sept. 1994.

[72] M. Mouly and M.-B. Pautet, *The GSM system for mobile communications.* 1982.

[73] A. Naguib, B. Khalaj, A. Paulraj, and T. Kailath, "Adaptive channel equalization for TDMA digital cellular communications using antenna arrays," in *Proc. ICASSP*, vol. 4, pp. 101–104, 1994.

[74] A. Nehorai, G. Su, and M. Morf, "Estimation of time differences of arrival by pole decomposition," *IEEE Trans. on ASSP*, vol. 31, no. 6, pp. 1478–1491, Dec. 1983.

[75] S. Nordholm, I. Claesson, and B. Bengtsson, "Adaptive array noise suppression of handsfree speaker input in cars," *IEEE Trans. on VT*, vol. 42, no. 4, pp. 514–518, Nov. 1993.

[76] T. Ohgane, N. Matsuzawa, T. Shimura, M. Mizuno, and H. Sasaoka, "BER performance of CMA adaptive array for high-speed GMSK mobile communication- a description of measurements in central Tokyo," *IEEE Trans. on VT*, vol. 42, no. 4, pp. 484–485, Nov. 1993.

[77] S.J. Olszanskyj and A.W. Bojanczyk, "Compact Givens representation of the orthogonal factor in recursive least squares," in *Proc. of the 5th SIAM conference on applied linear algebra* (J.G. Lewis, ed.), Snowbird (Utah), pp. 255–259, June 1994.

[78] B. Ottersten and T. Kailath, "Direction-of-arrival estimation for wide-band signals using the ESPRIT algorithm," *IEEE Trans. on ASSP*, vol. 38, no. 2, pp. 317–327, Feb. 1990.

[79] B. Ottersten, M. Viberg, and T. Kailath, "Analysis of subspace fitting and ML techniques for parameter estimation from sensor data," *IEEE Trans. on SP*, vol. 40, no. 3, pp. 590–600, Apr. 1992.

[80] B. Ottersten, M. Viberg, P. Stoica, and A. Nehorai, "Exact and large sample maximum likelihood techniques for parameter estimation and detection in array processing," in *Radar array processing* (S. Haykin, J. Litva, and T. Shepherd, eds.), pp. 99–151, Springer-Verlag, 1993.

[81] B. Porat, *Digital processing of random signals.* Prentice-Hall, 1994.

[82] J.G. Proakis, *Digital communications.* McGraw - Hill, 2nd ed., 1989.

[83] I. Proudler and J.G. McWhirter, "Algorithmic engineering in adaptive signal processing II - worked examples," *IEE Proceedings VIS*, vol. 141, no. 1, pp. 19–26, Feb. 1994.

[84] P. Quinton and Y. Robert, eds., *Algorithms and parallel VLSI architectures II.* Elsevier, 1992.

[85] G. Raleigh, S. Diggavi, V. Jones, and A. Paulraj, "A blind adaptive transmit antenna algorithm for wireless communication," tech. rep., ISL, Stanford University, 1995.

[86] B. Rao and K. Arun, "Model based processing of signals: a state space approach," *IEEE Proceedings*, vol. 80, no. 2, pp. 283–309, Feb. 1992.

[87] D.V. Bhaskar Rao and S.Y. Kung, "A state space approach for the 2D harmonic retrieval problem," in *Proc. ICASSP*, vol. 4, pp. 10.1–10.3, 1988.

[88] P.A. Regalia, "An adaptive unit-norm filter with applications to signal analysis and Karhunen-Loève transformations," *IEEE Trans. on CAS*, vol. 37, no. 5, pp. 646–649, May 1990.

[89] P.A. Regalia and Ph. Loubaton, "Rational subspace estimation using adaptive lossless filters," *IEEE Trans. on SP*, vol. 40, no. 10, pp. 2392–2405, Oct. 1992.

[90] R.A. Roberts and C.T. Mullis, *Digital signal processing.* Addison-Wesley, 1987.

[91] R. Roy, *ESPRIT: Estimation of signal parameters via rotational invariance techniques.* PhD thesis, Stanford University, Aug. 1987.

[92] R. Roy and T. Kailath, "ESPRIT - estimation of signal parameters via rotational invariance techniques," *IEEE Trans. on ASSP*, vol. 37, no. 7, pp. 984–995, July 1989.

[93] L. Scharf, "The SVD and reduced rank signal processing," *Signal Processing*, vol. 25, pp. 113–133, 1991.

[94] L.L. Scharf, *Statistical Signal Processing.* Addison-Wesley, 1991.

