Chapter 8
Numerical Stability Properties

Phillip Regalia and Richard Le Borne

Abstract Designers of algorithms must not only solve the problem of interest, but do so using methods which are robust under perturbations in the data as well as the intermediate parameters of the method. More generally, it is often the case that the actual problem of interest is too complicated to solve directly; simplifying assumptions are necessary. At each stage, from problem identification, to the setup of the problem to be solved using some method, to the ultimate algorithm to be implemented in code, perturbations and their effects must be anticipated and analyzed. Stability is the property that assesses the level of robustness to perturbations that is required before the computed solution given by an algorithm can be used with confidence. The origin of the perturbation can vary, as pointed out above. What is important, however, is to have analysis that supports the premise that a small change in the problem results in a small change to the solution.

8.1 Introduction

The use of the QR-decomposition (vs. QR-algorithm) depends greatly on its reputation for providing consistently usable results. When an algorithm's reputation suffers, whether deserved or not, users often seek an alternative. It is therefore important, if not essential, to first establish the criteria in which a reliable solution can be guaranteed. For example, the reputation of Gaussian elimination suffered greatly in the 1940s because of its inability to always provide a usable solution. It was not until an analysis performed by J. H. Wilkinson [1], that introduced the

Phillip Regalia
Catholic University of America, Washington, DC – USA
e-mail: regalia@cua.edu

Richard Le Borne
Tennessee Technological University, Cookeville, TN – USA
e-mail: rleborne@tntech.edu

J.A. Apolinário Jr. (ed.), *QRD-RLS Adaptive Filtering*,
DOI 10.1007/978-0-387-09734-3_8,

relationship between perturbations and the conditioning of the problem, before confidence in Gaussian elimination could be re-established. In this chapter, we will focus our attention on stability and what it means in the context of the development process that begins with the clear articulation of the problem and ends with the computer implementation of the algorithm.

8.2 Preliminaries

The usefulness of a computed solution is a statement assessing its numerical accuracy. When the estimate of the desired solution is a vector, required is a means for assessing its accuracy. The vector norm assigns a single number to a vector and this property is very useful for assessing the error in a given quantity. There are different vector norms, but the most often used are considered equivalent in $\mathbb{R}^N$ in that one norm is within a constant factor of another. For example, the Euclidean norm, or two-norm, is defined as the square-root of the sum of squares of vector elements. For the vector of filter weights, $\mathbf{w} = [w_1, w_2, \ldots, w_N]^{\mathrm{T}}$, its Euclidean norm, $\|\mathbf{w}\|_2$ is given by

$$\|\mathbf{w}\|_2 = \sqrt{w_1^2 + w_2^2 + \cdots + w_N^2} \tag{8.1}$$

$$= \sqrt{\textstyle\sum_{i=1}^{N} w_i^2} \tag{8.2}$$

For matrices, the norm again is used to associate a single number to it but its definition is chosen to be compatible with vector norms. For example, the matrix norm associated with the Euclidean norm is called the spectral norm. For $\mathbf{X} \in \mathbb{R}^{N\times M}$, its spectral norm, $\|\mathbf{X}\|_2$, is defined as

$$\|\mathbf{X}\|_2 = \max_{\|\mathbf{w}\|_2=1} \|\mathbf{X}\mathbf{w}\|_2 \tag{8.3}$$

$$= \sqrt{\lambda_{max}(\mathbf{X}^{\mathrm{T}}\mathbf{X})} \tag{8.4}$$

$$= \sigma_{max}(\mathbf{X}), \tag{8.5}$$

where $\lambda_{max}(\mathbf{X}^{\mathrm{T}}\mathbf{X})$ is the maximum eigenvalue of $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ and $\sigma_{max}(\mathbf{X})$ is the largest singular value of the data matrix $\mathbf{X}$. The singular value of a matrix will next be defined. For a full discussion on vector and matrix norms a good source is [2, Chapter 2].

Suppose at time index k, we are given the data matrix $\mathbf{X}(k) \in \mathbb{R}^{(k+1)\times(N+1)}$ of rank r. Then there are unitary matrices $\mathbf{U}(k) \in \mathbb{R}^{(k+1)\times(k+1)}$ and $\mathbf{V}(k) \in \mathbb{R}^{(N+1)\times(N+1)}$ where $\mathbf{U}^{\mathrm{T}}(k)\mathbf{U}(k) = \mathbf{I} \in \mathbb{R}^{(k+1)\times(k+1)}$ and $\mathbf{V}^{\mathrm{T}}(k)\mathbf{V}(k) = \mathbf{I} \in \mathbb{R}^{(N+1)\times(N+1)}$ and a diagonal matrix $\mathbf{\Sigma}(k) \in \mathbb{R}^{(k+1)\times(N+1)}$ where $\mathbf{\Sigma}(k) = \mathrm{diag}(\sigma_1(k), \sigma_2(k), \ldots, \sigma_r(k), 0, \ldots, 0) = \mathrm{diag}(\mathbf{\Sigma}_+(k), 0, \ldots, 0)$ with $\sigma_1(k) \geq \sigma_2(k) \geq \ldots \geq \sigma_r > 0$. Then the singular value decomposition is given by:

Definition 1 (Singular Value Decomposition).

$$\mathbf{X}(k) = \mathbf{U}(k)\boldsymbol{\Sigma}(k)\mathbf{V}^{\mathrm{T}}(k) \tag{8.6}$$

$$= \mathbf{U}(k)\begin{bmatrix}\boldsymbol{\Sigma}_{+}(k) & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\mathbf{V}^{\mathrm{T}}(k) \tag{8.7}$$

