

TECHNISCHE UNIVERSITÄT WIEN

DISSERTATION

**Recursive Algorithms for Adaptive Transversal
Filters: Optimality and Time-Variance**

ausgeführt zum Zwecke der Erlangung des akademischen Grades
eines Doktors der technischen Wissenschaften

eingereicht an der Technischen Universität Wien
Fakultät für Elektrotechnik
von

*Dipl.-Ing. Gernot Kubin
Lindengasse 49/1/6, A-1070 Wien
geboren am 24. Juni 1960 in Wien
Matrikel Nr. 7725476*

Wien, im Juni 1990

Kurzfassung

Diese Arbeit behandelt eine *vereinheitlichte Theorie* für den Entwurf und die Analyse rekursiver Algorithmen zur Adaption digitaler Filter mit transversaler Struktur. Zunächst wird der weitverbreitete Ansatz der *Fehlerminimierung* als Entwurfskriterium untersucht und aufgezeigt, daß damit keine schlüssige Ableitung praxisgerechter Algorithmen von einem Optimalitätskriterium her möglich ist. Der Grund liegt in der Unvereinbarkeit der Annahme einer zeitlich unveränderlichen Anwendungsumgebung für die Optimalitätsdefinition zum einen mit der praktischen Forderung der Nachführung des adaptiven Filters in zeitveränderlichen Umgebungen zum anderen. Der hier vorgeschlagene deterministische Ansatz für den Algorithmenentwurf geht über die reine Fehlerminimierung hinaus, indem er die *zeitliche Variation der Koeffizienten des adaptiven Filters miteinbezieht*.

In der Folge kann gezeigt werden, daß eine Fülle von Algorithmen dieser neuartigen, vereinheitlichten Beschreibung genügt und auch einige oft ad hoc eingeführte Anpassungen der Algorithmen in diesem Rahmen ohne Näherung ableitbar sind. Darunter fällt auch die Verwendung von *Fehlerfiltern*, wie sie speziell bei der Adaption rekursiver Filterstrukturen üblich ist.

Im zweiten Teil der Arbeit wird zunächst die Anwendung der beschriebenen Algorithmenklasse zur Nachführung des adaptiven Filters in zeitveränderlichen Umgebungen untersucht. Das Ergebnis ist eine *Beschreibung des Nachführverhaltens als Filterung* des von der Umgebung vorgegebenen, jedoch nicht direkt beobachtbaren Zeitverlaufs der Koeffizienten eines Referenzmodells. Bedingt durch die allgemeine Struktur der Adaptionalgorithmen ist diese Filterung stets linear und von erster Ordnung. Um eine Berücksichtigung von Vorwissen über das zu erwartende Zeitverhalten zu ermöglichen, ist eine Erweiterung der Algorithmenstruktur um sogenannte *Koeffizientenfilter* erforderlich. Dadurch kann das Nachführverhalten praxisgerecht als lineare Filterung höherer Ordnung bzw. auch als nichtlineare Filterung maßgeschneidert werden. Eine Reihe aktueller Vorschläge derartiger Koeffizientenfilter wird abschließend auf Basis des entsprechend erweiterten vereinheitlichten Optimalitätskriteriums diskutiert.

Abstract

This thesis presents a *unified theory* for the design and analysis of recursive algorithms for the adaptation of transversal digital filters. First, the widely used *error minimization approach* to algorithm design is investigated and it is shown not to allow a coherent derivation of practical algorithms from an optimality criterion. The reason is found in the incompatibility of the assumption of a time-invariant application environment for the optimality definition and of the practical demand on the adaptive filter for tracking in time-varying environments. The present proposal for a deterministic approach to algorithm design goes beyond mere error minimization in that the time variation of the coefficients of the adaptive filter is included as well.

In the sequel a wealth of algorithms is shown to fulfil this novel unified description and several algorithm modifications, which often appear ad hoc, are derived without invoking approximations. This covers also the utilization of *error filters* which is common practice in the adaptation of recursive filter structures.

The second part of the thesis is devoted to the application of the described algorithm class to the tracking of time-varying environments. As a result, the tracking behaviour can be described as a filtering operation on the time evolution of the coefficients of a reference model which is imposed through the environment but not directly observable. Due to the general structure of the adaptation algorithms, this learning filter is always linear and of first order. To facilitate the incorporation of prior knowledge about the expected time evolution, the algorithm structure needs to be extended with so-called *coefficient filters*. This allows to tailor the tracking behaviour in response to practical demands as linear higher-order filtering or nonlinear filtering. In conclusion, a series of topical proposals for such coefficient filters is discussed on the basis of the accordingly extended unified optimality criterion.

Contents

Kurzfassung	i
Abstract	ii
Acknowledgment	v
1 Introduction	1
1.1 Adaptive Signal Processing	1
1.2 Recursive Algorithms	2
1.3 Transversal Digital Filters	4
2 Optimal Recursive Adaptation	7
2.1 Introduction	7
2.2 Error Minimization	10
2.2.1 The Stochastic Paradigm	10
2.2.1.1 Instantaneous Techniques	11
2.2.1.2 Time-Averaging Techniques	12
2.2.2 The Deterministic Paradigm	14
2.2.2.1 Instantaneous Techniques	14
2.2.2.2 Time-Averaging Techniques	16
2.2.3 Summary of the Error Minimization Approach	17
2.3 Joint Recursive Optimality	18
2.3.1 Introducing the General Framework	18
2.3.1.1 An Illustrative Example	19
2.3.1.2 The General Framework	21
2.3.2 LMS-Type Algorithms	25
2.3.2.1 The LMS Algorithm	25
2.3.2.2 The Projection Algorithm	26
2.3.3 Individual Coefficient Adaptation	28
2.3.3.1 The Variable Step Algorithm	28
2.3.3.2 The Individual Adaptation Algorithm	29
2.3.3.3 The Signed Regressor Algorithm	30
2.3.4 RLS-Type Algorithms	31
2.3.4.1 The LMS/Newton Algorithm	32
2.3.4.2 Self-Orthogonalizing Algorithms	34
2.3.4.3 ‘Exact’ Least Squares Algorithms	35
2.3.4.4 Directional Forgetting Algorithms	40
2.3.4.5 Other Modified RLS Algorithms	43

2.3.5	Modified Error Measures	44
2.3.5.1	Signed Error Algorithms	44
2.3.5.2	Filtered Error Algorithms	47
2.3.6	Summary of the Joint Recursive Optimality Framework	55
2.4	Criticism and Conclusion	58
3	Adaptation in Time-Varying Environments	61
3.1	Introduction: Convergence and Tracking	61
3.1.1	Convergence in Time-Invariant Environments	62
3.1.2	Large Parameter Variation and Detection of Jumps	64
3.1.3	Smooth Time Variation and Tracking	65
3.2	Tracking of Smooth Time Variations	66
3.2.1	Coefficient Vector Evolution in a Transform Domain	66
3.2.2	Misalignment Vector and Learning Filter Interpretation	73
3.2.3	Summary of Tracking Analysis	75
3.3	Models for the Time Evolution of Filter Coefficients	76
3.3.1	The Deterministic Paradigm: Band-Limited Evolution	77
3.3.2	The Stochastic Paradigm: Smoothness Priors	80
3.4	Filtered Coefficient Algorithms	83
3.4.1	Introduction: In-Loop Filtering	83
3.4.2	Coefficient Prediction	87
3.4.3	Generalized Leakage	91
3.4.4	Multi-Step Algorithms	92
3.4.5	Post-Filtering Algorithms	94
3.5	Comparison and Criticism	96
4	Conclusion	99
4.1	Summary of Main Results	99
4.2	Directions for Further Research	102
A	Recursive Least Squares Algorithms	105
A.1	Derivation of a Recursive Solution for the Filter Coefficients	105
A.2	The Matrix Inversion Lemma	106
A.3	Positive Definiteness of the Auto-Correlation Matrix	107
B	Joint Recursive Optimality	111
B.1	The Quadratic Error Case	111
B.2	The Error Magnitude Case	114
C	Quadratic Analysis of Tracking Properties	117
C.1	Excess Error Due to Observation Noise	117
C.2	Excess Error Due to Coefficient Lag	120
	Bibliography	125
	Curriculum Vitae	145

“Muchas veces tomé la pluma para escribilla, y muchas la dejé, por no saber lo que escribiría.”

MIGUEL DE CERVANTES, Prólogo,
El Ingenioso Hidalgo Don Quijote de la Mancha, Madrid, 1605.

Acknowledgment

This thesis integrates varied influences from several people. I want to express my sincere thanks to all of them for their encouragement and support:

First of all, *Prof. Dr. W. Mecklenbräuker* has supervised my work throughout with the type of discreet guidance that is maybe the key ingredient to create a stimulating and freethinking atmosphere. He also provided a thorough scrutiny of my results and thereby contributed substantially to the final form of this thesis¹. My second advisor, *Prof. Dr. M. Deistler* was instrumental in orientating me towards a deeper understanding of the statistical basis of adaptive signal processing. True enough, both advisors have shown themselves most generous in complying with my exactions as to time limits in the final rush to complete this work.

Within my working environment at the University of Technology, Vienna, particular contributions have been made by *Dr. G. Doblinger* in keeping up a steady exchange of ideas on adaptive filtering and developing a simulation software package that proved so useful in experimenting with these ideas. *Prof. Dr. H. Weinrichter* and *Dr. F. Hlawatsch* have proof-read parts of the typescript and *Dipl.-Ing. W. Krattenthaler* has introduced me to the art of L^AT_EX typesetting. Several colleagues took the burden of my regular duties in the final stage of this work and thereby gave evidence of their friendship and team-spirit.

An important initial momentum to get engaged in adaptive filtering was my visiting period at the *Natuurkundig Laboratorium* in Eindhoven. The weekly meetings of the ‘adaptive filter club’ with its lively discussions provided a solid basis to get started in a field then new to me. Other ‘outside’ influences came from various people who provided me with references to work either unknown to me or not yet available. Such support has been given by *Dr. S. Bittanti*, *Dr. J. Bellegarda*, *Dr. R. Bitmead* and *Dr. M. Niedźwiecki*.

This thesis might also have been written without the help of my wife Maria. I am glad it was not. Her assistance in typing the text was maybe her least contribution to making the work progress smoothly. All the other ‘little’ things are more difficult to express and are aptly paraphrased as loving participation.

¹Of course, full responsibility for any remaining mistakes or erroneous arguments rests with me.

Chapter 1

Introduction

- The purpose of this chapter is to explain the title of the thesis: the concepts of adaptive signal processing, recursive algorithms, and transversal digital filters are commented upon.

1.1 Adaptive Signal Processing

The topic of this thesis is centered within the rapidly growing field of *adaptive signal processing* as documented in the large number of recently published text books [4, 14, 49, 81] [91, 96, 176, 224, 238] and special issues of journals [214, 215, 216, 217, 218] devoted to that subject. These references also give insight into the wide range of applications for adaptive signal processing, including channel equalization, echo and noise cancellation, signal enhancement and restoration, system identification and signal analysis, source coding for digital speech and image transmission, adaptive antenna arrays, and many more. In an attempt to cover such a variety of systems the following general definition¹ is proposed:

Definition 1 *A signal processing system is called adaptive if it processes different input signals in different, signal-dependent ways while pursuing an objective common to all.*

At the core of many adaptive signal processing systems resides an *adaptive filter*, i.e. a filter whose parameters are controlled by an *adaptation algorithm*. Figure 1.1 shows that parameter adaptation may depend on the input signal, the output signal, or an optional reference signal. From the first two signal dependencies, the *non-linearity* of the overall adaptive system follows—no matter whether the filter structure without adaptation would be classified as linear or nonlinear. This non-linearity gives rise to a couple of non-trivial questions regarding the adaptive system’s ‘optimal’ design and its stability. Approaching the problem from another point-of-view, questions regarding the ‘tracking’ capabilities in time-varying environments arise when regarding the adaptive filter *as if* controlled in a time-varying way.

Following this introductory chapter, Chapter 2 sets out to clarify the theoretical basis of what is actually meant by *optimality* in adaptive filter design and introduces a unified

¹This definition covers also *property-restoring* algorithms [224, Chapter 6] and *structurally adaptive systems* [222]. For a general discussion of adaptivity see also [238, Chapter 1].

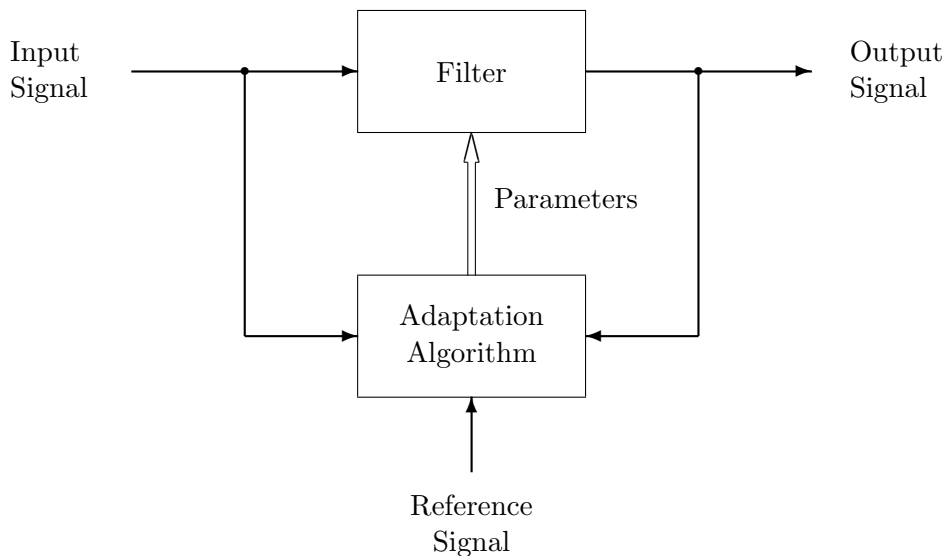


Figure 1.1: General adaptive filter configuration.

framework for a wide class of adaptation algorithms. Chapter 3 will shift the focus towards application of adaptive filters in time-varying environments. After a general analysis of tracking properties, the concept of *coefficient filters* is introduced that allows to tailor adaptation behaviour to specific prior knowledge originating from the application. Both parts of this thesis will persistently make use of both the deterministic and the stochastic point of view in an attempt to examine their respective assets. Chapter 4 concludes with a discussion of open problems and suggestions for further research.

1.2 Recursive Algorithms

In order to adjust the parameters of an adaptive filter, an adaptation algorithm is needed. Such algorithms can be discussed along several lines: the objective they are designed for, their performance, or their mode of operation. While the former two points will receive full attention in later chapters, the last is shortly addressed here.

Off-line operation refers to the mode where the execution of the algorithm is decoupled from data collection. This mode is a priori designed for a finite amount of data which are simultaneously available in the machine's memory. Therefore, off-line algorithms may use the same data several times in a re-optimization procedure to find the optimal parameter adjustment. This possibility allows to differentiate algorithms which provide a *direct solution* for the parameters from those which perform an *iterative search* [81, pp. 303–308].

On-line operation refers to the mode where the algorithm is executed at the same time while data are collected from their inputs. This mode is a priori designed to run continuously for an infinite period of time. This fact imposes certain restrictions on the

algorithm structure as only a finite amount of data can be memorized. Old data is, therefore, discarded continuously and cannot be used for re-optimization at a later stage. This ‘forgetting’ process can be organized in two manners [96, pp. 6–7]:

- *Block processing* deals with a whole block of data stored in a temporary buffer. The filter parameters are once adjusted for this particular block of data and remain unchanged until the next block of data² is moved into the buffer. Most importantly, *for each data block the optimal parameters are found independently of all previous blocks*. Thus old data is totally forgotten when moving from one block to the next. There is no long-term memory in such an algorithm. This feature is most beneficial for the long-term stability of adaptation algorithms, cf. also the general remarks in [250, pp. 4–5].
- *Recursive processing*³ finds an optimal parameter adjustment usually for each new incoming sample of data⁴. Most importantly, *this adjustment is made dependent on previous adjustments*. To this end, the algorithm propagates a set of *state variables* which represent a condensed history of the data (e.g. the latest parameter values, certain time-average statistics such as the auto-correlation of the data etc.). Old data is only partially forgotten in the state-transition equations which form the new state from a combination of old states with the *innovation* contained in the new data sample⁵. These algorithms have a long-term memory which is the source of potential long-term stability problems as mentioned before. Therefore, special care has to be taken in the design of partial forgetting as found in recursive algorithms, see e.g. Section 2.3.4.4.

Table 1.1 summarizes the discussion of adaptation algorithms w.r.t. their mode of operation⁶.

²The blocks may be spaced contiguously or with—sometimes considerable—overlap.

³For a comprehensive theoretical analysis of recursive processing see also [91, Section 6.2], [116, pp. 136–139] [150, Section 1.4] and [229].

⁴There are exceptions, of course, such as the ‘block adaptation’ LMS algorithm [47] and [224, p. 188] where the actual filter parameter adjustment occurs only after accumulation of the results of several successive observations. In spite of its name this algorithm processes the data recursively

⁵E.g. by simple averaging as done in algorithms with ‘exponential forgetting’.

⁶This table may be complemented with the following examples for the various operation modes:

- *Off-line processing, direct solution*: consider the design of an optimum *Wiener filter* for signals with known or estimated a priori statistics [91, Chapter 3].
- *Off-line processing, iterative search*: A typical example is *adaptive equalization* [196] for transmission over an unknown yet time-invariant channel. In that case, a training sequence or *preamble* can be transmitted prior to the actual data sequence and used for *channel identification*. The identification algorithm may be organized in an iterative mode, e.g. using the method of steepest descent or Newton’s method [238, Chapter 4].
A second example is the design of a *non-uniform quantizer* for minimum quantization noise power [99, Section 4.4]. Given a sufficiently long training sequence, the iterative *Lloyd-Max-algorithm* [145, 151] adapts the quantizer parameters to the long-term statistics of the signal. With this algorithm, successive iterations process the same training sequence in an attempt to re-optimize the quantizer parameters.
- *On-line block processing*: A prominent application area is speech signal processing in general [198]. The assumption of *short-time stationarity* for speech signals leads to a variety of short-time signal analysis and modeling approaches which use blocks or ‘frames’ of speech data with typical lengths

Table 1.1: ADAPTATION ALGORITHMS AND THEIR MODE OF OPERATION

Off-line	Direct solution
	Iterative search
On-line	Block processing
	Recursive processing

The key source of misunderstanding in such a discussion is to get mixed up in the use of the terms ‘iterative’ and ‘recursive’. This section should have provided some grounds for their distinction. This thesis will exclusively deal with recursive algorithms for on-line operation.

1.3 Transversal Digital Filters

Besides the adaptation algorithm, the *filter structure* constitutes an essential feature of an adaptive signal processing system. The reasons to choose a *digital filter* for signal processing are manifold and well-known [188, 199]. In *adaptive signal processing* the high flexibility of digital filters is the key issue: real-time adjustment of the filter parameters (or even the filter structure) is easy with a programmable digital computer implementation and allows for sophisticated adaptation laws⁷.

The reasons to concentrate on the *transversal filter structure* in this thesis mainly stem from two roots:

1. Transversal filters have a couple of advantageous properties for digital signal processing in general, cf. [188, Chapters 5 and 9] and [199, Chapters 5 and 9]:
 - simple control of finite-precision arithmetic effects,
 - high flexibility w.r.t. transfer function specification,
 - guaranteed stability both with time-invariant and time-varying parameters,
 - simple implementation on vector-oriented processors like standard digital signal processor chips due to the inner-product structure of the filter equation.
2. For adaptive signal processing, the advantage of the transversal filter structure originates again from the *inner-product structure* of the filter equation. The filter output is computed as a linear combination of delayed input signal samples, see Fig. 1.2. The weights in the linear combiner are the adapted parameters. Thus the effects of parameter adjustment are directly observable at the output, without delay or other

of 20 ms each. *Linear predictive speech coders* and several other modern parametric and hybrid speech coders [212] work in an adaptive block processing mode.

- *On-line recursive processing*: The most-widely used algorithm here is WIDROW’s LMS algorithm [238, Chapter 6]. A cornucopia of recursive algorithms will, anyway, appear in subsequent chapters of this thesis.

⁷An account of analog (sampled-data) adaptive filters can be found in [49, Sections 7.3 and 7.4].

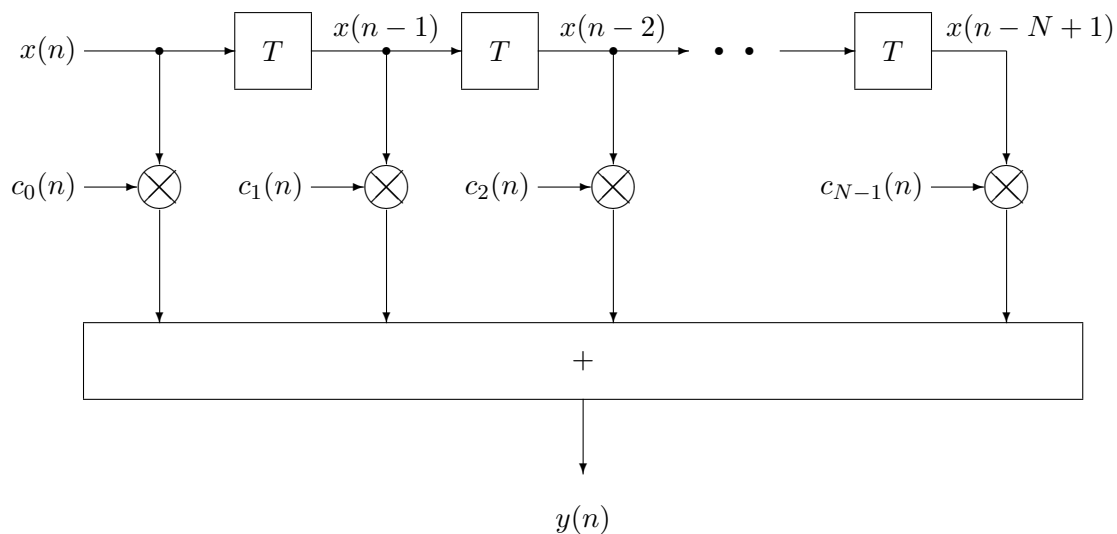


Figure 1.2: Transversal filter structure.

linear/nonlinear distortions. This allows for a simple derivation, analysis and implementation of the associated adaptation algorithms which get more involved for other structures such as lattice filters [71] [81, pp. 389–400], cascade-form filters [39, 69, 159], or state space filters [100]. It should, however, be emphasized that many of the results of this thesis apply as well to non-transversal filter structures if they can be described as a general *linear combiner structure where the filter parameters directly enter as weights*⁸. Some comments on this topic will appear in Subsection 2.3.5 and Chapter 4.

⁸This *linear adaptation class* was first described by GERSHO in [74] as referenced in [75]. See also [238, Chapter 2]. Important members of this class are direct-form IIR-filters in their equation error formulation [102, 167, 209], CHANG's generalized filter structure for data orthonormalization [35, 192], non-linear Volterra filters [115, 211, 244], and the class of adaptive filters described by PALMIERI and BONCELET [189] which incorporates generalized non-linear preprocessing of the data such as memoryless nonlinearities or combined time and rank order operations.

Chapter 2

Optimal Recursive Adaptation

• The purpose of this chapter is two-fold: To assess the standard approaches for derivation and interpretation of adaptation algorithms, both in their stochastic and deterministic settings, and to present the new and versatile *joint recursive optimality* approach. It will be shown that many adaptation algorithms trade time variance of the filter coefficients for error signal power.

2.1 Introduction

As adaptive signal processing systems are designed by engineering people there is a tendency towards calling such designs optimal. The ensuing question—*optimal with respect to what?*—sometimes remains with quite unsatisfactory answers. It is the purpose of this chapter to assess the tools for the derivation and interpretation of adaptation algorithms as currently in wide-spread use. These tools can largely be subdivided into two classes, belonging either to the *stochastic paradigm* or to the *deterministic paradigm*. Both will be shown to have their fallacies. As a second step, a new *unified framework* for algorithm design and analysis is proposed¹ that circumvents some of the conventional pitfalls.

To set a working environment for the subsequent discussion, a detailed outline of the adaptive signal processing system is presented, see Figure 2.1.

- The *input* to the adaptive filter is a scalar real-valued discrete-time signal $x(n)$ where n is the time index.
- This signal is fed into the chain of *delay elements* of the transversal filter structure as illustrated in Fig. 1.2. At time n , the samples $x(n), x(n-1), \dots$ through $x(n-N+1)$ are simultaneously present in the filter. They are combined into the N -component *data vector*²

$$\mathbf{x}(n) = [x(n), x(n-1), \dots, x(n-N+1)]^T. \quad (2.1)$$

- The filter output $y(n)$ is computed from the inner product of the data vector $\mathbf{x}(n)$

¹Several aspects of this framework have been published before, cf. [119, 120] and the summary discussion in [14, Section 7.10].

²The superscript T denotes *transposition* of vectors and matrices.

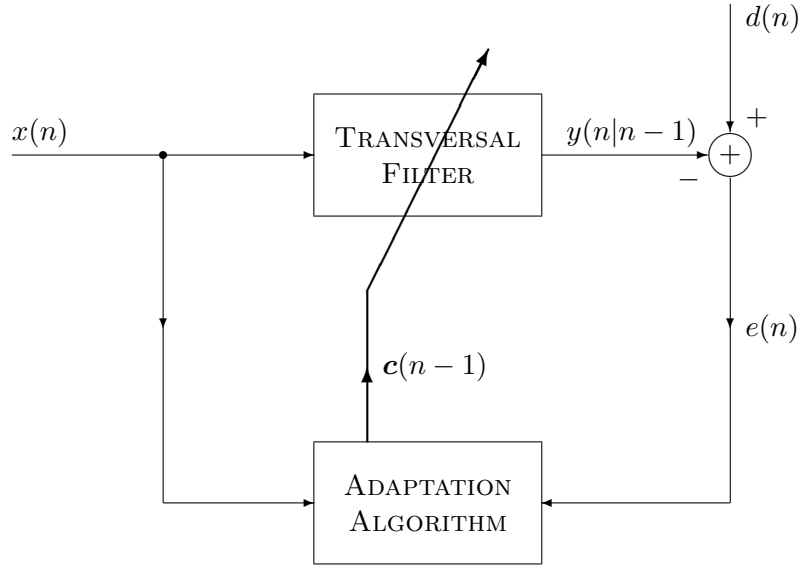


Figure 2.1: Outline of adaptive transversal filter (Bold lines denote vectors).

and the coefficient vector \mathbf{c}

$$y(n) = \mathbf{c}^T \mathbf{x}(n) = \sum_{k=0}^{N-1} c_k x(n-k) \quad (2.2)$$

- The filter operation is performed prior to the update of the coefficients. Therefore, the precise time dependence of the coefficient vector is denoted by $\mathbf{c}(n-1)$ and the corresponding filter output is called the *a priori filter output*

$$y(n|n-1) = \mathbf{c}^T(n-1) \mathbf{x}(n) = \sum_{k=0}^{N-1} c_k(n-1) x(n-k) \quad (2.3)$$

- This filter output is compared to its ‘desired’ value³ which is provided as the *reference signal* $d(n)$. The difference between the reference signal and the filter output is called the *a priori error*

$$e(n) = d(n) - y(n|n-1) = d(n) - \mathbf{c}^T(n-1) \mathbf{x}(n) \quad (2.4)$$

- This error signal reflects the *innovation* contained in both the new signal sample $d(n)$ and $\mathbf{x}(n)$, i.e. the amount of information not predictable from the past [91, pp. 90–92, 269–287] [109, 110]. This innovation may be attributed e.g. to time-varying changes or noise in the adaptive filter’s application environment. For the present, the terms ‘a priori error’ and ‘innovation’ will however be regarded as equivalent while their difference will become clear at a later stage, cf. Remark 1 of Subsection 2.3.5.2. on page 48.

³There are also adaptation algorithms which do not require an explicit reference signal such as GRIFITH’S P-vector algorithm [238, pp. 412–414] or the property restoring algorithms of [224, Chapter 6]. They will not receive further consideration here.

- The innovation is the main input to the *coefficient adaptation algorithm*. The other input is the data vector $\mathbf{x}(n)$ (and possibly some of its time-average statistics). Combining these two inputs with the most recent coefficient vector $\mathbf{c}(n-1)$, the update law for the computation of the coefficient vector $\mathbf{c}(n)$ usually displays the following general pattern [102][103, p. 5] [119]:

$$\begin{bmatrix} \text{new coefficient} \\ \text{vector} \end{bmatrix} = \begin{bmatrix} \text{old coefficient} \\ \text{vector} \end{bmatrix} + \begin{bmatrix} \text{step} \\ \text{size} \end{bmatrix} \cdot \begin{bmatrix} \text{(function of)} \\ \text{innovation} \end{bmatrix} \cdot \begin{bmatrix} \text{(function of)} \\ \text{data vector} \end{bmatrix} \quad (2.5)$$

This pattern will receive a direct explanation in Section 2.3 within the *joint recursive optimality* framework. For now, it is exemplified by two algorithms which serve as a kind of reference in the subsequent discussion:

1. The **Least Mean Squares Algorithm (LMS)**⁴

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mu e(n)\mathbf{x}(n) \quad (2.6)$$

where the *step size* μ is a real, time-independent scalar.

This is obviously the simplest form to implement the general pattern (2.5). It is also the algorithm which has found the widest application so far [238, Part IV]

2. The *Exponentially Weighted Recursive Least Squares Algorithm (RLS)* [91, Chapter 8][96, pp. 66–75]

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mathbf{R}^{-1}(n)e(n)\mathbf{x}(n) \quad (2.7a)$$

$$\mathbf{R}^{-1}(n) = \frac{1}{\lambda} \left[\mathbf{R}^{-1}(n-1) - \frac{\mathbf{R}^{-1}(n-1)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)}{\lambda + \mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \right] \quad (2.7b)$$

$$\text{with } 0 < \lambda \leq 1. \quad (2.7c)$$

This algorithm requires the recursive propagation of the ‘step-size’ matrix $\mathbf{R}^{-1}(n)$ which has the interpretation of being the inverse of an auto-correlation matrix estimate $\mathbf{R}(n)$ computed from an exponentially weighted time average:

$$\mathbf{R}(n) = \lambda\mathbf{R}(n-1) + \mathbf{x}(n)\mathbf{x}^T(n) \quad (2.8)$$

Eq. (2.8) can be shown to be equivalent to Eq. (2.7b)[cf. Appendix A].

In the following section, the conventional approach to the derivation of algorithms (2.6) and (2.7a-c) is presented. This approach is generally phrased in terms of the minimization of some measure of the error signal⁵.

⁴Proper algorithm initialization is assumed throughout this thesis and only discussed in selected cases.

⁵For a thorough discussion of various approaches to algorithm derivation see also [150, Chapter 2] and [228, Chapters 4 and 5].

2.2 Error Minimization

The design of the update algorithm for an adaptive filter starts out with a simple purpose in mind: the filter output $y(n)$ should approximately match its ‘desired’ form, i.e. the reference signal $d(n)$. There are several ways to quantify the mismatch between the output and the reference signal, i.e. to measure the error signal. They may be classified as either *stochastic* or *deterministic*, emphasizing either the *time-average* or *instantaneous* aspect. The following Subsections reflect this dual classification.

2.2.1 The Stochastic Paradigm

The stochastic paradigm in error minimization is often used in textbooks as a first introduction to adaptation algorithms. It can be recognized from the occurrence of *expectation operators* in the derivation of the algorithms⁶. These operators are usually defined on stationary stochastic processes which serve as embedding processes for the given discrete-time signals, in particular for the input signal $x(n)$ and the reference signal $d(n)$. Due to the underlying *stationarity assumption*, the following 4 step procedure is required for algorithm derivation:

1. Define the time-invariant ‘optimum’ coefficient vector \mathbf{c}_{opt} as the one to minimize the *expected squared error*

$$E\{[d(n) - \mathbf{c}_{opt}^T \mathbf{x}(n)]^2\} \stackrel{!}{=} \min \quad (2.9)$$

or a similar error measure defined in terms of expectation.

2. Define a strategy to find the solution to the minimization problem (2.9). Both direct or iterative search solutions are possible. These solutions presuppose the knowledge of the second-order statistics of the processes underlying $x(n)$ and $d(n)$.
3. Replace the known statistics in the algorithms with time-average or instantaneous estimates computed from the actual signals $x(n)$ and $d(n)$. This replacement results in recursive algorithms which continuously refine the estimate of the optimum vector using a ‘decreasing step-size’ sequence.
4. The algorithms derived in steps 1 through 3 are *a posteriori* recognized to be unsuited to tracking of time-varying changes because the decreasing step-size sequence makes the filter coefficients settle at a time-invariant optimum. Time-varying application environments constitute, however, one of the major incentives to develop adaptive signal processing systems. Therefore, the underlying stationarity assumption is thrown over and *ad hoc* modifications are crafted onto the algorithms to allow for continuous tracking using an approximately ‘constant step-size’ sequence.

The discussion of this procedure is started with an algorithm which performs an iterative search in the direction of instantaneous estimates of the *error gradient*: the LMS algorithm.

⁶This should be distinguished from the stochastic approach to *performance analysis* of adaptation algorithms where performance indicators such as convergence speed are averaged over a given signal class or stochastic process with reference to a given deterministic adaptation algorithm.

2.2.1.1 Instantaneous Techniques

An iterative search strategy for solving the minimization problem (2.9) starts out with a preliminary solution for the coefficient vector \mathbf{c}_{k-1} obtained from iteration $k-1$. To improve on that solution, the *gradient* of the cost function is evaluated for the given coefficient vector \mathbf{c}_{k-1} :

$$\begin{aligned} \nabla_{\mathbf{c}} E\{[d(n) - \mathbf{c}^T \mathbf{x}(n)]^2\} \Big|_{\mathbf{c}=\mathbf{c}_{k-1}} &= 2E\{-\mathbf{x}(n)[d(n) - \mathbf{c}_{k-1}^T \mathbf{x}(n)]\} \\ &= 2(\mathbf{R}\mathbf{c}_{k-1} - \mathbf{p}) \end{aligned} \quad (2.10a)$$

with the *cross-correlation vector*

$$\mathbf{p} = E\{d(n)\mathbf{x}(n)\} \quad (2.10b)$$

and the *auto-correlation matrix*

$$\mathbf{R} = E\{\mathbf{x}(n)\mathbf{x}^T(n)\} \quad (2.10c)$$

Moving in the direction of the *negative* gradient (2.10a), the new iteration computes

$$\mathbf{c}_k = \mathbf{c}_{k-1} + \mu_k(\mathbf{p} - \mathbf{R}\mathbf{c}_{k-1}) \quad (2.11)$$

where μ_k is a real, scalar step-size sequence. A sufficient condition for the convergence of (2.11) is the boundedness of the step size according to

$$0 < \mu_k < \frac{2}{\lambda_{max}} \quad (2.12)$$

where λ_{max} is the *maximum eigenvalue* of the auto-correlation matrix \mathbf{R} , cf. [96, p. 43].

The gradient search is an example of step 2 of the general procedure. Entering step 3, instantaneous estimates are used to replace the ensemble averages of the ‘true’ statistics (2.10b) and (2.10c). These estimates are obtained by *omitting* the expectation operators in the respective equations:

$$\hat{\mathbf{p}}(n) = d(n)\mathbf{x}(n) \quad (2.13a)$$

cross-correlation vector

$$\hat{\mathbf{R}}(n) = \mathbf{x}(n)\mathbf{x}^T(n) \quad (2.13b)$$

auto-correlation matrix

They are plugged into (2.11) while—at the same time—the *iteration index* k is reinterpreted as a *time index*, i.e. $k \rightarrow n$:

$$\begin{aligned} \mathbf{c}(n) &= \mathbf{c}(n-1) + \mu(n)[d(n)\mathbf{x}(n) - \mathbf{x}(n)\mathbf{x}^T(n)\mathbf{c}(n-1)] \\ &= \mathbf{c}(n-1) + \mu(n)e(n)\mathbf{x}(n) \end{aligned} \quad (2.14a)$$

$$\text{with } e(n) = d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) \quad (2.14b)$$

This algorithm has the following interpretation: whenever a new estimate of the required statistics is available from the observed signals, a further step of the iterative gradient search is executed. Thus the algorithm has been turned from an iterative (off-line) algorithm for known statistics (2.11) into a recursive (on-line) algorithm (2.14a,b) which performs an implicit time-average estimation of the a priori unknown statistics. This transition from iteration to recursion is referred to as *stochastic approximation* [96, pp. 49–53] [150, pp.42–48] [251, pp. 34–37] and is only consistent if

1. the existence of a time-invariant optimum is guaranteed (stationarity assumption), and if
2. the step-size sequence $\mu(n)$ satisfies the conditions [96, p. 53][251, p. 36]:

$$\mu(n) > 0 \quad (2.15a)$$

$$\sum_{n=1}^{\infty} \mu(n) = \infty \quad (2.15b)$$

$$\sum_{n=1}^{\infty} \mu^2(n) < \infty. \quad (2.15c)$$

Condition (2.15c) implies a *vanishing step size* $\mu(n)$ for $n \rightarrow \infty$. Thus the algorithm is unsuited to tracking of time-varying environments.

Step 4 of the general procedure consists in an *ad hoc* modification of the algorithm that removes this restriction. To this end, the step size is set to a *constant value* for all times n :

$$\mu(n) = \mu \quad \text{for all } n. \quad (2.16)$$

This choice makes (2.14a) identical to the ‘Least Mean Squares’ algorithm of (2.6). It is also referred to as a *stochastic gradient algorithm*. As this final form of the algorithm is not coherently derived from the optimization problem (2.9), it is not amazing that the algorithm *does not meet* the original objective of error minimization!

- Even in strictly stationary environments the LMS algorithm exhibits some *excess mean square error* or *misadjustment* exceeding the theoretical least squares value of (2.9). This effect is due to the statistical coefficient fluctuations introduced via the non-vanishing step size and remaining even after initial convergence of the algorithm [238, Chapter 6].
- On the other hand, the LMS algorithm can track slow time variations of the application environment reasonably well as will be shown in Chapter 3. This performance is again *not* to be expected from the original algorithm design objective which is directed towards a unique time-invariant optimum coefficient vector.

2.2.1.2 Time-Averaging Techniques

This approach is discussed within the context of a direct solution to the optimization problem (2.9):

$$\begin{aligned} E\{[d(n) - \mathbf{c}_{opt}^T \mathbf{x}(n)]^2\} &\stackrel{!}{=} \min \\ \Rightarrow \nabla_{\mathbf{c}} E\{[d(n) - \mathbf{c}^T \mathbf{x}(n)]^2\} \Big|_{\mathbf{c}=\mathbf{c}_{opt}} &= \\ 2(\mathbf{R}\mathbf{c}_{opt} - \mathbf{p}) &\stackrel{!}{=} 0 \\ \Leftrightarrow \mathbf{c}_{opt} &= \mathbf{R}^{-1}\mathbf{p} \end{aligned} \quad (2.17)$$

where definitions (2.10b) and (2.10c) have been used and where the existence of the inverse of the autocorrelation matrix \mathbf{R} is assumed⁷. The solution (2.17) requires first the knowledge of the statistics \mathbf{R} and \mathbf{p} of the process underlying $\mathbf{x}(n)$ and second the inversion of a $N \times N$ matrix. This equation is the result of step 2 ('direct solution') of the general procedure within the stochastic paradigm.

Proceeding into step 3, time-average estimates are used to replace the true statistics of (2.17), cf. [77] and [238, pp. 157–163]. These can be obtained by replacing the expectation operators with averages over the accumulated observations:

$$\hat{\mathbf{p}}(n) = \sum_{k=0}^n d(k)\mathbf{x}(k) = \hat{\mathbf{p}}(n-1) + d(n)\mathbf{x}(n) \quad (2.18a)$$

$$\text{and } \hat{\mathbf{R}}(n) = \sum_{k=0}^n \mathbf{x}(k)\mathbf{x}^T(k) = \hat{\mathbf{R}}(n-1) + \mathbf{x}(n)\mathbf{x}^T(n) \quad (2.18b)$$

Note that in these estimates a normalizing $1/n$ factor has been dropped for simplicity as it cancels out after insertion of (2.18a,b) into (2.17):

$$\mathbf{c}(n) = \hat{\mathbf{R}}^{-1}(n)\hat{\mathbf{p}}(n) \quad (2.19)$$

which is already a computable form of the 'direct-solution time-average' algorithm as it does not presuppose knowledge of the process statistics. It still requires the full $N \times N$ matrix inversion. For practical purposes it is, therefore, desirable to turn (2.19) into a recursive algorithm paralleling the recursive definition of the time-average estimates (2.18a,b). To this end, a two-step procedure derives first a recursion for $\mathbf{c}(n)$ in terms of $\mathbf{c}(n-1)$ and next a recursion for $\hat{\mathbf{R}}^{-1}(n)$ in terms of $\hat{\mathbf{R}}^{-1}(n-1)$. Details are given in Appendix A. The result is the following algorithm:

$$\mathbf{c}(n) = \mathbf{c}(n-1) + e(n)\hat{\mathbf{R}}^{-1}(n)\mathbf{x}(n) \quad (2.20a)$$

$$\text{where } e(n) = d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) \quad (2.20b)$$

$$\text{and } \hat{\mathbf{R}}^{-1}(n) = \hat{\mathbf{R}}^{-1}(n-1) - \frac{\hat{\mathbf{R}}^{-1}(n-1)\mathbf{x}(n)\mathbf{x}^T(n)\hat{\mathbf{R}}^{-1}(n-1)}{1 + \mathbf{x}^T(n)\hat{\mathbf{R}}^{-1}(n-1)\mathbf{x}(n)} \quad (2.20c)$$

The a priori filtering operation and error computation (2.20b) arises implicitly from the development of the recursive algorithm. Note that there is *no* such filtering/error computation in the fully equivalent direct-solution algorithm (2.19) with (2.18a,b). The choice between the two equivalent algorithm forms is governed by application requirements: If the application can do with a single 'good' estimate $\mathbf{c}(n)$ that is evaluated only after accumulating the statistics $\hat{\mathbf{R}}(n)$ and $\hat{\mathbf{p}}(n)$ over a sizable time period (*initial training* period or *adaptation delay*) then the direct solution is more efficient than the recursive form. On the other hand, if the filter should be operated right from the beginning—i.e. with the still imperfect estimates of the searched optimum—then the recursive form is the preferred choice.

⁷Otherwise the *pseudo-inverse* of \mathbf{R} may be used, cf. [91, pp. 331–341][248]. An analysis of the properties of auto-correlation matrices can be found in [14, Chapter 3] and [91, Chapter 2].

In both cases the algorithm is tuned to the underlying stationarity assumption which is reflected in the ‘step size’ of algorithm (2.20a). It equals the matrix-valued expression

$$\frac{\hat{\mathbf{R}}^{-1}(n-1)}{1 + \mathbf{x}^T(n)\hat{\mathbf{R}}^{-1}(n-1)\mathbf{x}(n)}$$

which converges to zero with increasing time n while $\mathbf{c}(n)$ approaches the time-invariant \mathbf{c}_{opt} defined in (2.17).

Step 4 of the general procedure provides the necessary modifications for time-varying environments. Instead of the *growing memory* of the time averages (2.18a,b), a *decaying memory* or even *finite memory* is used in the modified estimates. The *exponential* decay exemplifies this approach:

$$\hat{\mathbf{p}}(n) = \sum_{k=0}^n \lambda^{n-k} d(k) \mathbf{x}(k) = \lambda \hat{\mathbf{p}}(n-1) + d(n) \mathbf{x}(n) \quad (2.21a)$$

$$\text{and } \hat{\mathbf{R}}(n) = \sum_{k=0}^n \lambda^{n-k} \mathbf{x}(k) \mathbf{x}^T(k) = \lambda \hat{\mathbf{R}}(n-1) + \mathbf{x}(n) \mathbf{x}^T(n) \quad (2.21b)$$

$$\text{with } 0 < \lambda \leq 1. \quad (2.21c)$$

λ is the *forgetting factor* of the memory. Note that just as in (2.18a,b), a normalizing $(1 - \lambda)$ factor⁸ has been omitted as it is canceled out when solving for $\mathbf{c}(n)$ in 2.19. This simple modification turns⁹ the recursive algorithm of (2.20a-c) into the *exponentially weighted recursive least squares algorithm* (2.7a-c) of Section 2.1. This modified algorithm *does not meet* the original design objective of error minimization: just as for the LMS algorithm, an excess mean square error or misadjustment exceeding the theoretical least squares minimum of (2.9) persists even after convergence in stationary environments [14, pp. 195–199][49, p. 39] [96, pp. 294–296]. In time-varying environments the modified autocorrelation estimate (2.21b) results in a non-vanishing step size which allows continuous tracking as described in Section 3.2. This property could *not* be expected from the original objective (2.9) either.

2.2.2 The Deterministic Paradigm

The deterministic paradigm of error minimization is characterized through the definition of an ‘exact’ (i.e. non-probabilistic) expression for an error measure related to the filter coefficient vector $\mathbf{c}(n)$ at a particular time instant n . This measure is directly given in terms of samples of the signal sequences $\{\mathbf{x}(n)\}$ and $\{d(n)\}$. The discussion is again split into ‘instantaneous’ techniques and ‘time-averaging’ techniques.

2.2.2.1 Instantaneous Techniques

In Section 2.1 the a priori error has been defined as

$$e(n) = d(n) - y(n|n-1) = d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n)$$

⁸I.e. the normalization by the *effective window length* [14, pp. 63–65] [179, 193].

⁹For a derivation see Appendix A.

i.e. as the difference of the reference signal and the filter output computed with the ‘old’ coefficient vector $\mathbf{c}(n-1)$. In search for a coefficient update, it is natural to define the ‘new’ coefficient vector $\mathbf{c}(n)$ such that the error is reduced. To evaluate this reduction, the error ‘after the update’ or the *a posteriori error* is defined as

$$\epsilon(n) = d(n) - y(n|n) = d(n) - \mathbf{c}^T(n)\mathbf{x}(n) \quad (2.22)$$

where $y(n|n)$ is the filter output computed with the same input vector $\mathbf{x}(n)$ but with the updated coefficient vector $\mathbf{c}(n)$. The instantaneous techniques of the deterministic paradigm aim all at a minimal a posteriori error $\epsilon(n)$. As the corresponding optimality criterion

$$\epsilon(n) = d(n) - \mathbf{c}^T(n)\mathbf{x}(n) = 0 \quad (2.23)$$

has no unique solution in the coefficient vector $\mathbf{c}(n)$, several algorithm alternatives are considered in the literature:

Instantaneous Gradient Algorithms parallel the development of stochastic gradient algorithms in considering the gradient of the instantaneous squared error. A recursive update is obtained by moving the coefficient vector $\mathbf{c}(n-1)$ in the direction of the negative error gradient evaluated at $\mathbf{c} = \mathbf{c}(n-1)$ [14, pp. 97–100] and the so-called ‘MIT-rule’ in adaptive control [9] [10, pp. 106–111]:

$$\begin{aligned} \nabla_{\mathbf{c}}[d(n) - \mathbf{c}^T\mathbf{x}(n)]^2 \Big|_{\mathbf{c}=\mathbf{c}(n-1)} &= 2[d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n)](-\mathbf{x}(n)) \\ \Rightarrow \mathbf{c}(n) &= \mathbf{c}(n-1) + \mu\epsilon(n)\mathbf{x}(n) \end{aligned} \quad (2.24)$$

where (2.4) has been used. The resulting algorithm is easily recognized as the LMS algorithm (2.6). The present derivation provides the following interpretation: moving the coefficient vector in the direction of the negative error gradient will in general reduce this error—at least when considering the current observations. This error reduction is contingent upon a proper choice of μ :

$$\begin{aligned} |\epsilon(n)| &< |e(n)| \\ \left| d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) - (\mathbf{c}(n) - \mathbf{c}(n-1))^T\mathbf{x}(n) \right| &< |e(n)| \\ \Leftrightarrow |e(n)| \left| 1 - \mu\|\mathbf{x}(n)\|^2 \right| &< |e(n)| \\ \Leftrightarrow 0 < \mu < \frac{2}{\|\mathbf{x}(n)\|^2} \end{aligned} \quad (2.25)$$

Thus the instantaneous gradient approach gives insight into meaningful bounds for the step size μ , yet it does not provide grounds to explain either the excess error after convergence or the algorithm’s tracking abilities.

Zero-Forcing Algorithms¹⁰ are designed to meet (2.23) exactly. One approach consists in introducing the additional constraint [81, pp. 50–51]

$$\|\mathbf{c}(n) - \mathbf{c}(n-1)\|^2 \stackrel{!}{=} \min \quad (2.26)$$

¹⁰Cf. the corresponding terminology in channel equalizer design for zero intersymbol interference [141, pp. 325–329].

to obtain the unique solution

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \frac{e(n)\mathbf{x}(n)}{\|\mathbf{x}(n)\|^2} \quad (2.27)$$

with the time-varying step size $1/\|\mathbf{x}(n)\|^2$. This algorithm is known as the *projection algorithm*. Several algorithm modifications are in use, e.g. to avoid division by zero in case of $\|\mathbf{x}(n)\| = 0$, a small positive constant may be added to the denominator. Another modification [45] pre-multiplies the step size with a factor less than 1 so as to render the trade-off between misadjustment and convergence speed more flexible. MIKHAEL *et al.* specify in [168, 169] a general set of algorithms which meet (2.23) exactly:

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mathbf{M}(n)e(n)\mathbf{x}(n) \quad (2.28)$$

where $\mathbf{M}(n)$ is a *diagonal matrix* satisfying¹¹

$$\mathbf{x}^T(n)\mathbf{M}(n)\mathbf{x}(n) = 1 \quad (2.29)$$

The choice of the elements of $\mathbf{M}(n)$ is, however, based on heuristic arguments and allows no direct explanation of the algorithm properties, either.