[95] S. Schell, D. Smith, and W. Gardner, "Blind channel identification using 2nd order cyclostationary statistics," in *Signal Processing VII: Theory and Appl., Proc. EUSIPCO*, pp. 716–719, 1994.

[96] R.O. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. on AP*, vol. 34, no. 3, pp. 276–280, Mar. 1986.

[97] S. Seidel, T. Rappaport, S. Jain, M. Lord, and R. Singh, "Path loss, scattering and multipath delay statistics in four European cities for digital cellular and microcellular radiotelephone," *IEEE Trans. on VT*, vol. 40, pp. 721–730, Nov. 1991.

[98] T.J. Shan, M. Wax, and T. Kailath, "On spatial smoothing for direction-of-arrival estimation of coherent signals," *IEEE Trans. on ASSP*, vol. 33, no. 4, pp. 806–811, Aug. 1985.

[99] T. Shepherd and J. McWhirter, "Systolic adaptive beamforming," in *Radar array processing* (S. Haykin, J. Litva, and T. Shepherd, eds.), pp. 153–247, Springer-Verlag, 1993.

[100] T.J. Shepherd and J.G. McWhirter, "A pipelined array for linearly constrained least-squares optimisation," in *Proc. IMA Conference on Mathematics in Signal Processing*, pp. 457–483, 1985.

[101] D. Slock and C. Papadias, "Blind radio channel identification and equlaization based on oversampling and/or antenna arrays," in *Proc. of COST 229 Workshop* (D. Docampo and A. Figueiras, eds.), pp. 103–107, Oct. 1994.

[102] G.W. Stewart, "An updating algorithm for subspace tracking," Tech. Rep. CS-TR 2494, Dept. of Computer Science, Univ. of Maryland, 1990.

[103] P. Stoica, M. Viberg, and B. Ottersten, "Instrumental variable approach to array processing in spatially correlated noise fields," *IEEE Trans. on SP*, vol. 42, no. 1, pp. 121–133, Jan. 1994.

[104] G. Su and M. Morf, "The signal subspace approach for multiple wide-band emitter location," *IEEE Trans. on ASSP*, vol. 31, no. 6, pp. 1502–1521, Dec. 1983.

[105] G. Su and M. Morf, "Modal decomposition signal subspace algorithms," *IEEE Trans. on ASSP*, vol. 34, no. 3, pp. 585–602, June 1986.

[106] S. Swales, M. Beach, D. Edwards, and J. McGeehan, "The performance enhancement of multibeam adaptive base-station antennas for cellular land mobile radio systems," *IEEE Trans. on VT*, vol. 39, no. 1, pp. 56–67, Feb. 1990.

[107] A. Swindlehurst and T. Kailath, "A performance analysis of subspace-based methods in the presence of model errors, Part I: the MUSIC algorithm," *IEEE Trans. on SP*, vol. 40, no. 7, pp. 1758–1773, July 1992.

[108] A. Swindlehurst, B. Ottersten, R. Roy, and T. Kailath, "Multiple invariance ESPRIT," *IEEE Trans. on SP*, vol. 40, no. 4, pp. 867–881, Apr. 1992.

[109] S. Talwar, M. Viberg, and A. Paulraj, "Blind estimation of multiple co-channel digital signals using an antenna array," *IEEE SP letters*, vol. 1, pp. 29–31, 1994.

[110] J. Tugnait, "Time delay estimation with unknown spatially correlated Gaussian noise," *IEEE Trans. on SP*, vol. 41, no. 2, pp. 549–558, Feb. 1993.

[111] S. Unnikrishna Pillai and B. Ho Kwon, "Forward/backward spatial smoothing techniques for coherent signal identification," *IEEE Trans. on SP*, vol. 37, no. 1, pp. 8–14, Jan. 1989.

[112] R. Urick, *Principles of underwater sound.* McGraw-Hill, 2nd ed., 1975.

[113] A.J. van der Veen and E. Deprettere, "Parallel VLSI matrix pencil algorithm for high resolution direction finding," *IEEE Trans. on SP*, vol. 39, no. 2, pp. 383–394, Feb. 1991.

[114] A.J. van der Veen, E. Deprettere, and A. Swindlehurst, "Subspace-based signal analysis using singular value decomposition," *IEEE Proceedings*, vol. 81, no. 9, pp. 1277–1308, Sept. 1993.

[115] A.J. van der Veen, P. Ober, and E. Deprettere, "Azimuth and elevation computation in high resolution DOA estimation," *IEEE Trans. on SP*, vol. 40, no. 7, pp. 1828–1832, July 1992.