Here, $\sigma_i(k),\ i = 1,\ldots,r$ are the singular values of $\mathbf{X}(k)$ and for

$$\mathbf{U}(k) = [\mathbf{u}_1(k),\ldots,\mathbf{u}_{k+1}(k)] \tag{8.8}$$

$$\mathbf{V}(k) = [\mathbf{v}_1(k),\ldots,\mathbf{v}_{N+1}(k)], \tag{8.9}$$

we have that the $\mathbf{u}_i(k),\ i = 1,\ldots,k+1$ and $\mathbf{v}_j(k),\ j = 1,\ldots,N+1$ are, respectively, the left and right singular vectors associated with the singular values:

$$\mathbf{X}^{\mathrm{T}}(k)\mathbf{u}_i(k) = \sigma_i\mathbf{v}_i(k), \quad i = 1,\ldots,r \tag{8.10}$$

$$\mathbf{X}^{\mathrm{T}}(k)\mathbf{u}_i(k) = 0, \quad i = r+1,\ldots,k+1, \tag{8.11}$$

and,

$$\mathbf{X}(k)\mathbf{v}_i(k) = \sigma_i\mathbf{u}_i(k), \quad i = 1,\ldots,r \tag{8.12}$$

$$\mathbf{X}(k)\mathbf{v}_i(k) = 0, \quad i = r+1,\ldots,N+1 \tag{8.13}$$

It also holds that the square of the ith right singular value, $\sigma_i(k),\ i = 1,\ \ldots,\ N+1$ is equal to the ith eigenvalue of the correlation matrix, $\mathbf{X}^{\mathrm{T}}(k)\mathbf{X}(k)$, that is, $\lambda_i(\mathbf{X}^{\mathrm{T}}(k)\mathbf{X}(k)) = \sigma_i^2$, and $\mathbf{X}^{\mathrm{T}}(k)\mathbf{X}(k)\mathbf{v}_i(k) = \lambda_i\mathbf{v}(k) = \sigma_i^2\mathbf{v}_i(k),\ i = 1,\ \ldots,\ N+1$.

The pseudo-inverse, or generalized inverse, $\mathbf{X}^{\dagger}(k)$ can then be defined using the singular value decomposition:

Definition 2 (Pseudo-Inverse).

$$\underbrace{\mathbf{X}^{\dagger}(k)}_{(N+1)\times(k+1)} = \mathbf{V}(k)\underbrace{\begin{bmatrix}\boldsymbol{\Sigma}_{+}^{-1}(k) & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}}_{(N+1)\times(k+1)}\mathbf{U}^{\mathrm{T}}(k) \tag{8.14}$$

Consider the least-squares filtering problem which seeks to find the weight vector $\mathbf{w}(k)$ such that

$$\mathbf{X}(k)\,\mathbf{w}(k) = \mathbf{d}(k) \tag{8.15}$$

with $\mathbf{w}(k)$ satisfying

$$\min_{\mathbf{w}} \|\mathbf{d}(k) - \mathbf{X}(k)\,\mathbf{w}\|_2. \tag{8.16}$$

It is well known [2, Chapter 3] that $\mathbf{w}(k)$ solves (8.15) if $\mathbf{w}(k) = \mathbf{X}^{\dagger}(k)\mathbf{d}(k) + (\mathbf{I} - \mathbf{X}^{\dagger}(k)\mathbf{X}(k))\mathbf{z}$, where $\mathbf{z} \in \mathbb{R}^{(N+1)\times 1}$ is arbitrary. For (8.16), $\mathbf{z}$ must obviously be the zero vector so that

$$\mathbf{w}(k) = \mathbf{X}^{\dagger}(k)\,\mathbf{d}(k). \tag{8.17}$$

Note that for the case that $\mathbf{X}(k)$ has full rank, i.e., $\operatorname{rank}(\mathbf{X}(k)) = N+1$, then $\mathbf{X}^{\dagger}(k)\mathbf{X}(k) = \mathbf{I}^{(N+1)\times(N+1)}$ and the solution is unique and can be given in terms of the normal equations:

$$\mathbf{X}^{\mathrm{T}}(k)\,\mathbf{X}(k)\,\mathbf{w}(k) = \mathbf{X}^{\mathrm{T}}(k)\,\mathbf{d}(k) \tag{8.18}$$

$$\mathbf{w}(k) = \left(\mathbf{X}^{\mathrm{T}}(k)\,\mathbf{X}(k)\right)^{-1}\mathbf{X}^{\mathrm{T}}(k)\,\mathbf{d}(k) \tag{8.19}$$

$$\mathbf{w}(k) = \mathbf{X}^{\dagger}(k)\,\mathbf{d}(k). \tag{8.20}$$

Returning to our interest in whether the computed results from a recursive least-squares method are usable, we turn to the cause for perturbations in a result that, in theory, should not affect the convergence properties. The amount of deviation in a computed quantity from its exact value, be it the filter weights or the filter *a posteriori* residuals, can be affected from two sources: the nature of the problem and the method chosen to solve it. For recursive least-squares, we are interested in computing a sequence of least-squares solutions $\mathbf{w}(k)$, $k = N, N+1, \ldots$, where, at time index k, we have the overdetermined system of equations $\mathbf{X}(k)\,\mathbf{w}(k) \approx \mathbf{d}(k)$ in which we are interested in determining either $\mathbf{w}(k)$, the least squares filter weights, or the minimal-valued filter residuals, $\boldsymbol{\varepsilon}(k) = \mathbf{d}(k) - \mathbf{X}(k)\,\mathbf{w}(k)$. Before the stability of a method can be assessed, however, the sensitivity of the problem to changes in the data must be studied. Clear terminology is needed for this distinction.