2.2.2.2 Time-Averaging Techniques

These techniques look for a coefficient vector update that minimizes the time average of *several* squared ‘a posteriori errors’ evaluated over a window of past data samples. The corresponding deterministic error measure is defined as a function of the unknown coefficient vector \mathbf{c} and the time index n as

$$J(n, \mathbf{c}) = \sum_{k=0}^n [d(k) - \mathbf{c}^T \mathbf{x}(k)]^2 \quad (2.30)$$

and the optimal coefficient vector $\mathbf{c}(n)$ is chosen such that

$$\mathbf{c}(n) \stackrel{!}{=} \arg \min_{\mathbf{c} \in R^N} J(n, \mathbf{c}) \quad (2.31)$$

The solution to the minimization problem (2.31) is given by (2.19) with (2.18a,b). Just as in the stochastic paradigm, the recursive procedure (2.20a-c) can be used equivalently to compute $\mathbf{c}(n)$ in terms of $\mathbf{c}(n-1)$. This includes the definition of the a priori error (2.20b). Note that this error is the only error signal actually computed during the recursive operation of the algorithm while the ‘a posteriori errors’

$$d(k) - \mathbf{c}^T(n)\mathbf{x}(k) \quad \text{for } k < n$$

are only implicit quantities defined for the derivation of the algorithm from the error measure (2.30). Although this measure depends on the time index n , it is important to

¹¹Cf. equation (22) of [169]. Note that the minor differences from (2.29) are well understood from the fact that MIKHAEL *et al.* consider an adaptive IIR model in its equation-error formulation which results in two separate adaptive transversal filter structures. Furthermore they employ an explicit factor of 2 in the definition of the step size parameter.

understand that the optimum coefficient vector $\mathbf{c}(n)$ in (2.31) is defined as *time-invariant* over the whole data window stretching from $k = 0$ to $k = n$. As this window is steadily growing and as all error contributions receive equal weight, this algorithm cannot but converge to a time-invariant ‘optimum’ coefficient vector. Paralleling the stochastic paradigm once again, the algorithm has to be modified to be applicable to time-varying environments. In contrast to the stochastic paradigm, this modification cannot be applied at an intermediate step of algorithm derivation (i.e. after having determined the direct solution to the underlying optimization problem), but it is introduced right from the beginning in the error measure. Instead of a uniform error weighting recent errors receive more weight than older ones e.g. following an exponential weighting law:

$$J(n, \mathbf{c}) = \sum_{k=0}^n \lambda^{n-k} \left[d(k) - \mathbf{c}^T \mathbf{x}(k) \right]^2 \quad \text{with } 0 < \lambda < 1. \quad (2.32)$$

The updated coefficient vector $\mathbf{c}(n)$ is again defined by (2.31) with the error measure replaced by the above. The result is precisely the exponentially weighted recursive least squares algorithm (2.7a-c) of Section 2.1.

Summarizing this development, the exponentially weighted recursive least squares algorithm meets a deterministic time-averaged error minimization objective and is, for this reason, also called an ‘exact least squares’ technique. This fact should however not eclipse the above observations that

- the algorithm minimizes a measure of a posteriori errors which are irrelevant to the adaptive filter implementation, and that
- the coefficient vector $\mathbf{c}(n)$ is only ‘optimum’ if defined as time-invariant over a time window of a certain length.

Therefore, this deterministic derivation method does not allow any prediction of the tracking behaviour of the algorithm in time-varying environments, either. As explained later in Section 2.3.4, the optimality criterion (2.31) is only useful to establish the rapid *initial convergence* property of this algorithm.

2.2.3 Summary of the Error Minimization Approach

Error minimization has been shown to be a ubiquitous approach to the optimality design of adaptive filter algorithms. However, the preceding discussion has little bearing on the optimality with respect to performance of the described algorithms. This gap between the derivation or the design of an algorithm and its performance analysis is a major characteristic of the error minimization approach. Setting aside the instantaneous techniques of the deterministic paradigm discussed in Subsection 2.2.2.1, algorithm design is focussed on a time-invariant coefficient vector which is chosen such that a quadratic error measure is minimized. The performance analyses show that such ‘optimal’ algorithms do not respond to the latent needs of adaptive filter applications: they should not only work with unknown constant environments but with time-varying environments, too. To this end modified algorithms are virtually always used in practice. These modifications bring about the requested performance but sacrifice the explanatory power of a coherent derivation from first principles.

Instantaneous techniques of the deterministic paradigm constitute an exception to the above discussion. Zero-forcing algorithms, in particular, *do not focus on a time-invariant optimum coefficient vector but on the exact zeroing of the a posteriori error*. The algorithm derivation allows for a time-varying coefficient vector from the beginning, but leaves no room for *noise* which is an unfailing ingredient of any application environment. Driving the error exactly to zero cannot be meaningful from this point-of-view and inevitably results in coefficient fluctuations in excess of those found in other algorithms, cf. the results of the tracking analysis (3.23) and (C.16).

2.3 Joint Recursive Optimality

2.3.1 Introducing the General Framework

So far, two extremely opposed design principles have been encountered:

- *zero time variation* (time invariance) of the filter coefficients as a constraint for further error minimization
- *zero error* as a constraint for further specification of the coefficient vector update, e.g. as minimization of the time variation of the coefficient vector (as in the projection algorithm (2.26) and (2.27)).

Under the superficial opposition of these two principles hides an important common feature:

A meaningful design of an adaptation algorithm necessarily postulates a desired behaviour of *both* the error signal *and* the coefficient vector.

This fact is by no means surprising and can further be highlighted by two ‘nonsense’ algorithms which disregard this dual requirement.

Nonsense Adaptation Algorithm ‘A’. This design postulates a zero a posteriori error $\epsilon(n)$ without specifying any constraint on the coefficient vector $\mathbf{c}(n)$. One possible candidate for this design is given by

$$\epsilon(n) = 0 \tag{2.33a}$$

$$\mathbf{c}(n) = \frac{d(n)\mathbf{x}(n)}{\|\mathbf{x}(n)\|^2} \tag{2.33b}$$

$$y(n|n) = d(n) \tag{2.33c}$$

which results in a ‘perfect’ or ‘error-free’ adaptation at the price of a most rapidly varying coefficient vector $\mathbf{c}(n)$. This is rarely a desired behaviour as most applications look for a smooth time evolution of the coefficient vector. If the adaptive filter is for instance used as a signal analysis or signal modelling tool, the above algorithm is unlikely to show any explanatory power for the problem at hand. Note furthermore that applications such as adaptive noise or echo cancellation require a non-vanishing error signal as it forms the effective output signal of such systems.

Nonsense Adaptation Algorithm ‘B’. This second design postulates zero time variation of the coefficient vector $\mathbf{c}(n)$ without specifying any constraint on the error signal $\epsilon(n)$. One possible candidate for this design is given by

$$\epsilon(n) = d(n) \quad (2.34a)$$

$$\mathbf{c}(n) = \mathbf{c}(0) = \mathbf{0} \quad (2.34b)$$

$$y(n|n) = 0 \quad (2.34c)$$

which results in a perfectly time-invariant coefficient vector at the price of a high error signal power. This behaviour is again rarely desired in applications and bears no explanatory power for signal analysis or modelling, either. Note furthermore that neither algorithm ‘A’ nor ‘B’ produces any signal ($\epsilon(n)$ or $y(n|n)$) different from those already available without using the adaptive filter!

The *Joint Recursive Optimality* approach consists in carefully specifying the desired behaviour of both the time variation of the coefficients and the error signal. It is kept flexible by not aiming at one of the above extreme constraints (zero time variation or zero error) as these have been recognized as lying far from practical requirements. Therefore, the design is described as a *joint optimality* approach. It is a *recursive optimality* principle in that it incorporates the recursive operation of the resulting algorithms already in its definition: it assumes that the adaptive filter has been operated up to time instant $(n - 1)$ and defines the optimal update at time n only in terms of the accumulated deterministic history of the observations. Its rationale is first outlined in the context of the simple LMS algorithm to give as clear a picture as possible. Later on, it will be generalized to more advanced adaptation laws.

2.3.1.1 An Illustrative Example

In Section 2.2.2.1, the a posteriori error has been defined as

$$\epsilon(n) = d(n) - \mathbf{c}^T(n)\mathbf{x}(n) \quad (2.35)$$

Taking the time variation of the filter coefficients and the power of the error signal jointly into account, the optimality criterion is defined as

$$\|\mathbf{c}(n) - \mathbf{c}(n - 1)\|^2 + v^{-1}\epsilon^2(n) \stackrel{!}{=} \min \quad (2.36)$$

where v is a positive real weight controlling the trade-off between the coefficient vector variation (measured by the squared norm of its first difference) and the squared a posteriori error. The minimization problem (2.36) has the following explanation in words:

”Look for a new set of filter coefficients that do not differ much from the previous set while *simultaneously* keeping the error power as small as possible.”

The solution to the minimization problem is rendered *anschaulich* from geometrical considerations¹²[45] [240]. In Figure 2.2 the hyperplane π is defined perpendicular to the data vector $\mathbf{x}(n)$ as the geometrical locus of all vectors \mathbf{c} yielding zero a posteriori error $\epsilon(n)$:

$$\pi := \left\{ \mathbf{c} \in R^N \mid \epsilon(n) = d(n) - \mathbf{c}^T \mathbf{x}(n) = 0 \right\}$$

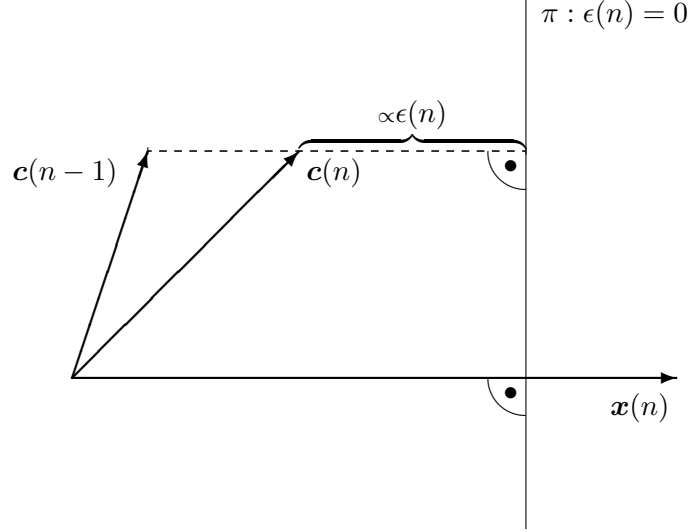


Figure 2.2: Geometrical interpretation of the joint recursive optimality criterion (2.36).

Moving $\mathbf{c}(n)$ out of this plane produces a non-zero $\epsilon(n)$ which is a direct measure of the perpendicular distance between $\mathbf{c}(n)$ and π . Given the previous coefficient vector $\mathbf{c}(n-1)$, (2.36) imposes a dual requirement on the locus of $\mathbf{c}(n)$:

- small geometrical distance from $\mathbf{c}(n-1)$ and
- small a posteriori error $\epsilon(n)$ i.e. small perpendicular distance from π .

These requirements are jointly met if $\mathbf{c}(n-1)$ is moved into $\mathbf{c}(n)$ along the direction perpendicular to π , i.e. parallel to the data vector $\mathbf{x}(n)$. Thereby the perpendicular distance from π can be brought down from the a priori error $e(n)$ to the a posteriori error $\epsilon(n)$ with the minimum ‘effort’ as far as coefficient variation is concerned. The precise size of this reduction follows in unique manner from (2.36), cf. Appendix B, and gives rise to the following adaptation algorithm:

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \frac{e(n)}{v + \|\mathbf{x}(n)\|^2} \mathbf{x}(n) \quad (2.37a)$$

$$\epsilon(n) = \frac{v}{v + \|\mathbf{x}(n)\|^2} e(n) \quad (2.37b)$$

Equs. (2.37a,b) recursively relate both posterior quantities $\mathbf{c}(n)$ and $\epsilon(n)$ to the prior quantities $\mathbf{c}(n-1)$ and $e(n)$. *The innovation observed at time n (i.e. the a priori error $e(n)$) is totally shared among the posterior error $\epsilon(n)$ and the time variation of the coefficients ($\mathbf{c}(n) - \mathbf{c}(n-1)$).* This is confirmed by evaluating

$$[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{x}(n) + \epsilon(n) = e(n) \quad (2.38)$$

$$\text{and } \|\mathbf{c}(n) - \mathbf{c}(n-1)\|^2 + v^{-1} \epsilon^2(n) = e^2(n) \quad (2.39)$$

¹²An exact analytical treatment of (the generalized form of) problem (2.36) is given in Appendix B.

where the first equality holds by definition of the quantities involved (no matter which adaptation algorithm is used) whereas the second equality holds only for the joint recursive optimality solution (2.37a,b). The weight v can be used to control the shares taken up by $\epsilon(n)$ and $[\mathbf{c}(n) - \mathbf{c}(n-1)]$ to account for the innovation. This is directly reflected in the performance analysis of algorithm (2.37a,b) in noisy time-varying environments cf. [236] and Section 3.2. The algorithm constitutes a variant of the projection algorithm discussed before and is also known as the *normalized LMS algorithm*.

Before generalizing the joint recursive optimality approach to a whole class of algorithms, a couple of remarks regarding this first simple example is appropriate:

Remark 1. The criterion (2.36) is also proposed by ROBERTS and MULLIS in [199, p. 279]. They use it to derive algorithm (2.37a,b) but do not carry the concept to the general framework presented below.

Remark 2. Similar criteria trading smoothness of coefficient variation for error power appear in the literature on algorithm design for tracking purposes [112, 113, 177, 178, 235] as discussed in Chapter 3. These references fail, however, in recognizing that already the standard algorithms such as LMS, RLS, and many more do satisfy a deterministic criterion of this form. This simple result makes up the rest of this chapter.

Remark 3. WIDROW, WINTER and BAXTER give in [240] an interpretation of the LMS algorithm as "*achieving the needed error correction with the smallest magnitude of weight vector change*". This statement is a re-phrasing of the optimality principle (2.36). Furthermore, the authors note that it has the potential of being generalized for the adaptation of layered neural nets which include even non-differentiable non-linearities. This is another instance of the wide scope of joint recursive optimality.

Remark 4. The derivation of (2.37a,b) from (2.36) follows a coherent, deterministic line. The optimality principle and the algorithm are *exactly equivalent*. Neither does it involve any approximation of stochastically defined quantities as in the stochastic paradigm of algorithm derivation nor does it make any direct postulate about the step size of the coefficient vector update as found in the instantaneous techniques of the deterministic paradigm. Unlike the exact least squares techniques, no claim is necessary about the optimality of $\mathbf{c}(n)$ for time indices less or greater than n . In particular, no reference is made to error signals which never occur in the adaptive filter's implementation.

2.3.1.2 The General Framework

The optimality criterion as given in (2.36) is tailored to the specific algorithm (2.37a,b). This approach is easily generalized to cover a wide class of algorithms. In this section a time-varying matrix-valued weighting parameter is first introduced into the optimality criterion. Later on in Section 2.3.5, the functional form of the criterion is changed from quadratic measures to others, including the concept of *error filters*. Chapter 3 will further enrich the approach with the concept of *coefficient filters* on the basis of 'hypermodels' describing the evolution of the coefficient vector with time. *Regressor filters* as they appear

in adaptation algorithms for non-transversal filter structures will receive only marginal comments.

As for now, (2.36) is rewritten with a time-varying matrix-valued weighting parameter as in

$$[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c}(n) - \mathbf{c}(n-1)] + \gamma^{-1}(n) \epsilon^2(n) \stackrel{!}{=} \min \quad (2.40)$$

where $\mathbf{G}(n)$ is called the *coefficient weighting matrix* and $\gamma(n)$ the *error weight*. These weighting parameters must be chosen subject to the following constraints:

$$\mathbf{G}(n) > 0, \quad \text{i.e. a symmetric and positive definite matrix,} \quad (2.41a)$$

$$\gamma(n) > 0 \quad (2.41b)$$

and are normalized¹³ such that

$$\gamma(n) + \mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n) = 1. \quad (2.42)$$

Paralleling the argument for the solution of (2.36), the generalized joint recursive optimality algorithm is found in Appendix B as

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mathbf{G}(n) e(n) \mathbf{x}(n) \quad (2.43a)$$

$$\epsilon(n) = \gamma(n) e(n) \quad (2.43b)$$

The solution (2.43a,b) is nicely summarized in the signal flow graph Figure 2.3¹⁴.

The interpretation of this signal flow graph sets out with the reference signal $d(n)$ which is compared to its ‘predicted value’, i.e. the a priori computed filter output $y(n|n-1)$. Their difference $e(n)$ defines the innovation observed at time n . This innovation is shared among the a posteriori error $\epsilon(n)$ and the coefficient vector increment $\mathbf{c}(n) - \mathbf{c}(n-1)$. To this end the innovation is scaled with the factors $\gamma(n)$ and $\mathbf{G}(n)\mathbf{x}(n)$, respectively. In this context, the error weight $\gamma(n)$ is also called the *conversion factor* (which converts the a priori error into the a posteriori error¹⁵) and the coefficient weighting matrix is also called the *gain matrix* (which controls the gain vector $\mathbf{G}(n)\mathbf{x}(n)$ of the coefficient update loop, a terminology arising from Kalman filter theory). In general, the latter multiplication induces a rotation of the data vector $\mathbf{x}(n)$. The actual coefficient update is done in a simple accumulator structure summing up the coefficient increments over time. The *innovation sharing among the time variation of the coefficients and the error signal power* is confirmed for this general algorithm structure by evaluating the cost function minimum attained for the solution (2.43a,b):

¹³The solution to the optimization problem (2.40) is unaffected by the choice of a *scale factor* common to both weights $\mathbf{G}(n)$ and $\gamma(n)$. In (2.42) a particular normalization is chosen which simplifies some of the notational burden in the following.

¹⁴Bold lines denote vector-valued signals.

¹⁵This conversion factor is of considerable importance in the context of recursive least squares algorithms [91, pp. 418–419] but is introduced here in the more general context of joint recursive optimality.

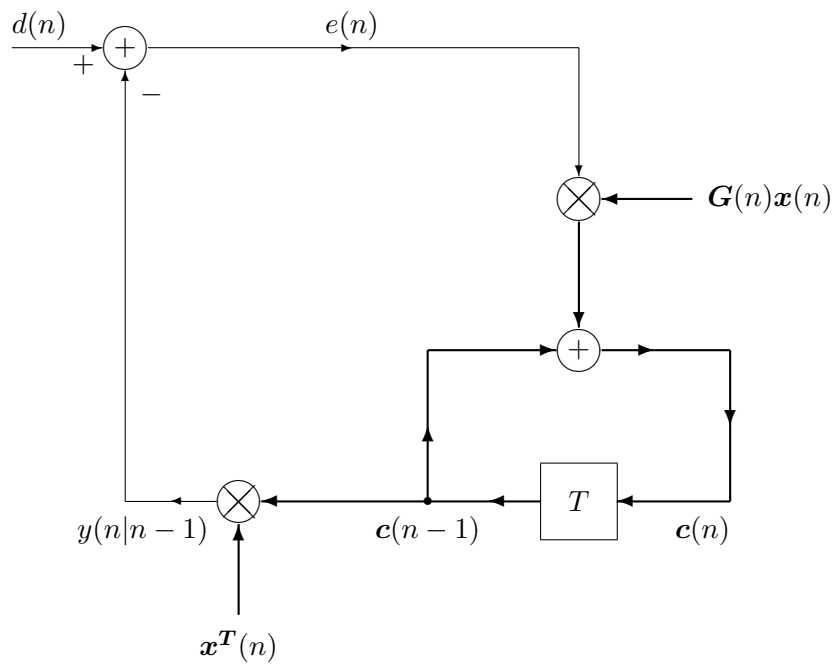


Figure 2.3: Signal flow graph for general joint recursive optimality adaptation algorithm (*a priori* quantities only).

1. Computing $\epsilon(n)$ with the help of the conversion factor provides a computationally efficient means to evaluate the a posteriori filter output from

$$y(n|n) = d(n) - \epsilon(n) \quad (2.45)$$

without performing the additional filtering operation $y(n|n) = \mathbf{c}^T(n)\mathbf{x}(n)$. This constitutes an attractive alternative in adaptive IIR filters, cf. Section 2.3.5.

2. Following the reasoning in the context of instantaneous gradient algorithms, a *deterministic stability bound* similar to (2.25) can be derived:

$$|\epsilon(n)| \leq |e(n)| \Leftrightarrow |\gamma(n)| \leq 1 \quad (2.46)$$

This inequality provides a fast means to check whether the choice of the weighting parameters $\gamma(n)$ and $\mathbf{G}(n)$ allows stable operation of the algorithm¹⁶.

In the following subsections, a large variety of algorithms falling into the joint recursive optimality class is discussed along a line of increasing complexity.

2.3.2 LMS-Type Algorithms

The discussion is started with ‘LMS-type’ algorithms, i.e. algorithms following the adaptation pattern (2.5) with a *scalar-valued* step size. They are obtained from (2.40) when specializing to coefficient weighting matrices $\mathbf{G}(n)$ proportional to the identity matrix.

2.3.2.1 The LMS Algorithm [237] [238, Chapter 6]

Let

$$\mathbf{G}(n) = \mu \mathbf{I} \quad (2.47a)$$

$$\gamma(n) = 1 - \mu \|\mathbf{x}(n)\|^2 \quad (2.47b)$$

to get from (2.43a,b)

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mu e(n)\mathbf{x}(n) \quad (2.48a)$$

$$\epsilon(n) = (1 - \mu \|\mathbf{x}(n)\|^2)e(n) \quad (2.48b)$$

Remark 1. For sufficiently small μ , i.e.

$$\mu \|\mathbf{x}(n)\|^2 \ll 1$$

$\epsilon(n)$ equals approximately $e(n)$, and from that $y(n|n) \approx y(n|n-1)$. Therefore the a posteriori output $y(n|n)$ is usually not computed as it provides no significant improvement over the a priori output $y(n|n-1)$.

¹⁶This condition should be regarded as necessary but generally not sufficient for algorithm stability.

Remark 2. The deterministic stability constraint is obtained from (2.46) as

$$|\gamma(n)| \leq 1 \Leftrightarrow 0 \leq \mu \leq \frac{2}{\|\mathbf{x}(n)\|^2} \quad (2.49)$$

This constraint shows that the range of the signal power $\|\mathbf{x}(n)\|^2$ must be known beforehand to allow an adequate choice of the step size μ .

Remark 3. The optimality criterion (2.40) reads with (2.47a,b)

$$(1 - \mu\|\mathbf{x}(n)\|^2)\|\mathbf{c}(n) - \mathbf{c}(n-1)\|^2 + \mu\epsilon^2(n) \stackrel{!}{=} \min \quad (2.50)$$

The trade-off between the time variation of the filter coefficients and the error power depends on the instantaneous signal power $\|\mathbf{x}(n)\|^2$. Assuming $1 - \mu\|\mathbf{x}(n)\|^2 > 0$, this dependency reflects the confidence put in the reference signal sample $d(n)$ given the current data vector $\mathbf{x}(n)$. For small instantaneous signal power $\|\mathbf{x}(n)\|^2$ little coefficient variation is allowed, as the instantaneous signal-to-noise ratio is assumed to be poor. In that case, the algorithm rather relies on previous coefficient estimates and allows for a large error power. In case of high input signal power, the corresponding behaviour is reversed.

Remark 4. Assume μ is chosen in the range

$$\frac{1}{\|\mathbf{x}(n)\|^2} \leq \mu \leq \frac{2}{\|\mathbf{x}(n)\|^2}. \quad (2.51)$$

Then the stability bounds (2.49) are still met, but $\gamma(n) = 1 - \mu\|\mathbf{x}(n)\|^2 \leq 0$ violates the positivity constraint (2.41b) on the error weight. Therefore, the algorithm (2.48a,b) is *no more* minimizing the sum of the time variance of the filter coefficients and the error power, yet it *maximizes the weighted difference* of the two expressions

$$(\mu\|\mathbf{x}(n)\|^2 - 1)\|\mathbf{c}(n) - \mathbf{c}(n-1)\|^2 - \mu\epsilon^2(n) \stackrel{!}{=} \max$$

For sure, such an optimality criterion does not make sense. This is also seen from the analysis of the misadjustment reached with μ satisfying (2.51). The sign alternation of the error signal according to (2.48b) induces some extra coefficient fluctuations when compared to a μ in the range $0 \leq \mu < 1/\|\mathbf{x}(n)\|^2$ chosen such that in both cases the algorithm has the same convergence time constant ('geometric ratio of convergence'), exhibiting either an 'underdamped' or an 'overdamped' convergence behaviour, cf. [91, pp. 213–216] [96, p. 44] [238, pp. 49–50].

2.3.2.2 The Projection Algorithm [14, pp. 117–121] [22] [81, pp. 50–54]

Let

$$\mathbf{G}(n) = \frac{\mu}{\|\mathbf{x}(n)\|^2} \mathbf{I} \quad (2.52a)$$

$$\gamma(n) = 1 - \mu \quad (2.52b)$$

to get from (2.43a,b)

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \frac{\mu}{\|\mathbf{x}(n)\|^2} e(n) \mathbf{x}(n) \quad (2.53a)$$

$$\epsilon(n) = (1 - \mu)e(n) \quad (2.53b)$$

Remark 1. For $\mu \ll 1$, one obtains again $y(n|n) \approx y(n|n-1)$.

Remark 2. The deterministic stability constraint is obtained from (2.46) with (2.52b) as

$$0 \leq \mu \leq 2 \quad (2.54)$$

which is independent from the input data vector $\mathbf{x}(n)$. Similarly, the convergence time constant and misadjustment of this algorithm are basically independent from the input signal power [22]. This feature makes the algorithm very useful in case of signals with unknown or rapidly varying power level (e.g. speech signals).

Remark 3. In the limiting case $\mu \rightarrow 1$, one obtains

$$\epsilon(n) = 0 \quad (2.55)$$

i.e. a zero-forcing algorithm, cf. page 15. This special case is considered when justifying the designation ‘projection algorithm’. Larger values of μ should not be considered, as they would violate the positivity constraint (2.41b) on the error weight.

Remark 4. The optimality criterion (2.40) reads with (2.52a,b)

$$(1 - \mu)\|\mathbf{c}(n) - \mathbf{c}(n-1)\|^2 \|\mathbf{x}(n)\|^2 + \mu \epsilon^2(n) \stackrel{!}{=} \min \quad (2.56)$$

Here, the trade-off between time variance and error power takes into account the different *scaling* of coefficients and signals in the adaptive filter. A fair comparison of the two types is achieved through multiplication of the coefficient variation $\|\mathbf{c}(n) - \mathbf{c}(n-1)\|$ with the length of the data vector $\|\mathbf{x}(n)\|$. Thereby, both contributions to the trade-off are approximately evaluated at the filter output. This explains the independence of the misadjustment from the signal power as the related excess error is also defined at the filter output only.

Remark 5. For $\|\mathbf{x}(n)\|^2 = 0$, (2.53a) cannot be computed. As such *silence intervals* may stretch over considerable periods of time in real-world signals (e.g. non-active intervals in voice transmission systems), they must be handled properly. Either the adaptation process is suspended during such periods by setting

$$\mathbf{c}(n) = \mathbf{c}(n-1) \quad \text{if} \quad \|\mathbf{x}(n)\| = 0 \quad (2.57)$$

or a positive constant is added to the denominator

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \frac{\mu}{v + \|\mathbf{x}(n)\|^2} e(n) \mathbf{x}(n) \quad \text{with} \quad v > 0 \quad (2.58)$$

Note that for $\mu = 1$ (2.58) is identical to the normalized LMS algorithm (2.37a) and obeys again the general joint recursive optimality principle (2.40) with

$$\mathbf{G}(n) = \frac{\mu}{v + \|\mathbf{x}(n)\|^2} \mathbf{I} \quad (2.59a)$$

$$\gamma(n) = \frac{v + (1 - \mu)\|\mathbf{x}(n)\|^2}{v + \|\mathbf{x}(n)\|^2}. \quad (2.59b)$$

2.3.3 Individual Coefficient Adaptation

Stepping up in algorithm complexity, algorithms are discussed which follow the general pattern (2.5) and have an *individual step size* for each filter coefficient. They are obtained from (2.40) when specializing to diagonal coefficient weighting matrices $\mathbf{G}(n)$.

2.3.3.1 The Variable Step Algorithm [90]

Let

$$\mathbf{G}(n) = \text{diag}\{\mu_0(n), \dots, \mu_{N-1}(n)\} = \begin{pmatrix} \mu_0(n) & 0 & \cdots & 0 \\ 0 & \mu_1(n) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mu_{N-1}(n) \end{pmatrix} \quad (2.60a)$$

$$\gamma(n) = 1 - \sum_{i=0}^{n-1} \mu_i(n) x^2(n-i) \quad (2.60b)$$

to get

$$c_i(n) = c_i(n-1) + \mu_i(n) e(n) x(n-i), \quad i = 0, 1, \dots, N-1 \quad (2.61a)$$

$$\epsilon(n) = \left[1 - \sum_{i=0}^{N-1} \mu_i(n) x^2(n-i) \right] e(n) \quad (2.61b)$$

Remark 1. The individual step size constants $\mu_i(n)$ are themselves continuously adapted according to the observed sign changes in the ‘correlation signals’ $e(n)x(n-i)$. This allows faster convergence for the same misadjustment when compared to the standard LMS algorithm.

Remark 2. A variant of the variable step algorithm is the so-called ‘ μ -vector LMS-algorithm’ considered by CHEN and VANDEWALLE in [36]. They propose to select $\mu_i(n)$ in (2.60a) according to

$$\mu_i(n) = (1 - \delta)^i \mu_0(n) \quad \text{for } i = 1, 2, \dots, N-1 \quad (2.62)$$

where $0 \leq \delta < 1$ and $\mu_0(n)$ is normalized by an estimate of the input signal power. The authors claim that this choice improves the tracking properties of the algorithm w.r.t. the conventional LMS algorithm as the factor $(1 - \delta)^i$ automatically discounts delayed input data samples $x(n-i)$ with increasing delay i .

Remark 3. The deterministic stability bound (2.46) reads with (2.60b)

$$0 \leq \sum_{i=0}^{N-1} \mu_i(n) x^2(n-i) \leq 2 \quad (2.63)$$

which is met *a fortiori* if

$$0 \leq \mu_i(n) \leq \frac{2}{x^2(n-i)} \quad (2.64)$$

The positivity constraint on $\gamma(n)$ (2.41b) suggests to further reduce the upper bound by a factor of 2.

Remark 4. The optimality criterion (2.40) reads with (2.60a,b) as

$$\sum_{i=0}^{N-1} \mu_i^{-1}(n) |c_i(n) - c_i(n-1)|^2 + \left[1 - \sum_{i=0}^{N-1} \mu_i(n) x^2(n-i) \right]^{-1} \epsilon^2(n) \stackrel{!}{=} \min \quad (2.65)$$

which reflects the individual control of the time variation of the coefficients $c_i(n)$ as mediated through the individual step size parameters $\mu_i(n)$.

2.3.3.2 The Individual Adaptation Algorithm [168, 169]

This algorithm is originally described for the application in an equation-error based adaptive IIR-filter problem. For simplicity, the equations are reduced to the adaptive transversal filter here. Let

$$\begin{aligned} \mathbf{G}(n) &= \frac{1}{v + \sum_{i=0}^{N-1} |x(n-i)|^3} \text{diag}\{|x(n)|, |x(n-1)|, \dots, |x(n-N+1)|\} \\ &= \frac{1}{v + \sum_{i=0}^{N-1} |x(n-i)|^3} \begin{pmatrix} |x(n)| & 0 & \cdots & 0 \\ 0 & |x(n-1)| & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & |x(n-N+1)| \end{pmatrix} \end{aligned} \quad (2.66a)$$

$$\gamma(n) = \frac{v}{v + \sum_{i=0}^{N-1} |x(n-i)|^3} \quad (2.66b)$$

to get

$$c_i(n) = c_i(n-1) + \frac{|x(n-i)|}{v + \sum_{i=0}^{N-1} |x(n-i)|^3} e(n) x(n-i) \quad i = 0, \dots, N-1 \quad (2.67a)$$

$$\epsilon(n) = \frac{v}{v + \sum_{i=0}^{N-1} |x(n-i)|^3} e(n) \quad (2.67b)$$

Remark 1. MIKHAEL *et al.* consider only the limit case $v \rightarrow 0$. This results in a zero-forcing algorithm with $\epsilon(n) = 0$. A numerically sound implementation of (2.67a), however, suggests to retain $v > 0$ so as to avoid division by zero during *silence intervals*.

Remark 2. The deterministic stability bound (2.46) is always met if v is non-negative.

Remark 3. The optimality criterion (2.40) reads with (2.66a,b) after canceling a common denominator simply as

$$\sum_{i=0}^{N-1} |c_i(n) - c_i(n-1)|^2 |x(n-i)|^{-1} + v^{-1} \epsilon^2(n) \stackrel{!}{=} \min \quad (2.68)$$

which reflects the design objective stated by the authors in [168, 169]: to allow for strong adaptation (i.e. faster time variation) of those coefficients which correspond to large instantaneous values of $|x(n-i)|$ (i.e. large input signal amplitudes).

Remark 4. The same authors do generalize their approach to arbitrary diagonal weighting matrices:

$$\begin{aligned} \mathbf{G}(n) &= \frac{1}{v + \sum_{j=0}^{N-1} \mu_j(n) x^2(n-j)} \text{diag}\{\mu_0(n), \dots, \mu_{N-1}(n)\} \\ &= \frac{1}{v + \sum_{j=0}^{N-1} \mu_j(n) x^2(n-j)} \begin{pmatrix} \mu_0(n) & 0 & \cdots & 0 \\ 0 & \mu_1(n) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mu_{N-1}(n) \end{pmatrix} \end{aligned} \quad (2.69a)$$

$$\gamma(n) = \frac{v}{v + \sum_{j=0}^{N-1} \mu_j(n) x^2(n-j)} \quad (2.69b)$$

$$(2.69c)$$

which gives rise to

$$c_i(n) = c_i(n-1) + \frac{\mu_i(n)}{v + \sum_{j=0}^{N-1} \mu_j(n) x^2(n-j)} e(n) x(n-i) \quad i = 0, \dots, N-1 \quad (2.70a)$$

$$\epsilon(n) = \frac{v}{v + \sum_{j=0}^{N-1} \mu_j(n) x^2(n-j)} e(n) \quad (2.70b)$$

Letting $v \rightarrow 0$ results in the general class of *zero-forcing algorithms* described in Subsection 2.2.2.1

2.3.3.3 The Signed Regressor Algorithm

Let

$$\begin{aligned} \mathbf{G}(n) &= \mu \text{diag}\{|x(n)|^{-1}, |x(n-1)|^{-1}, \dots, |x(n-N+1)|^{-1}\} \\ &= \mu \begin{pmatrix} |x(n)|^{-1} & 0 & \cdots & 0 \\ 0 & |x(n-1)|^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & |x(n-N+1)|^{-1} \end{pmatrix} \end{aligned} \quad (2.71a)$$

$$\gamma(n) = 1 - \mu \sum_{i=0}^{N-1} |x(n-i)| \quad (2.71b)$$

to get

$$c_i(n) = c_i(n-1) + \mu e(n) \operatorname{sgn}(x(n-i)) \quad \text{for } i = 0, 1, \dots, N-1 \quad (2.72a)$$

$$\epsilon(n) = \left(1 - \mu \sum_{i=0}^{N-1} |x(n-i)| \right) e(n) \quad (2.72b)$$

Remark 1. From (2.46) with (2.71b) the deterministic stability bound is

$$0 \leq \mu \leq 2 \left(\sum_{i=0}^{N-1} |x(n-i)| \right)^{-1} \quad (2.73)$$

where the positivity of $\gamma(n)$ requires to reduce the upper bound by a factor of 2 again. Comparing this bound to the bound (2.49) of the conventional LMS algorithm, the *Eucclidean* or L_2 norm of the data vector is replaced here with the *city-block* or L_1 norm of the same vector.

Remark 2. The signed regressor algorithm is attractive through its simple implementation. Note that the coefficient adaptation (2.72a) does not require any true multiplication if μ is chosen equal to a power of 2. In that case only shift and sign operations are needed in a fixed-point implementation.

Remark 3. The optimality criterion (2.40) reads with (2.71a,b) as

$$\left(1 - \mu \sum_{i=0}^{N-1} |x(n-i)| \right) \sum_{i=0}^{N-1} |c_i(n) - c_i(n-1)|^2 |x(n-i)| + \mu \epsilon^2(n) \stackrel{!}{=} \min \quad (2.74)$$

Apparently, no simple interpretation of this criterion is possible and therefore no prediction of actual algorithm performance can be made. This difficulty is also reflected in analyses found in the literature [45, 50, 128, 129, 206, 207, 208, 245, 246]. The application of this algorithm is, to a good extent, motivated from the desire to implement an LMS-like algorithm with reduced computational complexity.

2.3.4 RLS-Type Algorithms

Stepping up in algorithm complexity again, algorithms are discussed which follow the general pattern (2.5) and have a *matrix-valued* step size. They are obtained from (2.40) when allowing general (i.e. non-diagonal) coefficient weighting matrices $\mathbf{G}(n)$. As will be shown later, this weighting allows to rotate the data vector prior to the coefficient update such that the convergence behaviour is made independent of the second-order statistics of the input signal. The whole algorithm class receives its name from its most prominent member, the recursive least squares algorithm that has repeatedly served as an illustrative example in Section 2.2.

2.3.4.1 The LMS/Newton Algorithm [238, pp. 142–147]

This algorithm presupposes the a priori knowledge of the ‘true’ auto-correlation matrix \mathbf{R} of the input data vector $\mathbf{x}(n)$, cf. (2.10c). This matrix is directly used as coefficient vector weight:

$$\mathbf{G}(n) = \mu \mathbf{R}^{-1} \quad (2.75a)$$

$$\gamma(n) = 1 - \mu \mathbf{x}^T(n) \mathbf{R}^{-1} \mathbf{x}(n) \quad (2.75b)$$

resulting in

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mu e(n) \mathbf{R}^{-1} \mathbf{x}(n) \quad (2.76a)$$

$$\epsilon(n) = [1 - \mu \mathbf{x}^T(n) \mathbf{R}^{-1} \mathbf{x}(n)] e(n) \quad (2.76b)$$

Remark 1. The deterministic stability bound is obtained from (2.46) with (2.75b) as

$$0 \leq \mu \leq 2 \mathbf{x}^T(n) \mathbf{R}^{-1} \mathbf{x}(n) \quad (2.77)$$

whereas the positivity constraint (2.41b) reduces the upper bound by a factor of two.

Remark 2. The very fact of requiring a priori knowledge of the matrix \mathbf{R} limits its practical applicability. First, the existence of such a (time-invariant) matrix can only be assumed for stationary sequences $\{x(n)\}$. Second, the need to use an adaptive filter stems from the fact that the involved statistics are not totally known a priori. However— as shown in [174] for a special case in adaptive channel equalization—, it is sometimes possible to define an approximate \mathbf{R} -matrix a priori. This matrix improves the algorithm’s performance with respect to the LMS algorithm which does not follow the optimal ‘Newton’ search direction. WIDROW and WALACH elaborate in [239] on this algorithm so as to provide a theoretical reference for the performance analysis of the more practical RLS algorithms described below.

Remark 3. The optimality criterion (2.40) reads with (2.75a,b) as

$$\mu^{-1} [\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{R} [\mathbf{c}(n) - \mathbf{c}(n-1)] + (1 - \mu \mathbf{x}^T(n) \mathbf{R}^{-1} \mathbf{x}(n))^{-1} \epsilon^2(n) \stackrel{!}{=} \min \quad (2.78)$$

The interpretation of the two contributions is quite straight-forward:

- The term $[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{R} [\mathbf{c}(n) - \mathbf{c}(n-1)]$ is the coefficient variation as seen at the filter output if averaged over the ensemble of the process underlying $\{x(n)\}$:

$$\begin{aligned} & [\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{R} [\mathbf{c}(n) - \mathbf{c}(n-1)] = \\ & = [\mathbf{c}(n) - \mathbf{c}(n-1)]^T E \{ \mathbf{x}(n) \mathbf{x}^T(n) \} [\mathbf{c}(n) - \mathbf{c}(n-1)] \\ & = E \left\{ \left| [\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{x}(n) \right|^2 \right\} \\ & = E \left\{ |y(n|n) - y(n|n-1)|^2 \right\} \end{aligned} \quad (2.79)$$

where both the stationarity and the *independence assumption* have been invoked [163]. This first term describes the effect of the coefficient variation on the filter output in a more direct manner than does (2.56) for the projection algorithm. The latter considered only the magnitude of the data vector $\|\mathbf{x}(n)\|^2$ while the LMS/Newton algorithm considers also the *directional information* as evidenced in the inner product evaluation $[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{x}(n)$. This improvement makes the trade-off between convergence speed and final misadjustment independent from the *eigenvalue spread of the auto-correlation matrix* while the projection algorithm achieves only independence from the signal power (i.e. from $\text{tr}\{\mathbf{R}(n)\}$, cf. [238, p. 147]).

- The error weight $\gamma(n) = [1 - \mu \mathbf{x}^T(n) \mathbf{R}^{-1} \mathbf{x}(n)]$ can be related to the *log-likelihood* of the data vector $\mathbf{x}(n)$ if it is assumed to be a jointly Gaussian random vector with known correlation matrix \mathbf{R} [71] [91, pp. 418–419]. If the likelihood of the data vector is small the conversion factor or *likelihood variable* $\gamma(n)$ turns out to be small, too, and vice versa. In the case of likely input data, the algorithm allows for larger a posteriori errors while keeping the coefficient fluctuation small. If unlikely data re-appear, the algorithm is alerted via a large $\gamma^{-1}(n)$ to allow for faster coefficient variation so as to follow possible time-varying changes.

Remark 4. An alternative implementation of (2.76a) results from considering data orthonormalization at the adaptive filter input:

$$\tilde{\mathbf{x}}(n) = \mathbf{R}^{-1/2} \mathbf{x}(n) \quad (2.80a)$$

$$\tilde{\mathbf{c}}(n) = \mathbf{R}^{T/2} \mathbf{c}(n) \quad (2.80b)$$

where $\mathbf{R}^{1/2}$ is a matrix satisfying $\mathbf{R}^{1/2} \mathbf{R}^{T/2} = \mathbf{R}$. Then, from (2.76a) with (2.4)

$$\begin{aligned} e(n) &= d(n) - \mathbf{c}^T(n-1) \mathbf{x}(n) \\ &= d(n) - \tilde{\mathbf{c}}^T(n-1) \tilde{\mathbf{x}}(n) \end{aligned} \quad (2.81a)$$

$$\tilde{\mathbf{c}}(n) = \tilde{\mathbf{c}}(n-1) + \mu e(n) \tilde{\mathbf{x}}(n) \quad (2.81b)$$

Implementation requires now

1. N time-invariant FIR filters to perform the data orthonormalizing transform (2.80a),
2. a conventional LMS algorithm working in the transform domain (2.81a,b), but
3. *no back-transformation* of the coefficients $\tilde{\mathbf{c}}(n)$ to compute the filter output signal¹⁷.

This algorithm has been proposed in [35]. The interpretation of running a conventional LMS algorithm on the orthonormalized data illustrates once more why the convergence behaviour of this algorithm is independent from the auto-correlation of the input data.

¹⁷Back-transformation would only be required in the case of system identification where the actual coefficients $\mathbf{c}(n)$ are the desired output of the adaptive processing system.

2.3.4.2 Self-Orthogonalizing Algorithms

These algorithms aim at data normalization by an approximate auto-correlation matrix $\mathbf{R}(n)$ estimated from the input data sequence $x(n)$. The matrix estimate is chosen subject to complexity constraints. While GITLIN and MAGEE use an error feedback mechanism in [77] to continuously adapt the inverse matrix $\mathbf{R}^{-1}(n)$, it is otherwise customary [158, 190] [176, Chapter 4] to recursively estimate only the auto-correlation *function* of the input data out of which the full auto-correlation matrix is constructed based on the stationarity assumption (i.e. the matrix structure is prespecified as *Toeplitz*.) In this latter approach, fast (e.g. Levinson-type) matrix inversion techniques build on this matrix structure to generate efficient adaptation algorithms. This procedure is sometimes coupled with a block-wise update of the filter coefficients to reduce the computational cost even further¹⁸.

Summarizing, the algorithms are obtained from

$$\begin{aligned}\mathbf{G}(n) &= \mu \mathbf{R}^{-1}(n) \\ \gamma(n) &= 1 - \mu \mathbf{x}^T(n) \mathbf{R}^{-1}(n) \mathbf{x}(n)\end{aligned}$$

as

$$\begin{aligned}\mathbf{c}(n) &= \mathbf{c}(n-1) + \mu e(n) \mathbf{R}^{-1}(n) \mathbf{x}(n) \\ \epsilon(n) &= [1 - \mu \mathbf{x}^T(n) \mathbf{R}^{-1}(n) \mathbf{x}(n)] e(n)\end{aligned}$$

Remark 1. The deterministic stability bound for the parameter μ is again obtained as

$$0 \leq \mu \leq 2 \mathbf{x}^T(n) \mathbf{R}^{-1}(n) \mathbf{x}(n)$$

while the positivity constraint on $\gamma(n)$ reduces the upper bound by a factor of two.

Remark 2. These algorithms can be considered as approximations of algorithm (2.76a,b) and share its data orthonormalization property. Therefore, the eigenvalue spread or conditioning of the input auto-correlation matrix has little influence on the convergence behaviour of the algorithm, cf. [158, 190]. In comparison to the exact least squares algorithms discussed below, there is however a major drawback: self-orthogonalizing algorithms do not necessarily show the *rapid initial convergence property* or *algebraic convergence* which is described for the exact least squares methods e.g. in [96, pp. 293–302] and [79, 175]. This property is apparently contingent upon the possibility to interpret the algorithm as the recursive solution to a deterministic least squares problem defined on the given signal samples, cf. Section 2.2.2 on the time-averaging techniques of the deterministic paradigm. *Such interpretation is not possible with self-orthogonalizing algorithms.* An advantage over exact least squares algorithms in time-varying environments has recently been demonstrated in [18] where the flexibility found in the choice of the gain multiplier μ is exploited to improve the tracking performance of the algorithm.

¹⁸Cf. the two previous references and [220] for related adaptation algorithms in the context of lattice filters.

2.3.4.3 ‘Exact’ Least Squares Algorithms

Let

$$\mathbf{G}(n) = \lambda_2(n)\mathbf{R}^{-1}(n) \quad (2.82a)$$

$$\gamma(n) = 1 - \lambda_2(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n)\mathbf{x}(n) \quad (2.82b)$$

where

$$\mathbf{R}(n) = \lambda_1(n)\mathbf{R}(n-1) + \lambda_2(n)\mathbf{x}(n)\mathbf{x}^T(n) \quad (2.83)$$

is a recursive auto-correlation matrix estimate with, in general, time-varying data weighting factors $\lambda_1(n)$ and $\lambda_2(n)$ ¹⁹. This equation constitutes a generalization of the ‘exponential memory’ estimator and will further be referred to as data weighting *with general linear first-order memory* [136]. From (2.43a,b) one gets

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \lambda_2(n)e(n)\mathbf{R}^{-1}(n)\mathbf{x}(n) \quad (2.84a)$$

$$\epsilon(n) = [1 - \lambda_2(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n)\mathbf{x}(n)]e(n) \quad (2.84b)$$

whereas the *matrix inversion lemma* of Appendix A allows the recursive propagation of $\mathbf{R}^{-1}(n)$ from (2.83) as

$$\mathbf{R}^{-1}(n) = \frac{1}{\lambda_1(n)} \left\{ \mathbf{R}^{-1}(n-1) - \frac{\lambda_2(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)}{\lambda_1(n) + \lambda_2(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \right\} \quad (2.85)$$

Remark 1. From (2.85)

$$\mathbf{R}^{-1}(n)\mathbf{x}(n) = \frac{\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}{\lambda_1(n) + \lambda_2(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \quad (2.86)$$

and thereby $\gamma(n)$ may be redefined in terms of the matrix estimate at time $n-1$

$$\gamma(n) = 1 - \lambda_2(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n)\mathbf{x}(n) = \frac{\lambda_1(n)}{\lambda_1(n) + \lambda_2(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \quad (2.87)$$

In a similar way, the coefficient update (2.84a) may be rewritten as

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \lambda_2(n)e(n) \frac{\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}{\lambda_1(n) + \lambda_2(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \quad (2.88)$$

Remark 2. The deterministic stability bound is from (2.87)

$$\left| \frac{\lambda_1(n)}{\lambda_1(n) + \lambda_2(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \right| \leq 1 \quad (2.89)$$

¹⁹The inclusion of the data weighting factor $\lambda_2(n)$ in the definition of the coefficient weighting matrix $\mathbf{G}(n)$ is not accidental but necessary to allow the interpretation of the resulting algorithms as exact solutions to a weighted least squares problem as in (2.97)

From Appendix A and [87, Theorem 5.1], positive definiteness of the matrix estimate $\mathbf{R}(n)$ (2.83) can be assured for all $n \geq 0$ iff

$$\mathbf{R}(0) > 0 \quad (\text{a positive definite initial matrix}) \quad (2.90a)$$

$$\lambda_1(n) > 0 \quad \text{for all } n > 0 \quad (2.90b)$$

$$\lambda_2(n) > -\frac{\lambda_1(n)}{\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \quad \text{whenever } \|\mathbf{x}(n)\| > 0 \quad (2.90c)$$

This set of conditions is assumed to be satisfied as positive definiteness of the matrix estimate is a prerequisite for three important issues in least squares algorithms:

1. It coincides with constraint (2.41a) on the coefficient weighting matrix to make the algorithm fit into the joint recursive optimality framework²⁰.
2. It allows to interpret $\mathbf{R}(n)$ as an auto-correlation matrix estimate which has to be *positive* semi-definite by definition.
3. It is intimately related to the stability of the algorithm. If the matrix estimate contains zero or close-to zero eigenvalues, the algorithm gain grows beyond all limits and destabilizes the algorithm.