[116] A.J. van der Veen and A. Paulraj, "A constant modulus factorization technique for smart antenna applications in mobile communications," in *Advanced Signal Processing Algorithms, Architectures and Implementations V, Proc. of SPIE* (F.T. Luk, ed.), vol. 2296, San Diego, USA, pp. 230–41, July 1994.

[117] A.J. van der Veen, S. Talwar, and A. Paulraj, "Blind identification of FIR channels carrying multiple finite alphabet signals," in *Proc. ICASSP ( To Appear)*, p. 4, May 1995.

[118] H.W. van Dijk and E.F. Deprettere, "Systematic exploration of the space of Jacobi algorithms," in *Advanced Signal Processing V, Proc. SPIE* (F.T. Luk, ed.), pp. 388–402, July 1994.

[119] S. Van Huffel and J. Vandewalle, *The total least squares problem: computational aspects and analysis*, vol. 9 of *Frontiers in applied mathematics*. SIAM, 1991.

[120] P. Van Overschee and B. De Moor, "Subspace algorithms for the stochastic identification problem," *Automatica*, vol. 29, no. 3, pp. 649–660, 1993.

[121] P. Van Overschee and B. De Moor, "N4SID: subspace algorithms for the identification of combined deterministic-stochastic systems," *Automatica*, vol. 30, no. 1, pp. 75–93, Jan. 1994.

[122] L. Vandenberghe and S. Boyd, "Semidefinite programming," Submitted to *SIAM Review*, July 1994.

[123] J. Vandewalle and B. De Moor, "A variety of applications of the singular value decomposition in identification and signal processing," in *SVD and Signal Processing: Algorithms, Applications and Architectures* (E. Deprettere, ed.), pp. 43–91, 1988.

[124] F. Vanpoucke and M. Moonen, "Robust adaptive beamforming using an adjustable constraint: a parallel solution," in *Proc. of ProRISC Workshop on Circuits, Systems and Signal Processing*, Houthalen, Belgium, pp. 193–198, Mar. 1993.

[125] F. Vanpoucke and M. Moonen, "A systolic algorithm for robust adaptive beamforming using an adjustable constraint," in *Advanced Signal Processing Algorithms, Architectures and Implementations IV, Proc. of SPIE* (F.T. Luk, ed.), vol. 2027, San Diego, USA, pp. 398–409, July 1993.

[126] F. Vanpoucke and M. Moonen, "A state space algorithm for wide-band source localization," in *Proc. of 28th annual Asilomar conference on signals, systems and computers (To appear)*, Pacific Grove, USA, p. 5, Nov. 1994.

[127] F. Vanpoucke and M. Moonen, "A state space method for direction finding of wide-band emitters," in *Signal Processing VII, Theory and applications, Proc. of EUSIPCO*, vol. 2, pp. 780–783, Sept. 1994.

[128] F. Vanpoucke and M. Moonen, "Parallel and stable spherical subspace tracking," in *Proc. of ICASSP (To appear)*, Detroit, USA, p. 4, May 1995.

[129] F. Vanpoucke and M. Moonen, "A systolic algorithm for robust adaptive LCMV beamforming with an adjustable constraint," *To appear in IEEE Trans. on AES*, p. 31, 1995.

[130] F. Vanpoucke, M. Moonen, and Y. Berthoumieu, "An efficient subspace algorithm for 2-D harmonic retrieval," in *Proc. of ICASSP*, vol. 4, Adelaide, Australia, pp. 461–464, Apr. 1994.

[131] F. Vanpoucke, M. Moonen, and E. Deprettere, "Parallel and adaptive high resolution direction finding," in *Advanced Signal Processing Algorithms, Architectures and Implementations III, Proc. of SPIE* (F.T. Luk, ed.), vol. 1770, San Diego, USA, pp. 219–230, July 1992.

[132] F. Vanpoucke, M. Moonen, and E. Deprettere, "A generalized ESPRIT algorithm for direction finding of multiple wide-band emitters," in *Proc. of COST 229 workshop on adaptive methods and emergent techniques for signal processing and communications* (D. Docampo and A. Figueiras, eds.), Bayona, Spain, pp. 180–189, June 1993.

[133] F. Vanpoucke, M. Moonen, and E. Deprettere, "Systolic high resolution direction finding of multiple wide-band emitters," in *VLSI Signal Processing VI, Proc. of IEEE workshop on VLSI signal processing* (E. Eggermont et al., ed.), Veldhoven, The Netherlands, pp. 307–315, Oct. 1993.