8.2.1 Conditioning, forward stability, and backward stability

Depending on our purposes, we may not be interested in the determination of the least-squares solution $\mathbf{w}(k)$ directly but only the least-squares residuals, $\varepsilon(k)$. This choice can, and in the case of recursive least-squares does, have an impact regarding the sensitivity of the problem to perturbations in the input data. In general, this sensitivity inherent to the problem is regarded as the *conditioning* of the problem and is the first step in assessing the quality of a method.

Definition 3. A problem is well-conditioned if small changes in the data invoke only small changes in the solution. Otherwise, the problem is considered to be ill-conditioned.

For example, for linear square systems of equations in which the number of unknowns equals the number of equations, $\mathbf{Ax} = \mathbf{b}$, for $\mathbf{A} \in \mathbb{R}^{n\times n}$ of full rank, $\mathbf{x}, \mathbf{b} \in \mathbb{R}^{n\times 1}$, the sensitivity of the solution, $\mathbf{x}$, to changes in the elements in $\mathbf{A}$ is measured through the condition number, $\kappa_2(\mathbf{A})$, of the matrix and is defined by

$$\kappa_2(\mathbf{A}) = \|\mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2. \tag{8.21}$$

This quantity gives a measurement to the proximity of $\mathbf{A}$ to a singular matrix. When $\mathbf{A}$ is nearly singular, a small change in its entries could have a very profound change in its solution $\mathbf{x}$, regardless of the method chosen to solve the problem!

Suppose that because of inaccuracies whose origin is purposely left vague, instead of determining $\mathbf{w}(k)$, we have computed $\hat{\mathbf{w}}(k)$. Whether it is acceptable to use $\hat{\mathbf{w}}(k)$ in place of $\mathbf{w}(k)$ is often not answered through direct measurement of the absolute or relative error, $\|\mathbf{w}(k) - \hat{\mathbf{w}}(k)\|_2$ or $\|\mathbf{w}(k) - \hat{\mathbf{w}}(k)\|_2 / \|\mathbf{w}(k)\|_2$, $\|\mathbf{w}(k)\|_2 \neq 0$, respectively (since we would simply use $\mathbf{w}(k)$!). However, through error analysis techniques it is sometimes possible to bound this quantity by parameters that are computable. When this is the case, the analysis bounding the absolute or relative error is termed a *forward* or *direct* error analysis. If the bounded quantity is small enough over general operating conditions, the computed solution $\hat{\mathbf{w}}(k)$ is deemed usable and the method employed to solve the problem is considered *forward stable*.

Definition 4. A method for solving $\mathbf{Xw} = \mathbf{d}$ is forward stable if it has a small forward error. That is, the computed result $\hat{\mathbf{w}}$ satisfies the condition that $\frac{\|\mathbf{w}-\hat{\mathbf{w}}\|_2}{\|\mathbf{w}\|_2}$ (with $\|\mathbf{w}\|_2 \neq 0$) is small.

Often, a forward analysis is too difficult to achieve useful results. When this is the case, an alternative approach for the error analysis, termed a *backward* error analysis, may be considered. For this, the computed solution $\hat{\mathbf{w}}(k)$ is interpreted as the exact solution to some other problem. Considering this new problem to be a perturbation of the original problem defines the perspective for the analysis. When the perturbed problem is near enough to the original problem, the computed solution is considered to be *backward* stable. When there are many possible problems in which $\hat{\mathbf{w}}(k)$ is the exact solution, the smallest perturbation from the original problem is chosen.

Definition 5. A method for solving a problem is backward stable if its backward error is small. That is, its computed solution is the exact solution to a slightly perturbed problem.

The analysis to measure the effects of finite-precision is often an application of the results found from the stability analysis of the method; the perturbations are defined to model the effects of computer arithmetic and finite-precision representation. At this level, the implementation of the method could include variations for handling the storage and numerical computations. The focus, then, would be to tailor the implementation of the method to exploit the capabilities and avoid the handicaps given by a computer representation environment.

To summarize the above, the attention is usually on the forward error since this translates directly to the usefulness of the computed solution. When a direct analysis to bound the forward error is not possible or too difficult, a useful means for interpreting and connecting the concepts of conditioning and forward and backward errors, when defined in a consistent way is [3, Chapter 1, p. 10],

$$\text{Forward Error} \lesssim (\text{Condition Number}) \times (\text{Backward Error}).$$

For a well-conditioned problem, a small backward error implies a small forward error. But an ill-conditioned problem could lead to a misinterpretation of a small backward error since here there could still be a large forward error. A method is forward or backward stable if it produces a small forward or backward error, respectively. But a backward stable method does not necessarily produce a usable computed solution if the problem itself is sensitive to perturbations.

8.3 The Conditioning of the Least-Squares Problem

Signal processing problems, in particular least-squares problems, are recursively updated in time as new measurement data is received. On a more abstract setting, this can be formulated as a non-linear mapping that produces a sequence of vectors that (hopefully) approximate with increasing accuracy some desired solution. Determining the stability of this mapping, i.e., the study performed on the mapping to determine the degree of continuity under the effect of perturbations, is the initial goal of an analysis. Without additional specifics regarding continuity, the interpretation of a stability analysis may be left to the reader; an unnecessary consequence that should be avoided. To this end, it is often the case when analyzing the effects of perturbations to make the following distinction: Conditioning refers to the problem that is to be solved, while stability refers to either the method or its implementation as an algorithm. This means that it is possible to get a bad computed solution because of the nature of the problem or because of the manner chosen to solve it. An analysis assessing only the method/algorithm will be incomplete without an analysis of the problem.