The set of conditions (2.90a-c) allows to rearrange (2.89) as

$$\begin{aligned} \lambda_1(n) &\leq \lambda_1(n) + \lambda_2(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n) \\ \Leftrightarrow \lambda_2(n) &\geq 0 \end{aligned} \quad (2.91)$$

This is the ‘extra’ condition governing the choice of the data weighting factor so as to meet the deterministic stability bound for the coefficient adaptation loop. Note that the positivity constraint on $\gamma(n)$ (2.41b) is fulfilled a fortiori from (2.87) with (2.90b) and (2.91).

Remark 3. The optimality criterion (2.40) reads with (2.82a,b)

$$[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{R}(n) [\mathbf{c}(n) - \mathbf{c}(n-1)] + \lambda_2(n)\gamma^{-1}(n)\epsilon^2(n) \stackrel{!}{=} \min \quad (2.92)$$

or equivalently with (2.86)

$$\lambda_1(n)[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{R}(n-1) [\mathbf{c}(n) - \mathbf{c}(n-1)] + \lambda_2(n)\epsilon^2(n) \stackrel{!}{=} \min \quad (2.93)$$

Expanding (2.83) by iterated substitution gives

$$\mathbf{R}(n) = \lambda_1(n)\mathbf{R}(n-1) + \lambda_2(n)\mathbf{x}(n)\mathbf{x}^T(n) = \sum_{k=0}^n w(n,k)\mathbf{x}(k)\mathbf{x}^T(k) \quad (2.94)$$

with the *time-varying recursive window function*

$$w(n,k) = \begin{cases} \lambda_1(n)w(n-1,k) & \text{for } n > k \\ \lambda_2(k) & \text{for } n = k \\ 0 & \text{for } n < k \end{cases} \quad (2.95)$$

²⁰To see this from (2.82a), the positivity constraint on $\lambda_2(n)$ as given in (2.91) has to be taken into account as well.

Unlike (2.78), (2.79) for the LMS/Newton algorithm, the optimality criterion (2.92) can be reformulated now *without invoking any independence assumption* as

$$\sum_{k=0}^n w(n, k) \left| [\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{x}(k) \right|^2 + \lambda_2(n) \gamma^{-1}(n) \epsilon^2(n) \stackrel{!}{=} \min \quad (2.96)$$

The first term in this expression represents the amount of coefficient variation *as seen at the filter output* if averaged over the weighted history of the input signal $\mathbf{x}(k)$ for $k = n, n-1, \dots, 1, 0$. Thus the particular sample path from the actual experiment is taken to evaluate the average ‘directional distribution’ of the input data vector. This property justifies to call the exact RLS algorithms the deterministic counterpart of the LMS/Newton algorithm. The observed trade-off between convergence speed and misadjustment is again independent from the eigenvalue spread of the auto-correlation matrix which is defined as a *time-average statistics* here!

Remark 4. Equation (2.96) is not counter-intuitive in case of adaptation in time-varying environments. Note that the weighted average is only computed over the input data vectors $\mathbf{x}(k)$ to produce an appropriate (localized) statistical matrix serving as coefficient weighting matrix $\mathbf{G}(n)$. *The coefficients $\mathbf{c}(n)$ still enter with their instantaneous first difference only* which is adequate to allow tracking of a time-varying ‘reference’ coefficient vector, see Chapter 3. Note, in particular, that the criterion (2.96) does *not* maintain that $\mathbf{c}(n)$ be optimal over the whole time interval running from $k = 0$ to n . This follows from its purely recursive definition. Compare this interpretation to the conventional ‘exact’ or deterministic least squares criterion (2.31) with (2.30) which—after generalizing to the window function (2.95)—reads as

$$\sum_{k=0}^n w(n, k) [d(k) - \mathbf{c}^T(n) \mathbf{x}(k)]^2 \stackrel{!}{=} \min \quad (2.97)$$

This exact least squares criterion is rather counter-intuitive when applied in time-varying situations because $\mathbf{c}(n)$ is postulated to provide the best fit for the reference signal $d(k)$ in terms of the data vector $\mathbf{x}(k)$ over the complete growing history of the observations.

The true merit of criterion (2.97) lies in explaining the rapid initial convergence of these algorithms²¹ [14, pp. 195–199][96, pp. 293–302][79, 175]. This explains the superiority of exact least squares algorithms w.r.t. this specific feature but gives no indication on their tracking behaviour which—in fact—is similar to the other RLS algorithms as predicted from criterion (2.96).

Remark 5. Stability of the coefficient adaptation loop is contingent upon the conditions (2.90b, 2.91). These constraints are however insufficient to guarantee stability of the recursive matrix computation (2.85) as well. To this end, further restrictions are to be made. They can be phrased as conditions on the class of admissible signals $x(n)$ that will keep the estimator stable for a specified data weighting scheme $\lambda_1(n)$, $\lambda_2(n)$. Such conditions are known as *persistency of excitation* conditions. A different approach to

²¹Approximately $2N$ time steps when a good signal-to-noise ratio and an almost time-invariant environment is assumed.

the same problem tries to specify admissible data weighting schemes that will keep the estimator stable for a specified signal class. Such schemes are scarce, however, and some instances are discussed under the label ‘directional forgetting’ in the following subsection.

Remark 6. There exists an apparent multitude of algorithms falling into the exact least squares class with general linear first-order data weighting. They are tabulated in the following Table 2.1 along with references to the relevant literature which should be consulted for a more detailed description of the algorithms. Some comments are given in the following remarks.

Remark 7. There are roughly four classes of exact recursive least squares algorithms:

1. The **growing memory** RLS (# 1) is a class of its own as it performs no data weighting at all, i.e. $\lambda_1(n) = \lambda_2(n) = 1$. As it is a decreasing gain algorithm, its application is restricted to strictly time-invariant environments.
2. The **exponential data weighting** RLS algorithms (#’s 2 to 6) have a data weighting factor unequal 1 only in the feedback loop of the estimator which uniformly affects the whole history of the estimate *up to* (#’s 2 and 3) or *including* (#’s 4 to 6) the present instant. In the latter case, the weighting process becomes clarified if the definition of the recursive estimator (2.83) is rewritten as in

$$\begin{aligned} \mathbf{R}(n) &= \lambda(n)\mathbf{R}(n-1) + \lambda(n)\mathbf{x}(n)\mathbf{x}^T(n) \\ \Leftrightarrow \mathbf{R}(n) &= \lambda(n)[\mathbf{R}(n-1) + \mathbf{x}(n)\mathbf{x}^T(n)] \end{aligned}$$

where the latter equation confirms the purely recursive nature of the data weighting process. An interpretation of the difference between the first group of algorithms # 2, 3 and the second group # 4 to 6 considers whether the inclusion of the new data vector $\mathbf{x}(n)$ occurs *after or before forgetting* old data represented in $\mathbf{R}(n-1)$ [118, pp. 135–137]. For both cases, if $\lambda(n)$ is constant, the forgetting process of these estimators is exponential.

3. The **selective data weighting** RLS algorithms (#’s 7 to 9) put a weighting factor unequal 1 only on the current input data $\mathbf{x}(n)\mathbf{x}^T(n)$. Therefore, the amount of weighting that a particular data vector $\mathbf{x}(n)$ receives is determined once forever through the choice of $\lambda_2(n)$. As $\lambda_1(n) = 1$ and $\lambda_2(n) \geq 0$ for stability reasons, *these algorithms ‘never forget’ and thus they are only applicable in strictly time-invariant environments.*
4. The **mixed data weighting** RLS algorithms (#’s 10 to 13) have in general both weighting factors set unequal 1 and are from that the most flexible ones. The particular choice $\lambda_2(n) = 1 - \lambda_1(n)$ (#’s 11 and 12) provides an asymptotically unbiased auto-correlation matrix estimate $\mathbf{R}(n)$ for stationary inputs $\mathbf{x}(n)$. A further advantage of this choice is that during periods with $\lambda_1(n) \approx 1$ the estimate $\mathbf{R}(n)$ is not growing beyond all bounds (and thereby driving the gain matrix $\mathbf{G}(n) = \mathbf{R}^{-1}(n)$ to zero) but keeps its latest value until $\lambda_1(n)$ assumes lower values again. This is in contrast to recursive data weighting RLS algorithms where the update may stall after extended periods with $\lambda_1(n) \approx 1$.

Table 2.1: EXACT LEAST SQUARES ALGORITHMS WITH LINEAR FIRST-ORDER DATA WEIGHTING.

#	Name	$\lambda_1(n)$	$\lambda_2(n)$	Reference
1.	RLS, growing memory ^a	1	1	[81, p. 58]
2.	RLS, exponential memory	λ	1	[91, table 8.1]
3.	RLS, time-varying exponential memory ^b	$\lambda(n)$	1	[81, p. 64], [247]
4.	RLS, variable forgetting factor ^c	$\lambda(n)$	$\lambda(n)$	[70, 205, 234]
5.	RLS, bounded trace ^d	$\lambda(n)$	$\lambda(n)$	[48]
6.	RLS, constant trace ^e	$\lambda(n)$	$\lambda(n)$	[140]
7.	Orthogonal projection algorithm ^f	1	v^{-1}	[56], [81, pp. 54–58]
8.	RLS, selective data weighting ^g	1	$\lambda(n)$	[81, pp. 62–64], [130], [150, Eq. (2.19)]
9.	RLS, set membership weighting ^h	1	$\lambda(n)$	[53]
10.	Optimal bounding ellipsoid (OBE) algorithm ⁱ	$\sigma^{-2}(n)$	$\sigma^{-2}(n)\lambda(n)$	[68, 97]
11.	Modified OBE algorithm ^j	$1 - \lambda(n)$	$\lambda(n)$	[51]
12.	Stochastic Newton algorithm	$1 - \lambda(n)$	$\lambda(n)$	[69], [150, Eq. (2.93)]
13.	RLS, general forgetting factors ^k	$\lambda(n)$	$\lambda(n)/v$	[139]

^a $0 \ll \lambda < 1$

^b $0 < \lambda(n) < 1$

^c $\lambda(n) = \max\{\lambda_{min}, 1 - \gamma(n)e^2(n)/\Sigma_0\}$ where $\lambda_{min}, \Sigma_0 > 0$

^dIf $\text{trace}\{\mathbf{R}^{-1}(n)\} \leq c$: $\lambda(n)$ as in # 4, otherwise $\lambda(n) = 1$

^e $\lambda(n)$ chosen such that $\text{trace}\{\mathbf{R}^{-1}(n)\} = \text{constant}$. This suggestion originates from E. IRVING as referenced in [87, 88, 140].

^f $0 \leq v \ll 1$

^g $\lambda(n) \geq 0$

^h $\lambda(n) \geq 0$, selected according to a ‘minimum volume’ criterion

ⁱ $\lambda(n) \geq 0$, selected according to a ‘minimum volume’ or a ‘minimum trace’ criterion; $\sigma(n) \sim$ ellipsoid

size

^j $0 \leq \lambda(n) \leq 1 - \delta$, selected according to ‘minimum volume’ criterion

^k $0 < \lambda(n) < 1 - \delta$ where $\delta, v > 0$

Remark 8. In several cases it is possible to perform an equivalence transformation between the referenced algorithms. This involves proper *rescaling* of the auto-correlation matrix estimate. As an example consider the orthogonal projection algorithm and define

$$\mathbf{R}'(n) = v\mathbf{R}(n)$$

With this definition the algorithm becomes equivalent to the growing memory RLS algorithm. The scale factors of such transformations depend on the weighting factors $\lambda_1(n)$ and/or $\lambda_2(n)$ (in the preceding case on $\lambda_2(n) = v^{-1}$). It is however possible that $\lambda_2(n)$, in particular, comes close to zero or infinity in which case the corresponding scale factors get either undefined or—at least—numerically ill-defined. In the present example the scale factor v may assume the value zero to describe the ‘zero-forcing’ variant of the orthogonal projection algorithm but which has no counterpart in the growing memory RLS case. Summarizing, such algorithm equivalence is only a first guideline to their interpretation but does not dispense from an individually conducted performance analysis.

Remark 9. A further instance of scaling equivalence is found in the class of *set membership RLS algorithms* (#’s 9 to 11). All of them minimize the volume of an ellipsoid defining the set of filter coefficient vectors consistent both with the observed data and an a priori bounded noise assumption. This is equivalent to minimizing $\det\{\mathbf{R}^{-1}(n)\}$ in [68, 97] or $\det\{\sigma^2(n)\mathbf{R}^{-1}(n)\}$ in [53]. The alternative minimization of the ellipsoid parameter $\sigma^2(n)$ is discussed in [51]. *All three algorithms are exactly equivalent if the appropriate time-varying scale factors are employed.* This equivalence class does however show considerable intra-class differences when applied in time-varying environments. While the algorithm in [53] is not suited to such tracking purposes, the algorithm [51] is. This difference originates again from ill-defined scale factors which effectively break the superficial equivalence.

2.3.4.4 Directional Forgetting Algorithms

Let

$$\mathbf{G}(n) = \beta(n)\mathbf{R}^{-1}(n) \quad (2.98a)$$

$$\gamma(n) = 1 - \beta(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n)\mathbf{x}(n) \quad (2.98b)$$

where

$$\mathbf{R}(n) = \mathbf{R}(n-1) + [\beta(n) - \alpha(n)]\mathbf{x}(n)\mathbf{x}^T(n) \quad (2.99)$$

is an auto-correlation matrix estimate consistent with (2.83) if $\lambda_1(n) = 1$ and $\lambda_2(n) = \beta(n) - \alpha(n)$ is selected. The jointly optimal recursive algorithm (2.43a,b) turns with (2.98a,b) into

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \beta(n)e(n)\mathbf{R}^{-1}(n)\mathbf{x}(n) \quad (2.100a)$$

$$\epsilon(n) = [1 - \beta(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n)\mathbf{x}(n)]e(n) \quad (2.100b)$$

and the matrix inversion lemma of Appendix A allows the recursive propagation of $\mathbf{R}^{-1}(n)$ from

$$\mathbf{R}^{-1}(n) = \mathbf{R}^{-1}(n-1) - \frac{\mathbf{R}^{-1}(n-1)\mathbf{x}(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)}{[\beta(n) - \alpha(n)]^{-1} + \mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \quad (2.101)$$

Remark 1. From (2.101)

$$\mathbf{R}^{-1}(n)\mathbf{x}(n) = \frac{\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}{1 + [\beta(n) - \alpha(n)]\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \quad (2.102)$$

which allows to express the conversion factor in terms of $\mathbf{R}^{-1}(n-1)$

$$\gamma(n) = 1 - \beta(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n)\mathbf{x}(n) = \frac{1 - \alpha(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}{1 + [\beta(n) - \alpha(n)]\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \quad (2.103)$$

as well as the coefficient update

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \beta(n)e(n) \frac{\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}{1 + [\beta(n) - \alpha(n)]\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \quad (2.104)$$

Remark 2. The deterministic stability bound is from (2.103)

$$\left| \frac{1 - \alpha(n)\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}{1 + [\beta(n) - \alpha(n)]\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \right| \leq 1 \quad (2.105)$$

From (2.90c) $\mathbf{R}(n)$ is guaranteed to be positive definite iff $\beta(n)$ and $\alpha(n)$ are chosen subject to

$$\beta(n) - \alpha(n) > -\frac{1}{\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}$$

For such a choice of $\alpha(n)$ and $\beta(n)$ the denominator on the left-hand side of (2.105) is guaranteed to be positive so that after rearranging this inequality results in

$$\alpha(n) \leq \frac{1}{\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} + \beta(n)/2 \quad \text{and} \quad (2.106a)$$

$$\beta(n) \geq 0 \quad (2.106b)$$

The useful range for $\alpha(n)$ is further restricted from the positivity constraint on $\gamma(n)$, i.e.

$$\alpha(n) < \frac{1}{\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \quad (2.107)$$

This latter bound is also used by HÄGGLUND in [87, Eq. (5.27)] where a different argument is used for its justification.

Remark 3. Just as in the case of exact least squares algorithms, the optimality criterion (2.40) can be rephrased as

$$\sum_{k=0}^n w(n, k) \left| [\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{x}(n) \right|^2 + \beta(n)\gamma^{-1}(n)\epsilon^2(n) \stackrel{!}{=} \min \quad (2.108)$$

where

$$w(n, k) = \begin{cases} \beta(k) - \alpha(k) & \text{for all } n \geq k \\ 0 & \text{for all } n < k \end{cases} \quad (2.109)$$

is the data-selective window function. This optimality interpretation ensures again that the trade-off between the convergence speed and the misadjustment is independent from the eigenvalue spread of the auto-correlation matrix.

Remark 4. The choice of $\mathbf{G}(n)$ in (2.98a) is *not consistent* with (2.82a) of the exact least squares algorithms when considering $\lambda_2(n) = \beta(n) - \alpha(n)$. Therefore, no optimality criterion similar to (2.97) can be found that has (2.100a,b) as its solution. Even though, these algorithms do exhibit rapid initial convergence just as exact least squares algorithms do. A direct explanation to this fact is still lacking but might be possible within KULHAVÝ's Bayesian approach to the derivation of directional forgetting algorithms [127].

Remark 5. Remarkably, the stability bounds (2.106b, 2.107) allow for a *negative data weighting factor* $\lambda_2(n) = \beta(n) - \alpha(n) < 0$ in the matrix estimator (2.99)²². This is in contrast with all exact RLS algorithms where $\lambda_2(n) \geq 0$ is required for stability of the coefficient adaptation loop. This difference stems again from the above-mentioned different definition of the coefficient weighting matrix $\mathbf{G}(n)$ in the two types of algorithms, compare (2.82a) and (2.98a). Note furthermore that directional forgetting algorithms use a selective data weighting scheme in their matrix estimator. Unlike the corresponding exact least squares algorithms (#'s 7 to 9 in Table 2.1) they are, however, able to track time-varying environments as they do forget old data whenever $\lambda_2(n) < 0$ or $\beta(n) < \alpha(n)$. The forgetting process is better understood when confronting the directional forgetting estimator (2.99)

$$\mathbf{R}(n) = \mathbf{R}(n-1) + \beta(n)\mathbf{x}(n)\mathbf{x}^T(n) - \alpha(n)\mathbf{x}(n)\mathbf{x}^T(n) \quad (2.110)$$

with the conventional one (2.83)

$$\mathbf{R}(n) = \mathbf{R}(n-1) + \lambda_2(n)\mathbf{x}(n)\mathbf{x}^T(n) - (1 - \lambda_1(n))\mathbf{R}(n-1) \quad (2.111)$$

Each update is composed of three terms: first the old estimate $\mathbf{R}(n-1)$, then the second term proportional to the tensor product $\mathbf{x}(n)\mathbf{x}^T(n)$ of the current input data vector (a rank 1 matrix with positive scaling factor in either case), and the negative third term which represents the forgetting of old data. The exact least squares algorithms exhibit a uniform or 'isotropic' forgetting of all dimensions of the column space of $\mathbf{R}(n-1)$ (i.e. the forgetting term is of full rank N), whereas the directional forgetting algorithms exhibit a selective or 'anisotropic' forgetting in a single vector space dimension given by the current input vector $\mathbf{x}(n)$, i.e. their forgetting term is of rank 1. This feature is the basis for the naming of the algorithm class. Several members of this class as referenced in the literature are compiled into Table 2.2.

Remark 6. Entries # 2 to 5 of Table 2.2 turn into the growing memory RLS algorithm for $\lambda \rightarrow 1$. This is why they may be seen as least squares techniques with appropriately chosen (i.e. stable) forgetting schemes. Their merit is to sustain a numerically sound inverse matrix estimate even for input signals $x(n)$ which are *not persistently exciting*²³.

Remark 7. Just as for exact least squares algorithms, scaling equivalence of some of the algorithms can be established. In particular, the directional forgetting algorithm with

²²Just as in the case of exact least squares algorithms, the stability bounds for the coefficient adaptation loop ensure a fortiori the positive definiteness of the matrix estimate.

²³Cf. [29, 5] for an introduction to the concept of persistency of excitation.

Table 2.2: DIRECTIONAL FORGETTING ALGORITHMS.

#	Name	$\beta(n)$	$\alpha(n)$	Reference
1.	HÄGGLUND's class ^a	$v^{-1}(n)$	$\alpha(n)$	[87, 88]
2.	KULHAVÝ's directional forgetting class ^b	$\lambda(n)$	$\frac{1 - \lambda(n)}{\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}$	[126, 127]
3.	Prediction-error controlled directional forgetting ^c	$\lambda(n)$	$\frac{1 - \lambda(n)}{\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}$	[26, 27]
4.	Directional forgetting, constant forgetting factor ^d	λ	$\frac{1 - \lambda}{\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}$	[30]
5.	Selective memory algorithm ^e	1	$\frac{1 - \lambda}{\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}$	[121, 122]

^a $v(n) > 0$ is the 'measurement variance', $\alpha(n)$ is chosen in [87, Eq. (5.27)] to make $\mathbf{R}^{-1}(n)$ converge to a matrix proportional to the identity matrix.

^bOptimal choice of $\lambda(n) > 0$ from [126, Eqs. (25–27)] or [127, Eq. 37].

^c $\lambda(n) = \max\{\lambda_{min}, 1 - \gamma(n)e^2(n)/\Sigma_0\}$ where $\lambda_{min}, \Sigma_0 > 0$ or a similar choice as described in [27, Eqs. (12,13)]

^d $0 < \lambda \leq 1$.

^e $0 \leq \lambda \leq 1$. For $\lambda \rightarrow 0$, a zero-forcing algorithm with non-diagonal gain matrix is obtained.

constant forgetting factor (entry # 4 of Table 2.2) is obtained from the selective memory algorithm (entry # 5) with the definition

$$\mathbf{R}'(n) = \lambda\mathbf{R}(n). \quad (2.112)$$

This scale transformation becomes ill-defined for $\lambda \rightarrow 0$ in which case only the *selective memory approach* allows a stable propagation of the inverse matrix estimator (2.101).

2.3.4.5 Other Modified RLS Algorithms

Besides the class of directional forgetting algorithms there are several other modified recursive least squares algorithms which *have no interpretation as minimizing an exact least squares criterion*. They arise naturally from modifying the recursive autocorrelation matrix estimate $\mathbf{R}(n)$ *without modifying the coefficient adaptation law consistently*. These algorithms are generally obtained from the joint recursive optimality criterion (2.40) with

$$\begin{aligned} \mathbf{G}(n) &= \frac{\mu(n)\mathbf{R}^{-1}(n-1)}{v(n) + \mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \\ \gamma(n) &= \frac{v(n) + (1 - \mu(n))\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}{v(n) + \mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)} \end{aligned}$$

where $\mathbf{R}^{-1}(n-1)$ is the inverse auto-correlation matrix estimated without including the current input data vector $\mathbf{x}(n)$. From (2.43a,b)

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mu(n)e(n)\frac{\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}{v(n) + \mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}$$

$$\epsilon(n) = \frac{v(n) + (1 - \mu(n))\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}{v(n) + \mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}e(n)$$

Note that for a positive definite matrix estimate $\mathbf{R}^{-1}(n-1)$ both the positivity constraints (2.41a,b) and the deterministic stability bound (2.46) are met if the parameters $v(n)$ and $\mu(n)$ are chosen subject to

$$0 < \mu(n) < 1 + \frac{v(n)}{\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)}$$

The following algorithms belong to this class:

- Periodic covariance resetting [81, p. 65]
- The ‘linear’ forgetting algorithm [81, p. 67] [118, pp. 53–56]
- The ‘vector variable forgetting factor’ algorithm [203]
- Directional forgetting via explicit eigenvector decomposition [118, Chapter 4]
- ‘Exponential forgetting and resetting’ [204]

Due to the diversity of the matrix estimator modifications in the above algorithms, an individual analysis of each algorithm is required in that case and can be found in the given references.

2.3.5 Modified Error Measures

So far the joint recursive optimality principle has been founded on the quadratic criterion (2.40) and a variety of algorithms emerged from specifying the coefficient weighting matrix $\mathbf{G}(n)$ and the error weight $\gamma(n)$. Of course, the fundamental idea of trading time variation of the coefficients for error signal power can be cast into other mathematical forms, too. The resulting algorithms constitute further instances of joint recursive optimality which is generalized in this subsection to include *non-quadratic measures* of the a posteriori error and measures of a *filtered error signal*.

2.3.5.1 Signed Error Algorithms

The derivation of these algorithms is started from (2.40) *by replacing the squared a posteriori error $\epsilon^2(n)$ with its magnitude $|\epsilon(n)|$* :

$$\boxed{[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c}(n) - \mathbf{c}(n-1)] + 2|\epsilon(n)| \stackrel{!}{=} \min} \quad (2.113)$$

where

$$\mathbf{G}(n) > 0 \quad (2.114)$$

is the positive definite coefficient weighting matrix²⁴. In Appendix B the unique solution to this optimization problem is found to consist of two algorithm ‘branches’:

²⁴The criterion (2.113) has already been scaled such that the notational burden is minimized in the following. Implicitly, the error weight has been set equal to 1/2.

$$\begin{aligned} \text{Branch A:} \quad & \text{if } |e(n)| \geq \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) \\ & \text{then } \mathbf{c}(n) = \mathbf{c}(n-1) + \text{sgn}[e(n)]\mathbf{G}(n)\mathbf{x}(n) \quad (2.115a) \end{aligned}$$

$$\epsilon(n) = e(n) - \text{sgn}[e(n)]\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) \quad (2.115b)$$

$$\text{Branch B:} \quad \text{if } |e(n)| < \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)$$

$$\text{then } \mathbf{c}(n) = \mathbf{c}(n-1) + e(n)\frac{\mathbf{G}(n)\mathbf{x}(n)}{\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)} \quad (2.115c)$$

$$\epsilon(n) = 0 \quad (2.115d)$$

Remark 1. The two branches of the algorithm are entered depending on the magnitude of the a priori error $|e(n)|$. The splitting of the optimum in (2.113) occurs because the cost function has a discontinuity in its derivative w.r.t. the coefficient vector $\mathbf{c}(n)$ on the hyperplane defined by $\epsilon(n) = d(n) - \mathbf{c}^T(n)\mathbf{x}(n) \stackrel{!}{=} 0$. Branch B is entered for small a priori errors $e(n)$. In that case, the update (2.115a) would move the coefficient vector $\mathbf{c}(n-1)$ into a vector $\mathbf{c}(n)$ ‘on the other side of the discontinuity plane’ whereby both the coefficient variation $[\mathbf{c}(n) - \mathbf{c}(n-1)]$ and the magnitude of the a posteriori error $|\epsilon(n)|$ are only increased beyond the values assumed for the zero-forcing solution (2.115c,d).

Remark 2. It is often desired to avoid the more involved computation of branch B. One remedy is to apply the adaptation law of branch A unconditionally. This can be justified as follows:

- If $|e(n)| \geq \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)$ branch A is the actual optimum.
- If $\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)/2 \leq |e(n)| < \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)$ the adaptation law A is no more optimal but satisfies still the deterministic stability bound $|\epsilon(n)| \leq |e(n)|$. Thus the error is at least not increased from the use of (2.115a), cf. also [45, Eq. (30)].
- If $|e(n)| < \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)/2$ the adaptation law A is neither optimal nor stable. Applying (2.115a) in that case will overshoot the desired settling value of the coefficient vector $\mathbf{c}(n)$. This leads to an increase of the a posteriori error $|\epsilon(n)|$ up to a maximum of $\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)$. This increase will—at least in a time-invariant environment—only persist for the moment, as the next adaptation step is likely to fall in the stability region of adaptation law A (which forms an embedding of the unstable region of small a priori errors).
- Summarizing this analysis, the unconditional use of algorithm branch A induces extra coefficient vector fluctuations which however remain bounded to magnitudes on the order of $|\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)|^2$.

Remark 3. A more conservative simplification of the optimum algorithm (2.115a,b) keeps the distinction of the two branches A and B, but reduces the computations in B to

a simple wait-and-see operation²⁵:

$$\begin{aligned} \textbf{Branch A:} \quad & \text{if } |e(n)| \geq 3/4\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) \\ & \text{then } \mathbf{c}(n) = \mathbf{c}(n-1) + \text{sgn}[e(n)]\mathbf{G}(n)\mathbf{x}(n) \end{aligned} \quad (2.116a)$$

$$\epsilon(n) = e(n) - \text{sgn}[e(n)]\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) \quad (2.116b)$$

$$\begin{aligned} \textbf{Branch B:} \quad & \text{if } |e(n)| < 3/4\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) \\ & \text{then } \mathbf{c}(n) = \mathbf{c}(n-1) \end{aligned} \quad (2.116c)$$

$$\epsilon(n) = e(n) \quad (2.116d)$$

where the branching threshold has been selected to achieve minimum cost function values when inserting (2.116a,b) or (2.116c,d) into the criterion (2.113). Implementation of this modified algorithm avoids the division necessary in (2.115c) and is only slightly more complex than the fore-mentioned unconditional use of branch A while achieving a lower misadjustment.

Remark 4. The **sign algorithm** [23, 45, 60, 61, 75, 92, 162, 208, 231, 251] is obtained from selecting

$$\mathbf{G}(n) = \mu\mathbf{I} \quad (2.117)$$

which yields from an unconditional use of (2.115a)

$$\begin{aligned} \mathbf{c}(n) &= \mathbf{c}(n-1) + \mu\text{sgn}[e(n)]\mathbf{x}(n) \\ \epsilon(n) &= e(n) - \mu\text{sgn}[e(n)]\|\mathbf{x}(n)\|^2 \end{aligned}$$

A simple conditional variant would halt the coefficient update iff $|e(n)| < 3/4\mu\|\mathbf{x}(n)\|^2$.

Remark 5. The **sign-sign algorithm** [34, 58, 60, 92, 152, 245, 246] [99, pp. 305–307] is obtained from selecting

$$\begin{aligned} \mathbf{G}(n) &= \mu\text{diag}\{|x(n)|^{-1}, |x(n-1)|^{-1}, \dots, |x(n-N+1)|^{-1}\} \\ &= \mu \begin{pmatrix} |x(n)|^{-1} & 0 & \cdots & 0 \\ 0 & |x(n-1)|^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & |x(n-N+1)|^{-1} \end{pmatrix} \end{aligned}$$

which yields from an unconditional use of (2.115a):

$$\begin{aligned} c_i(n) &= c_i(n-1) + \mu\text{sgn}[e(n)]\text{sgn}[x(n-i)] \quad \text{for } i = 0, \dots, N-1 \\ \epsilon(n) &= e(n) - \mu\text{sgn}[e(n)] \sum_{i=0}^{N-1} |x(n-i)| \end{aligned}$$

²⁵This strategy may be called a *stopping rule* or a *deadzone algorithm* [81, pp. 88–91][103, pp. 166–167]. The effect of halting the coefficient update turns out to be similar to the *blocking* or *stalling effect* in finite-word length implementations of the conventional LMS algorithm [14, pp. 106–112].

A simple conditional variant would halt the coefficient update iff

$$|e(n)| < 3/4\mu \sum_{i=0}^{N-1} |x(n-i)|.$$

Sign-sign algorithms have found their way into a CCITT standard [34][99, Section 6.5.3] for digital speech transmission by *adaptive differential pulse code modulation* (ADPCM) where they are used to adapt both the transversal and the recursive part of the predictor filter. Even for this standard, stability problems (in particular w.r.t. the recursive filtering part) have been reported for its original version [104]. The revised version of the standard is only able to achieve stability by resetting the algorithm if implausible internal states are observed. This lesson should be taken serious when designing adaptation algorithms by ‘simplifying’ approximations to known algorithms: it is hard to extrapolate the actual behaviour of the new algorithm as its analysis may not be simplified at all!

Remark 6. It is commonplace to justify the use of signed error algorithms from complexity considerations. Both algorithms considered above can be implemented without multiplications which simplifies the necessary hardware considerably. GERSHO noted in [75] that the sign algorithm can also be interpreted as a ‘stochastic gradient’ algorithm based on a *mean magnitude criterion*. This observation corresponds nicely to the joint recursive optimality criterion (2.113). TSYPKIN considers in [251, pp. 56–62, 101–137] the even more general aspect of how to select the optimum error functional (mean square, mean magnitude, etc.) on the basis of either complete or limited knowledge of the probability density function of the noise as expressed in the modeling assumptions of the application. His discussion is restricted to strictly time-invariant environments while the joint recursive optimality approach will allow to port such issues to the time-varying case as well.

2.3.5.2 Filtered Error Algorithms

These algorithms arise from the optimality criterion (2.40) if the squared a posteriori error $\epsilon^2(n)$ is replaced by a squared *filtered error* $\epsilon_f^2(n)$ according to

$$[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c}(n) - \mathbf{c}(n-1)] + \gamma^{-1}(n) \epsilon_f^2(n) \stackrel{!}{=} \min \quad (2.118)$$

with the coefficient weighting matrix $\mathbf{G}(n)$ and the error weight $\gamma(n)$ chosen subject to (2.41a,b) and with the definitions of

- the *filtered error*

$$\epsilon_f(n) = \epsilon(n) - \epsilon(n|n-1) \quad (2.119a)$$

- the *predicted error*

$$\epsilon(n|n-1) = H(n, q^{-1})\epsilon(n-1) \quad (2.119b)$$

where $H(n, q^{-1})$ is in general a stable, causal, linear time-varying operator defined in terms of the unit-delay or *time-shift operator*²⁶ q^{-1} . The *decomposition of the error filter* (2.119a) as given by

$$\epsilon_f(n) = \epsilon(n) - \epsilon(n|n-1) = \left(1 - H(n, q^{-1})q^{-1}\right) \epsilon(n) \quad (2.120)$$

is justified from several sources:

1. It is common in the literature [136] to require the error filter operator to be a stable, causal ratio of two *monic* polynomials. Such an operator may, however, always be decomposed as in (2.120).
2. A different scaling of the error filter (2.120) would not provide an extra degree of freedom for the specification of an algorithm from (2.118) because it is lost in the choice of the coefficient weighting matrix $\mathbf{G}(n)$ anyway.
3. The interpretation of the new error measure $\epsilon_f^2(n)$ is intuitively appealing when (2.119a) is taken into account: it measures the difference between the actual a posteriori error and its prediction from past a posteriori errors. Note that $\epsilon(n|n-1)$ is the *predicted error signal* and not the *prediction error signal*²⁷.

With the definition of the *innovation signal*

$$i(n) = e(n) - \epsilon(n|n-1) = e(n) - H(n, q^{-1})\epsilon(n-1) \quad (2.121)$$

one can write the unique solution to the minimization problem (2.118) with (2.119a,b) as

$$\mathbf{c}(n) = \mathbf{c}(n-1) + i(n)\mathbf{G}(n)\mathbf{x}(n) \quad (2.122a)$$

$$\epsilon(n) = \epsilon(n|n-1) + \gamma(n)i(n) \quad (2.122b)$$

Remark 1. In this context, the distinction of a *a priori error* $e(n)$ and *innovation* $i(n)$ becomes essential, cf. the discussion in the sequel of (2.4). From (2.121), the innovation is

$$\begin{aligned} i(n) &= e(n) - \epsilon(n|n-1) = d(n) - y(n|n-1) - \epsilon(n|n-1) \\ &= d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) - H(n, q^{-1})\epsilon(n-1). \end{aligned} \quad (2.123)$$

The innovation $i(n)$ reflects the part of the observed reference signal sample $d(n)$ that can neither be predicted from the previous coefficients $\mathbf{c}(n-1)$ nor from the previous error $\epsilon(n-1)$. This innovation (and not the a priori error as in (2.43a,b) is shared between the

²⁶This notation is preferred over a z -transform notation for two reasons. The first reason is to allow for time-varying filters $H(n, q^{-1})$ and the second is that the time-recursive optimality criterion and the further algorithm developments are best formulated in the time-domain.

²⁷The expression *prediction error* is often used to designate the a priori error signal $e(n)$ [31, Chapter 8], [81, pp. 303–319], [150, pp. 24–32].

coefficient increment $[\mathbf{c}(n) - \mathbf{c}(n-1)]$ and the ‘error increment’ $\epsilon(n) - \epsilon(n|n-1) = \epsilon_f(n)$ in the adaptation algorithm (2.122a,b). Note that by definition:

$$[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{x}(n) + \epsilon_f(n) = i(n). \quad (2.124)$$

Evaluating the cost function minimum in (2.118) which is achieved for the general solution (2.122a,b), the innovation sharing property is re-confirmed:

$$[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c}(n) - \mathbf{c}(n-1)] + \gamma^{-1}(n) \epsilon_f^2(n) = i^2(n) \quad (2.125)$$

The working of filtered error adaptation algorithms is summarized in the signal flow graph of Figure 2.5.

Remark 2. From the above, the described class of algorithms (2.122a,b) may be called either ‘with error filtering’ or ‘with error prediction’. Both conceptions will be exemplified below. A further attribute of this class is ‘with augmented error’, a terminology coined by MONOPOLI in [171]. It refers to the simple fact that the a priori error signal $e(n)$ is *augmented* with the predicted error $\epsilon(n|n-1)$ prior to entering the actual coefficient update equations. Note furthermore that all joint recursive optimality algorithms as described in previous subsections can be considered special cases of the error filtering framework *if the predicted error signal is set to zero*.

Remark 3. *Error prediction* is a terminology appropriate in speech signal analysis. A common approach is *linear predictive analysis* [157][198, Chapter 8] which results from using a transversal ‘analysis filter’ where the reference signal $d(n)$ is chosen equal to the upcoming input signal sample $x(n+1)$:

$$d(n) = x(n+1). \quad (2.126)$$

With such a choice, the analysis filter tries to predict the speech sample $x(n+1)$ from the previous samples $x(n), \dots, x(n-N+1)$ stored in the delay elements of the filter. The filter output is an estimate of $x(n+1)$ using the filter coefficients at time n :

$$\hat{x}(n+1) = y(n|n) = \mathbf{c}^T(n) \mathbf{x}(n) \quad (2.127)$$

The unpredictable residual $x(n+1) - \hat{x}(n+1)$ equals the a posteriori error $\epsilon(n)$. It can be shown that the task of the adaptive filter is basically to cancel the poles of the spectral envelope of the speech signal, resulting in an approximately whitened residual signal $\epsilon(n)$. This fact and the consideration of the natural speech production process give rise to a *signal generation model* which is composed of a white *excitation signal* resembling the residual signal $\epsilon(n)$ and of a *synthesis filter* characterized by the inverse transfer function of the associated analysis filter. It is usual practice to restrict the filter order N to low values (between 8 and 16 for speech signals sampled at 8 kHz) because this *short-term prediction* of up to 2 ms ahead covers most of the predictable information of the speech signal as contained in its spectral envelope.

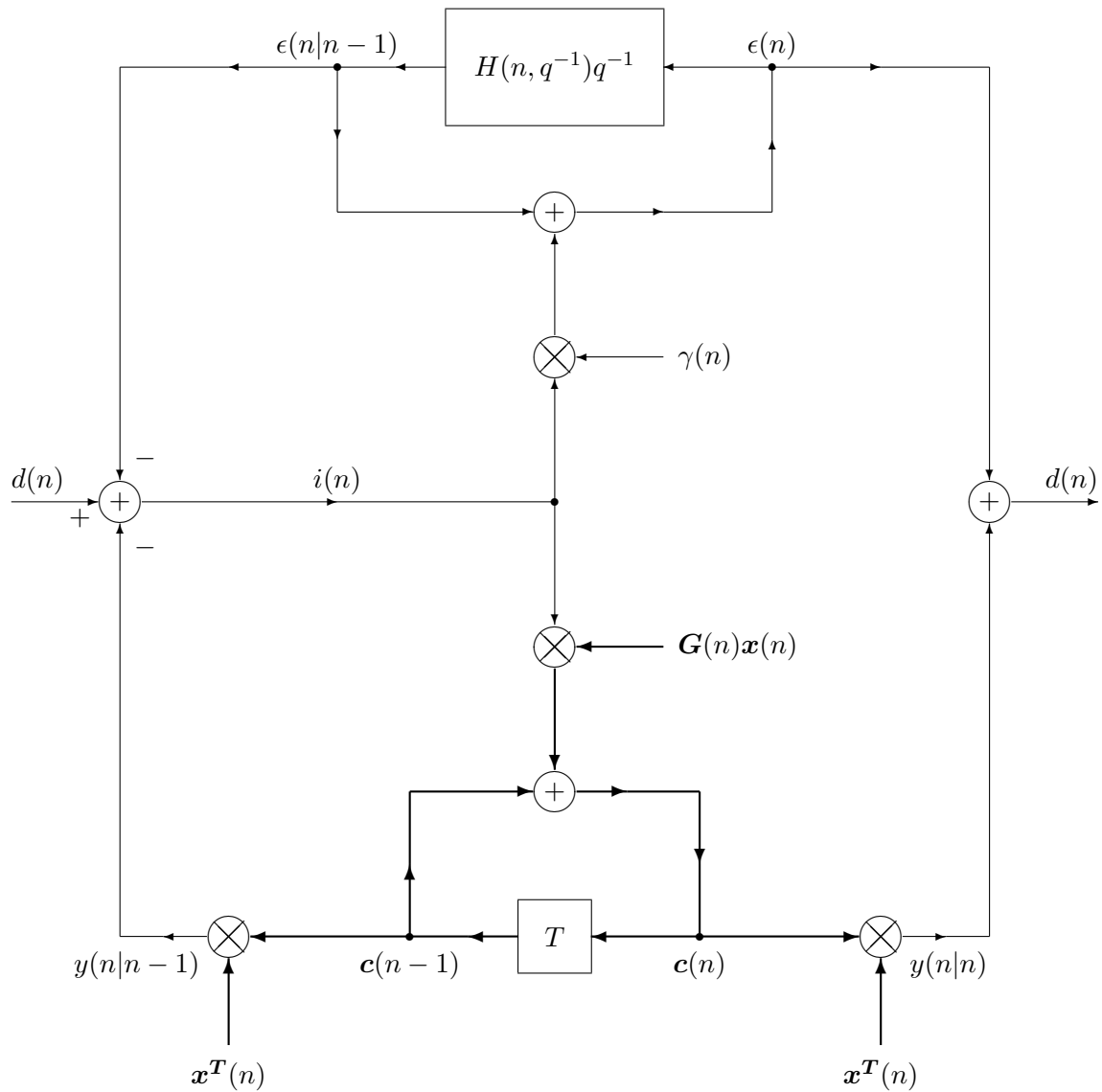


Figure 2.5: Signal flow graph for general joint recursive optimality algorithm *with error filtering* (a priori and a posteriori quantities displayed).

This practice disregards, however, the sizable amount of correlation that exists between successive periods of the speech signal (with a duration of some 10 ms) if it is produced ‘voiced’, i.e. with the vocal chords performing (quasi-) periodic oscillations. This correlation can be modeled through a *long-term predictor* [12] [99, pp. 312–317] which spans over a delay of some 40 to 120 sampling intervals at 8kHz. This additional predictor is both incorporated into the signal generation model and into the signal analysis set-up. Therefore, the original a posteriori error or residual signal $\epsilon(n)$ becomes itself a *predictable signal* and allows to define the predicted residual as

$$\epsilon(n|n-1) = H(n, q^{-1})\epsilon(n-1) = bq^{-L+1}\epsilon(n-1) \quad L = 40 \dots 120 \quad (2.128)$$

where both the coefficient b and the delay L are chosen to match the long-term correlation found in the time history of the residual signal $\epsilon(n)$. The filtered error signal

$$\epsilon_f(n) = \epsilon(n) - \epsilon(n|n-1) = \epsilon(n) - b\epsilon(n-L) \quad (2.129)$$

is the *unpredictable difference* between the residual $\epsilon(n)$ and its value predicted from the previous period of the speech signal. Modern speech coding techniques such as the standardized pan-european ‘RPE-LTP’ codec for digital mobile telephony [86, 230] exploit this dual-term prediction strategy and transmit instead of the original speech signal

- the coded short-term predictor coefficients $\mathbf{c}(n)$,
- the coded long-term predictor parameters b and L , and
- the coded filtered error $\epsilon_f(n)$.

These coders do determine the various predictor parameters by adaptation strategies which are out of the scope of this text (i.e. blockwise over a frame-length of some 20 ms)²⁸.

The application of recursive algorithms in such a speech signal analysis set-up raises the following problem [49, pp. 129–139]: if there is no long-term predictor used to model the residual signal $\epsilon(n)$ (i.e. $\epsilon(n|n-1) = 0$), then the short-term predictor coefficients $\mathbf{c}(n)$ will be disturbed by the (quasi-) periodic excitation during the voiced speech segments which leads to remarkable excursions from the otherwise reasonably smooth coefficient trajectories. The preceding discussion suggests to *use a long-term predictor within the adaptation loop of the short-term predictor* such that the update of $\mathbf{c}(n)$ depends via $i(n)$ on the predicted residual $\epsilon(n|n-1)$. This procedure can reduce the coupling effect between quasi-periodic excitation and slowly time-varying filtering. Studies in that vein have been undertaken in [95, 107, 108]. As to the author’s knowledge, a full implementation of (2.122a,b) with (2.128) has not been reported so far.

²⁸Their working can still be understood in terms of the optimality principle (2.118) [123]: trading of time variation of the short-term predictor coefficients for power of the unpredictable part in the residual of the speech signal (i.e. the ‘prediction gain’ [99, p. 253]). This trade-off is directly reflected in the bit rates allocated to code the two contributions. At present ‘hybrid’ coding methods such as the above-mentioned RPE-LTP codec and coders working at considerably lower bit rates (e.g. [114]) find an optimum by weighting both contributions more or less equally whereas earlier attempts focussed on only one of the two: in *vocoders* only the short-term predictor parameters are transmitted and a synthetic residual signal is supplied in the receiver; in *waveform coders* such as ADPCM only the residual signal is transmitted and the predictor coefficients are re-estimated from the reconstructed signal in the receiver [212].

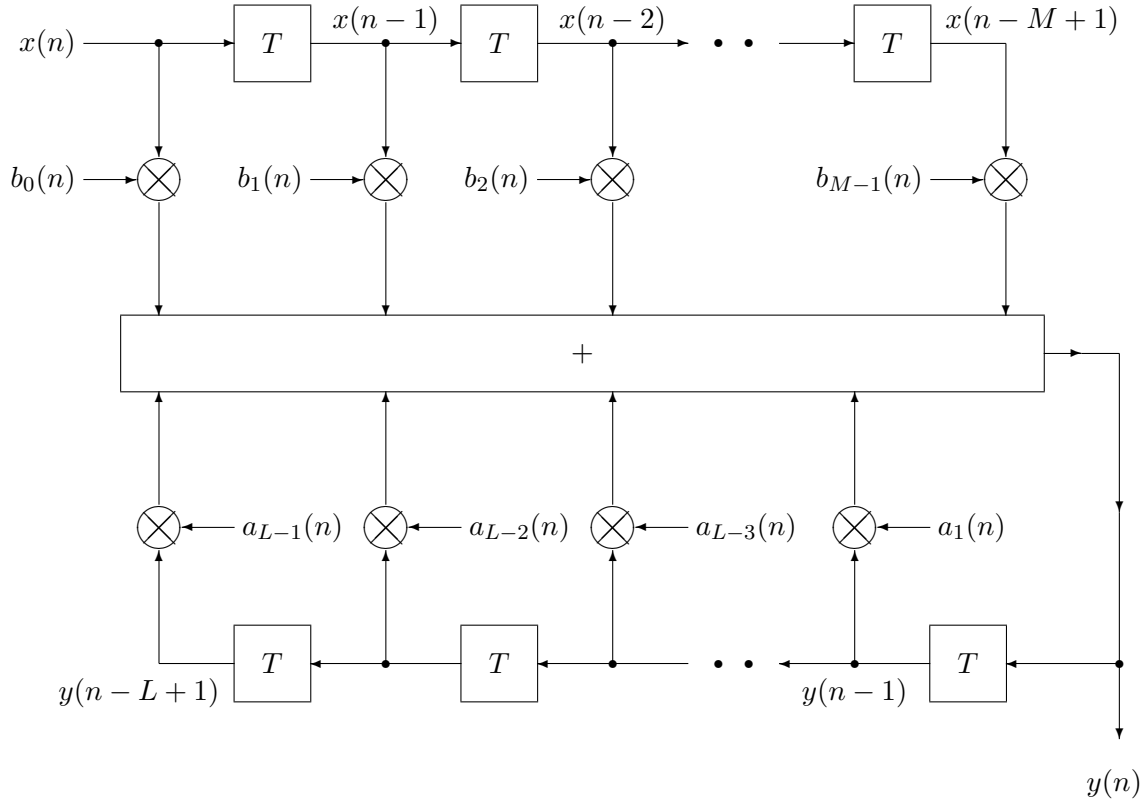


Figure 2.6: Direct-form IIR filter structure.

Remark 4. *Filtered error or augmented error algorithms are a standing paradigm in adaptive IIR (infinite impulse response) filtering [102, 209]. They originate mostly from POPOV’s hyperstability theory and its application to model-reference adaptive control [134]. These ideas have been ported and extended to applications in adaptive signal processing in [101] and related work as documented in [49, pp. 68–81][103, 224]. As this thesis focusses on the transversal filter structure, the case of IIR filters is touched upon only briefly.*

Figure 2.6 shows the direct-form structure of an IIR filter.