[134] F. Vanpoucke, M. Moonen, and E. Deprettere, "Direction finding of multiple wide-band emitters using state space modeling," *Signal Processing*, vol. 40, pp. 39–52, 1994.

[135] F. Vanpoucke, M. Moonen, and E. Deprettere, "Factored orthogonal matrix - vector multiplication with applications to parallel and adaptive eigenfiltering and SVD," Tech. Rep. 94-20, Katholieke Universiteit Leuven, Dept. E.E., 1994.

[136] F. Vanpoucke, M. Moonen, and E. Deprettere, "A numerically stable Jacobi array for parallel SVD updating," in *Advanced Signal Processing Algorithms, Architectures and Implementations V, Proc. of SPIE* (F.T. Luk, ed.), vol. 2296, San Diego, USA, pp. 403–412, July 1994.

[137] F. Vanpoucke, M. Moonen, and E. Deprettere, "Stable Jacobi SVD updating by factorization of the orthogonal matrix," in *SVD and Signal Processing III, Proc. of the 3rd international workshop on SVD and signal processing (To appear)* (M. Moonen and B. De Moor, eds.), Leuven, Belgium, Aug. 1994.

[138] F. Vanpoucke, M. Moonen, and J. Vandewalle, "Numerical issues and parallelism in adaptive beamforming methods," in *Systems and Networks, Mathematical Theory and Applications, Proc. of MTNS* (U. Helmke, R. Mennicken, and J. Saurer, eds.), vol. 2, Regensburg, Germany, pp. 905–908, Aug. 1993.

[139] F. Vanpoucke and A. Paulraj, "A harmonic noise model for direction finding in colored ambient noise," Tech. Rep. 95-01, Katholieke Universiteit Leuven, Dept. E.E., 1995.

[140] M. Viberg, *Subspace Fitting Concepts in Sensor Array Processing.* PhD thesis, Linköping University, 1989.

[141] M. Viberg, "Sensitivity of parametric direction finding to colored noise fields and undermodeling," *Signal Processing*, vol. 34, no. 2, pp. 207–222, Nov. 1993.

[142] M. Viberg, B. Ottersten, and T. Kailath, "Detection and estimation in sensor arrays using weighted subspace fitting," *IEEE Trans. on SP*, vol. 39, no. 11, pp. 2436–2449, Nov. 1992.

[143] J.E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. on Electronic Computers*, vol. 8, no. 3, pp. 330–334, Sept. 1959.

[144] A.M. Vural, "Effects of perturbations on the performance of optimum adaptive arrays," *IEEE Trans. on AES*, vol. 15, no. 1, pp. 76–87, Jan. 1979.

[145] H. Wang and M. Kaveh, "Coherent signal-subspace processing for the detection and estimation of angles of arrival of multiple wide-band sources," *IEEE Trans. on SP*, vol. 33, no. 4, pp. 823–831, Aug. 1985.

[146] C. Ward, P. Hargrave, and J. McWhirter, "A novel algorithm and architecture for adaptive digital beamforming," *IEEE Trans. on AP*, vol. 34, no. 3, pp. 338–346, Mar. 1986.

[147] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," *IEEE Trans. on ASSP*, vol. 33, no. 2, pp. 387–392, Apr. 1985.

[148] G. Xu and T. Kailath, "Direction of arrival estimation via exploitation of cyclostationarity - a combination of temporal and spatial processing," *IEEE Trans. on SP*, vol. 40, no. 2, pp. 1775–1785, July 1992.

[149] B. Yang, "Gradient based subspace tracking algorithms and systolic implementation," *Int. Journal of High Speed Electronics and Systems*, vol. 4, no. 2, pp. 203–218, June 1993.

[150] B. Yang, "Subspace tracking based on the projection approach and the recursive least squares method," in *Proc. ICASSP*, vol. 4, pp. 145–148, 1993.

[151] B. Yang and J.F. Böhme, "Linear systolic arrays for constrained least squares problems," in *Proc. IMA Conference on Mathematics in signal processing*, pp. 689–711, 1988.

[152] B. Yang and J.F. Böhme, "A multiple constrained adaptive beamformer and a systolic implementation," in *Signal Processing IV, Proc. EUSIPCO*, pp. 283–286, 1988.

[153] Z. Zhu and S. Haykin, "Radar detection using array processing," in *Radar array processing* (S. Haykin, J. Litva, and T. Shepherd, eds.), pp. 3–46, Springer-Verlag, 1993.