8.3.1 The conditioning of the least-squares problem

Before we formally state the least-squares problem, we introduce the following terminology. The following requires distinct notation to represent exact values from those which have been affected by perturbations. We will denote a perturbation by the δ symbol and the perturbed quantity by inserting a $\tilde{}$ above the symbol. For example, the presence of perturbations in the data matrix and the desired output will be denoted and defined by $\tilde{\mathbf{X}}(k) = \mathbf{X}(k) + \delta\mathbf{X}(k)$ and $\tilde{\mathbf{d}}(k) = \mathbf{d}(k) + \delta\mathbf{d}(k)$, respectively.

We now present two theorems from Golub and Wilkinson [4] that bound the effect of perturbations in $\mathbf{X}(k)$ and $\mathbf{d}(k)$ on the solutions to the least squares problem (8.15) and (8.16). For $\mathbf{X}(k)$ having full rank, we define its condition number κ_2 as

$$\kappa_2\big(\mathbf{X}(k)\big) \triangleq \frac{\sigma_1(k)}{\sigma_{N+1}(k)} \tag{8.22}$$

$$= \|\mathbf{X}(k)\|_2 \left\| \left[\mathbf{X}^{\mathrm{T}}(k)\mathbf{X}(k)\right]^{-1}\mathbf{X}^{\mathrm{T}}(k) \right\|_2 . \tag{8.23}$$

From the singular value decomposition it follows that,

$$\kappa_2(\mathbf{X}(k))^2 \triangleq \|\mathbf{X}(k)\|_2^2 \left\| \left[\mathbf{X}^{\mathrm{T}}(k)\mathbf{X}(k)\right]^{-1} \right\|_2 . \tag{8.24}$$

Theorem 1. *Let* $\mathbf{w}(k)$, $\boldsymbol{\varepsilon}(k)$, $\tilde{\mathbf{w}}(k)$, *and* $\tilde{\boldsymbol{\varepsilon}}(k)$ *satisfy*

$$\|\mathbf{X}(k)\,\mathbf{w}(k) - \mathbf{d}(k)\|_2 = \min_{\mathbf{w}} \|\boldsymbol{\varepsilon}(k)\|_2 \tag{8.25}$$

$$\|(\mathbf{X}(k) + \delta\mathbf{X}(k))\,\tilde{\mathbf{w}}(k) - (\mathbf{d}(k) + \delta\mathbf{d}(k))\|_2 = \min_{\tilde{\mathbf{w}}} \|\tilde{\boldsymbol{\varepsilon}}(k)\|_2 \tag{8.26}$$

where

$$\boldsymbol{\varepsilon}(k) = \mathbf{d}(k) - \mathbf{X}(k)\,\mathbf{w}(k), \text{ and}$$
$$\tilde{\boldsymbol{\varepsilon}}(k) = \left[\mathbf{d} + \tilde{\mathbf{d}}(k)\right] - \left[\mathbf{X}(k) + \delta\mathbf{X}(k)\right]\tilde{\mathbf{w}}(k).$$

If $\delta_{\max}$ *is given by*

$$\delta_{\max} = \max\left\{ \frac{\|\delta\mathbf{X}(k)\|_2}{\|\mathbf{X}(k)\|_2}, \frac{\|\delta\mathbf{d}(k)\|_2}{\|\mathbf{d}(k)\|_2} \right\} < \frac{\sigma_{N+1}(k)}{\sigma_1(k)} \tag{8.27}$$

and $\sin(\theta)$ *by*

$$\sin(\theta) = \frac{\rho_{\mathbf{w}(k)}}{\|\mathbf{d}(k)\|_2} \neq 1, \tag{8.28}$$

where $\rho_{\mathbf{w}(k)} \triangleq \|\mathbf{X}(k)\,\mathbf{w}(k) - \mathbf{d}(k)\|_2$ is the minimal least squares residual, then

$$\frac{\|\tilde{\mathbf{w}}(k) - \mathbf{w}(k)\|_2}{\|\mathbf{w}(k)\|_2} \leq \delta_{max} \left\{ \frac{2\kappa_2(\mathbf{X}(k)}{\cos(\theta)} + \tan(\theta)\kappa_2(\mathbf{X}(k))^2 \right\} + \mathscr{O}(\delta_{\max}^2) \quad (8.29)$$

and

$$\frac{\|\tilde{\boldsymbol{\varepsilon}}(k) - \boldsymbol{\varepsilon}(k)\|_2}{\|\mathbf{d}(k)\|_2} \leq \delta_{\max}\,(1 + 2\kappa_2(\mathbf{X}(k)))\min(1, k-N) + \mathscr{O}(\delta_{\max}^2). \quad (8.30)$$

Theorem 1 tells us that the sensitivity of the least squares filter residuals, $\varepsilon(k)$, are proportional to the conditioning, $\kappa_2(\mathbf{X}(k))$. Comparatively, the sensitivity of the filter weights to perturbations are proportional to the square of the conditioning, $\kappa_2^2(\mathbf{X}(k))$. This result pertains to the nature of the problem being solved and is independent of the method employed to solve it.

Under the conditions of Theorem 1 with the data matrix $\mathbf{X}(k)$ having full rank and no assumed perturbations in $\mathbf{d}(k)$, the conditioning of the least-squares problem, $\kappa_{LS}(\mathbf{X}(k), \mathbf{d}(k))$ is defined as [5, Chapter 1],

Definition 6.

$$\kappa_{LS}(\mathbf{X}(k), \mathbf{d}(k)) = \kappa_2(\mathbf{X}(k)) \left(1 + \kappa_2(\mathbf{X}(k)) \frac{\|\boldsymbol{\varepsilon}(k)\|_2}{\|\mathbf{X}(k)\|_2 \, \|\mathbf{w}(k)\|_2} \right). \quad (8.31)$$

From (8.31) it is seen that the conditioning of the least-squares problem, that is, the sensitivity of the least-squares problem to perturbations in the data matrix, depends on the *a posteriori* residual and thus on the right-hand-side vector $\mathbf{d}(k)$.