The output $y(n)$ is computed recursively from

$$y(n) = \sum_{l=1}^{L-1} a_l y(n-l) + \sum_{m=0}^{M-1} b_m x(n-m) \quad (2.130)$$

When the filter coefficients a_l and b_m are adapted and thereby made dependent on the signals involved, the question arises how to replace $y(n-l)$ in the recursion (2.130). Either the a posteriori estimate $y(n-l|n-l)$ or the (delayed) reference signal $d(n-l)$ may be used.

The latter approach results in the so-called *equation error formulation* [167] which is basically equivalent to the ‘linear combiner’ approach of adaptive transversal filters because (2.130) is turned into a non-recursive linear combination of observed signals only ($d(n-l)$ and $x(n-m)$). These signals are considered as reliable sources of information and

any error (i.e. mismatch between the computed output $y(n-l|n-l)$ and the ‘measured output’ $d(n)$) is attributed to errors in the filter equation or at the filter input but not to measurement noise at the filter output. This approach can also be described as a *linear regression* modeling of the filter output [150, pp. 33–36].

The other approach computes the output signal in terms of previous estimates $y(n-l|n-l)$ and makes the ‘linear combiner’ interpretation impossible. It is referred to as *pseudo-linear regression* modeling [150, pp. 48–51] or the *output error formulation*. This terminology stems from the system identification literature where the estimate $y(n-l|n-l)$ is esteemed as a reliable description of the plant’s output which is more trustworthy than the noisy measurements $d(n-l)$ of the same output. The output error formulation deserves further discussion as the coefficient adaptation algorithm turns out to be *non-linear in the coefficients* due to the dependence of the data vector (i.e. $y(n-l|n-l)$ and $x(n-m)$) on the coefficients. For further analysis, a concise notation of *compound vectors* of order $N = L + M - 1$ is introduced first.

- The *a posteriori* regressor vector

$$\phi(n) = [y(n-1|n-1), y(n-2|n-2), \dots, y(n-L+1|n-L+1), x(n), x(n-1), \dots, x(n-M+1)]^T \quad (2.131a)$$

- The compound coefficient vector

$$\mathbf{c}(n) = [a_1(n), a_2(n), \dots, a_{L-1}(n), b_0(n), b_1(n), \dots, b_{M-1}(n)]^T \quad (2.131b)$$

This notation allows to write e.g. the *a posteriori* output as

$$y(n|n) = \mathbf{c}^T(n)\phi(n) \quad (2.132)$$

With this notation the joint recursive optimality principle (2.118) can be applied in a straight-forward manner after replacing $\mathbf{x}(n)$ with $\phi(n)$. The result is again algorithm (2.122a,b). A common choice for the error filter or error predictor $H(n, q^{-1})$ is a time-invariant transversal filter

$$\epsilon(n|n-1) = H(n, q^{-1})\epsilon(n-1) = \sum_{l=1}^{L-1} h_l \epsilon(n-l) \quad (2.133a)$$

$$\epsilon_f(n) = [1 - H(n, q^{-1})q^{-1}]\epsilon(n) = [1 - \sum_{l=1}^{L-1} h_l q^{-l}]\epsilon(n) \quad (2.133b)$$

The coefficients h_l of the error predictor should be chosen such that the transfer function

$$\tilde{H}(z^{-1}) = \frac{1 - \sum_{l=1}^{L-1} h_l z^{-l}}{1 - \sum_{l=1}^{L-1} a_{ref,l} z^{-l}} \quad (2.134)$$

satisfies a certain *strictly positive real* condition where the coefficients $a_{ref,l}$ are the (unknown) coefficients in the recursive part of the system to be identified. For such a choice—which, of course, is a priori non-trivial—convergence of the filter output or even the filter coefficients to their ‘true’ values can be proved under rather general conditions [132, 148]²⁹. Various algorithm options are listed below:

²⁹Note that for $h_l = a_l(n-1)$, $l = 1, \dots, L-1$ the innovation signal of the above output error formulation turns into the a priori error signal of the corresponding equation error formulation [103, pp. 66–67][138][148, Section IX].

1. LANDAU's **Hyperstable Integral Adaptation Algorithm** [132, 133, 137, 135, 136] is obtained from setting

$$\begin{aligned}\mathbf{G}(n) &= \frac{\mathbf{R}^{-1}(n-1)}{1 + \boldsymbol{\phi}^T(n)\mathbf{R}^{-1}(n-1)\boldsymbol{\phi}(n)} \\ \gamma(n) &= \frac{1}{1 + \boldsymbol{\phi}^T(n)\mathbf{R}^{-1}(n-1)\boldsymbol{\phi}(n)}\end{aligned}$$

where

$$\mathbf{R}(n) = \lambda_1(n)\mathbf{R}(n-1) + \lambda_2(n)\boldsymbol{\phi}(n)\boldsymbol{\phi}^T(n)$$

is an auto-correlation matrix estimate consistent with (2.83) and where $0 < \lambda_1(n) \leq 1$, $0 \leq \lambda_2(n) < 2$ control the first-order data weighting filter. Note that there are two differences w.r.t. exact least squares algorithms:

- The data vector has been generalized to include the a posteriori filter outputs $y(n-l|n-l)$ which makes the coefficient update nonlinear in the coefficients.
- The definition of the coefficient weighting matrix is not consistent with (2.82a). Therefore, the exact least squares criterion (2.97) is not applicable.

With the above weighting factors, the optimality criterion (2.118) reads after simplification:

$$[\mathbf{c}(n) - \mathbf{c}(n-1)]^T \mathbf{R}(n-1)[\mathbf{c}(n) - \mathbf{c}(n-1)] + \epsilon_f^2(n) \stackrel{!}{=} \min$$

and the resulting algorithm is the same³⁰ as in [136, Eqs. (2.7, 3.9, 4.12, 4.14, 4.15)].

2. The **HARF Algorithm** (from 'hyperstable adaptive recursive filter') [101, Eqs. (3, 4, 5, 7, 8)] [49, pp. 68–81] is obtained from (2.118) with $\mathbf{x}(n)$ replaced by $\boldsymbol{\phi}(n)$ and

$$\begin{aligned}\mathbf{G}(n) &= \gamma(n)\text{diag}\{\mu_1(n), \dots, \mu_{L-1}(n), \rho_0(n), \dots, \rho_{M-1}(n)\} \\ &= \gamma(n) \begin{pmatrix} \mu_1(n) & 0 & \dots & \dots & 0 \\ 0 & \ddots & & & \vdots \\ \vdots & & \mu_{L-1}(n) & & \vdots \\ \vdots & & & \rho_0(n) & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & \dots & \dots & \dots & 0 & \rho_{M-1}(n) \end{pmatrix} \quad (2.135a)\end{aligned}$$

$$\gamma(n) = \left\{ 1 + \sum_{l=1}^{L-1} \mu_l(n)y^2(n-l|n-l) + \sum_{m=0}^{M-1} \rho_m(n)x^2(n-m) \right\}^{-1} \quad (2.135b)$$

³⁰Note for comparison that the error signals *a priori* error $e(n)$, *a posteriori* error $\epsilon(n)$, *innovation* $i(n)$, and *filtered error* $\epsilon_f(n)$ carry the following names in LANDAU's terminology: *a priori prediction error* $\epsilon^0(k+1)$, *a posteriori prediction error* $\epsilon(k+1)$, *a priori adaptation error* $\nu^0(k+1)$, and *a posteriori adaptation error* $\nu(k+1)$, respectively.

The optimality criterion (2.118) reads after simplification:

$$\sum_{l=1}^{L-1} \mu_l(n) [a_l(n) - a_l(n-1)]^2 + \sum_{m=0}^{M-1} \rho_m(n) [b_m(n) - b_m(n-1)]^2 + \epsilon_f^2(n) \stackrel{!}{=} \min \quad (2.136)$$

which allows to recognize the HARF algorithm as an *individual coefficient adaptation* variant of LANDAU's algorithm.

3. The **SHARF algorithm** (from 'simplified hyperstable adaptive recursive filter') [105, 138][103, p. 66-67] is obtained from the HARF algorithm for small step size parameters $\mu_l(n)$ and $\rho_m(n)$. 'Small' is defined by requiring the conversion factor to satisfy $0 \ll \gamma(n) \approx 1$. From that, both the a posteriori output signal and the a posteriori error can be approximated by their respective a priori values. Such approximation affects the regressor vector $\phi(n)$ and the error predictor which operates now on delayed a priori errors instead of a posteriori errors. In principle, it is possible to define an appropriately modified joint recursive optimality criterion that is met exactly by the SHARF algorithm. However, considering the above *small step size assumption*, this criterion provides no insights beyond the criterion (2.118) given before. For larger step sizes, it shows that the intuitively simple *a priori error filter* (as in the SHARF coefficient update equation, cf. [103, Eq. (8.29)]), is not natural to the optimality criterion where previous *a priori errors* have to be used to predict the current *a posteriori error*.
4. It is mentioned without proof that even more sophisticated error filtering schemes (e.g. with *adaptive error filtering* as in [106, 133]) can be embedded in an appropriately generalized joint recursive optimality criterion.

To conclude this excursion to adaptive IIR filters a generally applicable strategy for their *efficient implementation* is promoted [119]: within the output error formulation all algorithms do require the computation of *both* the a priori filter output $y(n|n-1)$ —to compute the innovation $i(n)$ —and the a posteriori output $y(n|n)$ —to form the a posteriori regressor vector $\phi(n)$. Either filter computation consists of $N = M + L - 1$ multiply/add operations. But the a posteriori output can often be obtained much cheaper from (2.45) as $y(n|n) = d(n) - \epsilon(n)$. While the computation of the a posteriori error $\epsilon(n)$ is not automatically provided for in actual adaptive IIR filter applications, this signal is available from (2.122b) at the cost of only one extra multiply/add operation over the computations already done for the coefficient updates in the above hyperstable algorithms. If the overall regressor length satisfies $N \gg 1$, the new strategy offers substantial computational savings with respect to the conventional direct a posteriori filtering operation. Thereby, the overall algorithm implementation cost comes close to the cost otherwise found in simplified algorithms (such as SHARF) which are only obtained after approximations to the desired algorithm structure. *Summarizing, when implementing adaptation algorithms in the output error formulation, the direct a posteriori filtering operation should be short-cut by direct computation of the a posteriori error whenever possible.*

2.3.6 Summary of the Joint Recursive Optimality Framework

The *joint recursive optimality* framework has been introduced in Subsection 2.3.1.1 within a simple geometrical interpretation of the normalized LMS algorithm. The general opti-

mality criterion as formulated in (2.40) leads to the general adaptation pattern (2.43a,b). A large variety of algorithms emerged in the following subsections within this pattern, with an increase in complexity from a scalar coefficient weight (LMS-type algorithms) through a diagonal coefficient weighting matrix (individual coefficient adaptation) to a full coefficient matrix (RLS-type algorithms). In a further stage of generalization, the error measure in the optimality criterion has been changed so that signed error and filtered error algorithms are obtained. Table 2.3 summarizes the definitions and relationships of the general framework including error filtering.

The underlying optimality principle has the following general properties:

1. It is *deterministic* as there are no expectation operators involved.
2. It is *recursive* in that an instantaneous optimum is defined at time n in terms of the previous coefficient vector of time $n - 1$.
3. It is directed towards an objective which is expressed *jointly* in terms of the desired behaviour of both the filter coefficients and the error.

Although deterministic in its nature, this framework is *open to account for statistical information as well* where the term ‘statistical’ carries the *non-probabilistic interpretation* advocated by GARDNER in [73]. Realizable recursive adaptation algorithms process such information virtually always on a time-average basis (rather than by construction of appropriate experiments allowing for the observation of ensemble averages). This information enters the optimality framework in two distinct manners:

1. The recursive operation of the algorithms constitutes *per se* an *implicit* memory stretching over a certain part of the time history of the observed signals $\{x(n)\}$ and $\{d(n)\}$. In other words, the optimal coefficient vector $\mathbf{c}(n)$ is not defined in absolute, i.e. strictly localized terms, but relative to the previous coefficient vector $\mathbf{c}(n - 1)$. Thus the algorithms accumulate global statistical information by recursively propagating their internal state variables.
2. The choice of the coefficient weighting matrix $\mathbf{G}(n)$ (and the error weight $\gamma(n)$) provides for *explicit* statistical design of the algorithms, and so does the error filter in Subsection 2.3.5. An important instance of this explicit design option is the choice of an inverse auto-correlation matrix estimate as the coefficient weighting matrix for RLS-type adaptation algorithms. This particular choice is *by no means definite within the joint recursive optimality framework on its own*, but deserves extrinsic justification from a (statistical) performance analysis.

Finally, the most striking feature of this framework is re-iterated here: *the joint local optimality framework provides a coherent derivation for a wide class of algorithms directly from an explicit optimality criterion*. It requires neither any modeling assumptions on the signals involved nor any simplifying approximations to arrive at practical algorithm structures as they are in actual use. This is in open contrast to the conventional derivation for most of the discussed algorithms. As this optimality is one of the few if not the only property shared by all of these adaptation algorithms, it suggests itself as the unifying root of a tree breaking into myriads³¹ of leaves.

³¹This score is due to [150, p. 9].

Table 2.3: JOINT RECURSIVE OPTIMALITY: DEFINITIONS AND RELATIONSHIPS.

1. Define the a priori error

$$e(n) = d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n).$$

2. Define the innovation

$$i(n) = e(n) - \epsilon(n|n-1)$$

where $\epsilon(n|n-1)$ is the error predicted from previous a posteriori errors $\epsilon(n-l)$, $l \geq 1$. In many cases $\epsilon(n|n-1) = 0$.

3. Choose a positive definite coefficient weighting matrix $\mathbf{G}(n)$ and a positive error weight $\gamma(n)$ with a normalization such that

$$\gamma(n) + \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) = 1.$$

4. Define the cost function

$$J(n, \mathbf{c}) = [\mathbf{c} - \mathbf{c}(n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c} - \mathbf{c}(n-1)] + \gamma^{-1}(n) [d(n) - \mathbf{c}^T \mathbf{x}(n) - \epsilon(n|n-1)]^2$$

and the optimization problem

$$\mathbf{c}(n) \stackrel{\dagger}{=} \arg \min_{\mathbf{c} \in R^N} J(n, \mathbf{c}).$$

5. The unique solution is found as

$$\begin{aligned} \mathbf{c}(n) &= \mathbf{c}(n-1) + i(n)\mathbf{G}(n)\mathbf{x}(n) \\ \epsilon(n) &= \epsilon(n|n-1) + \gamma(n)i(n). \end{aligned}$$

6. This solution achieves a cost function minimum of

$$J(n, \mathbf{c})|_{\mathbf{c}=\mathbf{c}(n)} = i^2(n).$$

7. If the a posteriori filter output is needed, it is often advantageously computed from

$$y(n|n) = d(n) - \epsilon(n) = d(n) - \epsilon(n|n-1) - \gamma(n)i(n).$$

2.4 Criticism and Conclusion

Having extolled the new joint recursive optimality framework over a considerable number of pages it is just fair to give a however short criticism of this approach, too. Most importantly, its merits should be assessed with respect to the conventional error minimization approach and to a couple of non-standard approaches as well.

In comparison to error minimization the following comments apply:

1. Both approaches give a *unifying guideline for the specification* of a large variety of adaptation algorithms. With joint recursive optimality the specification is guided by an explicit criterion that is met exactly by the algorithms. With error minimization the specification is guided by an optimality criterion that in practice is never satisfied exactly as either the underlying modeling assumptions are unrealistic (such as the existence of a time-invariant optimum coefficient setting) or the algorithm is slightly modified prior to its actual application.
2. Both approaches have their specific assets when it comes to *selecting a particular algorithm*. Within error minimization some of the global or average performance (e.g. the residual mean square error after convergence) can be predicted from the intended criterion. Within joint recursive optimality, a general analysis of tracking performance in time-varying environments is possible, see Chapter 3.
3. The joint recursive optimality framework bears much resemblance to the stochastic gradient approach of Section 2.2.1.1. This can be understood from the observation that moving the coefficient vector in the direction of the negative error gradient provides the means to reduce the error at minimum cost as far as coefficient variation is concerned [240]. The mutuality of the two approaches is again confined to convergence issues while the tracking properties of adaptation algorithms are not obvious from the gradient approach³². The gradient sets the direction *where* to move the filter coefficients but it does not tell *how far* to go in this direction³³.

The further comments pertain to *non-standard approaches* to the derivation of adaptation algorithms.

Recursive Prediction Error Methods. They are advocated as a general design principle in [31, Chapter 8] [150, pp. 88–98] and [226, 251]. First, a general explicit cost function is defined (in stochastic terms) as a measure of the ‘prediction error’, i.e. the a priori error. Realizable algorithms are obtained via a *stochastic approximation* procedure and resemble either stochastic gradient or stochastic Gauß/Newton algorithms. TSYPKIN’s approach [251] provides, in particular, the most coherent and complete framework inasmuch as all of the user’s choices (model structure, model order, ‘optimal’ cost functions, algorithm gains, etc.) are derived from explicit optimality considerations founded

³²Consider as an example the LMS algorithm and the effect of its step size μ in time-varying environments. If the algorithm is derived within the gradient approach, an extra analysis [236] is required to reveal the trade-off between time variation of the coefficients and error power, which is already present in the first principle of the joint recursive optimality approach.

³³Note that the opposite problem occurs with the zero-forcing algorithm design of Subsection 2.2.2.1. This design specifies exactly how far to move the coefficient vector, i.e. into the hyperplane defined by zero a posteriori error, but gives no indication of the optimal direction of the coefficient update

not only on the standard Gaussian assumption but on an analytical framework covering a wide range of process classes. *In these respects this approach goes far beyond the scope of the joint recursive optimality framework.* The only drawback is found in the stochastic approximation paradigm itself which is *per se* restricted to time-invariant problems and gives no guidelines on how to generalize to the time-varying case [150, p. 61]. Therefore, algorithms as simple as the LMS algorithm are not derivable from an explicit criterion within this approach but need to be explained from *ad hoc* ‘simplifications’ necessitated through practical tracking requirements.

The Control Theoretic Paradigm. It is the author’s personal impression³⁴ that there is want of taking optimality criteria as the starting point in a good deal of recursive adaptation algorithm design for control applications [10, 81, 134]. Rather, algorithms are postulated on the grounds of experience often drawn from off-line algorithm design or feedback system analysis. Only thereafter, a thorough analytical investigation of crucial algorithm properties such as stability or convergence is carried through and algorithms withstanding this scrutiny are proposed for application. This rationale is very attractive from an operational point-of-view, but it misses the *explanative power* of any optimality approach which provides both a key to the interpretation ‘what the algorithm is doing’ and—with a certain level of confidence—a justification why just this algorithm should actually solve the given application problem.

The Bayesian or Kalman Filter Approach [81, pp. 245–262][91, pp. 269–306][150, pp. 32–41]. GODARD in his 1974 paper [79] was the first to apply this approach to the development of a (growing memory) recursive least squares algorithm for adaptive transversal filters³⁵. He couched the problem in a stochastic state-space model where the unknown state corresponds to the unknown filter coefficients and the measurable observations correspond to the desired signal. The data vector is masqueraded as the measurement vector (or matrix). The optimal estimator for the filter coefficients is then defined either from a *minimum variance principle* or from a *maximum a posteriori principle* within a Bayesian framework [150, p. 32–41]. The following remarks pertain to this approach:

- It is entirely model based. Therefore, a specific algorithm is only optimal for specific statistical assumptions about the signals involved. There is no direct indication on how far from optimal the actual performance of an algorithm can be in a ‘mismatching’ statistical environment.
- The simple LMS algorithm needs already very specific modeling assumptions [19, Eqs. (84,92,94)]. This is counter-intuitive when considering the reasonable success of this algorithm in a very wide range of applications.
- A true asset of this approach is that it bears the potential for a time-varying description in its first principle. Simple random walk models or more specific *hypermmodels*

³⁴Of course, this impression *is* biased due to the author’s primary exposure to the signal processing world. Still, support for this view can also be found elsewhere, e.g. [102].

³⁵The development of the exponentially windowed RLS algorithm requires already quite tricky parameter choices within this framework, cf. [91, pp. 296–300]

[19, 20] can be exploited to capture the time-varying aspects of the application environment, see Chapter 3, and are easily incorporated in the state-space description of the problem.

- In comparison with the joint recursive optimality framework there are two major differences. First, the joint recursive optimality criterion is met deterministically by an algorithm no matter what statistical assumptions may apply. But if such assumptions are valid, then the deterministic optimality is readily interpreted in statistical terms, too. In contrast to that, the Kalman filter approach is a priori bound to a statistical interpretation. Second, joint recursive optimality allows the above-mentioned extensions towards hypermodels for time-varying environments, too. *Algorithm derivation* appears to be considerably simpler, though, as it bypasses some of the stochastic overhead in going from a design objective to the actual deterministic algorithm structure.

The chapter is concluded with a reminder what the joint recursive optimality framework *cannot explain*.

1. *Fast recursive least squares algorithms* [4, 14, 91, 96] cannot be understood from this framework alone. The designation ‘fast’ refers to the efficient computation of the vector-valued product $\mathbf{R}^{-1}(n)\mathbf{x}(n)$ as needed for the coefficient vector update of RLS algorithms. The ‘shifted structure’ of the data vector allows to organize these computations such that $O(N)$ operations are sufficient instead of $O(N^2)$ where N is the dimensionality of the data vector. As these algorithms are the exact equivalent of their computationally more involved counterparts, they do have an interpretation within joint recursive optimality. Yet, the intricate way how the recursive update of the coefficient weighting matrix $\mathbf{G}(n) = \mathbf{R}^{-1}(n)$ is circumvented cannot be predicted from this theory and is best clarified in a vector geometrical approach as described in [4].
2. Joint recursive optimality is a priori restricted to a *given complexity* in the filter structure, cf. the discussion in [81, pp. 275–283, 379–389]. This constraint is easily motivated from an applications point-of-view but should not divert from the quest for optimality in a less restricted context [81, pp. 248–275, 360–379] [251].
3. The joint recursive optimality framework has deliberately been confined to the *transversal filter structure*. As noted in the introduction, it can be generalized to the class of *linear adaptation algorithms*, i.e. to filter structures where the coefficients enter only as weights of a linear combiner which computes the filter output from otherwise arbitrarily (but non-adaptively) processed inputs. Even more, an extension to general filter structures can be achieved with the help of so-called ‘gradient filters’ that evaluate *the instantaneous sensitivities of the filter output with respect to the filter coefficients*³⁶. This idea has been successful in the development of stochastic gradient and/or RLS-type algorithms for a variety of filter structures [39, 69, 81, 159, 209, 251]. A unification within the joint recursive optimality framework is possible and will be given elsewhere.

³⁶To this end, an equivalent to TELLEGEN’s theorem ([44] and related work in [66]) can advantageously be applied to the adaptive filter which is regarded *as if* controlled in a time-varying way, cf. [241]

Chapter 3

Adaptation in Time-Varying Environments

• With a move towards application, the *tracking* properties of adaptation algorithms of the joint recursive optimality class are studied in a general setting and guidelines are developed to extend these algorithms through incorporation of time-varying models of the application environment. *Coefficient filters* emerge as a natural ingredient of the extended algorithms.

3.1 Introduction: Convergence and Tracking

While the previous chapter concentrated on the *theoretical basis* of algorithm design, the focus is now shifted towards practical application of such algorithms. The following two questions arise in this context:

1. What is the actual *performance* of the algorithm in a certain application?
2. How can the algorithm be *tailored* to specific needs of the application?

The typical applications of adaptive signal processing systems can be classified into two areas:

- *Adaptation in unknown but constant environments.*

The ‘classical’ problem of this area occurs in digital data communication via a *dial-up telephone line*: the unknown but basically time-invariant transfer characteristics of the line have to be compensated by an *adaptive channel equalizer* in the receiver [196]. In that case, the *initial convergence* speed of the adaptation algorithm is the crucial performance parameter as it determines the required ‘adaptation delay’, i.e. the interval of time that elapses between building up the connection and actually starting to transmit data. After initial convergence, the desired algorithm performance is specified from the steady-state equalization gain (in terms of intersymbol interference suppression [141, Chapter 8]), whereas the slow variation of the channel transfer function with time is usually less of a problem.

- *Adaptation in time-varying environments.*

A ‘classical’ problem of this class occurs in digital communication via a *fading HF*

channel [219]. Fading is mostly caused by multipath propagation over a time-varying medium (as is the case of radio links) or between moving transmitter/receiver combinations (as is the case in mobile radio applications). It results in both strong and rapid variations of the channel transfer function (including the gain) with time. Thus the steady-state operation of the adaptation algorithm must be tuned to *track* these variations and allow for appropriate equalization in the receiver [225, 243]. This tracking capability can, however, only be assessed in conjunction with the algorithm's ability to suppress the omnipresent noise of any real application environment. The initial convergence speed of such algorithms is of minor importance (putting aside the loose coupling that exists between this parameter and the tracking capabilities).

The dichotomy of adaptation in constant, but unknown, or in time-varying environments is not only valid w.r.t. the systems involved (the transmission channel in the above examples), but also w.r.t. the signals: the 'adaptive line enhancer' [238, pp. 354–361] tries to enhance a spectral line (i.e. a sinusoid) of *unknown* frequency which is buried in broadband noise whereas the 'adaptive linear predictive coder' [12, 76, 212] tries to predict the nonstationary speech signal using *time-varying* prediction parameters¹.

Coming back to the first question, the *actual algorithm performance* can be assessed along three major lines [19] which are treated individually in the next three subsections.

3.1.1 Convergence in Time-Invariant Environments

In time-invariant environments, several questions need to be answered:

- *Does the adaptation algorithm converge at all?* (Algorithm stability)

This question can be answered both from a deterministic or stochastic point of view. The first approach is fully developed for the assumption of periodic signals² in the *averaging theory* of [6] and [103, pp. 89–99]. Stochastic convergence studies in the literature assess the convergence of the first or second moment of the coefficient vector or even stronger forms, such as 'almost sure convergence' or convergence with probability 1, cf. [213]. The last approach, though mathematically more involved, appears to be most useful for practical purposes because it predicts the behaviour of the algorithm for 'almost all' possible input signals.

This ideal situation is not always achievable for practical adaptive algorithms as pertaining stability results are often valid for a restricted input signal class only. These conditions are referred to as *persistence of excitation* conditions and need to be assumed a priori. They are algorithm specific and there is virtually no means to check them for a given signal without running the algorithm with just this signal! To a limited extent, the excitation properties of a signal can be predicted from its

¹While emphasizing this dichotomy for the sake of clarity and simplicity, important applications which necessitate both adaptation types should not be overlooked. An example is acoustic echo-cancellation for the hands-free telephone [232] where both fast initial convergence and considerable tracking performance is required. Thus this dichotomy reflects different views taken in performance analysis rather than logically disjoint application environment descriptions.

²Of course, by allowing for increasingly large periods the results are transferable to the more practical non-periodic case as well.

statistical description such as its auto-correlation matrix or power spectral density [91, Chapter 2]. As the following questions make no sense unless convergence does occur, persistency of excitation will be assumed throughout.

- *Where does the algorithm converge to?*

The desired answer is that the algorithm converges to a ‘would-be’ optimum coefficient setting otherwise obtainable from the a priori solution of an optimum filter problem with known signal statistics (‘Wiener’ solution). Adaptation algorithms with non-decreasing gain at best converge in their mean to such an optimum, while the fluctuations of the filter coefficients around such an optimum sustain a finite, non-zero variance. If the noise component in the reference signal $d(n)$ is correlated with the input signal $x(n)$ convergence to a *biased* mean coefficient vector is to be expected for most algorithms [103, p. 31].

- *How fast does the algorithm converge?*

As an answer to this question, convergence is either characterized as being hyperbolic (e.g. for the growing memory RLS algorithm cf. [14, p. 198], [103, p. 37]) or as being exponential where the latter is the predominant scenario of non-decreasing gain algorithms [103, pp. 34-39]. In that case, a time constant can be related to the convergence curve. This constant is usually inversely proportional to the size of the algorithm gain. Furthermore, as convergence of the filter coefficient vector takes place in a multidimensional space, mutually independent modes of the convergence behaviour can be observed and characterized by individual time constants. They may depend on the eigenstructure of the auto-correlation matrix of the input data. Such dependence is removed by the appropriated choice of the coefficient weighting matrix as found in the RLS-type algorithms of the joint recursive optimality class of adaptation algorithms. The fore-going description of convergence speed refers to a situation ‘near convergence’, i.e. to the *asymptotic convergence behaviour* of the algorithm. The ‘near convergence’ condition is often invoked both during performance analysis and for algorithm design [251, pp. 42–47, Chapters 2, 3].

The asymptotic behaviour has to be distinguished from the *initial convergence behaviour*, which may differ considerably from the previous one: exact recursive least squares algorithms are notorious for their fast initial convergence³ but show an asymptotic behaviour similar to the LMS algorithm if the appropriate correspondence between exponential forgetting factor λ and the stepsize μ is established.

- *What residual error remains after convergence?*

Only decreasing gain algorithms can converge towards the ‘true’ optimum coefficient setting in a *noisy* environment. All other algorithms converge only into the vicinity of this optimum and continue with fluctuations of the coefficients around the optimum. These fluctuations can be measured either directly as the norm of the *coefficient misalignment vector* or indirectly in their effect on the filter output. This

³With the assumption of proper initialization, N time steps for a filter of order N in a time-invariant no-noise environment where the solution of the least-squares problem renders the true optimum, or approximately $2N$ time steps in a time-invariant noisy environment for convergence to within a 3 dB margin of the asymptotic residual squared error [14, p. 198].

latter approach allows to define the *excess mean square error* as the error contribution induced through the coefficient fluctuations at the filter output which would not be present if the coefficient did converge to the true optimum. This excess error is usually proportional to the algorithm gain. Therefore, adaptation algorithm design in time-invariant noisy environment is dictated from the *trade-off between convergence speed* (large algorithm gain for small time constant) *and precision of the coefficients* after convergence (small algorithm gain for small excess error). In this trade-off, joint recursive optimality is recognized again as a guideline to understand algorithm performance.

3.1.2 Large Parameter Variation and Detection of Jumps

When entering the field of time-varying environments, *abrupt* changes of this environment are usually distinguished from the *smooth* changes treated further below. While in the latter case tracking properties of the adaptation algorithm are studied, the large parameter ‘jumps’ of the former case necessitate to study at least the two following questions [20, Chapter 5]:

- *Is the algorithm able to detect parameter jumps at all?*
The recursive algorithms described in this thesis do not include decision devices and therefore do not provide such detection capabilities. Appropriate detectors have to be designed independently and may then be evaluated with respect to their fidelity in deciding upon the occurrence of a jump, their precision in locating it on the time axis, and with respect to their delay in obtaining such decisions.
- *How fast is the algorithm able to recover after a jump?*
The answer depends again on the capability of *detecting* such jumps. If they are detected (with a reasonable fidelity of course), then the algorithm may resume operation with a behaviour similar to its *initial convergence phase*. This can be achieved by proper reinitialization of the algorithm at the location of the jump and leads to a fast recovery in the case of exact RLS algorithms [43]. If, however, no detection device is available the algorithm will simply exhibit its asymptotic convergence properties, i.e. it will usually converge exponentially fast to the new parameter set, which is *slow* when compared to initial convergence! Due to the large parameter variation associated with a jump, this may induce time intervals of considerable length with a large error (either in the coefficient vector or in the adaptive filter output).

As this thesis is concerned with strictly recursive algorithms but not with the detection of abrupt changes of the application environments the reader is referred to the monograph by BASSEVILLE and BENVENISTE [13] as referenced in [20, p. 165] and to the surveys in [20, Chapter 5], [87, Chapter 4], [118, pp. 120–130] and [149]. Further references to this problem include [7, 8, 52, 98, 144, 125, 170, 194, 221, 242]. In addition to large parameter variation of the environment, abrupt changes may even occur in the structural description of the environment. To cope with such discontinuities or ‘jumps’, *structurally adaptive systems* have been proposed in [222].

3.1.3 Smooth Time Variation and Tracking

Having dealt with the two extreme poles of time variation in the application environment (i.e. strict time invariance and arbitrary, discontinuous changes), the intermediate case of smooth time variation is left over. It covers a wide range of practical environments and is *the* domain of recursive adaptation algorithms with non-decreasing gain. The formulation of the joint recursive optimality principle has already forboded this fact and the analysis in the following section will provide further support thereof. Performance evaluation in environments with smooth time variation must distinguish two sources of such variation:

- *Nonstationary input data.*

There is a number of applications which are characterized by a time-invariant statistical relationship between the input signal and the reference signal whereas these signals are individually described as nonstationary processes. Examples given in [62] include identification of linear time-invariant systems with nonstationary inputs, adaptive echo cancellation in telephone data transmission channels, and adaptive side lobe cancellation. In such environments, decreasing gain algorithms as described in Subsection 2.2.1 are the recommended choice. Even the simple LMS algorithm with decreasing step size provides almost sure convergence to the time-invariant filter coefficient optimum in that case [62]. While such an environment correctly is denoted as time-variant, the notion of *tracking* is not applicable in that case but only to the second type of time-variant environments.

- *Time-variant reference coefficient vector.*

In this case the statistical relationship between the input signal and the reference signal is itself time-varying. Such a situation may occur for instance in the *identification of a time-varying system* with measurable input $x(n)$ and an output $y(n)$ which can only be observed after disturbance by some additive measurement or observation noise $\eta(n)$. In that model, the reference signal $d(n)$ is composed of two terms:

$$d(n) = y(n) + \eta(n) \quad (3.1a)$$

$$\text{with } y(n) = \mathbf{c}_{ref}^T(n)\mathbf{x}(n) \quad (3.1b)$$

where the *reference coefficient vector* $\mathbf{c}_{ref}(n)$ accounts for the time variation of the system or ‘plant’ to be identified. The identification task can be formulated as “given the observed signal sequences $x(n)$ and $d(n)$, find an estimate $\mathbf{c}(n)$ of the coefficient vector that is as close as possible to the true coefficient vector $\mathbf{c}_{ref}(n)$ ”. The success of any solution to this problem is naturally evaluated in terms of some measure (e.g. the mean square norm) of the *misalignment vector*

$$\boldsymbol{\theta}(n) = \mathbf{c}(n) - \mathbf{c}_{ref}(n) \quad (3.2)$$

The picture is slightly different in the case of *joint process estimation* where (3.1a,b) is still assumed to hold but the task is formulated as “given the observed signal sequences $x(n)$ and $d(n)$, find an estimate $y(n|n)$ of the filter output that is as close as possible to $y(n)$.” In that case, the projection of the misalignment vector onto

the current input data vector is the natural basis for further performance evaluation of the algorithm:

$$\boldsymbol{\theta}^T(n)\mathbf{x}(n) = [\mathbf{c}(n) - \mathbf{c}_{ref}(n)]^T \mathbf{x}(n) = y(n|n) - y(n) \quad (3.3)$$

In both situations it is necessary that the adapted coefficient vector $\mathbf{c}(n)$ follows or *tracks* the time variation of the reference coefficient vector $\mathbf{c}_{ref}(n)$. *It will be shown that the tracking problem cannot be defined in absolute terms but only in relative statistical terms of both the reference coefficient vector $\mathbf{c}_{ref}(n)$ and the observation noise $\eta(n)$.* Of course, also the properties of the input sequence $x(n)$ enter into the analysis, in particular when it is modeled as a nonstationary process⁴. However, to elaborate on the crucial features of the tracking paradigm without too much notational and conceptual overhead, the simplifying assumption of stationary input data will be made wherever appropriate in the following.

What follows is organized according to the two questions formulated at the beginning of this section. First, a general performance analysis of algorithms satisfying the joint recursive optimality principle, will be made with emphasis on tracking of smooth time variations from noisy observations. Second, extensions of these algorithms will be discussed that allow to tailor them for specific needs in a given application. The specific flavour of such needs stems from a priori models describing the time evolution of the reference coefficient vector. These *hypermodels* are either known or assumed to be valid only for the particular environment under consideration. This approach will be formalized in terms of *coefficient filters* which are easily incorporated into the joint recursive optimality principle.

3.2 Tracking of Smooth Time Variations from Noisy Observations

3.2.1 Coefficient Vector Evolution in a Transform Domain

The analysis is started with the general algorithm structure of joint recursive optimality⁵ (2.43a,b):

$$\mathbf{c}(n) = \mathbf{c}(n-1) + e(n)\mathbf{G}(n)\mathbf{x}(n) \quad (3.4a)$$

$$\epsilon(n) = \gamma(n)e(n) \quad (3.4b)$$

where the a priori error $e(n)$ is

$$e(n) = d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) \quad (3.5)$$

⁴Cf. the studies in [25, 24, 72]. Moreover, in several situations the origin of the time-varying aspects of the application environment may not be separable *a posteriori* into the influence of non-stationary signals and the influence of time-varying systems. This occurs typically in digital communication problems where the receiver has only the received signal at its disposal which incorporates both effects of non-stationary transmission signals and time-varying channels.

⁵Error filtering as discussed in Subsection 2.3.5 is not included here as it would only complicate the present analysis of which the main purpose is to guide the reader towards the application of coefficient filters and *not* to provide a mess of quantitative tracking results for the large number of algorithms presented in the previous chapter.

Now, a *normalizing transformation* is introduced which acts simultaneously on the data vector $\mathbf{x}(n)$ and the coefficient vector $\mathbf{c}(n)$:

$$\tilde{\mathbf{x}}(n) = \mathbf{G}^{T/2}(n)\mathbf{x}(n) \quad (3.6a)$$

$$\tilde{\mathbf{c}}(n) = \mathbf{G}^{-1/2}(n)\mathbf{c}(n) \quad (3.6b)$$

where $\mathbf{G}^{1/2}(n)$ is a square root⁶ of the positive definite coefficient weighting matrix $\mathbf{G}(n)$ satisfying

$$\mathbf{G}^{1/2}(n)\mathbf{G}^{T/2}(n) = \mathbf{G}(n) \quad (3.7)$$

The adaptation algorithm reads with the transformed quantities:

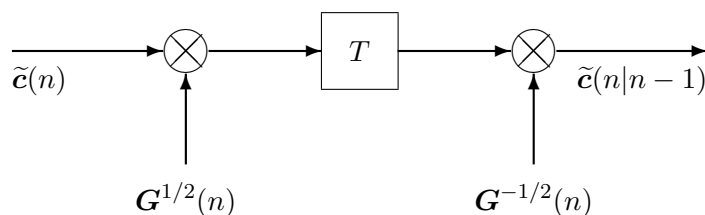
$$\tilde{\mathbf{c}}(n) = \mathbf{G}^{-1/2}(n)\mathbf{G}^{1/2}(n-1)\tilde{\mathbf{c}}(n-1) + e(n)\tilde{\mathbf{x}}(n) \quad (3.8a)$$

$$e(n) = d(n) - \tilde{\mathbf{c}}^T(n-1)\mathbf{G}^{T/2}(n-1)\mathbf{G}^{-T/2}(n)\tilde{\mathbf{x}}(n) \quad (3.8b)$$

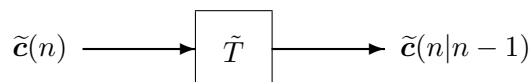
It is convenient to introduce the abbreviation

$$\tilde{\mathbf{c}}(n|n-1) = \mathbf{G}^{-1/2}(n)\mathbf{G}^{1/2}(n-1)\tilde{\mathbf{c}}(n-1) \quad (3.9)$$

for the generally time-varying scale transformation⁷ acting on the delayed coefficient vector $\tilde{\mathbf{c}}(n-1)$. An equivalent signal flow graph representation is



which will receive the short-hand notation of a ‘transforming delay’ \tilde{T} as in



⁶Note, that a square root matrix is only defined up to a *unitary matrix factor* [71, p. 831] [80, p. 395] [249, pp. 212–213]. This ambiguity can be resolved by imposing some additional constraint, such as symmetric or triangular structure, on the square-root matrix. For the present discussion it is irrelevant how the actual square root is defined. Also not necessary for the present development, direct computation of the square-root matrix is always possible: in case of LMS-type and *individual coefficient adaptation* algorithms, it is easy to evaluate the square roots of the diagonal matrix elements, in case of RLS-type algorithms the recursive propagation of the inverse coefficient matrix $\mathbf{G}^{-1}(n) = \mathbf{R}(n)$ can easily be transferred in the square-root domain as well, cf. POTTER’s algorithm as described in [150, pp. 327–328] and a related formula useful for directional forgetting algorithms in [121]. A similar strategy is found in the UDU^T -factorization proposed by BIERMAN [28].

⁷This is a first instance of a *coefficient vector predictor*, a general concept to be introduced in Section 3.4.

This is motivated below from considering the asymptotic behaviour of the algorithm for stationary input data $x(n)$ where the two square-root matrices $\mathbf{G}^{-1/2}(n)$ and $\mathbf{G}^{1/2}(n-1)$ cancel each other and the ‘transforming delay’ \tilde{T} becomes a simple delay T .

With (3.9), the algorithm (3.8a,b) reads as

$$\tilde{\mathbf{c}}(n) = \tilde{\mathbf{c}}(n|n-1) + e(n)\tilde{\mathbf{x}}(n) \quad (3.10a)$$

$$e(n) = d(n) - \tilde{\mathbf{c}}^T(n|n-1)\tilde{\mathbf{x}}(n) \quad (3.10b)$$

and is recognized as an LMS algorithm (with step size $\mu = 1$) working in the transform domain⁸. Compare this general result also to the special case of the LMS/Newton algorithm in Subsection 2.3.4.1.

Proceeding with the general analysis, the reference signal model (3.1a,b) is inserted first into (3.10b) to yield:

$$\begin{aligned} e(n) &= d(n) - \tilde{\mathbf{c}}^T(n|n-1)\tilde{\mathbf{x}}(n) \\ &= \mathbf{c}_{ref}^T(n)\mathbf{x}(n) + \eta(n) - \tilde{\mathbf{c}}^T(n|n-1)\tilde{\mathbf{x}}(n) \\ &= [\tilde{\mathbf{c}}_{ref}(n) + \tilde{\boldsymbol{\eta}}(n) - \tilde{\mathbf{c}}(n|n-1)]^T\tilde{\mathbf{x}}(n) \end{aligned}$$

where the *normalized noise vector* $\tilde{\boldsymbol{\eta}}(n)$ has been defined such that

$$\tilde{\boldsymbol{\eta}}(n) = \eta(n) \frac{\tilde{\mathbf{x}}(n)}{\|\tilde{\mathbf{x}}(n)\|^2} \Rightarrow \tilde{\boldsymbol{\eta}}^T(n)\tilde{\mathbf{x}}(n) = \eta(n) \quad (3.11)$$

and then, from (3.10a), the following important result is obtained:

$$\tilde{\mathbf{c}}(n) = \tilde{\mathbf{c}}(n|n-1) + \tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n)[\tilde{\mathbf{c}}_{ref}(n) + \tilde{\boldsymbol{\eta}}(n) - \tilde{\mathbf{c}}(n|n-1)]$$

$$\Leftrightarrow \tilde{\mathbf{c}}(n) = \mathbf{P}(n) [\gamma(n)\tilde{\mathbf{c}}(n|n-1) + (1-\gamma(n))(\tilde{\mathbf{c}}_{ref}(n) + \tilde{\boldsymbol{\eta}}(n))] + \mathbf{P}^\perp(n)\tilde{\mathbf{c}}(n|n-1) \quad (3.12)$$

where the projection operators $\mathbf{P}(n)$ and $\mathbf{P}^\perp(n)$ have been defined as follows:

- The *projection* onto the transformed data vector

$$\mathbf{P}(n) = \frac{\tilde{\mathbf{x}}(n)\tilde{\mathbf{x}}^T(n)}{\|\tilde{\mathbf{x}}(n)\|^2} \quad (3.13a)$$

- The projection onto its *orthogonal complement*

$$\mathbf{P}^\perp(n) = \mathbf{I} - \mathbf{P}(n) \quad (3.13b)$$

Furthermore, the identity

$$\begin{aligned} \|\tilde{\mathbf{x}}(n)\|^2 &= \tilde{\mathbf{x}}^T(n)\tilde{\mathbf{x}}(n) = \mathbf{x}^T(n)\mathbf{G}^{1/2}(n)\mathbf{G}^{T/2}(n)\mathbf{x}(n) \\ &= \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) = 1 - \gamma(n) \end{aligned} \quad (3.14)$$

⁸The fact that $\tilde{\mathbf{c}}(n|n-1)$ does not equal $\tilde{\mathbf{c}}(n-1)$ exactly will be ignored for the moment.

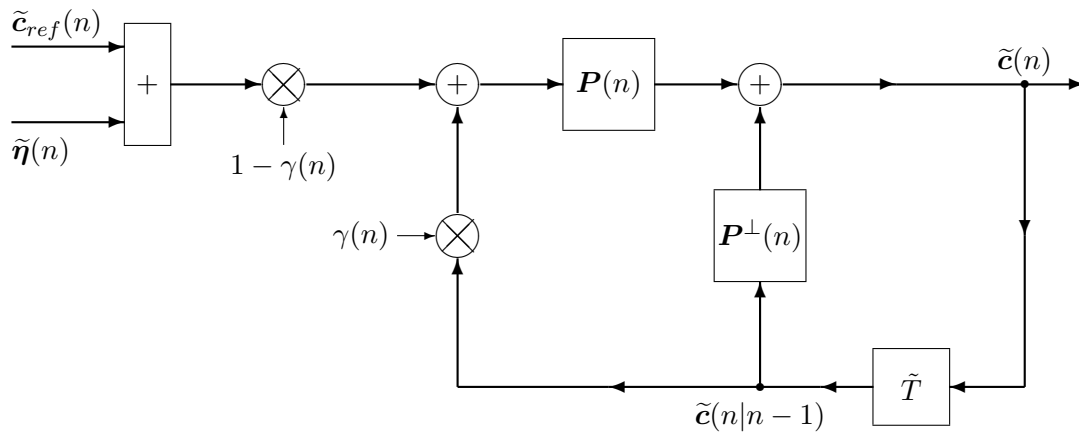


Figure 3.1: Transform domain signal flow graph related to the tracking properties of the general adaptation algorithm.

has been used to obtain (3.12). This result gives a clear picture of the tracking properties of the general adaptation algorithm (3.4a,b) when visualized as a signal flow graph as given in Figure 3.1. In the following remarks, this flow graph will serve as the firm basis to develop both formal aspects and intuition of tracking behaviour.

Remark 1. Equation (3.12) and the corresponding signal flow graph in Figure 3.1 are derived deterministically *without any approximation* and *without any assumption* on the properties of the time-varying reference coefficient vector $c_{ref}(n)$ or the noise $\eta(n)$. This is in contrast with the otherwise similar *filtering interpretation* of tracking properties discussed for the LMS algorithm in [236].

Remark 2. The signal flow graph block symbols $P(n)$ and $P^\perp(n)$ stand for a projection of the (coefficient) vector passing through the block onto the tranformed data vector or its orthogonal complement, respectively. From a computational point of view, such operation requires a full matrix \times vector multiplication in general. The ‘operator block’ notation is however preferred over a ‘matrix multiplier’ notation as it shows explicitly how the coefficient vector $\tilde{c}(n)$ is computed from its two components *parallel* and *orthogonal* to the current data $\tilde{x}(n)$. Note furthermore that the two operators add up to the identity operator

$$P(n) + P^\perp(n) = I$$

such that their effect can be interpreted as a kind of *soft switch* between the *parallel* and *orthogonal branch* of the feedback loop in the signal flow graph.

Remark 3. In the parallel branch of the feedback loop, a *weighted average* of the old information $\tilde{c}(n|n-1)$ and the new information $\tilde{c}_{ref}(n) + \tilde{\eta}(n)$ is computed. The weighting is controlled by the conversion factor $\gamma(n)$. If this factor is close to 1, the old information is propagated with little forgetting and little correction from the new information. If this

factor is small (or even close to 0), old information is partially forgotten and replaced by the equivalent amount of new information.

In the orthogonal branch of the feedback loop, old information is propagated unchanged as all observable new information is parallel to the transformed data vector $\tilde{\mathbf{x}}(n)$. The deterministic stability bound $|\gamma(n)| \leq 1$ is now easily understood from this signal flow graph as a sufficient condition to keep the overall gain of the feedback loop less or equal 1 and to guarantee thereby that the coefficient vector $\tilde{\mathbf{c}}(n)$ remains bounded for bounded inputs $\tilde{\mathbf{c}}_{ref}(n)$ and $\tilde{\boldsymbol{\eta}}(n)$.

Note furthermore that the ‘soft switch’ controls $\mathbf{P}(n)$ and $\mathbf{P}^\perp(n)$ are independent of the scaling of both the gain matrix $\mathbf{G}(n)$ and the data vector $\mathbf{x}(n)$ as their absolute size is cancelled in the definitions (3.13a,b). They do depend on the directional information in $\mathbf{G}(n)$ and $\mathbf{x}(n)$ only, or in short, on the *directional distribution* [118, p. 74] of the transformed data vector $\tilde{\mathbf{x}}(n)$. The ‘averaging’ control $\gamma(n)$ does however depend on the scaling of the gain matrix $\mathbf{G}(n)$ and of the data vector $\mathbf{x}(n)$ or, in short, on the *length*⁹ of the transformed data vector $\tilde{\mathbf{x}}(n)$. Thus these two statistical properties of the input data are manifested separately in the coefficient adaptation algorithm.

Remark 4. As stated before, the ‘soft switch’ $\mathbf{P}(n)$ depends on the *directional distribution* of the transformed data vector $\tilde{\mathbf{x}}(n)$.

For LMS-type algorithms, $\tilde{\mathbf{x}}(n)$ is collinear with $\mathbf{x}(n)$ and, therefore, this distribution will in general be non-uniform, i.e. there are directions of the N -dimensional input data vector space which occur more frequently than others. For stationary data this is due to the auto-correlation properties of the input data where ‘frequent’ directions correspond to large eigenvalues of the auto-correlation matrix and vice versa [14, Chapter 3]. This situation is usually described by transforming the data (and coefficient vector) into the eigencoordinates of the auto-correlation matrix and analysing the behaviour of the resulting N (decoupled) modes of the adaptation algorithm independently, cf. [236].

For RLS-type algorithms, $\tilde{\mathbf{x}}(n)$ is approximately *orthonormalized* through the transform $\mathbf{G}^{T/2}(n)\mathbf{x}(n)$ where $\mathbf{G}(n)$ is proportional to an inverse estimate of the auto-correlation matrix of the data. In that case, the directional distribution of $\tilde{\mathbf{x}}(n)$ is approximately uniform and all N modes exhibit approximately similar tracking behaviour.

Remark 5. A particular situation arises, if the data $\mathbf{x}(n)$ are taken from an M -dimensional subspace with $M < N$ [118, p. 80]. This occurs e.g. for an input signal that is composed of $M/2$ sinusoids and contains no random component [96, p. 57]. In such a situation no matrix transformation $\mathbf{G}^{T/2}(n)$ whatsoever can achieve a uniform directional distribution in $\tilde{\mathbf{x}}(n)$: the transformed data vector will lie in an M -dimensional subspace as well, and there exists an $(N - M)$ -dimensional complementary subspace of the coefficient vector space which *never* passes the ‘parallel branch’ of the feedback loop in Figure 3.1—or, in other words—which never can be tracked by any of the discussed algorithms. The

⁹In the LMS algorithm, this length depends on the instantaneous input signal power such that $\gamma(n)$ is large for small input power and vice versa. Therefore, the averaging of old and new information is made dependent on the instantaneous ‘signal-to-noise ratio’, i.e. the ratio of the (time-varying) input signal power to the (supposedly) time-invariant observation noise power. In RLS-type algorithms, this length is related to the likelihood of the input data as discussed in Subsection 2.3.4.1.

component of the initial coefficient vector that lies in this non-excited subspace will propagate forever undamped in the orthogonal branch of the feedback loop. This performance failure can be assessed from three viewpoints:

1. For *system identification* this misbehaviour is essential as the untracked component in the unexcited subspace will persist in the misalignment vector $\boldsymbol{\theta}(n)$ of (3.2).
2. *Signal estimation* is not affected through this misbehaviour as the unexcited subspace component of the coefficient vector is not observed at the filter output when the misalignment vector $\boldsymbol{\theta}(n)$ is projected onto the data vector $\boldsymbol{x}(n)$ as in (3.3).
3. From a *numerical point of view*, an undamped feedback loop is always prone to divergence due to accumulation of numerical errors generated by any finite-precision arithmetic implementation. This effect is observed in practice and known as *long-term drift* of the coefficient vector¹⁰ as described e.g. in [14, pp. 112–114] [42, 78] [96, p. 58].

Concluding this remark, it is mentioned that input data causing this type of tracking (and stability!) problem, are referred to as violating a *persistence of excitation* condition [5, 29] [31, p. 346, pp. 427–428] [103, Chapter 5] [131].

Remark 6. For many of the algorithms, the gain matrix $\mathbf{G}(n)$ only depends on $\boldsymbol{x}(n)$ but neither on $d(n)$ nor on $\boldsymbol{c}(n)$ ¹¹. In this case, the signal flow graph Figure 3.1 is recognized as a *linear first-order time-varying filter* operating simultaneously on both the reference coefficient vector $\tilde{\boldsymbol{c}}_{ref}(n)$ and the normalized noise vector $\tilde{\boldsymbol{\eta}}(n)$ to produce the coefficient vector $\tilde{\boldsymbol{c}}(n)$ at its output. This filter has been called a *learning filter* in [124] following the parlance of time-invariant environments where one refers to learning characteristics such as the *learning curve* etc. [236]. As both the reference coefficient and the noise undergo exactly the same filtering operation, tracking in the presence of noise is contingent upon (spectrally) different behaviour of the reference coefficient vector and the observation noise. This coincides with the frequent assumption of ‘slow’ variation of the reference coefficients (which are modeled as a vector-valued low-pass process or signal) and approximately white observation noise.

Remark 7. To proceed further in the analysis, the stationarity assumption is invoked for the input data $\boldsymbol{x}(n)$. In an asymptotic analysis, i.e. after dying out of the initial transients, the input-dependent control $\gamma(n)$ will be replaced by its expectation

$$\gamma = E\{\gamma(n)\}_{n \gg 1} \quad (3.15)$$

and the ‘transforming delay’ \tilde{T} by a simple delay T . Table 3.1 lists typical values of γ for various algorithm options. This table displays several important features of ‘typical’ adaptation algorithms:

¹⁰Counter-measures such as coefficient *leakage* will be discussed in Section 3.4.

¹¹Exceptions are found in the variable-step algorithm (2.60b,b) where the step-size parameters $\mu_i(n)$ depend on the correlation $e(n)x(n-i)$, in some exact RLS algorithms (Table 2.1, entries # 3, 4, 9, 10, and 11) where the data weighting factors depend on the error signal $e(n)$, and for the same reason, in the prediction-error controlled directional forgetting algorithms (Table 2.2, entries # 2 and 3).

Table 3.1: ASYMPTOTIC EXPECTED VALUE OF CONVERSION FACTOR FOR STATIONARY INPUT DATA.

#	Algorithm	$\gamma(n)$	$\gamma = E\{\gamma(n)\}_{ n \gg 1}$
1.	LMS algorithm	$1 - \mu \ \mathbf{x}(n)\ ^2$	$1 - \mu \text{trace}\{\mathbf{R}\}$
2.	Projection algorithm	$1 - \mu$	$1 - \mu$
3.	Zero-forcing algorithms	0	0
4.	Signed regressor algorithm	$1 - \mu \sum_{i=0}^{N-1} x(n-i) $	$1 - \mu N E\{ x(n) \}$
5.	LMS/Newton algorithm	$1 - \mu \mathbf{x}^T(n) \mathbf{R}^{-1} \mathbf{x}(n)$	$1 - \mu N$
6.	Exact RLS, growing memory and selective data weighting	$1 - \lambda_2(n) \mathbf{x}^T(n) \mathbf{R}^{-1}(n) \mathbf{x}(n)$	1
7.	Exact RLS, exponential and mixed data weighting	$1 - \lambda_2(n) \mathbf{x}^T(n) \mathbf{R}^{-1}(n) \mathbf{x}(n)$	λ_1^N
8.	RLS, directional forgetting	$1 - \beta(n) \mathbf{x}^T(n) \mathbf{R}^{-1}(n) \mathbf{x}(n)$	λ

1. Except for the LMS algorithm and the signed regressor algorithm, all other algorithms have an asymptotic value of γ which is *independent of the statistical characteristics* of the input data. Straight-forward comparison of those algorithms is, therefore, made possible if their parameters (such as the step size μ , the forgetting factor λ or the filter order N) are chosen such that a matching value of γ is obtained. As an example consider RLS algorithms with exponential forgetting and directional forgetting algorithms with constant forgetting factor. Their tracking properties are comparable [121], if their corresponding weighing factors are chosen subject to

$$\lambda_{exp}^N = \lambda_{dir} \quad (3.16)$$

Even in the context of the statistics-dependent LMS algorithm such equivalences have been established: the LMS and LMS/Newton algorithm have been shown in [239, Eq. (82)] to have comparable tracking properties, if their step-size parameters are chosen subject to

$$\mu_{LMS} = \frac{N}{\text{trace}\{\mathbf{R}\}} \mu_{LMS/Newton} \quad (3.17)$$

And finally, in [146, Eq. (5.2)] and [59, Eqs. (4.11), (4.17)] a similar equivalence is obtained between the LMS and the exponentially weighted RLS algorithm (with the forgetting factor λ close to 1) for

$$\lambda_{RLS} = 1 - \mu_{LMS} \frac{\text{trace}\{\mathbf{R}\}}{N} \quad (3.18)$$

which is again obtained from equating their γ -values (under the same assumption of λ close to 1)¹².

¹²ELEFThERIOU and FALCONER maintain in their paper [59] that RLS algorithms would still achieve a performance advantage if the filter length N is high and the maximum allowable step size μ of the LMS

2. *Algorithms with $\gamma = 1$ allow no tracking at all* as the input $\tilde{\mathbf{c}}_{ref}(n) + \tilde{\boldsymbol{\eta}}(n)$ to the signal flow graph Figure 3.1 is totally zeroed through the multiplier $1 - \gamma$. Prominent members of this ‘no tracking’ class of algorithms are all decreasing-gain algorithms, in particular the RLS algorithms with growing memory and with selective data weighting.
3. *Zero-forcing algorithms* are characterized by $\gamma = 0$ and therefore allow in a sense for the *fastest possible tracking* of all algorithms referenced in Table 3.1. This is understood from the effect γ has on the weighted average that is computed from the new information $\tilde{\mathbf{c}}_{ref}(n) + \tilde{\boldsymbol{\eta}}(n)$ and the old information $\tilde{\mathbf{c}}(n|n-1)$. Zero-forcing algorithms put maximum weight on the new information while minimizing the amount of smoothing over past observations. What superficially appears as an advantage is, however, not always realized as such in practice: to achieve a certain ‘quality’ of tracking (as measured in terms of the misalignment vector, cf. (3.2) and (3.3)) an optimum value of γ *between* 0 and 1 is desirable that allows the flexibility of trading ‘coefficient lag’ for ‘misadjustment due to observation noise’ as outlined below.

3.2.2 Misalignment Vector and Learning Filter Interpretation

The general analysis is continued from (3.12) with the development of an expression for the (transformed) misalignment vector $\tilde{\boldsymbol{\theta}}(n)$:

$$\begin{aligned}\tilde{\boldsymbol{\theta}}(n) &= \tilde{\mathbf{c}}(n) - \tilde{\mathbf{c}}_{ref}(n) \\ &= \mathbf{P}(n) [\gamma(n)(\tilde{\mathbf{c}}(n|n-1) - \tilde{\mathbf{c}}_{ref}(n)) + (1 - \gamma(n))\tilde{\boldsymbol{\eta}}(n)] \\ &\quad + \mathbf{P}^\perp(n) [\tilde{\mathbf{c}}(n|n-1) - \tilde{\mathbf{c}}_{ref}(n)]\end{aligned}\tag{3.19}$$

where for the *stationary asymptotic case* $\tilde{\mathbf{c}}(n|n-1) = \tilde{\mathbf{c}}(n-1)$. Therefore the difference

$$\tilde{\mathbf{c}}(n|n-1) - \tilde{\mathbf{c}}_{ref}(n) = \tilde{\boldsymbol{\theta}}(n-1) - [\tilde{\mathbf{c}}_{ref}(n) - \tilde{\mathbf{c}}_{ref}(n-1)]\tag{3.20}$$

is composed of the misalignment vector at time $(n-1)$ and the *first difference* of the reference coefficient vector. Inserting (3.20) into (3.19) yields with the asymptotic value of $\gamma(n)$:

$$\tilde{\boldsymbol{\theta}}(n) = [\gamma\mathbf{P}(n) + \mathbf{P}^\perp(n)] [\tilde{\boldsymbol{\theta}}(n-1) - \tilde{\mathbf{c}}_{ref}(n) + \tilde{\mathbf{c}}_{ref}(n-1)] + (1 - \gamma)\tilde{\boldsymbol{\eta}}(n)\tag{3.21}$$

The linearity of the recursion (3.21) in both its ‘inputs’ $\tilde{\mathbf{c}}_{ref}(n)$ and $\tilde{\boldsymbol{\eta}}(n)$ and its ‘output’ $\tilde{\boldsymbol{\theta}}(n)$ allows the decomposition into two separate recursions describing the misalignment $\tilde{\boldsymbol{\theta}}_{lag}(n)$ due to the time-variant reference coefficients and the misalignment $\tilde{\boldsymbol{\theta}}_{noise}(n)$ due to the observation noise:

algorithm is, therefore, limited to small values to satisfy the stability constraint $|\gamma| \leq 1$ which—on the other hand—is satisfied for any choice of forgetting factor λ in exponentially weighted RLS algorithms. This argument is flawed as (3.18) is only valid for λ close to 1 whereas the exact formulae for γ in Table 3.1 show that any choice of $0 \leq \lambda \leq 1$ allows to find a stable value for μ such that the two algorithms have a matching γ and therefore similar tracking behaviour. BENALLAL and GILLOIRE have shown in [18] that the situation is even reversed in practice where the stability problems associated with the (implicit) inversion of the auto-correlation matrix in exact RLS algorithms limit the choice of the forgetting factor λ to values so close to 1 that the LMS algorithm actually achieves better tracking performance! A similar result is found in [165],too.

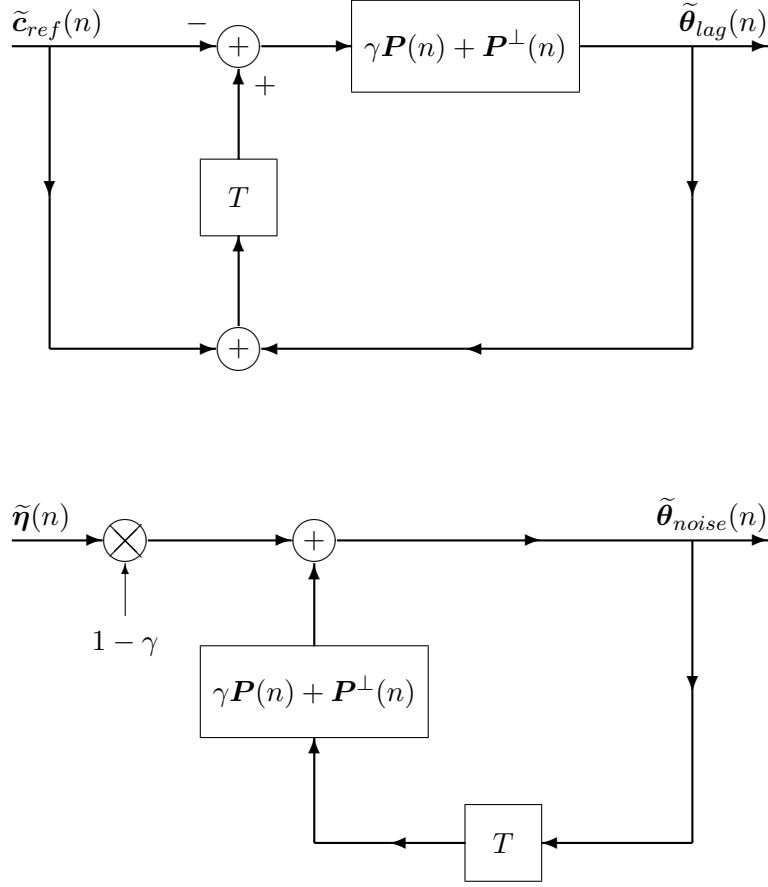


Figure 3.2: Transform domain signal flow graphs for the *lag filter* and *noise filter* of the general adaptation algorithm.

$$\begin{aligned} \tilde{\boldsymbol{\theta}}(n) &= \tilde{\boldsymbol{\theta}}_{lag}(n) + \tilde{\boldsymbol{\theta}}_{noise}(n) & (3.22a) \\ \tilde{\boldsymbol{\theta}}_{lag}(n) &= [\gamma \mathbf{P}(n) + \mathbf{P}^\perp(n)] [\tilde{\boldsymbol{\theta}}_{lag}(n-1) - \tilde{\mathbf{c}}_{ref}(n) + \tilde{\mathbf{c}}_{ref}(n-1)] & (3.22b) \\ \tilde{\boldsymbol{\theta}}_{noise}(n) &= [\gamma \mathbf{P}(n) + \mathbf{P}^\perp(n)] \tilde{\boldsymbol{\theta}}_{noise}(n-1) + (1-\gamma) \tilde{\boldsymbol{\eta}}(n) & (3.22c) \end{aligned}$$

The latter two equations correspond to the two first-order filters shown in the signal flow graphs of Figure 3.2. If all parameters of these two filters are regarded as frozen to their instantaneous values, a z -transform analysis becomes possible. The poles of both filters are controlled through the matrix $\gamma \mathbf{P}(n) + \mathbf{P}^\perp(n)$ which has $N-1$ eigenvalues equal to 1 and one eigenvalue equal to γ . The ‘lag filter’ has its zeroes all on the unit circle at $z=1$ whereas the ‘noise filter’ has its zeroes at $z=0$. The two filters play an antagonist role in determining the total misalignment $\tilde{\boldsymbol{\theta}}(n)$: the lag filter is high-pass and the noise-filter

is low-pass while their cut-off frequencies are simultaneously controlled by the common poles defined above.

It is important to realize that the decomposition of the total misalignment vector into a lag component and a noise component is always possible, without any approximation, due to the linearity of the considered coefficient adaptation algorithms alone.

This decomposition is also valid for the non-asymptotic case, cf. (3.19). Only when evaluating the (squared) norm or the covariance matrix of the total misalignment vector, further modeling assumptions are necessary, cf. for an example the random walk or autoregressive models used by [59, 236, 213] to describe the time evolution of the reference coefficient vector. The same remark is true for the evaluation of the norm of the misalignment vector ‘as seen’ at the filter output, i.e. the residual error $|\tilde{\boldsymbol{\theta}}^T(n)\tilde{\mathbf{x}}(n)|^2 = |\boldsymbol{\theta}^T(n)\mathbf{x}(n)|^2$. A general treatment of such second-order analysis has been attempted recently in [149]. In a similar vein, Appendix C shows how several of the results for the excess error due to time-variant reference coefficients and due to noise as found in [18, 45, 59, 146, 165, 236, 239] can be reproduced from specializing the general approach presented so far. Two rule-of-thumbs are worth mentioning here due to their general applicability:

The *excess error due to observation noise* (which is assumed to be white, zero-mean and independent of all other quantities involved) obeys the following (asymptotic) law:

$$\text{Avg} \left\{ \left(\boldsymbol{\theta}_{\text{noise}}^T(n-1)\mathbf{x}(n) \right)^2 \right\} = \frac{1-\gamma}{1+\gamma} \text{Avg} \{ \eta^2(n) \} \quad (3.23)$$

where *Avg* denotes a *sample-path average*, cf. Appendix C.1. This formula holds not only for small gains (i.e. $\gamma \approx 1$), but also for large gains. In particular, *zero-forcing algorithms* have $\gamma = 0$ and, from (3.23), show 100% misadjustment due to noise only (i.e. a 3 dB increase in error over the ‘ideal solution’ $\mathbf{c}(n) = \mathbf{c}_{\text{ref}}(n)$).

The *time constant* characterizing the first-order learning filter of the total excess error due to noise and coefficient lag is given by

$$\tau = \frac{N}{1-\gamma^2} \quad (3.24)$$

which holds for arbitrary gains, if the filter order is sufficiently high ($N \gg 1$). As $0 \leq \gamma^2 \leq 1$ a lower bound of $\tau \geq N$ follows.

3.2.3 Summary of Tracking Analysis

Summarizing this section on the tracking properties of the general adaptation algorithm (3.4a,b), the following main issues are enhanced once more:

1. The ‘tracking’ equation (3.12) is a first-order linear difference equation describing the adapted coefficients in terms of the reference coefficients and the observation noise.
2. Due to the *linearity* of the equation, the influence of these two inputs on the *misalignment vector* can be decomposed into two separate ‘learning filters’ operating independently.

3. As the tracking equation is of *first order*, the two filters are again describable as first-order high- and low-pass, respectively. Their cut-off frequencies are controlled only by the asymptotic conversion factor γ . The first-order behaviour is also recognized in the exponential convergence properties of these algorithms.
4. The *influence of the input data* vector on the filters is limited to
 - stability issues: long-term drift due to lack of persistent excitation, power constraints in several algorithms to guarantee $|\gamma| \leq 1$.
 - misalignment vector as a whole: only the coefficient misalignment shows non-uniform tracking speed and accuracy according to the degree of excitation of a particular mode (i.e. dimension of the associated vector space) which is related to the directional distribution or auto-correlation matrix of the input data.

The excess error observed at the filter output is, however, independent of this directional distribution. In this respect, RLS-type algorithms show no performance advantage over LMS-type algorithms, cf. also [14, p. 207], [59], [239].

5. The present analysis is valid for a whole class of algorithms following the general pattern (3.4a,b). This is different from previous studies¹³ which present more detailed results for specific algorithms. Only a few analyses of comparable scope have appeared [21, 19, 65, 64, 124, 149] in the attempt to cover the general features of linear first-order adaptation algorithms¹⁴. Their striking result is—in present terminology—, *apart from the single cut-off frequency parameter γ , there is no flexibility to tailor an algorithm to any a priori description of the time-varying application at hand*. This result necessitates to look for extensions of the basic algorithm pattern which is the topic of the next two sections.

3.3 Models for the Time Evolution of Filter Coefficients

The analysis in the previous section has revealed the general first-order tracking behaviour of most of the standard adaptation algorithms. This can be seen as a direct consequence of the underlying joint recursive optimality principle: it is phrased in terms of the *first difference* of the coefficient vector and of the power of the a posteriori error. Furthermore, the choice of the algorithm gain (or, to be more precise, the conversion factor $\gamma(n)$) determines the trade-off between the excess errors due to coefficient lag and due to observation noise, just as the trade-off between coefficient variation and a posteriori error is controlled through $\gamma(n)$ in the optimality criterion. This limited flexibility is, however, insufficient if prior knowledge on the specific nature of the time-variations is at hand. Consider the identification of a periodically time-varying system with (at least approximately) known period. As the standard adaptation algorithms react as first-order low-pass filters to the

¹³Cf. [59, 63, 72, 146, 22, 153, 155, 186, 213, 233, 236, 239].

¹⁴NIEDŹWIECKI covers in a series of papers [180, 181, 182] the *learning filter* characteristics of generally weighted least squares estimators which are shown to have an impulse response directly related to the weighting sequence used to define the least squares problem. These filters are, therefore, usually not restricted to first-order. The specific assets of such higher-order filters will be further discussed in Section 3.4.

reference coefficients, the related cut-off frequency has to be chosen well above the frequency specified through the known period. If this frequency is high, the excess error due to observation noise will also be high as it is filtered by the same low-pass filter as the reference coefficients! In such an application environment it would be desirable to *tailor* the adaptation algorithm such that it has a *narrow-band learning filter* matched to the period of the time-varying system. Generally speaking, the adaptation algorithms should be extended by the incorporation of prior models for the time evolution of the coefficient vector. As the filter coefficients themselves are parameters of a system or signal model, the time evolution models are called *hypermmodels* by BENVENISTE in [19, 20].

3.3.1 The Deterministic Paradigm: Band-Limited Evolution

Within the deterministic paradigm, the filter coefficients $\mathbf{c}(n)$ are regarded as deterministic functions that have a finite support in time. These functions can be *expanded into series* of (usually, but not necessarily) orthogonal base functions over the same finite time support. While such series expansion contains in principle an infinite number of terms, the a priori assumption of smooth time variations allows to truncate the series to a finite number of terms. This truncation essentially prescribes the *bandwidth of the time evolution* of the filter coefficients. The bandwidth is either chosen on physical grounds (e.g. in speech modeling where the time variations are expected to be no faster than 100 Hz because the velocity of speech organ movements is limited through their own inertia) or imposed from practical needs (e.g. in predictive speech coding where the time variation of the predictor parameters is directly related to the bit rate of the coder), cf. [15]. The clue of the approach is that, after insertion of the truncated series expansion into the filter equations, the problem can be reduced to the optimal choice of the *time-invariant expansion coefficients* for the given block of data.

To formalize this approach the filter equation is written as¹⁵:

$$\begin{aligned}\hat{y}(n) &= \mathbf{c}^T(n)\mathbf{x}(n) = \sum_{i=0}^{N-1} c_i(n)x(n-i) \\ &= \left[\sum_{i=0}^{N-1} c_i(n)q^{-i} \right] x(n)\end{aligned}\quad (3.25)$$

In this equation, the sequence of operations on the input signal $x(n)$ is

1. delay q^{-i}
2. multiplication with the coefficients $c_i(n)$
3. accumulation of the N products

If the filter were time-invariant, an equivalent *transposed structure* could be used for its implementation:

$$\hat{y}(n) = \left[\sum_{i=0}^{N-1} q^{-i}c_i(n) \right] x(n)\quad (3.26)$$

¹⁵The filter output is denoted as $\hat{y}(n)$ as it is computed from filter coefficients $\mathbf{c}(n)$ which are optimized over a whole block of data. This differs from the usual a posteriori filter output $y(n|n)$ which is obtained from the recursively computed filter coefficients.

Here, the sequence of operation is

1. multiplication of the input signal $x(n)$ with the coefficients $c_i(n)$
2. delay q^{-i} of the N products
3. accumulation of the delayed products.

Clearly, the two structures (3.25) and (3.26) are not equivalent in the time-varying case as

$$q^{-i}c_i(n) = c_i(n-i)q^{-i} \neq c_i(n)q^{-i} \quad (3.27)$$

It is common to ascribe different interpretations to the filter coefficients $c_i(n)$ in the two structures: in (3.25), they are regarded as a time-varying *memory function* of the filter whereas in (3.26) they act as a time-varying *impulse response* of the filter. The second interpretation is better suited to applications (such as time-varying spectral estimation [82],[83, pp. 69–87]) and reduces in the sequel the number of computations significantly¹⁶. Therefore this interpretation is adopted here.

Using a set of K base functions¹⁷ $\psi_k(n)$ the coefficients $c_i(n)$ are decomposed as

$$c_i(n) = \sum_{k=0}^{K-1} c_{ik}\psi_k(n) \quad (3.28)$$

where c_{ik} are a set of $K \times N$ time-invariant expansion coefficients. Insertion of this series expansion into (3.26) yields

$$\begin{aligned} \hat{y}(n) &= \left[\sum_{i=0}^{N-1} q^{-i} \sum_{k=0}^{K-1} c_{ik}\psi_k(n) \right] x(n) \\ &= \sum_{i=0}^{N-1} \sum_{k=0}^{K-1} c_{ik}q^{-i}\psi_k(n)x(n) \\ &= \sum_{i=0}^{N-1} \sum_{k=0}^{K-1} c_{ik}\psi_k(n-i)x(n-i) \end{aligned} \quad (3.29)$$

and with the definition of the K modulated input signals

$$x_k(n) = \psi_k(n)x(n) \quad k = 0, \dots, K-1 \quad (3.30)$$

(3.29) reads more compactly as

$$\hat{y}(n) = \sum_{i=0}^{N-1} \sum_{k=0}^{K-1} c_{ik}x_k(n-i). \quad (3.31)$$

This equation can be interpreted as an FIR filter operating on a vector-valued input signal to produce a scalar output. To see this, define

¹⁶By a factor of N when evaluating the products of *data* \times *base functions*.

¹⁷Various types of base functions have been considered in the literature, such as polynomials, sine/cosine functions or prolate spheroidal functions. For a comparison, see [83].

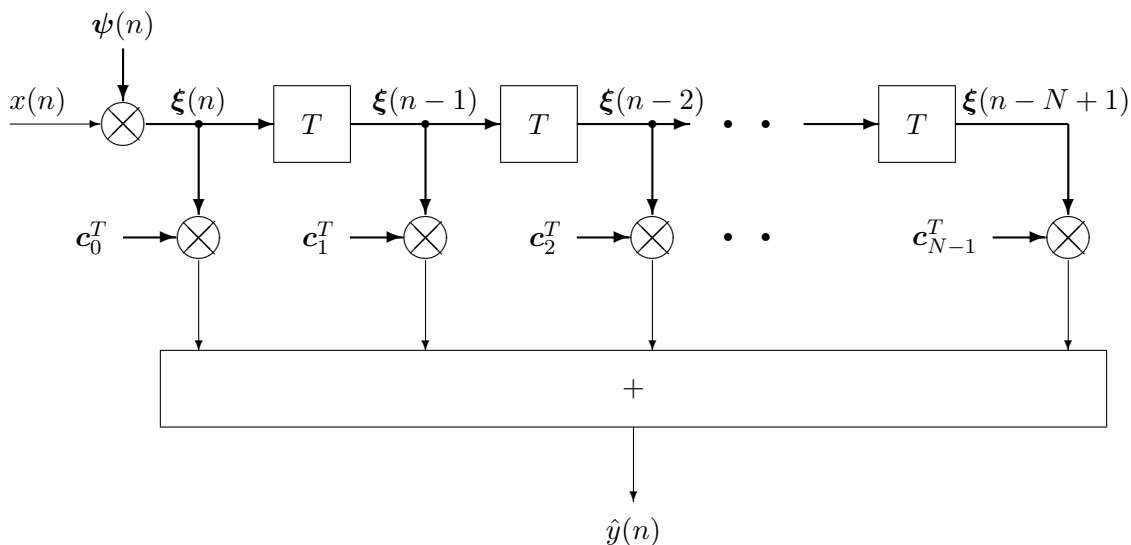


Figure 3.3: Signal flow graph of FIR filter with input modulated by K -dimensional base function vector $\psi(n)$.

- the *expansion coefficient vectors*

$$\mathbf{c}_i = [c_{i,0}, c_{i,1}, \dots, c_{i,K-1}]^T \quad i = 0, \dots, N-1 \quad (3.32)$$

- the *vector of modulated inputs*

$$\boldsymbol{\xi}(n) = \boldsymbol{\psi}(n)x(n) = [x_0(n), x_1(n), \dots, x_{K-1}(n)]^T \quad (3.33)$$

of dimension K each. Then (3.31) renders

$$\hat{y}(n) = \sum_{i=0}^{N-1} \mathbf{c}_i^T \boldsymbol{\xi}(n-i) \quad (3.34)$$

which corresponds to the signal flow graph in Figure 3.3.

And that's it¹⁸. Comparing (3.34) with the original filter equation (3.25), the problem of determining a set of *scalar time-varying* coefficients $c_i(n)$ has been transformed into the problem of determining a set of *vector-valued time-invariant* coefficients \mathbf{c}_i . In both cases these coefficients enter as the weights of a linear combiner structure. Therefore, the 'optimal' coefficients in (3.34) can be found using standard estimation techniques for time-invariant problems! This general approach to handle band-limited coefficient evolution has been around since the early 70's¹⁹. It has been brought to the attention of the signal processing community mainly through three seminal papers [82, 89, 147]. Since 1983, a growing number of papers is devoted to this technique out of which a number of

¹⁸A weak attempt to translate GRENIER's *tout est dit* in [83, p. 161].

¹⁹For an excellent overview of published work until 1984, see [83, pp. 149-158].

references is provided here [1, 3, 37, 38, 41, 67, 84, 85, 111, 161, 160, 183, 223]. When assessing the usefulness of this approach for recursive adaptation algorithm design, the following remarks are in order:

Remark 1. The approach is a priori designed for a finite block of data. This is a precondition for the validity of a truncated series expansion (3.28) of the (approximately band-limited) time-varying coefficients $c_i(n)$ in terms of time-invariant coefficients c_{ik} . There is no obvious method how adjacent blocks of data should be processed such that smooth transitions of the coefficient estimates are guaranteed at block boundaries²⁰. A possible alternative could emerge from allowing the expansion coefficients c_{ik} to be slowly time-varying on their own. Then, recursive, non-decreasing gain adaptation algorithms could be employed for their estimation instead of the block processing techniques suggested in the original formulation. As to the author's knowledge, an implementation of this rationale has not appeared, so far. First attempts to develop recursive versions of the band-limited coefficient evolution approach are reported in [3].

Remark 2. If the approach is used to track rapid variations of a reference coefficient vector $\mathbf{c}_{ref}(n)$, a rather large number of unknown variables (i.e. the $K \times N$ expansion coefficients c_{ik}) has to be estimated from the limited amount of data contained in the given block length. This problem gets more difficult with increasing bandwidth of the coefficients' time evolution, i.e. with the number K of base functions. It manifests itself in numerical ill-conditioning of the least-squares problem which has to be solved for c_{ik} [40]. LEE has shown how to reduce this problem through application of *regularization methods* in [142, 143]. In conclusion, the *band-limited evolution* model is useful only for a block-processing approach to the tracking of slowly time-varying coefficients or, more precisely to the related *coefficient matching* problem²¹. Recursive methods for the implementation of these ideas deserve further effort²²

3.3.2 The Stochastic Paradigm: Smoothness Priors

The idea to use smoothness priors in the estimation of time-varying parameters goes back to one of the fathers of the sampling theorem: E.T. WHITTAKER published in 1923 a method [235] that consists in the trade-off of *parameter smoothness and fidelity*.

²⁰LEE discusses in [142, pp. 92–93] such a method originally proposed by MCAULEY and QUATIERI in the context of a sinusoidal representation of speech [164]. He proposes to use continuity constraints at the block boundaries as a *regularization device* to lessen the numerical ill-conditioning problems of the standard least squares approach of GRENIER [82], cf. also Remark 2.

²¹This terminology is due to NIEDŹWIECKI [185] and distinguishes coefficient *tracking* from coefficient *matching* methods. While the first rely exclusively on past and present observations, the latter exploit also future observations to determine the optimal coefficient setting at a given time instant. Of course, the latter methodology is only applicable in off-line applications or, if a sufficient delay in determining the optimally matching coefficients is allowed.

²²The approach developed by BELLEGARDA [15, 16, 17] is much in the spirit of the present approach as it purports to impose a *maximum rate of change* constraint on the time evolution of the coefficients. It is implemented however, as a 'post-filtering' technique which will be described in Subsection 3.4.5.

Smoothness is measured in terms of the p -th difference $\nabla^{(p)}\mathbf{c}(n)$ of the parameters²³, where

$$\nabla^{(1)}\mathbf{c}(n) = \mathbf{c}(n) - \mathbf{c}(n-1) \quad (3.35a)$$

$$\nabla^{(2)}\mathbf{c}(n) = \mathbf{c}(n) - 2\mathbf{c}(n-1) + \mathbf{c}(n-2) \quad (3.35b)$$

$$\vdots \quad \quad \quad \vdots$$

$$\nabla^{(p)}\mathbf{c}(n) = (1 - q^{-i})^p \mathbf{c}(n) = \sum_{i=0}^p \binom{p}{i} (-1)^i \mathbf{c}(n-i) \quad (3.35c)$$

For a given *block of data*, the total smoothness cost is defined as the accumulated sum of squared p -th differences for some fixed order p . Fidelity is measured in terms of the actual error in explaining the observation $d(n)$ with the parameter vector $\mathbf{c}(n)$, i.e. in terms of $d(n) - \mathbf{c}^T(n)\mathbf{x}(n)$. The total fidelity cost function is expressed as the accumulated sum of the squared observation errors. WHITTAKER developed both a probabilistic interpretation of this approach as well as a deterministic method to solve for the unknown parameters over a given block of data. To this end, the optimum parameters are defined as minimizing a ‘joint optimality’ criterion that is a weighted sum of the total smoothness cost and the total fidelity cost. The weighting parameter can be used to emphasize one of the two conflicting constraints so as to achieve either better smoothness or fidelity.

The general idea of smoothness priors was re-invented by NEY in 1982 [177, 178]. He restricted the smoothness cost to be measured in terms of the first difference $\mathbf{c}(n) - \mathbf{c}(n-1)$ only, but considers several variants of possible non-quadratic smoothness and fidelity cost functions. By restricting the *range* of the coefficients $\mathbf{c}(n)$ to a *discrete set* he is able to develop a *recursive solution* to the ‘joint optimality’ problem (minimum weighted sum of smoothness and fidelity cost) by applying a dynamic programming technique. While his technique has the dual appeal of being both recursive and deterministic, it is unsuited to adaptive filtering applications, as even the simplistic assumption of a filter length $N = 10$ and a resolution of the filter coefficients to 8 bits requires to propagate $8^{10} \approx 10^9$ states in the dynamic programming algorithm. Therefore extensive pruning strategies would be required for such an algorithm to allow its actual implementation. Such studies have not been reported so far.

A further interpretation of WHITTAKER’s smoothness constraint has been given by LEE in [142]. He considers the solution to a least-squares problem which originates from unconstrained minimization of the total fidelity cost. This minimization problem is ill-posed as the occurrence of arbitrarily time-varying coefficients $\mathbf{c}(n)$ leads to a highly underdetermined system of equations. The additional smoothness constraint is regarded as a regularization procedure for this ill-posed problem. LEE’s work fails, however, in developing recursive solutions so that it is generally unsuited to adaptive filtering applications.

Only KITIGAWA and GERSCH develop *fully recursive* adaptation algorithms on the basis of WHITTAKER’s idea, cf. [112, 113]²⁴. They achieve this goal by embedding the problem in a *stochastic state-space description* and by using an extended Kalman filter approach for algorithm development. To this end, the following model of the time-varying environment is assumed:

$$\nabla^{(p)}\mathbf{c}_{ref}(n) = \boldsymbol{\zeta}(n) \quad (3.36a)$$

²³Note that WHITTAKER discusses only the scalar case $N = 1$. Here, the method is formulated for the general vector-valued case $N > 1$.

²⁴For a recent account of this work, see also [173].

$$d(n) = \mathbf{c}_{ref}^T(n)\mathbf{x}(n) + \eta(n) \quad (3.36b)$$

where $\zeta(n)$ is a vector-valued noise process and both $\zeta(n)$ and $\eta(n)$ are white and mutually uncorrelated. (3.36a) can be rewritten as

$$\mathbf{c}_{ref}(n) = \sum_{i=1}^p \binom{p}{i} (-1)^{i-1} \mathbf{c}_{ref}(n-i) + \zeta(n) \quad (3.37)$$

or, with the introduction of the pN -dimensional state vector

$$\mathbf{z}(n) = \begin{bmatrix} \mathbf{c}_{ref}(n) \\ \mathbf{c}_{ref}(n-1) \\ \vdots \\ \mathbf{c}_{ref}(n-p+1) \end{bmatrix} \quad (3.38)$$

one gets

$$\mathbf{z}(n) = \mathbf{F}\mathbf{z}(n-1) + \begin{bmatrix} \zeta(n) \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{bmatrix} \quad (3.39a)$$

$$d(n) = [\mathbf{x}^T(n), \mathbf{0}, \dots, \mathbf{0}] \mathbf{z}(n) + \eta(n) \quad (3.39b)$$

where the state transition matrix \mathbf{F} is of order $pN \times pN$ and contains only constants that depend on the difference order p . In the formulation of (3.39a,b), the Kalman filter methodology is directly applicable and results in the following coefficient adaptation algorithm, cf. [113, Eq. (2.7)], [112, Eqs. (25),(26)]:

$$\mathbf{c}(n|n-1) = (1 - (1 - q^{-1})^p) \mathbf{c}(n) = \sum_{i=1}^p \binom{p}{i} (-1)^{i-1} \mathbf{c}(n-i) \quad (3.40a)$$

$$\mathbf{c}(n) = \mathbf{c}(n|n-1) + (d(n) - \mathbf{c}^T(n|n-1)\mathbf{x}(n)) \mathbf{G}(n)\mathbf{x}(n) \quad (3.40b)$$

where $\mathbf{G}(n)$ is a matrix computed from the input data only²⁵. In their paper, KITAGAWA and GERSCH treat the following further topics within a block-oriented, non-recursive approach:

1. optimal choice of filter length N in case the model order is unknown
2. optimal choice of the *ratio* of the variances of the noise process $\zeta(n)$ and $\eta(n)$ controlling the time variation of the reference coefficients and the observation noise power, respectively.

²⁵The estimated variances of the noise processes $\zeta(n)$ and $\eta(n)$ also enter in the computation of the gain matrix.

3. Post-smoothing of the estimates $\mathbf{c}(n)$ over a finite block of data to compensate for the influence of the otherwise arbitrary initial conditions²⁶.

For the present purpose it is, however, sufficient to retain the two main features of the *recursive smoothing priors* algorithm (3.40a,b)

1. It contains a *coefficient vector predictor* (3.40a) which predicts the vector at time n from p previous instances of the same vector. This predictor is designed as a *smooth extrapolation* operator.
2. The coefficient update proper (3.40b) fits neatly into the joint recursive optimality pattern (3.4a,b), if $\mathbf{c}(n-1)$ is replaced by $\mathbf{c}(n|n-1)$.

These two statements will serve as one of the strong incentives to study ‘filtered coefficient algorithms’ from the joint recursive optimality point-of-view. Before entering this discussion, the present one on stochastic state-space models for the time evolution of filter coefficients is concluded with the following remarks:

Remark 1. If the smoothness constraint is formulated with the first-order difference ($p = 1$), then the standard joint recursive optimality structure is obtained from (3.40a,b). Still, there have been attempts to benefit from a state space formulation even in this case: a clever choice of the covariance matrix of the state noise $\zeta(n)$ can be exploited to model e.g. non-uniform variability of the N coefficients as proposed for speech signal modeling in [93, 94]. Another proposal has been reported in [172] where the state transition matrix of a $p = 1$ model is estimated from the data.

Remark 2. Further generalization of this approach is possible if the reference coefficients are not modeled as in (3.36a), but allowed to be stationary stochastic processes themselves which are modeled as the output of a known linear system driven by (white Gaussian) noise. Thus the reference signal $d(n)$ is embedded into a two-tiered model structure where the upper tier (the coefficient process) is justly referred to as a ‘hypermodel’. This line of thought has been followed by [154, 54] and led BENVENISTE to the development of so-called multistep algorithms in [19]. They will reappear in the next section on filtered coefficient algorithms.

3.4 Filtered Coefficient Algorithms

3.4.1 Introduction: In-Loop Filtering

In the last two sections, the following observations were made:

1. The standard adaptation algorithms of (3.4a,b) correspond to linear first-order learning filters.

²⁶It is shown in [112] that this post-smoothing operation drastically improves the results obtained in an off-line time-varying spectrum estimation application. Such smoothing is, however, not applicable to adaptive filtering applications which virtually always are operated on-line, cf. also [173].

2. The time evolution of the (reference) coefficient vector can often be modeled as a deterministic function or stochastic process with partially known properties.

The question arises whether there is a general avenue to match the ‘filtering’ properties of the adaptation algorithm to the coefficient evolution models. The answer is *yes* and will be phrased in terms of *coefficient filters*.

From a *structural point of view*, such coefficient filters can be inserted at various stages of the coefficient adaptation loop. The aim is always to achieve a better separation of the influences of the reference coefficient vector and the noise on the tracking behaviour of the algorithm. The many ways to apply coefficient filters to an adaptation algorithm can be subdivided into two main classes:

1. *In-loop filtering* places the coefficient filter *within* the feedback loop of coefficient adaptation, and will be further analysed in terms of
 - coefficient prediction
 - generalized leakage
 - multi-step algorithms
2. *Post-filtering* places the coefficient filter at the output of the feedback loop of coefficient adaptation and can be viewed as a post-processing technique striving to enhance the recursively computed coefficient estimates $\mathbf{c}(n)$. The description of such techniques is postponed to Subsection 3.4.5.

For now, in-loop filtering is the chosen technique and will be motivated from the study of the signal flow graph for the general joint recursive optimality adaptation algorithm²⁷ as given already in Figure 2.4 and reproduced here for convenience as Figure 3.4.

In this Figure two nodes A and B have been marked which denote the beginning and the end of the *vector-valued sub-graph* within the feedback loop of the adaptation algorithm. It is between these two nodes that any filters operating on the coefficient vector as a whole (and not on a projection) must be inserted. There are three distinct positions in this subgraph where such filters can be placed, cf. Figure 3.5:

1. In the feedforward branch of the accumulator loop: this case is referred to as *coefficient predictor*, see Subsection 3.4.2.
2. In the feedback branch of the accumulator loop: this case is referred to as *generalized leakage*, see Subsection 3.4.3.
3. Outside the accumulator loop: this case is referred to as a *multistep algorithm*. Out of the two possibilities, it is customary to place the filter before the accumulator loop in the signal flow from A to B, see Subsection 3.4.4.

In all three cases, a matrix-valued linear time-varying filter is inserted into the signal flow graph. The filter operators are referred to as $\mathbf{F}_p(n, q^{-1})$, $\mathbf{F}_l(n, q^{-1})$, and $\mathbf{F}_m(n, q^{-1})$, respectively. After the initial transients have faded away, the overall algorithm behaviour

²⁷The discussion is again restricted to adaptation algorithms *without* error filtering, although a generalization to algorithms with error filtering is possible and will be commented upon in Chapter 4.

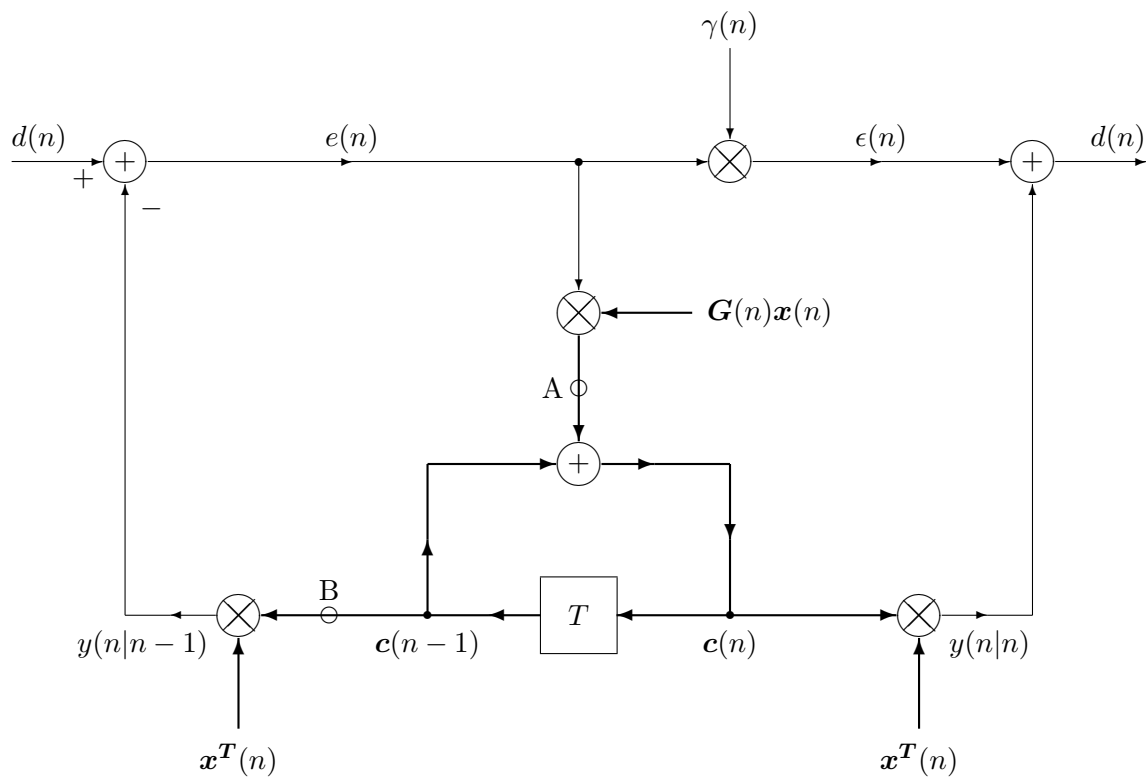


Figure 3.4: Signal flow graph of general joint recursive optimality algorithm without error filtering.

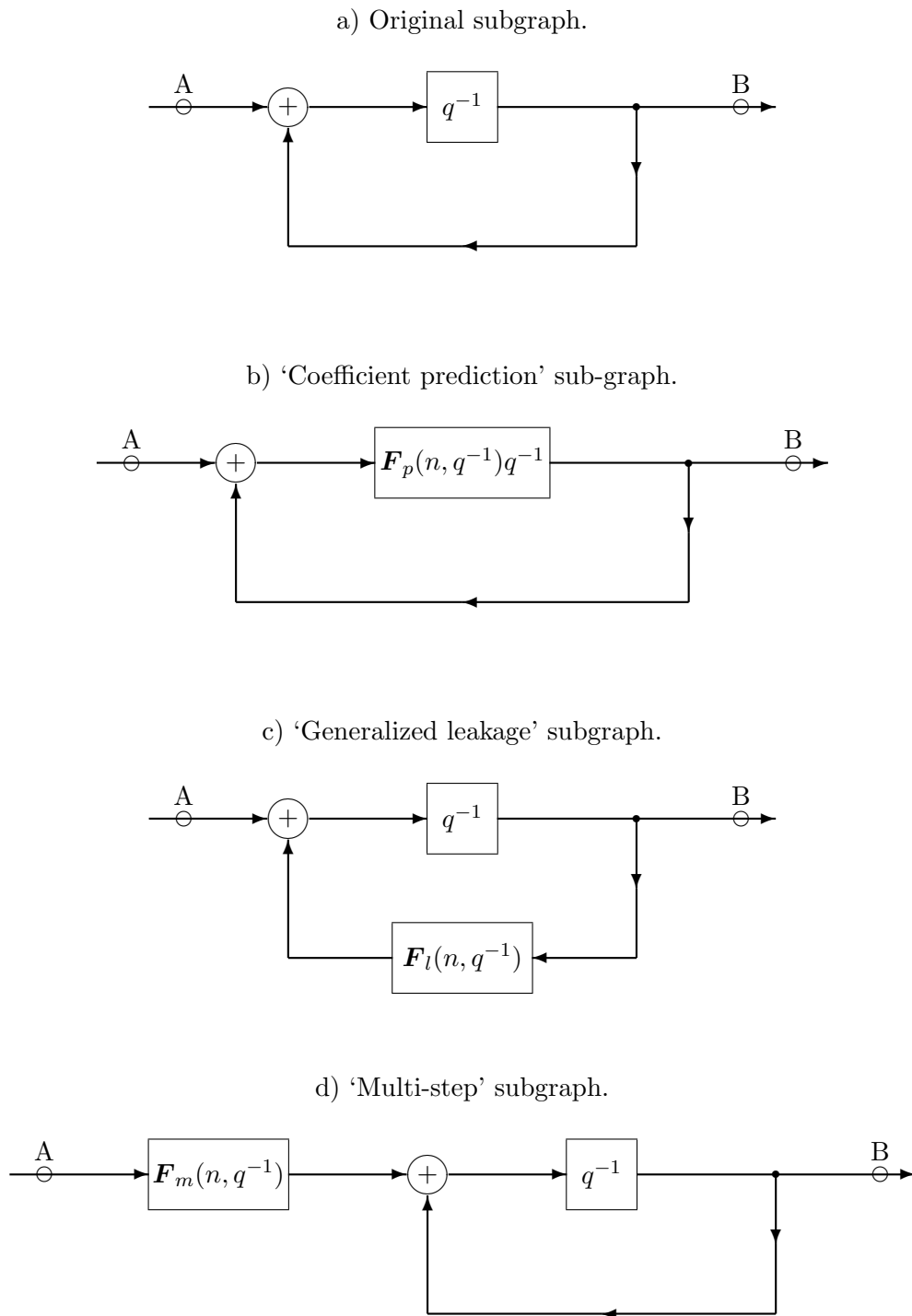


Figure 3.5: Vector-valued sub-graph of general adaptation algorithm and three possible modifications through coefficient filters.