8.3.2 Consistency, stability, and convergence

The recursive least-squares problem produces a sequence of solutions, $\mathbf{x}(k)$, $k = N+1, \ldots$, and it is because of this that the issue of convergence is of interest. Specifically, even though in a stationary environment the recursive least-squares solution converges, perturbations may significantly alter these theoretical properties to the extent that the computed solution may not converge.

Suppose the computed solution, $\hat{\mathbf{w}}(k)$ at time index k, is a very good approximation to the true least-squares solution $\mathbf{w}(k)$. Since the next computed solution,

$\hat{\mathbf{w}}(k+1)$, involves $\hat{\mathbf{w}}(k)$, it is natural to question the usefulness for our definition of a stable method. After a large number of iterations, the accumulative effect from each approximate solution could have disastrous consequences regarding the notion of convergence.

To this end, there is a need for additional conditions on the method before the sequence of computed solutions can be guaranteed to converge to a good approximation of the desired solution, be it the filter weights (the least-squares solution) or the filter *a posteriori* error (residual). In a recursive environment, the update parameters are typically non-linearly interlaced. Abstractly, let the parameters used in the method's update scheme be denoted as a non-linear mapping $\{f : A \longrightarrow B\}$ where A denotes the set of input data and parameters at the current state and B denotes the computed solution as well as the updated parameters to be used in the next recursion. The mapping f is strongly associated to the least-squares problem, and as such inherits any restrictions such as matrix structure (symmetry, close-to-Toeplitz, positive definitiveness, etc.). Any perturbations to this mapping must not interfere with this association. We will formalize this, but first we need to define what we mean by an equicontinuous mapping.

Definition 7. The set $\{B\}$ is *admissible* if for all $\mathbf{a} \in A$ $\phi(\mathbf{a}) = \mathbf{b} \in B$ and $\mathbf{a}, \mathbf{b}$ are associated to a least-squares problem.

Definition 8. The mapping $\{f : A \longrightarrow B\}$ is *continuous at* $\mathbf{a}$ if for all $\zeta > 0$ there exists an $\eta > 0$, such that $\| \phi(\hat{\mathbf{a}}) - \phi(\mathbf{a}) \| \leq \zeta$ whenever $\| \hat{\mathbf{a}} - \mathbf{a} \| \leq \eta$.

If we are interested in the convergence properties of all members f_δ in a neighborhood of f then we need the notion of consistency in addition to continuity.

Definition 9. The mapping f is *consistent* if for some $\eta > 0$, and all $\hat{\mathbf{a}} = \mathbf{a} + \delta\mathbf{a}$, $\| \delta\mathbf{a} \| < \eta$, it holds that $\phi(\hat{\mathbf{a}}) = \hat{\mathbf{b}} \in B$, B admissible.

Definition 10. $\Phi_\delta = f_\delta^{-1}$ is called an *approximation to the inverse mapping* f^{-1} if and only if f_δ is consistent with respect to the mapping f.

Theorem 2. *For ϕ_δ consistent with respect to f, a sufficient condition for convergence is the equicontinuity of Φ_δ.*

The proof of Theorem 2 can be found in ([6] p. 10).

8.4 The Recursive QR Least-Squares Methods

The QR-decomposition relies on a process that will replace selected non-zero entries of a matrix or vector with zeros. When this process is performed using orthogonal matrices, desirable properties concerning numerical stability result. We review first a direct stability analysis for the full QR decomposition adaptive filtering analysis, and then observe that the sufficient conditions for stable behavior reduce to a form of backward consistency. We then examine fast least-squares algorithms based on the QR decomposition. Although a direct stability analysis is considerably more complicated (if not intractable), backward consistency conditions can be obtained in a simple form, and relate to convergence via Theorem 2.

8.4.1 Full QR decomposition adaptive algorithm

For the full QR decomposition adaptive filtering algorithm, the time recursions absorb a new input vector $\mathbf{x}(k)$ and reference sample $d(k)$ at each time instant. The $N+1$ elements of $\mathbf{x}(k)$, however, need not derive from a tapped delay line, and indeed might derive from $N+1$ separate channels or sensor outputs, for example.

The basic update recursion for the triangular array appears as:

$$\begin{bmatrix} \mathbf{0}^{\mathrm{T}} \\ \mathbf{U}(k) \end{bmatrix} = \mathbf{Q}_\theta(k) \begin{bmatrix} \mathbf{x}^{\mathrm{T}}(k) \\ \lambda^{1/2}\mathbf{U}(k-1) \end{bmatrix}, \qquad \mathbf{U}(-1) = \mathbf{U}_{-1}, \tag{8.32}$$

in which $\mathbf{U}_{-1}$ is the initial condition on the triangular array (typically the zero array or a small multiple of the identity), and the orthogonal matrix $\mathbf{Q}_\theta(k)$ is chosen to null the entries of the top row of the array. In practice, roundoff errors will also contaminate the updated triangular array; if we denote by $\tilde{\mathbf{U}}(k)$ the finite-precision representation of the triangular array, the finite-precision counterpart of the basic update recursion may be written as