Table 3.2: EQUIVALENCE RELATIONSHIPS BETWEEN THE 3 TYPES OF COEFFICIENT FILTERS.

express	in terms of
$\mathbf{F}_p(n, q^{-1})$	$[1 + q^{-1}(1 - \mathbf{F}_l(n, q^{-1}))]^{-1}$
$\mathbf{F}_p(n, q^{-1})$	$(1 - q^{-1})^{-1} \mathbf{F}_m(n - 1, q^{-1}) \left[1 + \frac{q^{-1}}{1 - q^{-1}} \mathbf{F}_m(n - 1, q^{-1})\right]^{-1}$
$\mathbf{F}_l(n, q^{-1})$	$(\mathbf{F}_p(n, q^{-1})q^{-1})^{-1} [\mathbf{F}_p(n, q^{-1})(1 + q^{-1}) - 1]$
$\mathbf{F}_l(n, q^{-1})$	$(\mathbf{F}_m(n - 1, q^{-1})q^{-1})^{-1} [\mathbf{F}_m(n - 1, q^{-1}) - 1 + q^{-1}]$
$\mathbf{F}_m(n, q^{-1})$	$(1 - q^{-1}) (1 - \mathbf{F}_p(n + 1, q^{-1})q^{-1})^{-1} \mathbf{F}_p(n + 1, q^{-1})$
$\mathbf{F}_m(n, q^{-1})$	$(1 - q^{-1}) (1 - \mathbf{F}_l(n, q^{-1})q^{-1})^{-1}$

will be the same no matter which of the three coefficient filter types is realized *if the total filter operator* $\mathbf{F}_{BA}(n, q^{-1})$ *describing the linear transfer characteristics from node A to node B is the same.* From the following identities

$$\mathbf{F}_{BA}(n, q^{-1}) = (1 - \mathbf{F}_p(n, q^{-1})q^{-1})^{-1} \mathbf{F}_p(n, q^{-1})q^{-1} \quad (3.41a)$$

$$= (1 - q^{-1} \mathbf{F}_l(n, q^{-1}))^{-1} q^{-1} \quad (3.41b)$$

$$= \frac{q^{-1}}{1 - q^{-1}} \mathbf{F}_m(n, q^{-1}) \quad (3.41c)$$

an equivalence Table 3.2 can be compiled that allows to pass from one filter representation to another.

Remarkably, if any of the filters is chosen causal, also its two equivalent counterparts are automatically causal. A similar general statement with respect to stability of the three filter types has not been found. None of the examples in the following subsections encounters, however, stability problems. The discussion is started with coefficient prediction filters as they bear the strongest resemblance to the joint recursive optimality framework.

3.4.2 Coefficient Prediction

The coefficient prediction filter has been introduced in Figure 3.5 as simple replacement of the delay operator q^{-1} by the more complex filtering operator $\mathbf{F}_p(n, q^{-1})q^{-1}$. The output of the coefficient predictor is directly used to compute the a priori filter output (and error) in Figure 3.4. Therefore, it is appropriate to introduce the notation

$$\mathbf{c}(n|n - 1) = \mathbf{F}_p(n, q^{-1})q^{-1}\mathbf{c}(n) = \mathbf{F}_p(n, q^{-1})\mathbf{c}(n - 1) \quad (3.42)$$

where $\mathbf{c}(n|n - 1)$ is the coefficient vector at time n as predicted from the previous vectors $\mathbf{c}(n - i)$, $i \geq 1$ (if $\mathbf{F}_p(n, q^{-1})$ is assumed causal). Employing that notation, the adaptation algorithm with inserted prediction filter reads now:

$$\mathbf{c}(n) = \mathbf{c}(n|n-1) + e(n)\mathbf{G}(n)\mathbf{x}(n) \quad (3.43a)$$

$$\text{with } e(n) = d(n) - \mathbf{c}^T(n|n-1)\mathbf{x}(n) \quad (3.43b)$$

It is not surprising that this modified algorithm *satisfies*²⁸ *again a joint recursive optimality principle*, i.e.

$$[\mathbf{c}(n) - \mathbf{c}(n|n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c}(n) - \mathbf{c}(n|n-1)] + \gamma^{-1}(n) \epsilon^2(n) \stackrel{!}{=} \min \quad (3.44)$$

where the weighting parameters are again chosen subject to positivity constraints and normalized such that

$$\gamma(n) + \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) = 1 \quad (3.45)$$

Furthermore, one obtains again

$$\epsilon(n) = \gamma(n)e(n) \quad (3.46)$$

as in the case of the algorithm without coefficient filtering. Note that the modification in the optimality criterion (3.44) is non-trivial when compared to the no-filter case (2.40): in place of the previous coefficient vector $\mathbf{c}(n-1)$, the prediction $\mathbf{c}(n|n-1)$ is used now which may depend in an arbitrarily complex way²⁹ on all previous values $\mathbf{c}(n-i)$, $i \geq 1$. While the first term in the criterion (2.40) has been interpreted as a measure of the time variation of the filter coefficients, the first term in the criterion (3.44) is rather interpreted as a measure of the *fidelity of the coefficient predictor*. The remainder of this discussion is again organized in a series of remarks.

Remark 1. The two equations (3.43a) and (3.46) compute the ‘a posteriori’ quantities $\mathbf{c}(n)$ and $\epsilon(n)$ in terms of the ‘a priori’ quantities $\mathbf{c}(n|n-1)$ and $e(n)$. Therefore, a more precise notation suggests itself by writing $\mathbf{c}(n|n)$ in place of $\mathbf{c}(n)$. The simple notation $\mathbf{c}(n)$ is, however, retained for convenience as it is unequivocal as well.

Remark 2. From the interpretation of the optimality criterion (3.44), it is desirable to use the best possible predictor $\mathbf{F}_p(n, q^{-1})$. In all algorithms considered in Chapter 2 and Section 3.2, this predictor has been set equal to the identity operator, i.e.

$$\mathbf{c}(n|n-1) = \mathbf{c}(n-1) \quad (3.47a)$$

$$\mathbf{F}_p(n, q^{-1}) = \mathbf{I} \quad (3.47b)$$

²⁸For a proof, see Appendix B.

²⁹Actually, it is not necessary at all to assume that $\mathbf{c}(n|n-1)$ is obtained by linear filtering of the coefficient vectors $\mathbf{c}(n-i)$. Any kind of linear or non-linear predictor may be used to obtain $\mathbf{c}(n|n-1)$ in terms of the available observation history, including any side information available from additional observations etc. It is only necessary, that the value of $\mathbf{c}(n|n-1)$ is computable without taking the observations at time n into account.

In what situation is such a predictor optimal? From the conventional error minimization paradigm of algorithm derivation, one might suspect that this is the optimal predictor for a stationary environment, i.e. for a time evolution model of the filter coefficients that is constant for all times n . *This is not the case* as for such a model the optimal predictor would average over the entire observed history of $\mathbf{c}(n-i)$, $1 \leq i \leq n$, to obtain a better prediction of the constant coefficient vector than by simply perpetuating the latest estimate $\mathbf{c}(n-1)$ which is known to fluctuate around the optimum even after convergence has occurred in a stationary environment.

The description of the situation where the predictor (3.47b) is optimal is found in the probabilistic concept of a *martingale* process. Simply put, a martingale is a stochastic process of which the conditional expectation at time n , given its complete history of observations, is equal to its latest observed value³⁰. *If the reference coefficients obey a martingale law the predictor (3.47a,b) is optimal in a mean-square sense*, [191, pp.412–414]. An example of such behaviour is found in a pure random walk of the filter coefficients (which is somewhat unrealistic due to the divergence of its variance with time). From a more practical point of view, the martingale assumption reflects a *wait-and-see strategy*, which might be an optimal common sense strategy if *no* a priori information (besides a vague smoothness assumption) is available about the time evolution of the filter coefficients. This fact explains to a certain extent why these simple algorithms are the first choice for most adaptive system designers as reliable a priori information on the time evolution of filter coefficients is often scarce in practice.

Remark 3. If a state space model is used for the reference signal $d(n)$ where the unobserved state vector equals the reference coefficient vector,

$$\mathbf{c}_{ref}(n) = \mathbf{F}(n, n-1)\mathbf{c}_{ref}(n-1) + \boldsymbol{\zeta}(n) \quad (3.48a)$$

$$d(n) = \mathbf{x}^T(n)\mathbf{c}_{ref}(n) + \eta(n) \quad (3.48b)$$

then the optimal coefficient predictor equals the state transition matrix³¹:

$$\mathbf{c}(n|n-1) = \mathbf{F}(n, n-1)\mathbf{c}(n-1) \quad (3.49a)$$

$$\mathbf{F}_p(n, q^{-1}) = \mathbf{F}(n, n-1) \quad (3.49b)$$

Thus, the coefficient predictor does not modify the learning filter order of the adaptation algorithm! It only allows to incorporate known deterministical dynamics of the coefficient vector by specification of the transition matrix $\mathbf{F}(n, n-1)$ and, if the Kalman filter methodology is invoked to compute the gain matrix $\mathbf{G}(n)$, of the covariance properties of the random components $\boldsymbol{\zeta}(n)$ and $\eta(n)$.

Remark 4. Stepping up in complexity, the smoothness priors concept is included here again as an instance of coefficient prediction:

$$\mathbf{c}(n|n-1) = \left[1 - \left(1 - q^{-1}\right)^p\right] \mathbf{c}(n) \quad (3.50a)$$

³⁰Cf. e.g. [57, Chapter VII] but also the concise introductions of this concept in [31, pp. 799–802], [81, pp. 499–503], [150, p. 409, pp. 453–463], and [191, p. 477]. The concept originates from gambling theory where a game is called fair if the expected capital of a player after a game equals his capital just before placing his bet.

³¹The choice reflects implicitly a kind of ‘certainty equivalence’ principle.

$$\begin{aligned}
\mathbf{F}_p(n, q^{-1}) &= q \left[1 - (1 - q^{-1})^p \right] \\
&= \sum_{i=1}^p \binom{p}{i} (-1)^{i-1} q^{-i+1}
\end{aligned} \tag{3.50b}$$

The simplest non-trivial case is $p = 2$ where

$$\mathbf{F}_p(n, q^{-1}) = 2 - q^{-1} \tag{3.51}$$

and, with (3.42), the algorithm (3.43a,b) results in³²

$$\mathbf{c}(n) = 2\mathbf{c}(n-1) - \mathbf{c}(n-2) + e(n)\mathbf{G}(n)\mathbf{x}(n) \tag{3.52a}$$

$$e(n) = d(n) - [2\mathbf{c}(n-1) - \mathbf{c}(n-2)]^T \mathbf{x}(n) \tag{3.52b}$$

and satisfies the optimality criterion (3.44) in the following form:

$$[\mathbf{c}(n) - 2\mathbf{c}(n-1) + \mathbf{c}(n-2)]^T \mathbf{G}^{-1}(n) [\mathbf{c}(n) - \mathbf{c}(n-1) + \mathbf{c}(n-2)] + \gamma^{-1}(n) \epsilon^2(n) \stackrel{!}{=} \min. \tag{3.53}$$

Remark 5. The use of coefficient prediction filters to extend standard adaptation algorithms on the basis of a priori knowledge about the nature of the time-varying environment was proposed by the present author in [119]. Independent work [46, 166] considered the practical application of this concept to adaptive equalization of fading HF channels. The incorporation of simple predictive models for the fading process provides considerable equalizer performance improvement both when considering the simple LMS algorithm or an RLS-like Kalman algorithm as long as the signal-to-noise ratio at the filter input is higher than 30 dB [46]. For a lower signal-to-noise ratio, the quality of the ‘a posteriori estimates’ $\mathbf{c}(n)$ does not suffice to yield useful predictions $\mathbf{c}(n|n-1)$ even if the predictor model matches perfectly the ‘true’ reference system [166]. This shows the potential failure of the ‘certainty equivalence’ strategy as will be elaborated in the next remark.

Remark 6 The effect of the coefficient predictor on the tracking properties of the adaptation algorithm can be assessed from considering the modified asymptotic equation (3.21) governing the evolution of the misalignment vector $\tilde{\boldsymbol{\theta}}(n)$:

$$\begin{aligned}
\tilde{\boldsymbol{\theta}}(n) &= [\gamma \mathbf{P}(n) + \mathbf{P}^\perp(n)] \left[\mathbf{F}_p(n, q^{-1}) \tilde{\boldsymbol{\theta}}(n-1) - (\tilde{\mathbf{c}}_{ref}(n) - \mathbf{F}_p(n, q^{-1}) \tilde{\mathbf{c}}_{ref}(n-1)) \right] \\
&+ (1 - \gamma) \tilde{\boldsymbol{\eta}}(n)
\end{aligned} \tag{3.54}$$

If $\mathbf{F}_p(n, q^{-1})$ is the best possible predictor for $\mathbf{c}_{ref}(n)$ in terms of $\mathbf{c}_{ref}(n-1)$, then the driving term $\tilde{\mathbf{c}}_{ref}(n) - \mathbf{F}_p(n, q^{-1}) \tilde{\mathbf{c}}_{ref}(n-1)$ causing the misalignment due to lag is minimal. This ‘certainty equivalence’ choice of $\mathbf{F}_p(n, q^{-1})$ is however not the general optimum as the operator $\mathbf{F}_p(n, q^{-1})$ enters equation (3.54) a second time in front of $\tilde{\boldsymbol{\theta}}(n-1)$. Therefore, the choice of $\mathbf{F}_p(n, q^{-1})$ has to consider also the overall noise suppression properties to achieve an optimum with respect to total misalignment or excess error. This optimum may require a coefficient predictor which differs from the certainty equivalence choice.

³²This result is later compared to the so-called *momentum LMS* algorithm, see Subsection 3.4.3.

3.4.3 Generalized Leakage

The second option of placing a coefficient filter into the general adaptation algorithm leads to the following structure:

$$\begin{aligned} \mathbf{c}(n) &= \mathbf{F}_l(n, q^{-1})\mathbf{c}(n-1) + e(n)\mathbf{G}(n)\mathbf{x}(n) & (3.55a) \\ \text{with } e(n) &= d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) & (3.55b) \end{aligned}$$

From Table 3.2 one obtains the equivalent prediction filter as

$$\mathbf{F}_p(n, q^{-1}) = \left(1 + q^{-1}(1 - \mathbf{F}_l(n, q^{-1}))\right)^{-1} \quad (3.56)$$

and, therefore, the algorithm (3.55a,b) satisfies the joint recursive optimality principle (3.44) with

$$\mathbf{c}(n|n-1) = \left(1 + q^{-1}(1 - \mathbf{F}_l(n, q^{-1}))\right)^{-1} \mathbf{c}(n-1) \quad (3.57)$$

Remark 1. The simplest non-trivial instance of (3.55a,b) is

$$\mathbf{F}_l(n, q^{-1}) = \lambda \mathbf{I} \quad \text{with } 0 \ll \lambda < 1 \quad (3.58)$$

which results in the *leakage algorithm* [78]

$$\mathbf{c}(n) = \lambda \mathbf{c}(n-1) + e(n)\mathbf{G}(n)\mathbf{x}(n) \quad (3.59a)$$

$$e(n) = d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) \quad (3.59b)$$

where the gain matrix is usually adopted from the LMS algorithm as $\mathbf{G}(n) = \mu \mathbf{I}$. From (3.58) and (3.57), this algorithm meets the deterministic optimality criterion

$$\left[1 - \mu \|\mathbf{x}(n)\|^2\right] \left\| \mathbf{c}(n) - \frac{1}{1 + (1 - \lambda)q^{-1}} \mathbf{c}(n-1) \right\|^2 + \mu \epsilon^2(n) \stackrel{!}{=} \min \quad (3.60)$$

In comparison with the optimality criterion (2.50) of the conventional LMS algorithm, the leaky LMS criterion (3.60) evaluates the time variation of the coefficients from a filtered first difference of the coefficients whereas in (2.50) the filter is absent (to see this immediately, let $\lambda \rightarrow 1$ in (3.60)). The use of the leaky LMS algorithm in place of the conventional one is mostly motivated from its improved long-term stability for non-persistently exciting input data [78], [96, pp. 56–59], [103, pp. 160–163]. BELLANGER has shown in [14, pp. 112–114] that the introduction of leakage can also improve the *algorithm performance in a nonstationary* prediction application. This matches nicely the above observation w.r.t. the modified coefficient variation measure in the optimality criterion.

Remark 2. Another common choice in the literature is

$$\mathbf{F}_l(n, q^{-1}) = 1 + \alpha(1 - q^{-1}) \quad \text{with} \quad 0 \leq \alpha \leq 1 \quad (3.61)$$

which results in the *momentum LMS algorithm* [210, 227] (where $\mathbf{G}(n) = \mu\mathbf{I}$):

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mu e(n)\mathbf{x}(n) + \alpha(\mathbf{c}(n-1) - \mathbf{c}(n-2)) \quad (3.62a)$$

$$e(n) = d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) \quad (3.62b)$$

This algorithm purports to speed up convergence towards a time-invariant optimum (for the same final excess error) because it accelerates the coefficient vector movement through inclusion of the ‘momentum’ term $\alpha(\mathbf{c}(n-1) - \mathbf{c}(n-2))$. The optimality criterion (3.44) reads with (3.61) and (3.57)

$$\left[1 - \mu\|\mathbf{x}(n)\|\right]^2 \left\| \mathbf{c}(n) - \frac{1}{1 - \alpha q^{-1} + \alpha q^{-2}} \mathbf{c}(n-1) \right\|^2 + \mu \epsilon^2(n) \stackrel{!}{=} \min \quad (3.63)$$

The algorithm is rather popular in the emerging field of neural networks where the inclusion of a momentum term is standard in the so-called *backpropagation algorithm* [201]. The convergence and tracking analysis as presented in [210] reveals that the inclusion of the momentum term results in a *second-order learning filter*. Such filter may have complex poles which manifest themselves as oscillatory behaviour of the coefficient trajectories, especially if α gets close to 1 [200]. Another interesting comparison arises for $\alpha \rightarrow 1$ where the algorithm (3.62a,b) reads as

$$\mathbf{c}(n) = 2\mathbf{c}(n-1) - \mathbf{c}(n-2) + \mu e(n)\mathbf{x}(n) \quad (3.64a)$$

$$e(n) = d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) \quad (3.64b)$$

which is identical to the second-order smoothness priors algorithm (3.52a,b) apart from the error signal computation: here the a priori filter output is computed with the ‘old’ coefficients $\mathbf{c}(n-1)$ while in (3.52b) the a priori filter output is computed with the predicted coefficient vector $\mathbf{c}(n|n-1) = 2\mathbf{c}(n-1) - \mathbf{c}(n-2)$.

3.4.4 Multi-Step Algorithms

Multi-step algorithms constitute the third option of placing the coefficient filter into the general adaptation algorithm. From Figure 3.4 with Figure 3.5 one has³³

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \mathbf{F}_m(n, q^{-1}) (e(n)\mathbf{G}(n)\mathbf{x}(n)) \quad (3.65a)$$

$$e(n) = d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) \quad (3.65b)$$

³³The notation might be simplified by collapsing $\mathbf{F}_m(n, q^{-1})$ and $\mathbf{G}(n)$ into a single matrix-valued operator. This is avoided here for the sake of clarity when comparing to the other filtered coefficient algorithms.

and from Table 3.2

$$\mathbf{F}_p(n, q^{-1}) = (1 - q^{-1})^{-1} \mathbf{F}_m(n-1, q^{-1}) \left[1 + \frac{q^{-1}}{1 - q^{-1}} \mathbf{F}_m(n-1, q^{-1}) \right]^{-1} \quad (3.66)$$

and, therefore, *all multi-step algorithms satisfy the deterministic optimality criterion* (3.44) with

$$\mathbf{c}(n|n-1) = (1 - q^{-1})^{-1} \mathbf{F}_m(n-1, q^{-1}) \left[1 + \frac{q^{-1}}{1 - q^{-1}} \mathbf{F}_m(n-1, q^{-1}) \right]^{-1} \mathbf{c}(n-1) \quad (3.67)$$

Remark 1. The idea of using multi-step procedures in stochastic approximation algorithms for time-invariant environments goes back to TSYPKIN's work around 1970 [20, p. 163]. The aim was to accelerate convergence towards a (time-invariant) optimum by using a search direction which is the average of several successive (negative) gradient estimates. These ideas were further developed e.g. in [117, 202] but were confined to decreasing-gain algorithms.

Remark 2. An early paper [55] studies a *two-step* variant of LMS with non-decreasing gain by setting

$$\mathbf{F}_m(n, q^{-1}) = (\alpha + \beta q^{-1}) \mathbf{I} \quad (3.68)$$

The authors show an advantage of this algorithm w.r.t. convergence speed in a noisy time-invariant environment but give no indication of its tracking capabilities.

Remark 3. BELLANGER studies in [14, pp. 121–123] the properties of the *delayed LMS algorithm* with

$$\mathbf{F}_m(n, q^{-1}) = q^{-L} \mathbf{I} \quad (3.69)$$

where the delay L is chosen to ease implementation (e.g. through pipelining). For the special case of $L = 1$ and $\mathbf{G}(n) = \mu \mathbf{I}$, the optimality criterion (3.44) reads with (3.67) and (3.69) as

$$[1 - \mu \|\mathbf{x}(n)\|]^2 \left\| \mathbf{c}(n) - \frac{q^{-1}}{1 - q^{-1} + q^{-2}} \mathbf{c}(n-1) \right\|^2 + \mu \epsilon^2(n) \stackrel{!}{=} \min \quad (3.70)$$

which is—apart from the extra delay q^{-1} —the same as for the *momentum LMS algorithm* with $\alpha \rightarrow 1$, cf. (3.63). From the analysis in [14], performance is close to the conventional LMS algorithm and can be expected to be worse than LMS in time-varying environments.

Remark 4. Only BENVENISTE recognizes the general applicability of the algorithm structure (3.65a,b) to the tracking problem in time-varying environments. In [19], [20, pp. 140–161], a complete theory is developed how fairly general time evolution models for the reference coefficient vector can be cast into an (extended) state-space formalism and translated into the corresponding choice of an optimal multi-step filter $\mathbf{F}_m(n, q^{-1})$. The treatment includes the special case of periodically time-variant environments which

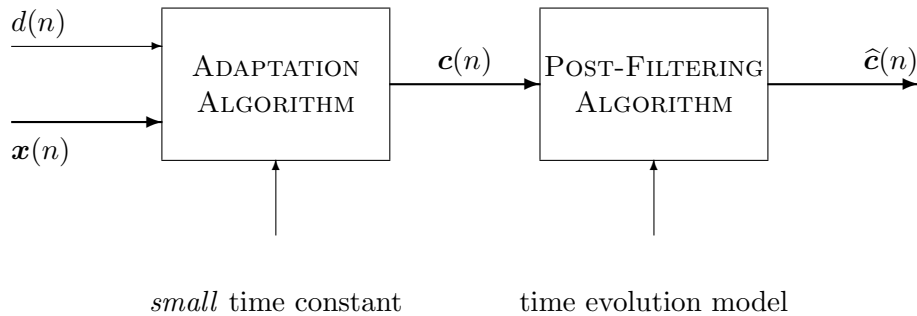


Figure 3.6: General approach to post-filtering of coefficient estimates.

has served as a starting point of the present discussion in Section 3.3. The analysis by BENVENISTE corroborates the view that only if reliable prior knowledge on the time evolution of the coefficients can be condensed in a ‘hypermodel’, the application of multi-step algorithms (or filtered coefficient algorithms in general) is justified. Otherwise—if such knowledge is unavailable—one-step procedures (without coefficient filtering) are the preferred choice.

3.4.5 Post-Filtering Algorithms

So far, three methods have been outlined how to incorporate (limited) prior knowledge in terms of coefficient filters into the feedback loop of the adaptation algorithm. For practical reasons it is sometimes desired to separate the coefficient adaptation process from the coefficient filtering process³⁴. A general approach to this problem consists in a two-stage procedure:

- In the first stage, an adaptation algorithm is chosen to compute estimates $\mathbf{c}(n)$ of the coefficient vector without any detailed modeling assumption. The only requirement is that the effective cut-off frequency of the associated learning filter be high enough to ensure that no information about the time-varying coefficients is lost. Of course, this entails an undue increase of the excess error due to noise, in general.
- In the second stage, a post-processing technique is chosen to improve on the quality of the estimates $\mathbf{c}(n)$. Only in this stage, a priori model assumptions on the time evolution of the reference coefficients are introduced to do away with some of the excess error due to noise as produced in the first-stage algorithm. The result is an enhanced coefficient vector estimate $\hat{\mathbf{c}}(n)$.

This general approach is summarized in the block diagram Figure 3.6.

If the application is not in system identification but in joint process estimation, an enhanced output signal $\hat{y}(n) = \hat{\mathbf{c}}^T(n)\mathbf{x}(n)$ has to be computed from an extra filtering operation. The interpretation of this general approach may be guided from considering the first-stage adaptation algorithm as a simple non-specific *front end that converts the noisy scalar information $d(n)$ about the reference coefficients $\mathbf{c}_{ref}(n)$ into a noisy vector-valued*

³⁴Examples will be given below after delineating the general post-filtering procedure.

information $c(n)$. The second-stage post-filtering algorithm builds on a vector-oriented model of the reference coefficients to remove the noise from its vector-valued input and produces the final estimates $\hat{c}(n)$.

Remark 1. The advantage of this approach lies mainly in its flexibility: the choice of the adaptation algorithm may concentrate on numerical stability and efficiency issues while leaving the actual coefficient modeling problem to the second stage. On the other hand, the post-filtering can be arbitrarily complex and there is no need to organize this second-stage processing in a recursive manner³⁵. It is even possible to run *several post-filtering algorithms in parallel*, using a single adaptation algorithm as a common front end: in that case either a decision device or a linear combiner reduces the parallel outputs of the post-filtering algorithms to a single optimum. This *multiple-model* approach is most beneficial in a situation where the parameters of the time-evolution model (such as the speed of coefficient variation) are not known precisely and the co-existence of several models improves the overall robustness of the adaptive filter, cf. the studies in [7, 166, 187, 184, 185].

Remark 2. Post-filtering techniques are rather popular in speech signal modeling where the front-end adaptation algorithm is usually a block processing linear predictive analysis filter operated under a short-time stationarity assumption. All time-varying detail of the coefficient evolution that is faster than the limit imposed through the block (or ‘frame’) length is a priori lost in this first stage. As an instance of second-stage processing a functional series expansion approach has been proposed in [2, 11, 156]. This technique is known as *temporal decomposition* and includes the signal-dependent optimization of the set of base functions which is used for series expansion. This demonstrates clearly the higher flexibility of a post-filtering technique in comparison to the related in-loop filtering technique, in that case the functional series expansion approach of the deterministic paradigm in 3.3.1. On the other hand, the method is implemented entirely off-line and is—not only from complexity considerations—refractory to a recursive implementation.

Remark 3. A further example from speech processing is *(hidden) Markov modeling* of speech parameters over time. The standard approach stages this type of time-evolution model as a post-filtering technique (cf. [197] for a tutorial on its use in speech recognition applications) after a conventional block processing adaptation algorithm. In a non-standard approach, PORITZ has investigated these models as an in-loop filtering technique by inserting them directly into the first-stage adaptation algorithm³⁶ [194, 195].

Remark 4. A theoretical foundation of post-filtering techniques in the context of recursive adaptation algorithms is given by NIEDŹWIECKI: In [180, 181, 182] he shows that the effect of the weighting function in least squares algorithms can be represented as the filtering of the reference coefficient vector with a (learning) filter whose impulse response

³⁵Of course, also the first-stage adaptation algorithm may be organized in a block-processing mode of operation!

³⁶A similar technique for ‘non-hidden’ Markov modeling of coefficient evolution has been described in [170].

equals the weighting function. In [185] he takes up BELLEGARDA's approach to coefficient post-filtering [15, 17] and argues that the design of a specific weighting function (or in-loop coefficient filter) for least-squares algorithms can be quite cumbersome if it is desired to retain the possibility of developing a recursive algorithm for this weighting scheme. Therefore, a standard, e.g. exponentially weighted least squares algorithm with sufficiently small forgetting factor should be used as the front-end algorithm whereas the specific, model-based coefficient filtering is executed as a second-stage operation. Furthermore, he is able to show that the *overall learning filter* of the two-stage system has an impulse response *which is the convolution of the impulse responses of each of the two stages*. If the first stage impulse response is short enough, the second one can be tailored easily to match the time-evolution model of the application. This decomposition of adaptation and filtering allows even to consider coefficient filter designs with prescribed phase or group-delay properties etc. These issues are still an open field for further research.

3.5 Comparison and Criticism

Starting from a general tracking analysis of the standard joint recursive optimality pattern, the need for coefficient filtering was recognized from the fact that the standard algorithms all behave as first-order linear learning filters. After a short review of available deterministic and stochastic techniques for modeling of the time evolution of the filter coefficients, the previous section has come up with a general tool for inclusion of such model-based information into an adaptation algorithm: coefficient filters are applied either within the feedback loop of the algorithm or as a post-processing technique to the first-stage adaptation process. Both approaches are well represented in the recent literature on algorithm design for tracking purposes and from a general point-of-view a decision upon the best choice seems to be difficult at present. A lot has been said in favour of the post-filtering approach in the preceding section, especially w.r.t. its general flexibility and to its efficiency in a multiple-model based description. Is there anything to win with in-loop coefficient filtering, though? Of course, there is. The post-filtering approach is tied to the assumption that no relevant information is lost in the first-stage algorithm, i.e. that the related time constant of the algorithm can be chosen small enough. There are, however, two fundamental lower limits to this choice:

1. There is a minimum learning time constant of any linear first-order adaptation algorithm: none of the standard algorithms considered in Section 3.2 can react faster than a zero-forcing algorithm which still has a time constant equal to the filter length N (cf. Eq. (3.24) with $\gamma = 0$). If the filter length N is high, as occurs e.g. in acoustic echo cancellation where N can easily be on the order of 10^3 , the maximum possible cut-off frequency of a standard adaptation algorithm may be too low to follow the time-variations of the environment in the first processing stage.
2. When approaching their minimum learning time constant, adaptation algorithms tend to get numerically unstable, in particular if they are of the least squares type.

Both problems may receive at least partial cure from in-loop coefficient filtering: if the coefficient predictor accounts e.g. for the rapid periodic variation of a time-evolution model, tracking may become possible even if the adaptation *relative* to the rapid periodic

variation³⁷ is of course also limited by a time constant greater or equal to the filter length. As to numerical problems, [142] has shown how the incorporation of in-loop coefficient filtering induces additional constraints that regularize an otherwise ill-posed problem.

Summarizing this argument, it is recommended not to strive for an early decision whether in-loop filtering or post-loop filtering is the preferable choice for a given application but to devote to both of them fair consideration.

As a final comment to coefficient filtering, the three in-loop techniques are compared: coefficient prediction, generalized leakage, and multi-step algorithms. In Section 3.4.1, the general equivalence of the three schemes in steady state has been shown from a simple signal flow graph equivalence. Now, while all three techniques have their archetypical representatives in actual applications, there are two reasons to favour the *coefficient prediction approach*:

1. It is this approach which allows to extend the joint recursive optimality criterion in the simplest possible way, i.e. by replacing the ‘old’ coefficient vector $\mathbf{c}(n-1)$ by a prediction $\mathbf{c}(n|n-1)$ computed directly with the coefficient predictor $\mathbf{F}_p(n, q^{-1})$. Of course, also the other two techniques can be cast into the joint recursive optimality framework, but then the required modifications are less obvious a priori (i.e. the respective coefficient filters $\mathbf{F}_l(n, q^{-1})$ and $\mathbf{F}_m(n, q^{-1})$ do not enter the criterion in a simple way that would be predicted without further inspection, cf. Eqs. (3.57) and (3.67)).
2. From a practical point-of-view, it is not at all desirable to restrict the time evolution models to be linear in the previous coefficients $\mathbf{c}(n-i), i \geq 1$. As a matter of fact, given any prediction $\mathbf{c}(n|n-1)$, the joint recursive optimality criterion (3.44) produces the algorithm (3.43a,b), no matter how $\mathbf{c}(n|n-1)$ is computed. The interpretation of (3.44) is then in terms of trading the coefficient prediction error for the filtering error power $\epsilon^2(n)$. That is why *any high-level coefficient evolution model can be incorporated into the lowest-level adaptation algorithm* if the model is expressed in a predictive form [119]. In principle, even knowledge-based methods as suggested in [149] might be used for this modeling purpose. A more profane use of non-linear coefficient predictors is found in adaptation algorithms with *coefficient projection into a prescribed region* (e.g. to ensure stability in case of certain adaptive IIR filters or generally to avoid overflow of the number representation) as discussed by [65, 64], [81, pp. 91–94], [103, pp. 54–56], [209], [251, pp. 178–182]. In that case, the predicted coefficient vector $\mathbf{c}(n|n-1)$ is just the projection of the previous vector $\mathbf{c}(n-1)$ into the allowed region. A simple description of these non-linear extensions of the coefficient filtering concept seems not to be possible with the generalized leakage or the multi-step algorithm approach. In conclusion, *coefficient prediction is suggested as the simplest and most generally applicable tool for customizing an adaptation algorithm to a hypermodel of the coefficient behaviour.*

³⁷An interpretation of this effect may be obtained by comparison with a stroboscope where a sampling technique is used to shift a rapidly time-varying process to a more convenient time scale. In the present example, the coefficient predictor may be seen as a (narrow) band-pass filter and tracking speed requirements are only defined w.r.t. the equivalent low-pass representation of the time evolution model.

Chapter 4

Conclusion

- Results are collected together and put into perspective to stimulate further research.

This conclusion completes a thesis which is organized in two main parts: the first one in Chapter 2 treats the theoretical basis for the optimal design of recursive algorithms for the adaptation of transversal filters while the second part in Chapter 3 treats the tracking behaviour of these algorithms in time-varying environments and suggests extensions of the algorithms to incorporate predictive coefficient evolution models. As both parts have received their individual concluding comments already, there is not much left over to say here. Therefore, this concluding chapter is confined to a summary of the main results of the theory of joint recursive optimality, and to some indications on future directions for research.

4.1 Summary of Main Results

The main result is that there is only *one* main result: virtually all algorithms discussed in this thesis can be obtained from a single deterministic optimality principle. The unifying aspect is the strong point of this general theory. A concise statement of this theory is provided in Table 4.1 and Figure 4.1.

Note that the algorithm overview in this table looks very similar to the previously given Table 2.3, yet there is one important difference: only the present table unifies the dual aspects of error prediction and coefficient prediction in a common framework. Within this framework, three parameters

- the coefficient predictor $\mathbf{F}(n, q^{-1})$
- the gain matrix $\mathbf{G}(n)$ and
- the error predictor $H(n, q^{-1})$

completely specify an algorithm.

Table 4.1: JOINT RECURSIVE OPTIMALITY WITH BOTH ERROR PREDICTION AND COEFFICIENT PREDICTION.

1. Define a coefficient predictor $\mathbf{c}(n|n-1)$ where two common suggestions are

$$\begin{aligned} \mathbf{c}(n|n-1) &= \mathbf{c}(n-1) \\ \text{or } \mathbf{c}(n|n-1) &= \mathbf{F}_p(n, q^{-1})\mathbf{c}(n-1). \end{aligned}$$

2. Define an error predictor $\epsilon(n|n-1)$ where two common suggestions are

$$\begin{aligned} \epsilon(n|n-1) &= 0 \\ \text{or } \epsilon(n|n-1) &= H(n, q^{-1})\epsilon(n-1). \end{aligned}$$

3. Compute the innovation

$$i(n) = d(n) - \mathbf{c}^T(n|n-1)\mathbf{x}(n) - \epsilon(n|n-1).$$

4. Choose a positive definite coefficient weighting matrix $\mathbf{G}(n)$ and a positive error weight $\gamma(n)$ with a normalization such that

$$\gamma(n) + \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) = 1.$$

5. Define the cost function

$$\begin{aligned} J(n, \mathbf{c}) &= [\mathbf{c} - \mathbf{c}(n|n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c} - \mathbf{c}(n|n-1)] \\ &\quad + \gamma^{-1}(n)[d(n) - \mathbf{c}^T \mathbf{x}(n) - \epsilon(n|n-1)]^2 \end{aligned}$$

and the optimization problem

$$\mathbf{c}(n) \stackrel{!}{=} \arg \min_{\mathbf{c} \in R^N} J(n, \mathbf{c}).$$

6. The unique solution is found as

$$\begin{aligned} \mathbf{c}(n) &= \mathbf{c}(n|n-1) + i(n)\mathbf{G}(n)\mathbf{x}(n) \\ \epsilon(n) &= \epsilon(n|n-1) + \gamma(n)i(n). \end{aligned}$$

7. This solution achieves a cost function minimum of

$$J(n, \mathbf{c})|_{\mathbf{c}=\mathbf{c}(n)} = i^2(n)$$

which is referred to as the *innovation sharing* property.

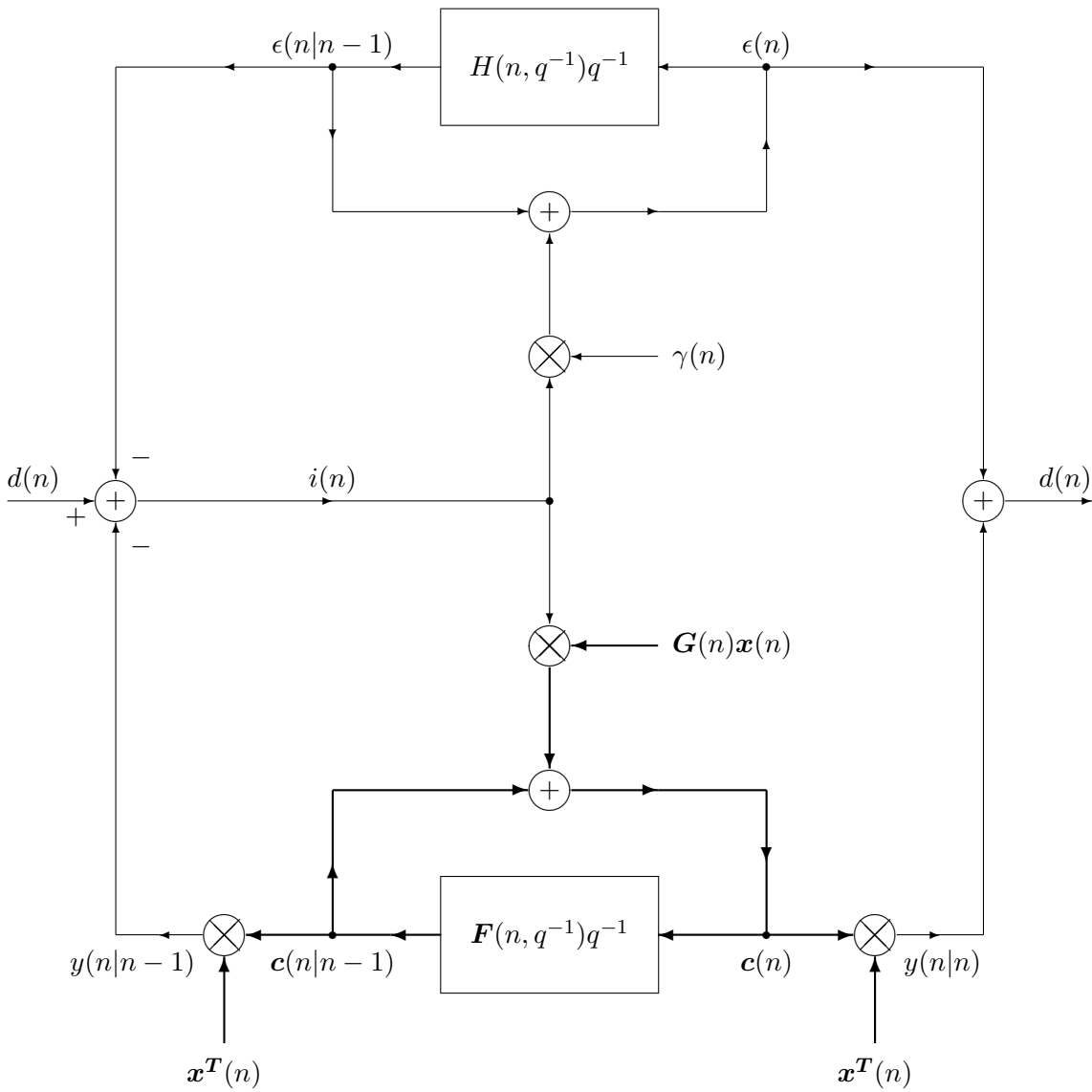


Figure 4.1: Signal flow graph of general joint recursive optimality algorithm with both coefficient and error prediction (a priori and a posteriori quantities displayed).

The specific choice of

$$\mathbf{c}(n|n-1) = \mathbf{c}(n-1) \Leftrightarrow \mathbf{F}(n, q^{-1}) = \mathbf{I} \quad (4.1a)$$

$$\epsilon(n|n-1) = 0 \Leftrightarrow H(n, q^{-1}) = 0 \quad (4.1b)$$

leads to a wide class of standard adaptation algorithms which includes as prominent members the LMS- and RLS-type algorithms. This choice is adequate if the coefficient evolution can be modeled as a martingale process and if the error can be modeled as a martingale difference process [31, p. 801]. Both assumptions represent the *common sense choice for the case of totally lacking prior knowledge*. This standard class of algorithms has been analysed in terms of tracking properties where the two asymptotic results of Section 3.2, Eq. (3.23) and (3.24):

$$M_{noise} = \frac{1-\gamma}{1+\gamma} \quad (4.2a)$$

$$\tau = \frac{N}{1-\gamma^2} \quad (4.2b)$$

allow a simple assessment of the quantitative tracking performance of such algorithms¹. Of course, the formulae (4.2a,b) appear as crude rule-of-thumbs in comparison with more sophisticated results on specific algorithms as found in the literature. However, they do convey the fundamental performance limitations of any algorithm falling into this class and motivate the application of filtered coefficient algorithms whenever some prior knowledge is available. Coefficient filtering is attractive both when applied in-loop or as a post-filtering operation. If it is applied in-loop the coefficient prediction formulation is the most general available.

4.2 Directions for Further Research

When developing a unifying framework for a large class of otherwise scattered items, the question arises whether this is the most general theory one can think of. In the present case of joint recursive optimality the answer is *no*. The potential towards further generalization originates from the present restriction to the transversal filter structure. As mentioned already at the end of Chapter 2, relaxation of this restriction leads to the extension of adaptation algorithms towards their general applicability in arbitrary filter structures. One possible recipe for this extension is overly simple when explained with the help of the signal flow graph Figure 4.1:

1. Replace the ‘a priori’ inner product computation in the lower left of the graph by the appropriate non-transversal filter structure to compute $y(n|n-1)$ in terms of $\mathbf{x}(n)$ and $\mathbf{c}(n|n-1)$ and do the same replacement at the a posteriori side of the graph.
2. Replace the gain vector $\mathbf{G}(n)\mathbf{x}(n)$ by a *filtered regressor* $\mathbf{G}(n, q^{-1})\mathbf{x}(n)$ which equals the sensitivity of the filter output $y(n)$ with respect to the coefficients $\mathbf{c}(n)$.

¹The two performance indices M_{noise} and τ bear the following interpretation: M_{noise} is the ‘misadjustment’ due to the excess error induced through the observation noise in the reference signal, τ is the time constant associated with the first-order learning filter description of the total excess error induced both through noise and coefficient lag.

As this recipe originates from simplifying approximations, further investigation is required to assess its merits. This need may serve as a directive for upcoming research.

A further point left out of consideration so far is the study of algorithm implementation under finite-precision arithmetic conditions. Although this is an issue that usually deserves an algorithm specific treatment, general problems in this field have already been touched upon in this thesis (long-term drift due to lack of persistent excitation). Further research might mark off general numerical properties related to the overall algorithm structure Figure 4.1 from more specific problems such as the stable propagation of an inverse correlation matrix estimator in RLS-type algorithms. The latter aspect has received some attention by the present author in [121, 122]. This work is not included here and will be the topic of a separate publication.

A final comment applies to the practical operation of adaptive signal processing systems in time-varying environments: so far, only little work is reported that could show an actual improvement obtained in a real-world application from the employment of time-evolution models and/or filtered coefficient algorithms. Practical needs are however rather urgent (as in the equalization of rapidly varying HF channels for digital communication or in the cancellation of acoustical echos in the rapidly varying environment of a handsfree telephone) and should bring about the willingness to adopt these still rather new concepts and to develop them further into neat engineering solutions.

Appendix A

Recursive Least Squares Algorithms

A.1 Derivation of a Recursive Solution for the Filter Coefficients

A procedure is presented to find a recursive solution for the direct-solution coefficient vector optimum (2.19)

$$\mathbf{c}(n) = \hat{\mathbf{R}}^{-1}(n)\hat{\mathbf{p}}(n) \quad (\text{A.1})$$

where

$$\hat{\mathbf{p}}(n) = \lambda_1(n)\hat{\mathbf{p}}(n-1) + \lambda_2(n)d(n)\mathbf{x}(n) \quad (\text{A.2a})$$

$$\hat{\mathbf{R}}(n) = \lambda_1(n)\hat{\mathbf{R}}(n-1) + \lambda_2(n)\mathbf{x}(n)\mathbf{x}^T(n) \quad (\text{A.2b})$$

are estimates of the cross-correlation vector \mathbf{p} and the auto-correlation matrix \mathbf{R} , respectively, and $\lambda_1(n)$, $\lambda_2(n)$ are the weighting factors of a general linear first-order memory of the estimators, cf. Subsection 2.3.4.3.

As a first step, from (A.1)

$$\lambda_1(n)\hat{\mathbf{R}}(n-1)\mathbf{c}(n-1) = \lambda_1(n)\hat{\mathbf{p}}(n-1) \quad (\text{A.3})$$

and inserting the recursions (A.2a,b)

$$[\hat{\mathbf{R}}(n) - \lambda_2(n)\mathbf{x}(n)\mathbf{x}^T(n)]\mathbf{c}(n-1) = \hat{\mathbf{p}}(n) - \lambda_2(n)d(n)\mathbf{x}(n) \quad (\text{A.4})$$

Premultiplying (A.4) with $\hat{\mathbf{R}}^{-1}(n)$ where the positive definiteness of the matrix $\hat{\mathbf{R}}(n)$ is assumed (see Section A.3 below), yields after rearranging:

$$\underbrace{\hat{\mathbf{R}}^{-1}(n)\hat{\mathbf{p}}(n)}_{\mathbf{c}(n)} = \mathbf{c}(n-1) + \lambda_2(n) \left(d(n) - \mathbf{x}^T(n)\mathbf{c}(n-1) \right) \hat{\mathbf{R}}^{-1}(n)\mathbf{x}(n) \quad (\text{A.5})$$

or, with the definition of the a priori error

$$e(n) = d(n) - \mathbf{c}^T(n-1)\mathbf{x}(n) \quad (\text{A.6})$$

the desired coefficient vector recursion is obtained

$$\mathbf{c}(n) = \mathbf{c}(n-1) + \lambda_2(n)e(n)\widehat{\mathbf{R}}^{-1}(n)\mathbf{x}(n) \quad (\text{A.7})$$

Note that the introduction of the a priori error occurs to the development without making any reference to the associated adaptive filter!

A.2 The Matrix Inversion Lemma

As a second step to develop the weighted recursive least squares algorithm from the defining equations (A.1) and (A.2a,b), a recursion is developed to propagate directly the inverse matrix estimate $\widehat{\mathbf{R}}^{-1}(n)$ without explicit inversion. To this end, the *matrix inversion lemma* is invoked that states:

Lemma A.1. Let \mathbf{x} and \mathbf{z} be two N -dimensional vectors with $\mathbf{x}^T\mathbf{z} \neq -1$. Then

$$\left[\mathbf{I} + \mathbf{z}\mathbf{z}^T\right]^{-1} = \mathbf{I} - \frac{\mathbf{z}\mathbf{x}^T}{1 + \mathbf{x}^T\mathbf{z}} \quad (\text{A.8})$$

whereby the inversion of an $N \times N$ matrix is reduced to a single scalar division.