$$\begin{bmatrix} \mathbf{0}^{\mathrm{T}} \\ \tilde{\mathbf{U}}(k) \end{bmatrix} = \mathbf{Q}_{\tilde{\theta}}(k) \begin{bmatrix} \mathbf{x}^{\mathrm{T}}(k) \\ \lambda^{1/2}\tilde{\mathbf{U}}(k-1) \end{bmatrix} + \begin{bmatrix} \mathbf{0}^{\mathrm{T}} \\ \delta\mathbf{U}(k) \end{bmatrix}, \qquad \tilde{\mathbf{U}}(-1) = \mathbf{U}_{-1}, \tag{8.33}$$

in which $\mathbf{Q}_{\tilde{\theta}}(k)$ is the product of rotations determined from the (finite-precision) triangularization of $\tilde{\mathbf{U}}(k)$, and the second term on the right-hand-side accounts for the difference between the first term, were it calculated in exact arithmetic, and the actual stored result on the left-hand side. We assume also that the initial condition $\mathbf{U}_{-1}$ admits an exact representation in finite-precision.

We examine first a direct stability analysis and then illustrate the connection with backward consistency concepts.

By squaring up either side of (8.32), we recover the familiar recursion

$$\begin{aligned}\mathbf{R}(k) &= [\mathbf{0}\ \ \mathbf{U}^{\mathrm{T}}(k)]\begin{bmatrix}\mathbf{0}^{\mathrm{T}}\\ \mathbf{U}(k)\end{bmatrix}\\ &= [\mathbf{x}(k)\ \ \lambda^{1/2}\mathbf{U}^{\mathrm{T}}(k-1)]\underbrace{\mathbf{Q}_{\theta}^{\mathrm{T}}(k)\,\mathbf{Q}_{\theta}(k)}_{\mathbf{I}}\begin{bmatrix}\mathbf{x}^{\mathrm{T}}(k)\\ \lambda^{1/2}\mathbf{U}(k-1)\end{bmatrix}\\ &= \lambda\,\mathbf{U}^{\mathrm{T}}(k-1)\,\mathbf{U}(k-1)+\mathbf{x}(k)\,\mathbf{x}^{\mathrm{T}}(k)\\ &= \lambda\,\mathbf{R}(k-1)+\mathbf{x}(k)\,\mathbf{x}^{\mathrm{T}}(k), \qquad \mathbf{R}(-1)=\mathbf{U}_{-1}^{\mathrm{T}}\mathbf{U}_{-1}. \end{aligned} \tag{8.34}$$

A similar squaring-up operation applied to (8.33) gives

$$\tilde{\mathbf{R}}(k)=\lambda\,\tilde{\mathbf{R}}(k-1)+\mathbf{x}(k)\,\mathbf{x}^{\mathrm{T}}(k)+\delta\mathbf{R}(k), \qquad \tilde{\mathbf{R}}_{-1}=\mathbf{U}_{-1}^{\mathrm{T}}\mathbf{U}_{-1}, \tag{8.35}$$

in which

$$\begin{aligned}\delta\mathbf{R}(k) &= [\mathbf{x}\ \ \lambda^{1/2}\tilde{\mathbf{U}}^{\mathrm{T}}(k-1)]\,\mathbf{Q}_{\tilde{\theta}}^{\mathrm{T}}(k)\begin{bmatrix}\mathbf{0}^{\mathrm{T}}\\ \delta\mathbf{U}(k)\end{bmatrix}\\ &+ [\mathbf{0}\ \ \delta\mathbf{U}^{\mathrm{T}}(k)]\,\mathbf{Q}_{\tilde{\theta}}(k)\begin{bmatrix}\mathbf{x}^{\mathrm{T}}(k)\\ \lambda^{1/2}\tilde{\mathbf{U}}(k-1)\end{bmatrix}\\ &+ [\delta\mathbf{U}(k)]^{\mathrm{T}}\delta\mathbf{U}(k)\end{aligned} \tag{8.36}$$

accounts for the roundoff errors injected at iteration k, once expressed in the covariance domain. The difference $\tilde{\mathbf{R}}(k)-\mathbf{R}(k)$ thus adheres to the recursion

$$[\tilde{\mathbf{R}}(k)-\mathbf{R}(k)]=\lambda\,[\tilde{\mathbf{R}}(k-1)-\mathbf{R}(k-1)]+\delta\mathbf{R}(k), \qquad \tilde{\mathbf{R}}(-1)-\mathbf{R}(-1)=\mathbf{0}. \tag{8.37}$$

This recursion admits the explicit solution

$$[\tilde{\mathbf{R}}(k)-\mathbf{R}(k)]=\sum_{n=0}^{k}\lambda^{k-n}\,\delta\mathbf{R}(n). \tag{8.38}$$

Now, if the finite-precision errors are bounded at each iteration, meaning that $\|\delta\mathbf{R}(n)\|\le B<\infty$ for some constant B, where $\|\cdot\|$ denotes any valid matrix norm, then the difference $\tilde{\mathbf{R}}(k)-\mathbf{R}(k)$ is easily bounded as follows:

$$\begin{aligned}\|\tilde{\mathbf{R}}(k)-\mathbf{R}(k)\| &= \left\| \sum_{n=0}^{k} \lambda^{n-k}\, \delta\mathbf{R}(n) \right\| \\ &\le \sum_{n=0}^{k} \lambda^{n-k}\, \|\delta\mathbf{R}(n)\| \\ &\le B \sum_{n=0}^{k} \lambda^{n-k} \\ &\le B\,\frac{1}{1-\lambda}\,, \qquad \text{for all } k, \end{aligned} \tag{8.39}$$

in which the final inequality is valid for $0 < \lambda < 1$. We observe, as expected, that a smaller value of B (corresponding to higher precision in the calculations) results in $\tilde{\mathbf{R}}(k)$ tracking its exact-arithmetic counterpart $\mathbf{R}(k)$ more closely. Conversely, choosing λ closer to one results in a less favorable distance bound between $\tilde{\mathbf{R}}(k)$ and $\mathbf{R}(k)$. This is because values of λ closer to one induce a greater effective memory of the algorithm, so that a given arithmetic error $\delta\mathbf{R}(n)$ will linger more prominently through successive iterations.

So what does this bound imply about the difference $\tilde{\mathbf{U}}(k)-\mathbf{U}(k)$? To answer this, we note that $\mathbf{U}(k)$ [respectively, $\tilde{\mathbf{U}}(k)$] is a Cholesky factor of $\mathbf{R}(k)$ [respectively, $\tilde{\mathbf{R}}(k)$]; the Cholesky factor becomes unique once we specify whether it is upper or lower triangular, with positive elements along the diagonal.[1] We note also that the Cholesky factor is a continuous function of a positive definite matrix argument (e.g., [7]). Thus, if $\mathbf{R}(k)$ and $\tilde{\mathbf{R}}(k)$ remain positive definite (to be addressed shortly), there exists a constant c such that the uniform bound $\|\tilde{\mathbf{R}}(k)-\mathbf{R}(k)\| \le B/(1-\lambda)$ for all k implies that

$$\|\tilde{\mathbf{U}}(k)-\mathbf{U}(k)\| \le \frac{cB}{1-\lambda}\,, \quad \text{for all } k. \tag{8.40}$$

Now, positive definiteness of $\mathbf{R}(k)$ and $\tilde{\mathbf{R}}(k)$ (which are never explicitly formed), reduces to a full rank condition on $\mathbf{U}(k)$ and $\tilde{\mathbf{U}}(k)$. In view of the triangular structure, this reduces, in turn, to either matrix having non-zero elements in each diagonal position, which is rather easily checked.

To summarize, the following three conditions:

1. Bounded arithmetic errors: $\|\delta\mathbf{U}(n)\| \le cB$ for all n [giving the numerator in (8.40)];
2. Forgetting factor strictly less than one: $0 < \lambda < 1$;
3. Full rank data: all diagonal elements of $\tilde{\mathbf{U}}(k)$ remain positive;

[1] One can also define a Cholesky factor with respect to the anti-diagonal, as in Chapter 3. We revert to the more conventional approach of triangular arrays with respect to the main diagonal in this chapter, so that the various statements to follow are more consistent with the cited references. The conclusions concerning stability carry over to algorithms based on an anti-diagonal Cholesky factorization as well, although the notations to describe intermediate quantities would change somewhat.

are sufficient to ensure that the finite-precision representation $\tilde{\mathbf{U}}(k)$ remains within a bounded distance from its exact arithmetic counterpart $\mathbf{U}(k)$ as k increases. The first two conditions are under the designer's control; the third condition is data dependent, and is usually captured as a *persistence of excitation* constraint. The formal definition, in the present context, takes the following form: Let $\sigma_1(k)$ and $\sigma_{N+1}(k)$ be the largest and smallest singular values, respectively, of the data matrix $\mathbf{X}(k)$. The data which build $\mathbf{X}(k)$ are persistently exciting of order $N+1$ if these extremal singular values are uniformly bounded, meaning that there exists positive constants κ_a and κ_b such that

$$\sigma_1(k) \leq \kappa_a < \infty, \quad \text{and} \quad \sigma_{N+1}(k) \geq \kappa_b > 0 \qquad \text{for all } k > k_0 \tag{8.41}$$

where k_0 is some starting time. These inequalities imply that the condition number of $\mathbf{X}(k)$ is bounded:

$$\frac{\sigma_1(k)}{\sigma_{N+1}(k)} \leq \frac{\kappa_a}{\kappa_b} < \infty, \qquad \text{for all } k > k_0 \tag{8.42}$$

and thus that the least-squares problem remains well-posed. Note that the ratio κ_a/κ_b can be shown equal to the quantity $\kappa_2(\mathbf{X}(k))$ from (8.23).

Fortunately, the persistence of excitation condition is easily checked in the orthogonal triangularization procedure:

Result 1 *If the input data are bounded, then persistent excitation holds if and only if there exists a constant $c > 0$ for which*

$$\cos\theta_i(l) \geq c, \qquad \textit{for all } i \textit{ and } l.$$

For the verification, we note that the triangular matrix $\mathbf{U}(\cdot)$ becomes rank deficient if and only if at least one of its diagonal elements vanishes. Thus let $\mathbf{U}_{ii}(l)$ denote the ith diagonal element of the Cholesky factor at any time l. The formula for the rotation angles which achieve the triangularization at an arbitrary time instant l is

$$\cos\theta_i(l) = \frac{\lambda^{1/2}\mathbf{U}_{ii}(l-1)}{\mathbf{U}_{ii}(l)}, \qquad i = 0, 1, \ldots, N-1, \tag{8.43}$$

which shows that $\cos\theta_i(l) = 0$ for some i if and only if $\mathbf{U}_{ii}(l-1) = 0$, i.e., if and only if $\mathbf{U}(l-1)$ is rank deficient. Thus if $\mathbf{U}(l-1)$ has full rank for all l, then $\cos\theta_i(l) > 0$ for all i and l, which is to say that $\cos\theta_i(l) = 0$ for some i and l if and only if the smallest singular value $\sigma_{N+1}(l-1) = 0$. By continuity arguments, bounding the smallest singular value $\sigma_{N+1}(l-1)$ away from zero for all l must likewise bound $\cos\theta_i(l)$ away from zero for all i and l, and vice versa.