Proof:

$$\begin{aligned} \text{Let } \left[\mathbf{I} + \mathbf{z}\mathbf{z}^T\right]^{-1} &= \mathbf{A} \\ \text{Then } \mathbf{A}\left[\mathbf{I} + \mathbf{z}\mathbf{z}^T\right] &= \mathbf{I} \\ \mathbf{A} + (\mathbf{A}\mathbf{z})\mathbf{z}^T &= \mathbf{I} \quad | \cdot \mathbf{z} \end{aligned} \quad (\text{A.9})$$

$$\begin{aligned} (\mathbf{A}\mathbf{z}) + (\mathbf{A}\mathbf{z})\mathbf{z}^T\mathbf{z} &= \mathbf{z} \quad | \div (1 + \mathbf{x}^T\mathbf{z}) \neq 0 \\ \mathbf{A}\mathbf{z} &= \frac{\mathbf{z}}{1 + \mathbf{x}^T\mathbf{z}} \end{aligned} \quad (\text{A.10})$$

Insertion of (A.10) into (A.9) completes the proof:

$$\begin{aligned} \mathbf{A} + \frac{\mathbf{z}}{1 + \mathbf{x}^T\mathbf{z}}\mathbf{z}^T &= \mathbf{I} \\ \mathbf{A} &= \mathbf{I} - \frac{\mathbf{z}\mathbf{z}^T}{1 + \mathbf{x}^T\mathbf{z}} \quad \text{Q.E.D.} \end{aligned}$$

Now, (A.2b) can easily be manipulated in a form such that (A.8) becomes applicable:

$$\begin{aligned} \widehat{\mathbf{R}}(n) &= \lambda_1(n)\widehat{\mathbf{R}}(n-1) \left[\mathbf{I} + \frac{\lambda_2(n)}{\lambda_1(n)}\widehat{\mathbf{R}}^{-1}(n-1)\mathbf{x}(n)\mathbf{x}^T(n) \right] \\ \widehat{\mathbf{R}}^{-1}(n) &= \left[\mathbf{I} + \underbrace{\frac{\lambda_2(n)}{\lambda_1(n)}\widehat{\mathbf{R}}^{-1}(n-1)\mathbf{x}(n)\mathbf{x}^T(n)}_{\mathbf{z}} \right]^{-1} \frac{1}{\lambda_1(n)}\widehat{\mathbf{R}}^{-1}(n-1) \end{aligned} \quad (\text{A.11})$$

where both the positivity of $\lambda_1(n)$ and $\widehat{\mathbf{R}}(n-1)$ have been assumed. Then, from (A.11) with (A.8)

$$\widehat{\mathbf{R}}^{-1}(n) = \left[\mathbf{I} - \frac{\frac{\lambda_2(n)}{\lambda_1(n)} \widehat{\mathbf{R}}^{-1}(n-1) \mathbf{x}(n) \mathbf{x}^T(n)}{1 + \mathbf{x}^T(n) \frac{\lambda_2(n)}{\lambda_1(n)} \widehat{\mathbf{R}}^{-1}(n-1) \mathbf{x}(n)} \right] \frac{1}{\lambda_1(n)} \widehat{\mathbf{R}}^{-1}(n-1)$$

$$\widehat{\mathbf{R}}^{-1}(n) = \frac{1}{\lambda_1(n)} \left[\widehat{\mathbf{R}}^{-1}(n-1) - \frac{\lambda_2(n) \widehat{\mathbf{R}}^{-1}(n-1) \mathbf{x}(n) \mathbf{x}^T(n) \widehat{\mathbf{R}}^{-1}(n-1)}{\lambda_1(n) + \lambda_2(n) \mathbf{x}^T(n) \widehat{\mathbf{R}}^{-1}(n-1) \mathbf{x}(n)} \right] \quad (\text{A.12})$$

which is the desired result, cf. the general update formula (2.85) and the special case with $\lambda_1(n) = \lambda_2(n) = 1$ in (2.20c).

A.3 Positive Definiteness of the Recursive Auto-Correlation Matrix Estimate

The following theorem is proved for recursive auto-correlation matrix estimates with first-order linear data weighting.

Theorem A.1. Let

$$\mathbf{R}(n) = \lambda_1(n) \mathbf{R}(n-1) + \lambda_2(n) \mathbf{x}(n) \mathbf{x}^T(n)$$

where $\mathbf{x}(n)$ is an arbitrary sequence of real N -dimensional vectors and $\lambda_1(n)$, $\lambda_2(n)$ are real-valued weighting factors. Then the set of *necessary and sufficient conditions* for $\mathbf{R}(n)$ to be positive definite for all $n \geq 0$ and any sequence $\mathbf{x}(n)$ is given by

$$\mathbf{R}(0) > 0 \quad (\text{A.13a})$$

$$\lambda_1(n) > 0 \quad \text{for all } n > 0 \quad (\text{A.13b})$$

$$\lambda_2(n) > -\frac{\lambda_1(n)}{\mathbf{x}^T(n) \mathbf{R}^{-1}(n-1) \mathbf{x}(n)} \quad \text{whenever } \|\mathbf{x}(n)\| > 0 \quad (\text{A.13c})$$

Proof: The proof is organized in two parts:

1. First it is shown that conditions (A.13a-c) are *sufficient* for positive definiteness of $\mathbf{R}(n)$. To this end, the quadratic form $\mathbf{v}^T \mathbf{R}(n) \mathbf{v}$ is shown to be positive for *any* choice of the vector \mathbf{v} .

- If $\mathbf{v}^T \mathbf{x}(n) \neq 0$ one has

$$\mathbf{v}^T \mathbf{R}(n) \mathbf{v} = \lambda_1(n) \mathbf{v}^T \mathbf{R}(n-1) \mathbf{v} + \lambda_2(n) [\mathbf{v}^T \mathbf{x}(n)]^2$$

$$\begin{aligned}
&\geq \lambda_1(n) \frac{[\mathbf{v}^T \mathbf{x}(n)]^2}{\mathbf{x}^T(n) \mathbf{R}^{-1}(n-1) \mathbf{x}(n)} + \lambda_2(n) [\mathbf{v}^T \mathbf{x}(n)]^2 \\
&= \underbrace{[\mathbf{v}^T \mathbf{x}(n)]^2}_{> 0} \underbrace{\left(\frac{\lambda_1(n)}{\mathbf{x}^T(n) \mathbf{R}^{-1}(n-1) \mathbf{x}(n)} + \lambda_2(n) \right)}_{> 0 \text{ from (A.13c)}} \\
&> 0
\end{aligned} \tag{A.14}$$

where the *Schwarz inequality*

$$\|\mathbf{a}\| \|\mathbf{b}\| \geq |\mathbf{a}^T \mathbf{b}| \tag{A.15}$$

has been used with

$$\mathbf{a} = \mathbf{R}^{T/2}(n-1) \mathbf{v} \tag{A.16a}$$

$$\mathbf{b} = \mathbf{R}^{-1/2}(n-1) \mathbf{x}(n) \tag{A.16b}$$

(A.16b) requires $\mathbf{R}(n-1)$ to have non-zero eigenvalues. Therefore, the proof is by induction, starting from (A.13a) with a positive definite $\mathbf{R}(0)$ and showing from (A.14) that for a positive definite $\mathbf{R}(n-1)$, $\mathbf{R}(n)$ can be guaranteed to be positive as well.

- If $\mathbf{v}^T \mathbf{x}(n) = 0$ one has

$$\mathbf{v}^T \mathbf{R}(n) \mathbf{v} = \lambda_1(n) \mathbf{v}^T \mathbf{R}(n-1) \mathbf{v} \tag{A.17}$$

which allows again with (A.13b) a proof by induction.

2. Next is shown that conditions (A.13a-c) are also *necessary* to guarantee a positive definite $\mathbf{R}(n)$ no matter what sequence $\mathbf{x}(n)$ is applied:

- The necessity of (A.13a): $\mathbf{R}(0) > 0$ is trivial to guarantee $\mathbf{R}(n) > 0$ for all $n \geq 0$.
- The necessity of (A.13b) is shown by an indirect argument. Assume $\lambda_1(n) \leq 0$ for some $n > 0$. Then

$$\mathbf{v}^T \mathbf{R}(n) \mathbf{v} = \lambda_1(n) \mathbf{v}^T \mathbf{R}(n-1) \mathbf{v} \leq 0 \tag{A.18}$$

if $\mathbf{v}^T \mathbf{x}(n) = 0$ and $\mathbf{R}(n-1) > 0$. Thus, if the matrix estimate has been positive definite up to time index $n-1$ the matrix $\mathbf{R}(n)$ will have at least one non-positive eigenvalue. Therefore (A.13b) is also necessary to hold for all $n > 0$.

- The necessity of (A.13c) is also shown by an indirect argument. Assume

$$\lambda_2(n) \leq -\frac{\lambda_1(n)}{\mathbf{x}^T(n) \mathbf{R}^{-1}(n-1) \mathbf{x}(n)}$$

for some $n > 0$. Then

$$\begin{aligned}
\mathbf{x}^T(n) \mathbf{R}(n) \mathbf{x}(n) &= \lambda_1(n) \mathbf{x}^T(n) \mathbf{R}(n-1) \mathbf{x}(n) + \lambda_2(n) (\mathbf{x}^T(n) \mathbf{x}(n))^2 \\
&\leq \lambda_1(n) \left[\mathbf{x}^T(n) \mathbf{R}(n-1) \mathbf{x}(n) - \frac{(\mathbf{x}^T(n) \mathbf{x}(n))^2}{\mathbf{x}^T(n) \mathbf{R}^{-1}(n-1) \mathbf{x}(n)} \right]
\end{aligned} \tag{A.19}$$

Now, the positive definiteness of $\mathbf{R}(n)$ should be guaranteed for any sequence $\mathbf{x}(n)$. A purposeful yet allowed choice of $\mathbf{x}(n)$ is parallel to an eigenvector of $\mathbf{R}(n-1)$. In this case

$$(\mathbf{x}^T(n)\mathbf{R}(n-1)\mathbf{x}(n)) \cdot (\mathbf{x}^T(n)\mathbf{R}^{-1}(n-1)\mathbf{x}(n)) = (\mathbf{x}^T(n)\mathbf{x}(n))^2 \quad (\text{A.20})$$

so that from (A.19)

$$\mathbf{x}^T(n)\mathbf{R}(n)\mathbf{x}(n) \leq 0 \quad (\text{A.21})$$

which shows that $\mathbf{R}(n)$ must have at least one non-positive eigenvalue. This completes the second part of the proof.

Appendix B

Joint Recursive Optimality

B.1 Solution of Optimization Problem— The Quadratic Error Case

In this section, the solution to the optimization problem defining the general joint recursive optimality coefficient adaptation algorithm is given. The criterion is specified in its most general form, i.e. including an error predictor $\epsilon(n|n-1)$ and a coefficient predictor $\mathbf{c}(n|n-1)$ as in Table 4.1. The simplified version (2.40), (2.118), and (3.44) can be obtained from setting

$$\mathbf{c}(n|n-1) = \mathbf{c}(n-1) \quad (\text{B.1a})$$

$$\text{and/or} \quad \epsilon(n|n-1) = 0 \quad (\text{B.1b})$$

The cost function to start with is defined as

$$J(n, \mathbf{c}) = [\mathbf{c} - \mathbf{c}(n|n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c} - \mathbf{c}(n|n-1)] + \gamma^{-1}(n) [d(n) - \mathbf{c}^T \mathbf{x}(n) - \epsilon(n|n-1)]^2 \quad (\text{B.2})$$

where $\mathbf{G}(n)$ is a *symmetric positive definite matrix*, $\gamma(n)$ a positive scalar and \mathbf{c} is the independent variable over which the minimization

$$\mathbf{c}(n) = \arg \min_{\mathbf{c} \in R^N} J(n, \mathbf{c}) \quad (\text{B.3})$$

is carried out. A *necessary condition* for the solution (B.3) is that the gradient of the cost function vanishes:

$$\nabla_{\mathbf{c}} J(n, \mathbf{c})|_{\mathbf{c}=\mathbf{c}(n)} \stackrel{!}{=} \mathbf{0} \quad (\text{B.4})$$

$$2\mathbf{G}^{-1}(n) [\mathbf{c}(n) - \mathbf{c}(n|n-1)] + \gamma^{-1}(n) 2[d(n) - \mathbf{c}^T(n) \mathbf{x}(n) - \epsilon(n|n-1)] (-\mathbf{x}(n)) = \mathbf{0}$$

$$[\mathbf{c}(n) - \mathbf{c}(n|n-1)] = \gamma^{-1}(n) \mathbf{G}(n) \mathbf{x}(n) [d(n) - \mathbf{c}^T(n) \mathbf{x}(n) - \epsilon(n|n-1)] \quad (\text{B.5})$$

From (B.5), $\mathbf{c}(n) - \mathbf{c}(n|n-1)$ must be collinear to $\mathbf{G}(n) \mathbf{x}(n)$ and therefore it can be written as

$$\mathbf{c}(n) - \mathbf{c}(n|n-1) = \alpha(n)\mathbf{G}(n)\mathbf{x}(n) \quad (\text{B.6})$$

Furthermore, with the definition of the a posteriori error

$$\epsilon(n) = d(n) - \mathbf{c}^T(n)\mathbf{x}(n) \quad (\text{B.7})$$

one has from (B.5) with (B.6)

$$\alpha(n)\mathbf{G}(n)\mathbf{x}(n) = \gamma^{-1}(n)\mathbf{G}(n)\mathbf{x}(n)[\epsilon(n) - \epsilon(n|n-1)]$$

$$\epsilon(n) = \epsilon(n|n-1) + \gamma(n)\alpha(n) \quad (\text{B.8})$$

Insertion of (B.6) into (B.5) allows to determine the scale factor $\alpha(n)$:

$$\begin{aligned} & \alpha(n)\mathbf{G}(n)\mathbf{x}(n) \\ &= \gamma^{-1}(n)\mathbf{G}(n)\mathbf{x}(n)[d(n) - (\mathbf{c}(n|n-1) + \alpha(n)\mathbf{G}(n)\mathbf{x}(n))^T\mathbf{x}(n) - \epsilon(n|n-1)] \\ & \alpha(n)\gamma(n) \\ &= d(n) - \mathbf{c}^T(n|n-1)\mathbf{x}(n) - \epsilon(n|n-1) - \alpha(n)\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) \end{aligned}$$

or, with the definition of the innovation signal

$$i(n) = d(n) - \mathbf{c}^T(n|n-1)\mathbf{x}(n) - \epsilon(n|n-1) \quad (\text{B.9})$$

the result

$$\alpha(n) = \frac{i(n)}{\gamma(n) + \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)} \quad (\text{B.10})$$

is obtained. (B.10) yields with (B.6) and (B.8) the general adaptation algorithm

$$\mathbf{c}(n) = \mathbf{c}(n|n-1) + \frac{i(n)}{\gamma(n) + \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)}\mathbf{G}(n)\mathbf{x}(n) \quad (\text{B.11a})$$

$$\epsilon(n) = \epsilon(n|n-1) + \frac{\gamma(n)}{\gamma(n) + \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)}i(n) \quad (\text{B.11b})$$

(B.11a,b) is derived from a necessary condition for a minimum of the cost function (B.3). A *sufficient condition* is found from considering the second derivative with respect to the coefficient vector \mathbf{c} , i.e. the matrix

$$\nabla_{\mathbf{c}}^2 J(n, \mathbf{c}) \Big|_{\mathbf{c}=\mathbf{c}(n)} \stackrel{!}{>} 0 \quad (\text{B.12})$$

must be positive definite. From (B.12) with (B.2):

$$2\mathbf{G}^{-1}(n) + \gamma^{-1}(n)2\mathbf{x}(n)\mathbf{x}^T(n) > 0$$

which is satisfied for arbitrary $\mathbf{x}(n)$ iff

$$\mathbf{G}(n) > 0 \quad (\text{B.13a})$$

$$\gamma(n) > 0 \quad (\text{B.13b})$$

If conditions (B.13a,b) are prescribed for the choice of the coefficient weighting matrix $\mathbf{G}(n)$ and the error weight $\gamma(n)$, the algorithm (B.11a,b) is guaranteed to achieve the unique minimum of the cost function $J(n, \mathbf{c})$.

A final remark to the general solution (B.11a,b) and the choice of the weighting parameters $\mathbf{G}(n)$, $\gamma(n)$ is appropriate. To this end, the value of the cost function $J(n, \mathbf{c})$ that is assumed for the minimum solution $\mathbf{c} = \mathbf{c}(n)$ is computed

$$\begin{aligned} J(n, \mathbf{c})|_{\mathbf{c}=\mathbf{c}(n)} &= [\mathbf{c}(n) - \mathbf{c}(n|n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c}(n) - \mathbf{c}(n|n-1)] \\ &\quad + \gamma^{-1}(n) [d(n) - \mathbf{c}^T(n) \mathbf{x}(n) - \epsilon(n|n-1)]^2 \\ &= \frac{i(n)}{\gamma(n) + \mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n)} \mathbf{x}^T(n) \mathbf{G}^T(n) \mathbf{G}^{-1}(n) \mathbf{G}(n) \mathbf{x}(n) \frac{i(n)}{\gamma(n) + \mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n)} \\ &\quad + \gamma^{-1}(n) [\epsilon(n) - \epsilon(n|n-1)]^2 \\ &= \frac{i^2(n) \mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n)}{(\gamma(n) + \mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n))^2} + \gamma^{-1}(n) \frac{\gamma^2(n)}{(\gamma(n) + \mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n))^2} i^2(n) \\ &= \frac{i^2(n)}{(\gamma(n) + \mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n))} (\mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n) + \gamma(n)) \end{aligned}$$

$$J(n, \mathbf{c})|_{\mathbf{c}=\mathbf{c}(n)} = \frac{i^2(n)}{\gamma(n) + \mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n)} \quad (\text{B.14})$$

It is seen from (B.14) that the cost function minimum is always proportional to the squared innovation signal $i^2(n)$. Just as in the adaptation algorithm (B.11a,b), the proportionality factor is again $[\gamma(n) + \mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n)]^{-1}$. Now, as the definition of an optimum coefficient vector $\mathbf{c}(n)$ in (B.3) is not changed by the inclusion of a (positive) scale factor in front of the cost function $J(n, \mathbf{c})$, it is suggested to *normalize* $J(n, \mathbf{c})$ such that its minimum value equals precisely $i^2(n)$. To this end, the weighting parameters $\mathbf{G}(n)$ and $\gamma(n)$ must be chosen subject to

$$\gamma(n) + \mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n) = 1 \quad (\text{B.15})$$

This choice has three advantages:

1. It resolves the ambiguity in the weighting parameters $\gamma(n)$ and $\mathbf{G}(n)$ (which would fake an extra degree of freedom in their common scale factor!).
2. It assigns a meaningful value to the cost function minimum

$$J(n, \mathbf{c})|_{\mathbf{c}=\mathbf{c}(n)} = i^2(n) \quad (\text{B.16})$$

3. It simplifies the general adaptation algorithm structure (B.11a,b) to

$$\mathbf{c}(n) = \mathbf{c}(n|n-1) + i(n)\mathbf{G}(n)\mathbf{x}(n) \quad (\text{B.17a})$$

$$\epsilon(n) = \epsilon(n|n-1) + \gamma(n)i(n) \quad (\text{B.17b})$$

B.2 Solution of Optimization Problem— The Error Magnitude Case

In (2.113) of Section 2.3.5, the original joint recursive optimality criterion is replaced with a criterion in terms of the a posteriori error magnitude $|\epsilon(n)|$:

$$J(n, \mathbf{c}) = [\mathbf{c} - \mathbf{c}(n|n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c} - \mathbf{c}(n|n-1)] + 2|d(n) - \mathbf{c}^T \mathbf{x}(n)| \quad (\text{B.18})$$

where $\mathbf{G}(n)$ is a symmetric positive definite matrix and more often than not, $\mathbf{c}(n|n-1) = \mathbf{c}(n-1)$. The solution to the optimization problem

$$\mathbf{c}(n) = \arg \min_{\mathbf{c} \in R^N} J(n, \mathbf{c}) \quad (\text{B.19})$$

is developed here. A necessary condition for the optimum is again

$$\nabla_{\mathbf{c}} J(n, \mathbf{c})|_{\mathbf{c}=\mathbf{c}(n)} \stackrel{!}{=} \mathbf{0} \quad (\text{B.20})$$

$$2\mathbf{G}^{-1}(n)[\mathbf{c}(n) - \mathbf{c}(n|n-1)] + 2\text{sgn}(d(n) - \mathbf{c}^T(n)\mathbf{x}(n))(-\mathbf{x}(n)) = \mathbf{0}$$

$$\mathbf{c}(n) = \mathbf{c}(n|n-1) + \mathbf{G}(n)\mathbf{x}(n)\text{sgn}(d(n) - \mathbf{c}^T(n)\mathbf{x}(n)) \quad (\text{B.21})$$

This is a nonlinear equation in $\mathbf{c}(n)$ which can be solved analytically, yet. From (B.21)

$$\begin{aligned} d(n) - \mathbf{c}^T(n)\mathbf{x}(n) \\ = d(n) - \mathbf{c}^T(n|n-1)\mathbf{x}(n) - \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)\text{sgn}(d(n) - \mathbf{c}^T(n)\mathbf{x}(n)) \end{aligned}$$

or, more compactly,

$$\Leftrightarrow \epsilon(n) + \text{sgn}[\epsilon(n)]\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) = e(n) \quad (\text{B.22})$$

$$\text{sgn}[\epsilon(n)](|\epsilon(n)| + \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)) = \text{sgn}[e(n)]|e(n)| \quad (\text{B.23})$$

which can be split into two equations

$$\text{sgn}[\epsilon(n)] = \text{sgn}[e(n)] \quad (\text{B.24a})$$

$$|\epsilon(n)| = |e(n)| - \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) \quad (\text{B.24b})$$

Only if the last two equations (B.24a,b) can be met simultaneously, the optimality equation (B.20) has a solution. From (B.24b) with $|\epsilon(n)| \geq 0$, it is concluded that

$$|e(n)| \geq \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) \quad (\text{B.25})$$

is a precondition for such a solution. In that case (which will be denoted as algorithm branch 'A'), (B.24a) can be inserted in (B.21) to yield

$$\mathbf{c}(n) = \mathbf{c}(n|n-1) + \text{sgn}[e(n)]\mathbf{G}(n)\mathbf{x}(n) \quad (\text{B.26})$$

and from (B.22) with (B.24a):

$$\epsilon(n) = e(n) - \text{sgn}[e(n)]\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n) \quad (\text{B.27})$$

It is easy to convince oneself that this solution actually achieves a cost function minimum as

$$\nabla_{\mathbf{c}}^2 J(n, \mathbf{c}) \Big|_{\mathbf{c}=\mathbf{c}(n)} = 2\mathbf{G}^{-1}(n) > 0 \quad (\text{B.28})$$

If $|e(n)| < \mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)$, algorithm branch 'B' is entered. In this case, the gradient of the cost function vanishes nowhere in the coefficient vector space so that a minimum of the cost function can only be attained at 'boundaries', in particular at the discontinuity of the derivative of the cost function which occurs for all $\mathbf{c}(n)$ with $\epsilon(n) = 0$. A zero-forcing algorithm is conjectured as a candidate solution in this case

$$\mathbf{c}(n) = \mathbf{c}(n|n-1) + e(n) \frac{\mathbf{G}(n)\mathbf{x}(n)}{\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)} \quad (\text{B.29a})$$

$$\epsilon(n) = 0 \quad (\text{B.29b})$$

and for a proof of its actual optimality it is shown that any vector $\mathbf{c}'(n) \neq \mathbf{c}(n)$ that lies also in the hyperplane $\epsilon(n) = 0$ yields a higher cost function value. First, decompose

$$\mathbf{c}'(n) = \mathbf{c}(n) + \mathbf{z}(n) \quad (\text{B.30a})$$

$$\text{where } \mathbf{z}^T(n)\mathbf{x}(n) = 0 \quad (\text{B.30b})$$

which follows from the constraint $\epsilon(n) = 0$ for both $\mathbf{c}(n)$ and $\mathbf{c}'(n)$. Next, (B.30a) is inserted into (B.18) to get

$$\begin{aligned} & J(n, \mathbf{c}) \Big|_{\mathbf{c}=\mathbf{c}'(n)} \\ &= [\mathbf{c}(n) + \mathbf{z}(n) - \mathbf{c}(n|n-1)]^T \mathbf{G}^{-1}(n) [\mathbf{c}(n) + \mathbf{z}(n) - \mathbf{c}(n|n-1)] + 2|\epsilon(n)| \\ &= \left[\frac{e(n)\mathbf{G}(n)\mathbf{x}(n)}{\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)} + \mathbf{z}(n) \right]^T \mathbf{G}^{-1}(n) \left[\frac{e(n)\mathbf{G}(n)\mathbf{x}(n)}{\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)} + \mathbf{z}(n) \right] + 0 \\ &= e^2(n) \frac{\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)}{(\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n))^2} + 2e(n) \underbrace{\frac{\mathbf{z}^T(n)\mathbf{G}^{-1}(n)\mathbf{G}(n)\mathbf{x}(n)}{\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)}}_{=0} + \underbrace{\mathbf{z}^T(n)\mathbf{G}^{-1}(n)\mathbf{z}(n)}_{>0 \text{ for } \mathbf{z}(n) \neq \mathbf{0}} \\ &> \frac{e^2(n)}{\mathbf{x}^T(n)\mathbf{G}(n)\mathbf{x}(n)} = J(n, \mathbf{c}) \Big|_{\mathbf{c}=\mathbf{c}(n)} \end{aligned} \quad (\text{B.31})$$

The last inequality completes the proof for the optimality of the solution $\mathbf{c}(n)$ within the hyperplane $\epsilon(n) = 0$. To check its global optimality, it suffices to check the cost function value at the only other ‘boundary’ of the problem, i.e. for $\mathbf{c}(n) \rightarrow \infty$. In this case, $J(n, \mathbf{c}) \rightarrow \infty$, too, and therefore, $\mathbf{c}(n)$ in (B.29a) achieves the global cost function minimum for branch ‘B’ of the algorithm. An overview of the whole algorithm with both branches ‘A’ and ‘B’ is given in (2.116a-d).

Appendix C

Quadratic Analysis of Tracking Properties

This appendix extends the tracking analysis of the general algorithm pattern (3.4a,b) to the analysis of the squared norm of the misalignment vector $\|\tilde{\boldsymbol{\theta}}(n)\|^2$. Unlike the linear analysis of Section 3.2, an exact decomposition into two contributions $\|\tilde{\boldsymbol{\theta}}_{lag}(n)\|^2$ and $\|\tilde{\boldsymbol{\theta}}_{noise}(n)\|^2$ is not possible without further assumptions on the mutual independence of the reference coefficient vector and the observation noise, as one has from (3.22a):

$$\|\tilde{\boldsymbol{\theta}}(n)\|^2 = \tilde{\boldsymbol{\theta}}^T(n)\tilde{\boldsymbol{\theta}}(n) = \|\tilde{\boldsymbol{\theta}}_{lag}(n)\|^2 + \|\tilde{\boldsymbol{\theta}}_{noise}(n)\|^2 + 2\tilde{\boldsymbol{\theta}}_{lag}^T(n)\tilde{\boldsymbol{\theta}}_{noise}(n) \quad (\text{C.1})$$

Now, if $\eta(n)$ and $\mathbf{c}_{ref}(n)$ are modeled as mutually orthogonal stochastic processes, i.e.

$$E\{\eta(m)\mathbf{c}_{ref}(n)\} = \mathbf{0} \quad \text{for all } m, n \quad (\text{C.2})$$

and if both $\eta(n)$ and $\mathbf{c}_{ref}(n)$ are assumed as statistically independent from the input data $\mathbf{x}(n)$ such that the filter equations (3.22b,c) are both linear, then the misalignment vector components $\tilde{\boldsymbol{\theta}}_{lag}(n)$ and $\tilde{\boldsymbol{\theta}}_{noise}(n)$ are mutually orthogonal as well:

$$E\{\tilde{\boldsymbol{\theta}}_{lag}(m)\tilde{\boldsymbol{\theta}}_{noise}^T(n)\} = \mathbf{0} \quad \text{for all } m, n \quad (\text{C.3})$$

In this case one obtains

$$E\{\|\tilde{\boldsymbol{\theta}}(n)\|^2\} = E\{\|\tilde{\boldsymbol{\theta}}_{lag}(n)\|^2\} + E\{\|\tilde{\boldsymbol{\theta}}_{noise}(n)\|^2\} \quad (\text{C.4})$$

and the two contributions can be evaluated individually as done in the next two sections where a partially deterministic approach to algorithm analysis is followed as proposed for a more specific case in [45].

C.1 Excess Error Due to Observation Noise

From (3.22c)

$$\begin{aligned} \|\tilde{\boldsymbol{\theta}}_{noise}(n)\|^2 &= \tilde{\boldsymbol{\theta}}_{noise}^T(n-1) \left(\gamma^2 \mathbf{P}(n) + \mathbf{P}^\perp(n) \right) \tilde{\boldsymbol{\theta}}_{noise}(n-1) + (1-\gamma)^2 \|\tilde{\boldsymbol{\eta}}(n)\|^2 \\ &\quad + 2(1-\gamma)\tilde{\boldsymbol{\eta}}^T(n) \left(\gamma \mathbf{P}(n) + \mathbf{P}^\perp(n) \right) \tilde{\boldsymbol{\theta}}_{noise}(n-1) \end{aligned} \quad (\text{C.5})$$

Now

$$\gamma^2 \mathbf{P}(n) + \mathbf{P}^\perp(n) = \mathbf{I} - (1 - \gamma^2) \mathbf{P}(n) \quad (\text{C.6})$$

and

$$\mathbf{P}^\perp(n) \tilde{\boldsymbol{\eta}}(n) = \mathbf{0} \quad (\text{C.7})$$

so that (C.5) yields

$$\begin{aligned} \|\tilde{\boldsymbol{\theta}}_{noise}(n)\|^2 &= \|\tilde{\boldsymbol{\theta}}_{noise}(n-1)\|^2 + \left\{ (\gamma^2 - 1) \tilde{\boldsymbol{\theta}}_{noise}^T(n-1) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) \right. \\ &\quad \left. + (1 - \gamma)^2 \|\tilde{\boldsymbol{\eta}}(n)\|^2 + 2\gamma(1 - \gamma) \tilde{\boldsymbol{\eta}}^T(n) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) \right\} \end{aligned} \quad (\text{C.8})$$

If it is assumed that the algorithm does not diverge¹, one has

$$0 < \|\tilde{\boldsymbol{\theta}}_{noise}(n)\|^2 < \tilde{\Theta}_{max}^2 \quad (\text{C.9})$$

for all n and some finite bound $\tilde{\Theta}_{max}^2$. *A fortiori* the running sum of the increments in the recursion (C.8) must stay within the same bounds:

$$\begin{aligned} 0 < \sum_{n=0}^{n_0} \left\{ (\gamma^2 - 1) \tilde{\boldsymbol{\theta}}_{noise}^T(n-1) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) + (1 - \gamma)^2 \|\tilde{\boldsymbol{\eta}}(n)\|^2 \right. \\ \left. + 2\gamma(1 - \gamma) \tilde{\boldsymbol{\eta}}^T(n) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) \right\} < \tilde{\Theta}_{max}^2 \end{aligned} \quad (\text{C.10})$$

From (C.10), a deterministic condition on the sample-path average *Avg* of the increments can be obtained:

$$\begin{aligned} \lim_{n_0 \rightarrow \infty} \frac{1}{n_0} \sum_{n=0}^{n_0} \left\{ (\gamma^2 - 1) \tilde{\boldsymbol{\theta}}_{noise}^T(n-1) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) + (1 - \gamma)^2 \|\tilde{\boldsymbol{\eta}}(n)\|^2 \right. \\ \left. + 2\gamma(1 - \gamma) \tilde{\boldsymbol{\eta}}^T(n) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) \right\} = \\ \text{Avg} \left\{ (\gamma^2 - 1) \tilde{\boldsymbol{\theta}}_{noise}^T(n-1) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) + (1 - \gamma)^2 \|\tilde{\boldsymbol{\eta}}(n)\|^2 \right. \\ \left. + 2\gamma(1 - \gamma) \tilde{\boldsymbol{\eta}}^T(n) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) \right\} = 0 \end{aligned} \quad (\text{C.11})$$

For a further evaluation all transformed quantities are expressed by their untransformed counterparts:

$$\begin{aligned} &\tilde{\boldsymbol{\theta}}_{noise}^T(n-1) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) \\ &= \boldsymbol{\theta}_{noise}^T(n-1) \mathbf{G}^{-T/2}(n-1) \frac{\mathbf{G}^{T/2}(n) \mathbf{x}(n) \mathbf{x}^T(n) \mathbf{G}^{1/2}(n)}{\mathbf{x}^T(n) \mathbf{G}(n) \mathbf{x}(n)} \mathbf{G}^{-1/2}(n-1) \boldsymbol{\theta}_{noise}(n-1) \\ &= \frac{1}{1 - \gamma} [\boldsymbol{\theta}_{noise}^T(n-1) \mathbf{x}(n)]^2 \end{aligned} \quad (\text{C.12})$$

¹This is easily shown from (C.8) for $\gamma < 1$ if the cross-correlation term $2\gamma(1 - \gamma) \tilde{\boldsymbol{\eta}}^T(n) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1)$ can be neglected, see the comment following (C.15).

$$\begin{aligned}
(1 - \gamma)^2 \|\tilde{\boldsymbol{\eta}}(n)\|^2 &= (1 - \gamma)^2 \eta(n) \frac{\tilde{\mathbf{x}}^T(n) \tilde{\mathbf{x}}(n)}{\|\tilde{\mathbf{x}}(n)\|^2 \|\tilde{\mathbf{x}}(n)\|^2} \eta(n) \\
&= (1 - \gamma) \eta^2(n)
\end{aligned} \tag{C.13}$$

$$\begin{aligned}
\tilde{\boldsymbol{\eta}}^T(n) \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) &= \eta(n) \frac{\tilde{\mathbf{x}}^T(n)}{\|\tilde{\mathbf{x}}(n)\|^2} \mathbf{P}(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) \\
&= \frac{\eta(n)}{1 - \gamma} \tilde{\mathbf{x}}^T(n) \tilde{\boldsymbol{\theta}}_{noise}(n-1) \\
&= \frac{\eta(n)}{1 - \gamma} \mathbf{x}^T(n) \mathbf{G}^{1/2}(n) \mathbf{G}^{-1/2}(n-1) \boldsymbol{\theta}_{noise}(n-1) \\
&= \frac{\eta(n)}{1 - \gamma} \mathbf{x}^T(n) \boldsymbol{\theta}_{noise}(n-1)
\end{aligned} \tag{C.14}$$

where the asymptotic equality $\mathbf{G}(n) \mathbf{G}^{-1}(n-1) = \mathbf{I}$ has been used several times. Insertion of these results into (C.11) yields

$$\text{Avg}\{[\boldsymbol{\theta}_{noise}^T(n-1) \mathbf{x}(n)]^2\} = \frac{2\gamma}{1 + \gamma} \text{Avg}\{\eta(n) \boldsymbol{\theta}_{noise}^T(n-1) \mathbf{x}(n)\} + \frac{1 - \gamma}{1 + \gamma} \text{Avg}\{\eta^2(n)\} \tag{C.15}$$

The ‘mixed’ or cross-correlation term on the right-hand side of (C.15) can be neglected in two cases:

1. $\gamma \rightarrow 0$, as is the case in zero-forcing algorithms. Then one obtains:

$$\text{Avg}\{[\boldsymbol{\theta}_{noise}^T(n-1) \mathbf{x}(n)]^2\} = \text{Avg}\{\eta^2(n)\} \tag{C.16}$$

which holds independently of the spectral properties of the noise sequence $\eta(n)$.

2. $\eta(n)$ is a white process with zero-mean. Then, from the ensuing orthogonality of $\eta(n)$ and $\boldsymbol{\theta}_{noise}(n-1)$, one obtains²

$$\text{Avg}\{[\boldsymbol{\theta}_{noise}^T(n-1) \mathbf{x}(n)]^2\} = \frac{1 - \gamma}{1 + \gamma} \text{Avg}\{\eta^2(n)\} \tag{C.17}$$

The left-hand side of (C.16) and (C.17), respectively, equals the excess error due to noise as observed at the (a priori) filter output. The ratio of this excess error and $\text{Avg}\{\eta^2(n)\}$ (the theoretical minimum of the total error which can only be achieved for $\mathbf{c}(n) = \mathbf{c}_{ref}(n)$) is called the *misadjustment due to noise*

$$M_{noise} = \frac{1 - \gamma}{1 + \gamma} \tag{C.18}$$

This result can be compared with results from algorithm-specific analyses found in the literature if the asymptotic γ -values are inserted from Table 3.1:

- LMS algorithm [236, Eq. (51)]

$$M_{noise} = \frac{\mu \text{trace}\{\mathbf{R}\}}{2 - \mu \text{trace}\{\mathbf{R}\}} \tag{C.19}$$

²This results holds with probability 1.

- Projection algorithm [22, Eq. (31)] [45, Eq. (38)]

$$M_{noise} = \frac{\mu}{2 - \mu} \quad (\text{C.20})$$

- Zero-forcing algorithms [18, Eq. (26)] [22, Eq. (31)] [45, Eq. (38)]

$$M_{noise} = 1 \quad (\text{C.21})$$

- LMS/Newton algorithms [239, Eq. (49)]

$$M_{noise} = \frac{\mu N}{2 - \mu N} \quad (\text{C.22})$$

- Exact RLS with growing memory or with selective data weighting (decreasing gain algorithms) [239, Eq. (23) with $N \rightarrow \infty$]

$$M_{noise} = 0 \quad (\text{C.23})$$

- Exact RLS with exponential or mixed data weighting [59, Eq. (3.21)] [146, Eq. (3.11)] [165, Eq. (19a)]

$$M_{noise} = \frac{1 - \lambda^N}{1 + \lambda^N} \approx \frac{N(1 - \lambda)}{2} \quad \text{for } 0 \ll \lambda < 1 \quad (\text{C.24})$$

As a final comment to this misalignment result, note that simulations in [59] show an excess error that is always slightly larger than predicted from an asymptotic theory. The explanation given by ELEFThERIOU and FALCONER in [59] can be carried over to the general analysis here: asymptotically one has

$$E\{\mathbf{G}(n)\mathbf{G}^{-1}(n-1)\}\Big|_{n \gg 1} = \mathbf{I} \quad (\text{C.25})$$

but the matrix product $\mathbf{G}(n)\mathbf{G}^{-1}(n-1)$ will still fluctuate around this identity matrix. These fluctuations can be described approximately as noise introduced into the feedback loop of the signal flow graph Figure 3.1 when replacing the ‘transforming delay’ \tilde{T} with a simple delay T . This extra noise leads to an increase of the overall excess error as compared to its theoretical value³.

C.2 Excess Error Due to Coefficient Lag

From (3.22b)

$$\begin{aligned} \|\tilde{\boldsymbol{\theta}}_{lag}(n)\|^2 &= [\tilde{\boldsymbol{\theta}}_{lag}(n-1) - \tilde{\mathbf{c}}_{ref}(n) + \tilde{\mathbf{c}}_{ref}(n-1)]^T \\ &\cdot [\gamma^2 \mathbf{P}(n) + \mathbf{P}^\perp(n)][\tilde{\boldsymbol{\theta}}_{lag}(n-1) - \tilde{\mathbf{c}}_{ref}(n) + \tilde{\mathbf{c}}_{ref}(n-1)] \end{aligned} \quad (\text{C.26})$$

Analysis of (C.26) is more involved than (C.5) as

³For the same reason, the misadjustment for the projection algorithm as predicted from (C.20) is smaller than the more accurate expression in [22, Eq. (31)]. This difference becomes, in particular, significant for small N ($N < 10$) where the fluctuations of the gain matrix $\mu/\|\mathbf{x}(n)\|^2 \mathbf{I}$ are non-negligible. For large N ($N \gg 1$) the present asymptotic theory delivers correct results.

- the first-difference of the reference coefficients $\tilde{\mathbf{c}}_{ref}(n) - \tilde{\mathbf{c}}_{ref}(n-1)$ is likely to be *correlated* with $\tilde{\boldsymbol{\theta}}_{lag}(n-1)$ unless a pure random walk model is assumed for the reference coefficients, and as
- it is desirable to evaluate the *time constant* of (C.26) because this constant characterizes the tracking behaviour in general while expressions for the asymptotic excess error due to coefficient lag always require rather specific modeling assumptions for $\mathbf{c}_{ref}(n)$.

Therefore analysis is restricted to the following simplifying assumptions

1. The *independence assumption* [163] is invoked so that $\tilde{\boldsymbol{\theta}}(n-1)$ can be assumed to be independent of $\tilde{\mathbf{x}}(n)$.
2. The reference coefficients are modeled as an autoregressive model with *large time constant* [213]. Then, in the limit of the time constant approaching infinity, $\tilde{\boldsymbol{\theta}}(n-1)$ and $\tilde{\mathbf{c}}_{ref}(n) - \tilde{\mathbf{c}}_{ref}(n-1)$ become independent as well.
3. Products of the form

$$(\mathbf{a}^T \mathbf{b})^2$$

are evaluated approximately as

$$\frac{1}{N} \|\mathbf{a}\|^2 \|\mathbf{b}\|^2$$

if \mathbf{a} and \mathbf{b} can be regarded as independent random vectors of dimension N each.

Under these assumptions one obtains from (C.26)

$$\begin{aligned} E\{\|\tilde{\boldsymbol{\theta}}_{lag}(n)\|^2\} &= E\{\|\tilde{\boldsymbol{\theta}}_{lag}(n-1)\|^2\} + E\{\|\tilde{\mathbf{c}}_{ref}(n) - \tilde{\mathbf{c}}_{ref}(n-1)\|^2\} \\ &\quad - \frac{1-\gamma^2}{\|\tilde{\mathbf{x}}(n)\|^2} E\left\{(\tilde{\boldsymbol{\theta}}_{lag}^T(n-1)\tilde{\mathbf{x}}(n))^2 + ([\tilde{\mathbf{c}}_{ref}(n) - \tilde{\mathbf{c}}_{ref}(n-1)]^T \tilde{\mathbf{x}}(n))^2\right\} \\ &= \left(1 - \frac{1-\gamma^2}{N}\right) E\left\{\|\tilde{\boldsymbol{\theta}}_{lag}(n-1)\|^2 + \|\tilde{\mathbf{c}}_{ref}(n) - \tilde{\mathbf{c}}_{ref}(n-1)\|^2\right\} \end{aligned} \quad (\text{C.27})$$

From (C.27), the time-constant τ for the convergence of the norm of the misalignment vector due to lag can be obtained as:

$$e^{-n/\tau} = \left(1 - \frac{1-\gamma^2}{N}\right)^n$$

$$\tau = -\frac{1}{\ln\left(1 - \frac{1-\gamma^2}{N}\right)} \approx \frac{N}{1-\gamma^2} \quad (\text{C.28})$$

where the last approximation is valid for $(1-\gamma^2)/N \ll 1$, a condition which is practically always satisfied—irrespective of the algorithm gain⁴—provided the filter length satisfies

⁴Note that $0 \leq \gamma^2 \leq 1$ for all stable algorithms.

$N \gg 1$. Furthermore, the asymptotic value of the norm of the (transformed) misalignment vector due to noise is obtained from (C.27) as

$$\begin{aligned} E\{\|\tilde{\boldsymbol{\theta}}_{lag}(n)\|^2\}_{n \gg 1} &= \left(\frac{N}{1-\gamma^2} - 1\right) E\{\|\tilde{\mathbf{c}}_{ref}(n) - \tilde{\mathbf{c}}_{ref}(n-1)\|^2\} \\ &\approx \tau E\{\|\tilde{\mathbf{c}}_{ref}(n) - \tilde{\mathbf{c}}_{ref}(n-1)\|^2\} \end{aligned} \quad (\text{C.29})$$

The asymptotic value of the excess error due to lag can be evaluated after back-transformation in the original domain from the following development:

$$\begin{aligned} E\left\{\left(\boldsymbol{\theta}_{lag}^T(n-1)\mathbf{x}(n)\right)^2\right\} &= E\left\{\left(\tilde{\boldsymbol{\theta}}_{lag}^T(n-1)\tilde{\mathbf{x}}(n)\right)^2\right\} \\ &= \frac{1}{N} E\left\{\|\tilde{\boldsymbol{\theta}}_{lag}(n-1)\|^2\right\} E\left\{\|\tilde{\mathbf{x}}(n)\|^2\right\} \\ &= \frac{1}{N} E\left\{\|\tilde{\boldsymbol{\theta}}_{lag}(n)\|^2\right\} E\left\{\|\tilde{\mathbf{x}}(n)\|^2\right\} \\ &= \frac{1}{N} \tau E\left\{\|\tilde{\mathbf{c}}_{ref}(n) - \tilde{\mathbf{c}}_{ref}(n-1)\|^2\right\} E\left\{\|\tilde{\mathbf{x}}(n)\|^2\right\} \\ &= \tau E\left\{\left([\tilde{\mathbf{c}}_{ref}(n) - \tilde{\mathbf{c}}_{ref}(n-1)]^T \tilde{\mathbf{x}}(n)\right)^2\right\} \\ &= \tau E\left\{\left([\mathbf{c}_{ref}(n) - \mathbf{c}_{ref}(n-1)]^T \mathbf{x}(n)\right)^2\right\} \\ &= \frac{1}{N} \tau E\left\{\|\mathbf{c}_{ref}(n) - \mathbf{c}_{ref}(n-1)\|^2\right\} E\left\{\|\mathbf{x}(n)\|^2\right\} \end{aligned}$$

$$E\left\{\left(\boldsymbol{\theta}_{lag}^T(n-1)\mathbf{x}(n)\right)^2\right\} = \tau \frac{\text{trace}\{\mathbf{R}\}}{N} E\left\{\|\mathbf{c}_{ref}(n) - \mathbf{c}_{ref}(n-1)\|^2\right\} \quad (\text{C.30})$$

The results (C.28) and (C.30) can be compared against more specific results derived in the literature:

- LMS algorithm

$$\tau = \frac{N}{\mu \text{trace}\{\mathbf{R}\} (2 - \mu \text{trace}\{\mathbf{R}\})} \quad (\text{C.31a})$$

$$E\left\{\left(\boldsymbol{\theta}_{lag}^T(n-1)\mathbf{x}(n)\right)^2\right\} = \frac{1}{2\mu} E\left\{\|\mathbf{c}_{ref}(n) - \mathbf{c}_{ref}(n-1)\|^2\right\} \quad (\text{C.31b})$$

where an approximation for small μ has been used in the last equation. This result coincides with [236, Eq. (80)] if

$$E\left\{\|\mathbf{c}_{ref}(n) - \mathbf{c}_{ref}(n-1)\|^2\right\} = N\sigma^2 \quad (\text{C.32})$$

is inserted from the reference coefficient model in [236, Fig. 10 with $a \rightarrow 1$]

- Projection algorithm

$$\tau = \frac{N}{\mu(2-\mu)} \quad (\text{C.33a})$$

$$E \left\{ \left(\boldsymbol{\theta}_{lag}^T(n-1) \mathbf{x}(n) \right)^2 \right\} = \frac{\text{trace}\{\mathbf{R}\}}{\mu(2-\mu)} E \left\{ \|\mathbf{c}_{ref}(n) - \mathbf{c}_{ref}(n-1)\|^2 \right\} \quad (\text{C.33b})$$

For the time constant cf. [45, Eq. (37)].

- Zero-forcing algorithm

$$\tau = N \quad (\text{C.34a})$$

$$E \left\{ \left(\boldsymbol{\theta}_{lag}^T(n-1) \mathbf{x}(n) \right)^2 \right\} = \text{trace}\{\mathbf{R}\} E \left\{ \|\mathbf{c}_{ref}(n) - \mathbf{c}_{ref}(n-1)\|^2 \right\} \quad (\text{C.34b})$$

Apparently the results in [18, Eqs. (20), (27)] are not applicable to the zero-forcing case (i.e. $\rho \rightarrow 1$, but unrestricted λ in the terminology of [18]) as they would obtain $\tau = 1$ (and a corresponding factor of $1/N$ in the excess error) which cannot be true for *any* adaptation algorithm.

- LMS/Newton algorithm

$$\tau = \frac{1}{\mu(2-\mu N)} \quad (\text{C.35a})$$

$$E \left\{ \left(\boldsymbol{\theta}_{lag}^T(n-1) \mathbf{x}(n) \right)^2 \right\} = \frac{\text{trace}\{\mathbf{R}\}}{2\mu} E \left\{ \|\mathbf{c}_{ref}(n) - \mathbf{c}_{ref}(n-1)\|^2 \right\} \quad (\text{C.35b})$$

cf. [239, Eq. (45b)] for the time constant and [239, Eq. (74)] for the excess error.

- Exact RLS with exponential data weighting

$$\tau \approx \frac{1}{2(1-\lambda)} \quad \text{if } 0 \ll \lambda < 1 \quad (\text{C.36a})$$

$$E \left\{ \left(\boldsymbol{\theta}_{lag}^T(n-1) \mathbf{x}(n) \right)^2 \right\} = \frac{\text{trace}\{\mathbf{R}\}}{2N(1-\lambda)} E \left\{ \|\mathbf{c}_{ref}(n) - \mathbf{c}_{ref}(n-1)\|^2 \right\} \quad (\text{C.36b})$$

cf. [59, Eqs. (4.11) and (4.13)], [146, Eq. (4.10)], and [165, Eq. (19b)].

From this comparison one can observe that even though rather strong assumptions have been necessary to arrive at the rule-of-thumbs (C.28) and (C.30) they are still quite generally applicable in practice.

Bibliography

- [1] D. ABOUTAJDINE, M. NAJIM, and M. OUADOU. “Time-varying parameter estimation: comparison of two classes of methods,” in I.T. Young et al., eds., *Signal Processing III: Theories and Applications*, pp. 361–364, North-Holland, Amsterdam etc., 1986.
- [2] G. AHLBOM, F. BIMBOT, and G. CHOLLET. “Modeling spectral speech transitions using temporal decomposition techniques,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’87*, pp. 13–16, Dallas (TX), April 1987.
- [3] G. ALENGRIN, M. BARLAUD, and J. MENEZ. “Unbiased parameter estimation of nonstationary signals in noise,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-34**(5):1319–1322, October 1986.
- [4] S.T. ALEXANDER. *Adaptive Signal Processing—Theory and Applications*. Springer-Verlag, New York, 1986.
- [5] B.D.O. ANDERSON. “Adaptive systems, lack of persistency of excitation and bursting phenomena,” *Automatica*, **21**(3):247–258, 1985.
- [6] B.D.O. ANDERSON et al. *Stability of Adaptive Systems: Passivity and Averaging Analysis*. The MIT Press, Cambridge (MA), 1986.
- [7] P. ANDERSSON. “Adaptive forgetting in recursive identification through multiple models,” *International Journal of Control*, **42**(5):1175–1193, 1985.
- [8] U. APPEL and A. VON BRANDT. “Recursive lattice algorithms with finite-duration windows,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’82*, pp. 647–650, Paris (France), May 1982.
- [9] K.J. ÅSTRÖM. “Theory and applications of adaptive control—a survey,” *Automatica*, **19**(5):471–486, 1983.
- [10] K.J. ÅSTRÖM and B. WITTENMARK. *Adaptive Control*. Addison-Wesley, Reading(MA), 1989.
- [11] B.S. ATAL. “Efficient coding of LPC parameters by temporal decomposition,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’83*, pp. 81–84, Boston (MA), 1983.

- [12] B.S. ATAL and M.R. SCHROEDER. "Adaptive predictive coding of speech signals," *The Bell System Technical Journal*, **49**(8):1973–1986, October 1970.
- [13] M. BASSEVILLE and A. BENVENISTE. *Detection of abrupt changes in signals and dynamical systems*. Vol. 77 of *Lecture Notes in Control and Information Sciences*, Springer-Verlag, Berlin, 1986.
- [14] M.G. BELLANGER. *Adaptive Digital Filters and Signal Analysis*. Marcel Dekker, New York and Basel, 1987.
- [15] J.R. BELLEGARDA and D.C. FARDEN. "Time-varying modelling of arbitrary non-stationary signals," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'88*, pp. 2200–2203, New York, April 1988.
- [16] J.R. BELLEGARDA and D.C. FARDEN. "Continuously adaptive linear predictive coding of speech," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'88*, pp. 347–350, New York, April 1988.
- [17] J.R. BELLEGARDA and D.C. FARDEN. "Time-varying signal modelling in arbitrary non-stationary environments," in J.L. Lacoume et al., eds., *Signal Processing IV: Theory and Applications*, pp. 1421–1424, North-Holland, Amsterdam, etc., 1988.
- [18] A. BENALLAL and A. GILLOIRE. "Improvement of the tracking capability of the numerically stable fast RLS algorithms for adaptive filtering," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'89*, pp. 1031–1034, Glasgow (Scotland), May 1989.
- [19] A. BENVENISTE. "Design of adaptive algorithms for the tracking of time-varying systems," *International Journal of Adaptive Control and Signal Processing*, **1**:3–29, 1987.
- [20] A. BENVENISTE, M. METIVIER, and P. PRIOURET. *Algorithmes adaptatifs et approximations stochastiques*. Masson, Paris (France), 1987.
- [21] A. BENVENISTE and G. RUGET. "A measure of the tracking capability of recursive stochastic algorithms with constant gains," *IEEE Transactions on Automatic Control*, **AC-27**(3):639–649, June 1982.
- [22] N.J. BERSHAD. "Analysis of the normalized LMS algorithm with Gaussian inputs," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-34**(4):793–806, August 1986.
- [23] N.J. BERSHAD. "On error-saturation nonlinearities in LMS adaptation," *IEEE Transactions on Acoustics, Speech, and Singal Processing*, **ASSP-36**(4):440–452, April 1988.
- [24] N.J. BERSHAD et al. "Tracking characteristics of the LMS adaptive line enhancer-response to a linear chirp signal in noise," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-28**(5):504–516, October 1980.

- [25] N.J. BERSHAD and O. MACCHI. “Comparison of RLS and LMS algorithms for tracking a chirped signal,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’89*, pp. 896–899, Glasgow (Scotland), May 1989.
- [26] D. BERTIN, S. BITTANTI, and P. BOLZERN. “A prediction-error directional forgetting technique for recursive estimation,” *Systems Science*, **11**(2):33–39, 1985.
- [27] D. BERTIN, S. BITTANTI, and P. BOLZERN. “Tracking of nonstationary systems by means of different prediction-error directional forgetting techniques,” in *Proceedings of the 2nd IFAC Workshop on Adaptive Systems in Control and Signal Processing*, pp. 91–96, Lund (Sweden), 1986.
- [28] G.J. BIERMAN. *Factorization methods for discrete sequential estimation*. Academic press, New York, 1977.
- [29] R.R. BITMEAD. “Persistence of excitation conditions and the convergence of some adaptive schemes,” *IEEE Transactions on Information Theory*, **IT-30**(2):183–191, March 1984.
- [30] S. BITTANTI, P. BOLZERN, and M. CAMPI. *Convergence and exponential convergence of identification algorithms with directional forgetting*. Report 88-002, Politecnico di Milano, Dipartimento di Elettronica, Milan (Italy), 1988. To appear in *Automatica*.
- [31] P.E. CAINES. *Linear Stochastic Systems*. John Wiley & Sons, New York etc., 1988.
- [32] G. CARAYANNIS, D. MANOLAKIS, and N. KALOUPSIDIS. “Fast Kalman type algorithms for sequential signal processing,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’83*, pp. 186–189, Boston (MA), 1983.
- [33] G. CARAYANNIS, D.G. MANOLAKIS, and N. KALOUPSIDIS. “A fast sequential algorithm for least-squares filtering and prediction,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-31**(6):1394–1402, December 1983. Reprinted also in L.H. Sibul, ed., *Adaptive Signal Processing*, pp. 178–186, IEEE Press, New York, 1987.
- [34] CCITT. *32 kbit/s adaptive differential pulse code modulation (ADPCM)*. Recommendation G.721, CCITT Study group XVIII, Geneva (Switzerland), July 1986. Draft revision of recommendation G.721 which supersedes the original proposal from October 1984.
- [35] R.W. CHANG. “A new equalizer structure for fast start-up digital communications,” *BSTJ*, **50**:1969–2017, July/August 1971.
- [36] J. CHEN and J. VANDEWALLE. “A μ -vector LMS adaptive system for enhancing nonstationary narrow-band signal,” in *Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS’88*, pp. 771–774, Helsinki, June 1988.

- [37] M.C. CHEVALIER, G. CHOLLET, and Y. GRENIER. "Speech analysis and restitution using time-dependent autoregressive models," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'85*, pp. 501–504, Tampa (FL), March 1985.
- [38] M.C. CHEVALIER and Y. GRENIER. "Autoregressive models with time-dependent log area ratios," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'85*, pp. 1049–1052, Tampa (FL), March 1985.
- [39] P.C. CHING and C.C. GOODYEAR. "Adaptive cascade filter for speech analysis," *IEE Proc.*, **130(E)**:11–18, January 1983.
- [40] G. CHOLLET. Private communication, 1985.
- [41] G. CHOLLET, Y. GRENIER, and S.M. MARCUS. "Temporal decomposition and non-stationary modeling of speech," in I.T. Young et al., eds., *Signal Processing III: Theories and Applications*, pp. 365–368, North-Holland, Amsterdam etc., 1986.
- [42] J.M. CIOFFI. "Limited-precision effects in adaptive filtering," *IEEE Transactions on Circuits and Systems*, **CAS-34(7)**:821–833, July 1987.
- [43] J.M. CIOFFI and T. KAILATH. "Fast, recursive-least-squares transversal filters for adaptive filtering," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-32(2)**:304–337, April 1984.
- [44] T.A.C.M. CLAASEN and W.F.G. MECKLENBRÄUKER. "On the transposition of linear time-varying discrete-time networks and its application to multirate digital systems," *Philips Journal of Research*, **33(1/2)**:78–102, 1978.
- [45] T.A.C.M. CLAASEN and W.F.G. MECKLENBRÄUKER. "Comparison of the convergence of two filter algorithms for adaptive FIR digital filters," *IEEE Transactions on Circuits and Systems*, **CAS-28(6)**:510–518, June 1981.
- [46] A.P. CLARK and R. HARUN. "Assessment of Kalman-filter channel estimators for an HF radio link," *IEE Proceedings*, **133-F(6)**:513–521, October 1986.
- [47] G.A. CLARK, S.K. MITRA, and S.R. PARKER. "Block implementation of adaptive digital filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-29**:744–752, June 1981.
- [48] A.O. CORDERO and D.Q. MAYNE. "Deterministic convergence of a self-tuning regulator with variable forgetting factor," *IEE Proceedings*, **128-D(1)**:19–23, January 1981.
- [49] C.F.N. COWAN and P.M. GRANT, eds. *Adaptive Filters*. Prentice-Hall, Englewood Cliffs (NJ), 1985.
- [50] C.F.N. COWAN and J. MAVOR. "Design and performance of a 256-point adaptive filter," *IEE Proceedings*, **127-F(3)**:179–184, June 1980.

- [51] S. DASGUPTA and Y.F. HUANG. "Asymptotically convergent modified recursive least-squares with data-dependent updating and forgetting factor for systems with bounded noise," *IEEE Transactions on Information Theory*, **IT-33**(3):383–392, May 1987.
- [52] A. DE LIMA VEIGA and Y. GRENIER. "A multi-step excited model for speech parameter trajectories," in *Proceedings of the IEEE International conference on Acoustics, Speech, and Signal Processing, ICASSP'88*, pp. 67–70, New York, April 1988.
- [53] J.R. DELLER. "Set membership identification in digital signal processing," *IEEE ASSP Magazine*, **6**(4):4–20, October 1989.
- [54] A. DEMBO and O. ZEITOUNI. "Maximum a posteriori estimation of time-varying ARMA processes from noisy observations," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-36**(4):471–476, April 1988.
- [55] C. DEVIEUX and R. PICKHOLTZ. "Adaptive equalization with a second order gradient algorithm," in *Proceedings of the Symposium on Computer Processing in Communications*, pp. 665–681, Polytechnic Institute of Brooklyn, Brooklyn(NY), April 1969.
- [56] C.E. DEVILA. "A recursive least-squares algorithm with data adaptive step size," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'89*, pp. 912–915, Glasgow (Scotland), May 1989.
- [57] J.L. DOOB. *Stochastic Processes*. John Wiley & Sons, New York, 1953.
- [58] D.L. DUTTWEILER. "Adaptive filter with nonlinearities in the correlation multiplier," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-30**(4):578–586, August 1982.
- [59] E. ELEFThERIOU and D.D. FALCONER. "Tracking properties and steady-state performance of RLS adaptive filter algorithms," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-34**(5):1097–1109, October 1986.
- [60] C.R. ELEVITCH et al. "Quiver diagrams and signed adaptive filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-37**(2):227–236, February 1989.
- [61] E. EWEDA. "Almost sure convergence of a decreasing gain algorithm for adaptive filtering," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-36**(10):1669–1671, October 1988.
- [62] E. EWEDA and O. MACCHI. "Convergence of the RLS and LMS adaptive filters," *IEEE Transactions on Circuits and Systems*, **CAS-34**(7):799–803, July 1987.
- [63] E. EWEDA and O. MACCHI. "Tracking error bounds of adaptive nonstationary filtering," *Automatica*, **21**(3):293–302, 1985.

- [64] D.C. FARDEN. "Tracking properties of adaptive signal processing algorithms," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-29**(3):439–446, June 1981.
- [65] D.C. FARDEN and K. SAYOOD. "Tracking properties of adaptive signal processing algorithms," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'80*, pp. 466–469, Denver (CO), April 1980.
- [66] A. FETTWEIS. "A general theorem for signal-flow networks, with applications," *Archiv für Elektronik und Übertragungstechnik*, **25**(12):557–561, 1971.
- [67] M.J. FLAHERTY. "Application of polynomial splines to a time-varying autoregressive model of speech," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'88*, pp. 2220–2223, New York, April 1988.
- [68] E. FOGEL and Y.F. HUANG. "On the value of information in system identification—bounded noise case," *Automatica*, **18**(2):229–238, 1982.
- [69] U. FORSSÉN and B. GUDMUNDSON. "Adaptive algorithms for arbitrary filter structures," in *Proceedings of the 1989 URSI International Symposium on Signals, Systems, and Electronics, ISSSE'89*, pp. 466–471, Erlangen(FRG), September 1989.
- [70] T.R. FORTESCUE, L.S. KERSHENBAUM, and B.E. YDSTIE. "Implementation of self-tuning regulators with variable forgetting factors," *Automatica*, **17**(6):831–835, 1981.
- [71] B. FRIEDLANDER. "Lattice filters for adaptive processing," *Proc. IEEE*, **70**:829–867, August 1982. Reprinted also in L.H. Sibul, ed., *Adaptive Signal Processing*, pp. 257–295, IEEE Press, New York, 1987.
- [72] W.A. GARDNER. "Nonstationary learning characteristics of the LMS algorithm," *IEEE Transactions on Circuits and Systems*, **CAS-34**(10):1199–1207, October 1987.
- [73] W.A. GARDNER. *Statistical Spectral Analysis—a Nonprobabilistic Theory*. Prentice-Hall Inc., Englewood Cliffs (NJ), 1988.
- [74] A. GERSHO. "Linear adaptation," in *Proceedings of the Symposium on Computer Processing in Communications*, pp. 653–664, Polytechnic Institute of Brooklyn, Brooklyn(NY), April 1969.
- [75] A. GERSHO. "Adaptive filtering with binary reinforcement," *IEEE Transactions on Information Theory*, **IT-30**(2):191–199, March 1984.
- [76] J.D. GIBSON, S.K. JONES, and J.L. MELSA. "Sequentially adaptive prediction and coding of speech signals," *IEEE Transactions on Communications*, **COM-22**(11):1789–1797, November 1974.
- [77] R.D. GITLIN and F.R. MAGEE, JR. "Self-orthogonalizing adaptive equalization algorithms," *IEEE Transactions on Communications*, **COM-25**(7):666–672, July 1977. Reprinted also in L.H. Sibul, ed., *Adaptive Signal Processing*, pp. 199–205, IEEE Press, New York, 1987.

- [78] R.D. GITLIN, H.C. MEADORS, and S.B. WEINSTEIN. “The tap leakage algorithm: an algorithm for the stable operation of a digitally implemented, fractionally spaced adaptive equalizer,” *The Bell System Technical Journal*, **61**(8):1817–1839, October 1982.
- [79] D. GODARD. “Channel equalization using a Kalman filter for fast data transmission,” *IBM Journal of Research and Development*, **18**:267–273, May 1974. Reprinted also in L.H. Sibul, ed., *Adaptive Signal Processing*, pp. 163–169, IEEE Press, New York, 1987.
- [80] G.H. GOLUB and C.F. VAN LOAN. *Matrix Computations*. The John Hopkins University Press, Baltimore (MD), 1983.
- [81] G.C. GOODWIN and K.S. SIN. *Adaptive Filtering Prediction and Control*. Prentice-Hall, Englewood Cliffs (NJ), 1984.
- [82] Y. GRENIER. “Time-dependent ARMA modeling of nonstationary signals,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-31**(4):899–911, August 1983.
- [83] Y. GRENIER. *Modélisation de signaux non-stationnaires*. Thèse pour le doctorat d’état, Université de Paris-Sud, Orsay (France), October 1984. N° d’ordre 2921.
- [84] Y. GRENIER. “Nonstationary ARMA models via simultaneous AR and MA estimation,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’86*, pp. 2339–2342, Tokyo (Japan), 1986.
- [85] Y. GRENIER and M.C. OMNES-CHEVALIER. “Autoregressive models with time-dependent log area ratios,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-36**(10):1602–1612, October 1988.
- [86] GSM. *GSM full rate speech transcoding*. Recommendation 6.10, CEPT/CCH groupe spécial mobile, March 1988. Draft 2.1.0.
- [87] T. HÄGGLUND. *New estimation techniques for adaptive control*. PhD thesis, Lund Institute of Technology, Lund (Sweden), 1983. Coden LUTFD2/(TFRT-1025)/1-120/(1983).
- [88] T. HÄGGLUND. “The problem of forgetting old data in recursive estimation,” in *Proceedings of the IFAC Workshop on Adaptive Systems in Control and Signal Processing*, pp. 213–214, San Francisco (CA), 1983.
- [89] M. HALL, A.V. OPPENHEIM, and A. WILLSKY. “Time-varying parameter modeling of speech,” *Signal processing*, **5**(3):267–285, May 1983.
- [90] R.W. HARRIS, D.M. CHABRIES, and F.A. BISHOP. “A variable step (VS) adaptive filter algorithm,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-34**(2):309–316, April 1986.
- [91] S. HAYKIN. *Adaptive Filter Theory*. Prentice-Hall, Englewood Cliffs (NJ), 1986.

- [92] D. HIRSCH and W.J. WOLF. "A simple adaptive equalizer for efficient data transmission," *IEEE Transactions on Communications*, **COM-18**(1):5–12, February 1970.
- [93] H. HÖGE. "Schätzung der LPC-Koeffizienten mit Hilfe des Kalmanfilters in der digitalen Sprachverarbeitung," *Frequenz*, **34**(5):149–152, 1980.
- [94] H. HÖGE. "Estimation of the dynamics of vocal tract parameters," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'81*, pp. 359–361, Atlanta (GA), March/April 1981.
- [95] H. HÖGE. "Adaptive estimation of some parameters of speech excitation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'82*, pp. 1314–1317, Paris (France), May 1982.
- [96] M.L. HONIG and D.G. MESSERSCHMITT. *Adaptive Filters: Structures, Algorithms, and Applications*. Kluwer Academic Publishers, Boston etc., 1984.
- [97] Y.F. HUANG. "A recursive estimation algorithm using selective updating for spectral analysis and adaptive signal processing," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-34**(5):1331–1334, October 1986.
- [98] R. ISERMANN and K.-H. LACHMANN. "Parameter-adaptive control with configuration aids and supervision functions," *Automatica*, **21**(6):625–638, 1985.
- [99] N.S. JAYANT and P. NOLL. *Digital Coding of Waveforms—Principles and Applications to Speech and Video*. Prentice-Hall, Englewood Cliffs (NJ), 1984.
- [100] D.A. JOHNS, W.M. SNELGROVE, and A.S. SEDRA. "State-space adaptive recursive filters," in *Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS'88*, pp. 2153–2156, Helsinki, June 1988.
- [101] C.R. JOHNSON, JR. "Convergence proof for a hyperstable adaptive recursive filter," *IEEE Transactions on Information Theory*, **IT-25**(6):745–749, November 1979.
- [102] C.R. JOHNSON, JR. "Adaptive IIR filtering: current results and open issues," *IEEE Transactions on Information Theory*, **IT-30**:237–250, March 1984.
- [103] C.R. JOHNSON, JR. *Lectures on Adaptive Parameter Estimation*. Prentice Hall, Englewood Cliffs (NJ), 1988.
- [104] C.R. JOHNSON, JR. and R.R. BITMEAD. "ADPCM: an object lesson in adaptive system theory," in *Proceedings of the IEEE/ASSP 1986 Digital Signal Processing Workshop*, pp. 2.3.1–2.3.2, Chatham (MA), October 1986.
- [105] C.R. JOHNSON, JR. et al. "SHARF convergence properties," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-29**(3):659–670, June 1981.
- [106] C.R. JOHNSON, JR. and T. TAYLOR. "Failure of a parallel adaptive identifier with adaptive error filtering," *IEEE Transactions on Automatic Control*, **AC-25**(6):1248–1250, December 1980.

- [107] P. KABAL and R.P. RAMACHANDRAN. “Joint solutions for the formant and pitch predictors in speech processing,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'88*, pp. 315–318, New York, April 1988.
- [108] P. KABAL and R.P. RAMACHANDRAN. “Joint optimization of linear predictors in speech coders,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-37**(5):642–650, May 1989.
- [109] T. KAILATH. “An innovations approach to least-squares estimation, part I: linear filtering in additive white noise,” *IEEE Transactions on Automatic Control*, **AC-13**:646–655, December 1968. Reprinted also in H.W. Sorenson, ed., *Kalman Filtering: Theory and Application*, pp. 55–64, IEEE Press, New York, 1985.
- [110] T. KAILATH and P. FROST. “An innovations approach to least-squares estimation, part II: linear smoothing in additive white noise,” *IEEE Transactions on Automatic Control*, **AC-13**:655–660, December 1968. Reprinted also in H.W. Sorenson, ed., *Kalman Filtering: Theory and Application*, pp. 252–257, IEEE Press, New York, 1985.
- [111] T. KIRYU and T. IJIMA. “Unbiased parameter estimation of non-stationary signals on the block processing,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'88*, pp. 2208–2211, New York, April 1988.
- [112] G. KITAGAWA. “Changing spectrum estimation,” *Journal of Sound and Vibration*, **89**(3):433–445, 1983.
- [113] G. KITAGAWA and W. GERSCH. “A smoothness priors time-varying AR coefficient modeling of nonstationary covariance time series,” *IEEE Transactions on Automatic Control*, **AC-30**(1):48–56, January 1985.
- [114] W.B. KLEIJN, D.J. KRASINSKI, and R.H. KETCHUM. “An efficient stochastically excited linear predictive coding algorithm for high quality low bit rate transmission of speech,” *Speech Communication*, **7**:305–316, 1988.
- [115] T. KOH and E.J. POWERS. “Second-order Volterra filtering and its application to nonlinear system identification,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-33**:1445–1455, December 1985.
- [116] P. KOPACEK. *Identifikation zeitvarianter Regelsysteme*. Friedr. Vieweg & Sohn, Braunschweig, Wiesbaden (FRG), 1978.
- [117] A.P. KOROSTELEV. “Multistep procedures of stochastic optimization,” *Avtomatika i Telemekhanika*, 82–90, May 1981. (Number 5).
- [118] F.J. KRAUS. *Das Vergessen in rekursiven Parameterschätzverfahren*. PhD thesis, Eidgenössische Technische Hochschule, Zürich (Switzerland), 1986. Diss. ETH Nr. 8012.

- [119] G. KUBIN. "Direct-form adaptive filter algorithms: a unified view," in I.T. Young et al., eds., *Signal Processing III: Theories and Applications*, pp. 127–130, North-Holland, Amsterdam etc., 1986.
- [120] G. KUBIN. "On local optimality of direct-form adaptive filter algorithms," in *Proceedings of the IEEE/ASSP 1986 Digital Signal Processing Workshop*, pp. 2.5.1–2.5.2, Chatham(MA), October 1986.
- [121] G. KUBIN. "Stabilization of the RLS algorithm in the absence of persistent excitation," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'88*, pp. 1369–1372, New York, April 1988.
- [122] G. KUBIN. "Robust estimation of the correlation matrix for adaptive filtering," in J.L. Lacoume et al., eds., *Signal Processing IV: Theories and Applications*, pp. 811–814, North-Holland, Amsterdam, etc., 1988.
- [123] G. KUBIN. "Stochastic versus deterministic concepts in adaptive systems design," in *Proceedings of the IEEE International Symposium on Circuits and Systems, IS-CAS'88*, pp. 39–42, Espoo (Finland), June 1988.
- [124] G. KUBIN. "Time-varying coefficient tracking and noise suppression properties of a class of adaptive algorithms," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'88*, pp. 1542–1545, New York, April 1988.
- [125] M.R. KUJATH, J. LI, and S.N. SARWAL. "Processing signals for estimating abrupt changes in system parameters," in M.H. Hamza, ed., *Applied Signal Processing*, pp. 85–88, Acta Press, Anaheim, Calgaray, Zurich, 1985.
- [126] R. KULHAVÝ. "Restricted exponential forgetting in real-time identification," in *Proceedings of the 7th IFAC/IFORS Symposium on Identification and System Parameter Estimation*, pp. 1143–1148, York (UK), July 1985.
- [127] R. KULHAVÝ. "Restricted exponential forgetting in real-time identification," *Automatica*, **23**(5):589–600, 1987.
- [128] R. KUMAR and J.B. MOORE. "Detection techniques in least squares identification," *Automatica*, **17**(6):805–819, 1981.
- [129] R. KUMAR and J.B. MOORE. "Adaptive equalization via fast quantized-state methods," *IEEE Transactions on Communications*, **COM-29**(10):1492–1501, October 1981.
- [130] R. KUMAR and J.B. MOORE. "Convergence of adaptive minimum variance algorithms via weighting coefficient selection," *IEEE Transactions on Automatic Control*, **AC-27**(1):146–153, February 1982.
- [131] T.L. LAI. "On the concept of excitation in least squares identification and adaptive control," *Stochastics*, **16**:227–254, 1986.

- [132] I.D. LANDAU. “Unbiased recursive identification using model reference adaptive techniques,” *IEEE Transactions on Automatic Control*, **AC-21**(2):194–202, April 1976.
- [133] I.D. LANDAU. “An addendum to ‘unbiased recursive identification using model reference adaptive techniques,’” *IEEE Transactions on Automatic Control*, **AC-23**(1):97–99, February 1978.
- [134] I.D. LANDAU. *Adaptive Control: The Model Reference Approach*. Marcel Dekker, New York, 1979.
- [135] I.D. LANDAU. “An extension to a stability theorem applicable to adaptive control,” *IEEE Transactions on Automatic Control*, **AC-25**(4):814–817, August 1980.
- [136] I.D. LANDAU. “A feedback system approach to adaptive filtering,” *IEEE Transactions on Information Theory*, **IT-30**(2):251–262, March 1984.
- [137] I.D. LANDAU and H.M. SILVEIRA. “A stability theorem with applications to adaptive control,” *IEEE Transactions on Automatic Control*, **AC-24**(2):305–312, April 1979.
- [138] M.G. LARIMORE, J.R. TREICHLER, and C.R. JOHNSON, JR. “SHARF: an algorithm for adapting IIR digital filters,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-28**(4):428–440, August 1980.
- [139] R. LOZANO LEAL. “Adaptive control with forgetting factor,” in *Proceedings of the IFAC World Congress*, pp. 83–88, Kyoto (Japan), 1981.
- [140] R. LOZANO LEAL and G.C. GOODWIN. “A globally convergent adaptive pole placement algorithm without a persistency of excitation requirement,” *IEEE Transactions on Automatic Control*, **AC-30**(8):795–798, August 1985.
- [141] E.A. LEE and D.G. MESSERSCHMITT. *Digital Communication*. Kluwer Academic Publishers, Boston, etc., 1988.
- [142] Y.-T. LEE. *A general time-varying model for speech signals and estimation of its parameters*. Report LEMS-40, Brown University, Division of Engineering, Laboratory for Engineering Man/Machine Studies, Providence (RI), 1988.
- [143] Y.-T. LEE and H.F. SILVERMAN. “On a general time-varying model for speech signals,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’88*, pp. 95–98, New York, April 1988.
- [144] S. LI. “On a class of nonstationary signals,” in *Proceedings of the IEEE International conference on Acoustics, Speech, and Signal Processing, ICASSP’88*, pp. 2192–2195, New York, April 1988.
- [145] Y. LINDE, A. BUZO, and R.M. GRAY. “An algorithm for vector quantizer design,” *IEEE Transactions on Communications*, **COM-28**:84–95, January 1984.

- [146] F. LING and J.G. PROAKIS. “Nonstationary learning characteristics of least squares adaptive estimation algorithms,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'84*, pp. 3.7.1–3.7.4, San Diego (CA), March 1984.
- [147] L.A. LIPORACE. “Linear estimation of nonstationary signals,” *Journal of the Acoustical Society of America*, **58**(6):1288–1295, December 1975.
- [148] L. LJUNG. “On positive real transfer functions and the convergence of some recursive schemes,” *IEEE Transactions on Automatic Control*, **AC-22**(4):539–551, August 1977.
- [149] L. LJUNG and S. GUNNARSSON. “Adaptation and tracking in system identification—a survey,” *Automatica*, **26**(1):7–21, January 1990.
- [150] L. LJUNG and T. SÖDERSTRÖM. *Theory and Practice of Recursive Identification*. The MIT Press, Cambridge (MA), 1983.
- [151] S.P. LLOYD. “Least squares quantization in PCM,” *IEEE Transactions on Information Theory*, **IT-28**:129–137, March 1982.
- [152] R.W. LUCKY. “Techniques for adaptive equalization of digital communication systems,” *The Bell System Technical Journal*, **45**(2):255–286, February 1966. Reprinted also in L.H. Sibul, ed., *Adaptive Signal Processing*, pp. 65–80, IEEE Press, New York, 1987.
- [153] O. MACCHI. “Optimization of adaptive identification for time-varying filters,” *IEEE Transactions on Automatic Control*, **AC-31**(3):283–287, March 1986.
- [154] G.A. MACK and V.K. JAIN. “A compensated-Kalman speech parameter estimator,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'85*, pp. 1129–1132, Tampa (FL), March 1985.
- [155] S. MARCOS and O. MACCHI. “Tracking capability of the least mean square algorithm: application to an asynchronous echo canceller,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-35**(11):1570–1578, November 1987.
- [156] S.M. MARCUS and R.A.J.M. VAN LIESHOUT. *Temporal decomposition of speech*. IPO Annual Progress Report, **19**: pp. 25–31, Instituut voor Perceptie Onderzoek, Eindhoven (The Netherlands), 1984.
- [157] J.D. MARKEL and A.H. GRAY, JR. *Linear Prediction of Speech*. Springer-Verlag, Berlin, Heidelberg, New York, 1976.
- [158] D.F. MARSHALL and W.K. JENKINS. “A fast quasi-Newton adaptive filtering algorithm,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'88*, pp. 1377–1380, New York, April 1988.
- [159] K.W. MARTIN and M.T. SUN. “Adaptive filters suitable for real-time spectral analysis,” *IEEE Transactions on Circuits and Systems*, **CAS-23**:218–229, February 1986.

- [160] G. MARTINELLI et al. "Identification of stable nonstationary lattice predictors by linear programming," *Proceedings of the IEEE*, **74**(5):759–760, May 1986.
- [161] G. MARTINELLI, G. ORLANDI, and L. PRINA RICOTTI. "Long term speech spectra of non-stationary AR models," in M.H. Hamza, ed., *Applied Signal Processing*, pp. 137–140, Acta Press, Anaheim, Calgaray, Zurich, 1985.
- [162] V.J. MATHEWS. "An efficient FIR adaptive filter using DPCM and the sign algorithm," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-37**(1):128–133, January 1989.
- [163] J.E. MAZO. "On the independence theory of equalizer convergence," *The Bell System Technical Journal*, **58**(5):963–993, May/June 1979.
- [164] R.J. MCAULAY and T.F. QUATIERI. "Speech analysis/synthesis based on a sinusoidal representation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-34**(4):744–754, August 1986.
- [165] S. McLAUGHLIN, B. MULGREW, and C.F.N. COWAN. "Performance comparison of least squares and least mean squares algorithms as HF channel estimators," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'87*, pp. 2105–2108, Dallas (TX), April 1987.
- [166] S. McLAUGHLIN, B. MULGREW, and C.F.N. COWAN. "The use of a priori knowledge for HF channel estimation," in J.L. Lacoume et al., eds., *Signal Processing IV: Theory and Applications*, pp. 371–374, North-Holland, Amsterdam, etc., 1988.
- [167] J.M. MENDEL. *Discrete Techniques of Parameter Estimation: The Equation Error Formulation*. Marcel Dekker, New York, 1973.
- [168] W.B. MIKHAEL et al. "Optimum adaptive algorithms with applications to noise cancellation," *IEEE Transactions on Circuits and Systems*, **CAS-31**(3):312–315, March 1984.
- [169] W.B. MIKHAEL et al. "Adaptive filters with individual adaptation of parameters," *IEEE Transactions on Circuits and Systems*, **CAS-33**(7):677–686, July 1986.
- [170] M. MILLNERT. "Identification of ARX models with markovian parameters," *International Journal of Control*, **45**(6):2045–2058, 1987.
- [171] R.V. MONOPOLI. "Model reference adaptive control with an augmented error signal," *IEEE Transactions on Automatic Control*, **AC-19**(5):474–484, October 1974.
- [172] H. MORIKAWA and H. FUJISAKI. "System identification of the speech production process based on a state-space representation," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-32**(2):252–262, April 1984.
- [173] A.T. MOSER and D. GRAUPE. "Identification of nonstationary models with application to myoelectric signals for controlling electrical stimulation of paraplegics," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-37**(5):713–719, May 1989.

- [174] K.H. MUELLER. "A new, fast-converging mean-square algorithm for adaptive equalizers with partial-response signaling," *The Bell System Technical Journal*, **54**(1):143–153, January 1975.
- [175] M.S. MUELLER. "On the rapid initial convergence of least-squares equalizer adjustment algorithms," *The Bell System Technical Journal*, **60**(10):2345–2358, December 1981.
- [176] B. MULGREW and C.F.N. COWAN. *Adaptive Filters and Equalisers*. Kluwer Academic Publishers, Boston etc., 1988.
- [177] H. NEY. "A dynamic programming technique for nonlinear smoothing," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'81*, pp. 62–65, Atlanta (GA), March/April 1981.
- [178] H. NEY. "Dynamic programming algorithm for optimal estimation of speech parameter contours," *IEEE Transactions on Systems, Man, and Cybernetics*, **SMC-13**(3):208–214, March/April 1983.
- [179] M. NIEDŹWIECKI. "On the localized estimators and generalized Akaike's criteria," *IEEE Transactions on Automatic Control*, **AC-29**(11):970–983, November 1984.
- [180] M. NIEDŹWIECKI. "On tracking properties of localized estimators," in I.T. Young et al., eds., *Signal Processing III: Theories and Applications*, pp. 1083–1086, North-Holland, Amsterdam etc., 1986.
- [181] M. NIEDŹWIECKI. "First-order tracking properties of weighted least squares estimators," *IEEE Transactions on Automatic Control*, **AC-33**(1):94–96, January 1988.
- [182] M. NIEDŹWIECKI. "On tracking characteristics of weighted least squares estimators applied to nonstationary system identification," *IEEE Transactions on Automatic Control*, **AC-33**(1):96–98, January 1988.
- [183] M. NIEDŹWIECKI. "Functional series modeling approach to identification of nonstationary stochastic systems," *IEEE Transactions on Automatic Control*, **33**(10):955–961, October 1988.
- [184] M. NIEDŹWIECKI. "Identification of nonstationary stochastic systems," in *Proceedings of the 27th IEEE Conference on Decision and Control, CDC'88*, Austin (TX), 1988. To appear in *IEEE Transactions on Automatic Control*, 1990.
- [185] M. NIEDŹWIECKI. "Identification of time-varying systems using combined parameter estimation and filtering," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-38**(4):679–686, April 1990.
- [186] M. NIEDŹWIECKI and L. GUO. "Non-asymptotic results for finite-memory WLS filters," in *Proceedings of the 28th IEEE Conference on Decision and Control, CDC'89*, Tampa (FL), November 1989.

- [187] M. NIEDŹWIECKI and Z. KOWALCZUK. “Improving parameter tracking properties of finite-memory adaptive filters,” in J.L. Lacoume et al., eds., *Signal Processing IV: Theory and Applications*, pp. 799–802, North-Holland, Amsterdam, etc., 1988.
- [188] A.V. OPPENHEIM and R.W. SCHAFER. *Digital Signal Processing*. Prentice-Hall, Englewood Cliffs, 1975.
- [189] F. PALMIERI and C.G. BONCELET, JR. “A class of nonlinear adaptive filters,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’88*, pp. 1483–1486, New York, April 1988.
- [190] G. PANDA et al. “A self-orthogonalizing efficient block adaptive filter,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-34**(6):1573–1582, December 1982.
- [191] A. PAPOULIS. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill International, Tokyo etc., 2nd edition, 1984.
- [192] H. PEREZ and S. TSUJII. “Adaptive filtering using orthogonal functions,” in J.L. Lacoume et al., eds., *Signal Processing IV: Theory and Applications*, pp. 575–578, North-Holland, Amsterdam, etc., 1988.
- [193] B. PORAT. “Second-order equivalence of rectangular and exponential windows in least-squares estimation of Gaussian autoregressive processes,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-33**(5):1209–1212, October 1985.
- [194] A.P. PORITZ. “Linear predictive hidden Markov models and the speech signals,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’82*, pp. 1291–1294, Paris (France), May 1982.
- [195] A.B. PORITZ. “Hidden Markov models: a guided tour,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’88*, pp. 7–13, New York, April 1988.
- [196] S.U.H. QURESHI. “Adaptive equalization,” *Proc. IEEE*, **73**:1349–1387, September 1985.
- [197] L.R. RABINER and B.H. JUANG. “An introduction to hidden Markov models,” *IEEE ASSP Magazine*, **3**(1):4–16, January 1986.
- [198] L.R. RABINER and R.W. SCHAFER. *Digital Processing of Speech Signals*. Prentice-Hall, Englewood Cliffs (NJ), 1978.
- [199] R.A. ROBERTS and C.T. MULLIS. *Digital Signal Processing*. Addison-Wesley, Reading(MA), 1987.
- [200] S. ROY. Private communication, 1988.
- [201] D.E. RUMELHART, G.E. HINTON, and R.J. WILLIAMS. “Learning internal representations by error propagation,” in D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, eds., *Parallel Distributed Processing—Explorations in the*

- Microstructure of Cognition, vol. 1: Foundations*, pp. 318–362, The MIT Press, Cambridge (MA), 1986.
- [202] A. RUSZCZYŃSKI and W. SYSKI. “Stochastic approximation method with gradient averaging for unconstrained problems,” *IEEE Transactions on Automatic Control*, **AC-28**(12):1097–1105, December 1983.
- [203] S. SAELID, O. EGELAND, and B. FOSS. “A solution to the blow-up problem in adaptive controllers,” *Modeling, Identification and Control*, **6**(1):39–56, 1985.
- [204] M.E. SALGADO, G.C. GOODWIN, and R.H. MIDDLETON. “Modified least squares algorithm incorporating exponential resetting and forgetting,” *International Journal of Control*, **47**(2):477–491, 1988.
- [205] S.P. SANOFF and P.E. WELLSTEAD. “Comments on: implementation of self-tuning regulators with variable forgetting factors,” *Automatica*, **19**(3):345–346, 1983.
- [206] W.A. SETHARES et al. “Excitation conditions for signed regressor least mean squares adaptation,” *IEEE Transactions on Circuits and Systems*, **CAS-35**(6):613–624, June 1988.
- [207] W.A. SETHARES and C.R. JOHNSON, JR. “Exciting conditions for quantized state adaptive algorithms,” in *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’87*, pp. 133–136, Dallas(TX), 1987.
- [208] W.A. SETHARES and C.R. JOHNSON, JR. “A comparison of two quantized state adaptive algorithms,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-37**(1):138–143, January 1989.
- [209] J.J. SHYNK. “Adaptive IIR filtering,” *IEEE ASSP Magazine*, **6**(2):4–21, April 1989.
- [210] J.J. SHYNK and S. ROY. “The LMS algorithm with momentum updating,” in *Proceedings of the IEEE International Symposium on Circuits and Systems*, pp. 2651–2654, Espoo (Finland), June 1988.
- [211] G.L. SICURANZA, A. BUCCONI, and P. MITRI. “Adaptive echo cancellation with nonlinear digital filter,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’84*, pp. 3.10.1–3.10.4, San Diego(CA), March 1984.
- [212] R.J. SLUYTER. “Digitization of speech,” *Philips Technical Review*, **41**(7/8):201–223, September 1984.
- [213] V. SOLO. “The limiting behavior of LMS,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-37**(12):1909–1922, December 1989.
- [214] “Special issue on adaptive systems,” *Proceedings of the IEEE*, **64**:1123–1243, August 1976.
- [215] “Special issue on adaptive signal processing,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-29**:625–775, June 1981.

- [216] “Special issue on adaptive filtering,” *IEEE Transactions on Information Theory*, **IT-30**:131–295, March 1984.
- [217] “Special issue on adaptive processing antenna systems,” *IEEE Transactions on Antennas and Propagation*, **AP-34**:273–462, March 1986.
- [218] “Special issue on adaptive systems and applications,” *IEEE Transactions on Circuits and Systems*, **CAS-34**:705–854, July 1987.
- [219] “Special issue on fading and multipath channel communications,” *IEEE Journal of Selected Areas in Communications*, **SAC-5**(2):65–311, February 1987.
- [220] P. STROBACH. “Pure order recursive least-squares ladder algorithms,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-34**(4):880–897, August 1986.
- [221] P. STROBACH. “Recursive covariance ladder algorithms for ARMA system identification,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-36**(4):560–580, April 1988.
- [222] J. SZTIPANOVITS. “Towards structural adaptivity,” in *Proceedings of the IEEE International Symposium on Circuits and Systems, ISCAS’88*, pp. 2359–2362, Espoo (Finland), June 1988.
- [223] A. TORICES-ARGUELLES and A.J. RUBIO-AYUSO. “On isolated word recognition systems using time dependent linear prediction,” in *Proceedings of the European Conference on Speech Technology*, pp. 280–283, Edinburgh (Scotland), September 1987.
- [224] J.R. TREICHLER, C.R. JOHNSON, JR., and M.G. LARIMORE. *Theory and Design of Adaptive Filters*. John Wiley & Sons, New York etc., 1987.
- [225] J.R. TREICHLER, M.G. LARIMORE, and S.L. WOOD. “Tracking speed requirements for time-varying adaptive channel equalizers,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’87*, pp. 2157–2160, Dallas (TX), April 1987.
- [226] E. TRULSSON and L. LJUNG. “Adaptive control based on explicit criterion minimization,” *Automatica*, **21**(4):385–399, 1985.
- [227] M.A. TUGAY and Y. TANIK. “Properties of the momentum LMS,” *Signal Processing*, **18**(2):117–127, October 1989.
- [228] H. UNBEHAUEN, B. GÖHRING, and B. BAUER. *Parameterschätzverfahren zur Systemidentifikation*. R. Oldenbourg Verlag, Munich, Vienna, 1974.
- [229] M. UNSER. “Recursion in short-time signal analysis,” *Signal Processing*, **5**:229–240, 1983.
- [230] P. VARY et al. “The European mobile radio speech codec,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’88*, pp. 227–230, New York, April 1988.

- [231] N.A.M. VERHOECKX, H.C. VAN DEN ELZEN, F.A.M. SNIJDERS, and P.J. VAN GERWEN. "Digital echo cancellation for baseband data transmission," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-27**(6):768–781, December 1979.
- [232] R. WEHRMANN. "Report on an international workshop on acoustic echo control held in Berlin (FRG) on 14 and 15 september 1989," *Signal Processing*, **19**(3):255–258, March 1990.
- [233] A. WEISS and D. MITRA. "Digital adaptive filters: Conditions for convergence, rates for convergence, effects of noise and errors arising from the implementation," *IEEE Transactions on Information Theory*, **IT-25**:637–652, November 1979.
- [234] P.E. WELLSTEAD and S.P. SANOFF. "Extended self-tuning algorithm," *International Journal of Control*, **34**(3):433–455, 1981.
- [235] E.T. WHITTAKER. "On a new method of graduation," *Proceedings of the Edinburgh Mathematical Society*, **41**:63–75, 1923.
- [236] B. WIDROW et al. "Stationary and nonstationary learning characteristics of the LMS adaptive filter," *Proc. IEEE*, **64**(8):1151–1162, August 1976. Reprinted also in L.H. Sibul, ed., *Adaptive Signal Processing*, pp. 187–198, IEEE Press, New York, 1987.
- [237] B. WIDROW and M.E. HOFF, JR. "Adaptive switching circuits," in *1960 IRE WESCON Convention Record*, pp. 96–104, IRE, New York, 1960. Reprinted also in J.A. Anderson and E. Rosenfeld, eds., *Neurocomputing: Foundations of Research*, pp. 126–134, The MIT Press, Cambridge(MA), 1988.
- [238] B. WIDROW and S.D. STEARNS. *Adaptive Signal Processing*. Prentice-Hall, Englewood Cliffs, 1985.
- [239] B. WIDROW and E. WALACH. "On the statistical efficiency of the LMS algorithm with nonstationary inputs," *IEEE Transactions on Information Theory*, **IT-30**(2):211–221, March 1984.
- [240] B. WIDROW, R.G. WINTER, and R.A. BAXTER. "Layered neural nets for pattern recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-36**(7):1109–1118, July 1988.
- [241] L.S. WILLIMANN. "Computation of the reponse-error-gradient of linear discrete filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-22**(1):62–65, February 1974.
- [242] B. WITTENMARK. "A two-level estimator for time varying parameters," *Automatica*, **15**:85–89, 1979.
- [243] S.L. WOOD, M.G. LARIMORE, and J.R. TREICHLER. "The impact of an adaptive equalizer's update behavior on the symbol error rate in a nonstationary environment," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP'88*, pp. 1608–1611, New York, April 1988.

- [244] Y. XIA. “Ein neuer LMS–Algorithmus mit schneller Konvergenz bei der Parameterschätzung mit Volterra–Filtern,” *Archiv für Elektronik und Übertragungstechnik*, **44**(1):59–64, 1990.
- [245] P. XUE and B. LIU. “Adaptive equalizer using finite-bit power-of-two quantizer,” in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP’84*, pp. 46.9.1–46.9.4, San Diego(CA), March 1984.
- [246] P. XUE and B. LIU. “Adaptive equalizer using finite-bit power-of-two quantizer,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, **ASSP-34**(6):1603–1611, December 1986.
- [247] M.B. ZARROP. “Variable forgetting factors in parameter estimation,” in *Proceedings of the IFAC Symposium on Identification and System Parameter Estimation*, pp. 1123–1126, Washington D.C., 1982.
- [248] G. ZIELKE. “Verlgemeinerte inverse Matrizen,” in *Jahrbuch Überblicke Mathematik 1983*, pp. 95–116, Bibliographisches Institut, Mannheim, 1983.
- [249] R. ZURMÜHL and S. FALK. *Matrizen und ihre Anwendungen. Teil 1: Grundlagen*. Springer-Verlag, Berlin, etc., 5th edition, 1984.
- [250] R. ZURMÜHL and S. FALK. *Matrizen und ihre Anwendungen. Teil 2: Numerische Methoden*. Springer-Verlag, Berlin, etc., 5th edition, 1986.
- [251] JA.S. ZYPKIN. *Grundlagen der informationellen Theorie der Identifikation*. Verlag Technik, Berlin, 1987. German translation from the Russian original: Ya.Z. Tsypkin, *Osnovy informacionnoj teorii identifikacii*, Izdatel’stvo ‘Nauka’, Moscow, 1984.

“...y cuando no sirva de otra cosa,
por lo menos, servirá aquel largo
catálogo de autores a dar de impro-
viso autoridad al libro.”

MIGUEL DE CERVANTES, Prólogo,
*El Ingenioso Hidalgo Don Quijote
de la Mancha*, Madrid, 1605.

Curriculum Vitae

GERNOT KUBIN was born as a son to Ing. Karl Kubin and his wife Herta Kubin, née Bilko, in Vienna, Austria, on June 24, 1960. After primary and secondary education at various places in Austria and the Federal Republic of Germany, he enrolled in the University of Technology, Vienna, in 1977 to study electrical engineering with specialization in communications engineering. In 1979 and 1982, he passed the first and second diploma examinations with highest honours, respectively, and graduated as Diplom-Ingenieur on June 23, 1982. Parallel studies were followed in general linguistics and finno-ugristics at the University of Vienna as well as in engineering physics at the University of Technology, Vienna. The latter led to some work on a quantum theoretic description for the scattering of Mößbauer radiation in single crystals in 1982/83.

Since 1983, he has been a research and teaching assistant at the Institut für Nachrichtentechnik und Hochfrequenztechnik of the University of Technology, Vienna. This period was interrupted in 1988/89 when he was on leave of absence with the Arbeiter-Samariter-Bund Österreichs (a Vienna-based ambulance organization) to complete his civil service.

His teaching activities have included classes on statistical communications theory and laboratory courses on transformer measurement and speech processing. Since 1986, he has lectured in EURASIP courses on adaptive filtering at the Universities of Technology in Vienna, Darmstadt (FRG), and Zurich (Switzerland).

Research activities have initially built on the outcome of his graduation project on speech synthesis. This work resulted in the development of an articulatory speech synthesizer for German that mimics the human speech organ movements. In 1985, he received an Erwin Schrödinger fellowship and spent seven months as a visiting scientist at the Natuurkundig Laboratorium (Philips Research Laboratories), Eindhoven, the Netherlands, where he was engaged in digital speech coding techniques at medium bit rates. In 1987, he was co-recipient of the Plansee award for his continued collaboration on the text-to-speech synthesis system GRAPHON.

While at Natuurkundig Laboratorium, interests into adaptive signal processing began to grow, too, and finally have led to the development of a unified theory of recursive algorithms for adaptive filtering and to the design of specific algorithms with improved numerical stability. Parts of this work are summarized for the present thesis. Current research interests continue to lie both in the fields of speech processing and adaptive signal processing.

Since July 17, 1987, he is married to Maria Kubin, née Seidl, and since April 28, 1988, they enjoy their little daughter Anna.