To treat the joint-process portion, recall that the basic structure takes the form of the orthogonal filter

$$\begin{bmatrix} e_{q_1}(k) \\ \mathbf{d}_{q_2}(k) \end{bmatrix} = \mathbf{Q}_\theta(k) \begin{bmatrix} d(k) \\ \lambda^{1/2}\mathbf{d}_{q_2}(k-1) \end{bmatrix}. \tag{8.44}$$

In a practical implementation that includes roundoff errors, the recursion instead takes the form

$$\begin{bmatrix} \tilde{e}_{q_1}(k) \\ \tilde{\mathbf{d}}_{q_2}(k) \end{bmatrix} = \mathbf{Q}_{\tilde{\theta}}(k) \begin{bmatrix} d(k) \\ \lambda^{1/2}\tilde{\mathbf{d}}_{q_2}(k-1) \end{bmatrix} + \begin{bmatrix} \delta e(k) \\ \delta\mathbf{d}_{q_2}(k) \end{bmatrix}, \tag{8.45}$$

in which $\mathbf{Q}_{\tilde{\theta}}(k)$ is the product of rotations determined from the (finite-precision) triangularization of $\tilde{\mathbf{U}}(k)$. Observe that both $\tilde{\mathbf{d}}_{q_2}(k)$ and $\tilde{\mathbf{d}}_{q_2}(k-1)$ occur in this equation, indicative of a feedback loop involving the state variables $\tilde{\mathbf{d}}_{q_2}(\cdot)$. As such, roundoff errors accumulated in $\tilde{\mathbf{d}}_{q_2}(k)$ will propagate in time. Similar to the development above, provided $\lambda < 1$, the feedback loop may be shown exponentially stable, inducing bounded error growth provided the injected roundoff error (modeled by the term $\delta\mathbf{d}_{q_2}(k)$ above) is bounded at each time instant.

For the deeper question of whether the computed $\tilde{\mathbf{d}}_{q_2}(k)$ has relevance to the underlying least-squares problem, return to the unperturbed system (8.44) and partition the transition matrix $\mathbf{Q}_\theta(k)$ as

$$\mathbf{Q}_\theta(k) = \begin{bmatrix} \gamma(k) & \mathbf{g}^{\mathrm{T}}(k) \\ \mathbf{f}(k) & \mathbf{E}(k) \end{bmatrix}, \tag{8.46}$$

in which $\mathbf{E}(k)$ is $(N+1)\times(N+1)$, $\mathbf{f}(k)$ is $(N+1)\times 1$, $\mathbf{g}(k)$ is $(N+1)\times 1$, and $\gamma(k)$ is a scalar. The state equation may then be solved "backwards in time" as

$$\begin{aligned} \mathbf{d}_{q_2}(k) = {} & \lambda^{(l+1)/2}\,\boldsymbol{\Phi}(k,k-l)\,\mathbf{d}_{q_2}(k-l) \\ & + \underbrace{\left[\mathbf{f}(k)\ \mathbf{E}(k)\mathbf{f}(k-1)\ \boldsymbol{\Phi}(k,k-1)\mathbf{f}(k-2)\ \cdots\ \boldsymbol{\Phi}(k,k-l+1)\mathbf{f}(k-l)\right]}_{\mathscr{C}(k,k-l)} \\ & \times \begin{bmatrix} d(k) \\ \lambda^{1/2}d(k-1) \\ \lambda\, d(k-2) \\ \vdots \\ \lambda^{l/2}d(k-l) \end{bmatrix}, \end{aligned} \tag{8.47}$$

in which

$$\boldsymbol{\Phi}(k,k-l) \triangleq \begin{cases} \mathbf{E}(k)\mathbf{E}(k-1)\cdots\mathbf{E}(k-l+1), & l \geq 1; \\ \mathbf{I}, & l = 0; \end{cases} \tag{8.48}$$

is the *state transition matrix* [8] from time $k-l$ to k, and $\mathscr{C}(k,k-l)$ is the *controllability matrix* [8] for the system, over the same time window. The system is said to be *uniformly controllable* [9] provided there exists a window length L, and constants

$a > 0$ and $b < \infty$, for which the Gramian of the controllability matrix $\mathscr{C}(k,k-L)$ is bounded and of full rank:

$$a\mathbf{I} \leq \mathscr{C}(k,k-L)\mathscr{C}^{\mathrm{T}}(k,k-L) \leq b\mathbf{I}, \qquad \text{for all } k \geq k_0, \tag{8.49}$$

where k_0 is some starting time. [Observe that $\sqrt{a}$ and $\sqrt{b}$ bound the extremal singular values of $\mathscr{C}(k,k-L)$.] Since the controllability matrix $\mathscr{C}(k,k-L)$ has dimensions $(N+1) \times (L+1)$, clearly we must have $L \geq N$ if the full rank condition is to hold.

The relevance of this condition is that, when satisfied, an arbitrary configuration for $\tilde{\mathbf{d}}_{q_2}(k)$ can be reached by an appropriate choice of the (exponentially weighted) reference vector

$$[d(k), \lambda^{1/2} d(k-1), \ldots, \lambda^{L/2} d(k-L)]^{\mathrm{T}}. \tag{8.50}$$

As such, even with numerical errors accumulated in $\tilde{\mathbf{d}}_{q_2}(k)$, the values so obtained may be considered the *exact* state produced by some reference vector, as required of admissibility defined in Section 8.3.2.

Now, since the matrix $\mathbf{Q}_{\tilde{\theta}}(k)$ is orthogonal for each k, one may show (e.g., [10])

$$\boldsymbol{\Phi}(k,k-L)\boldsymbol{\Phi}^{\mathrm{T}}(k,k-L) + \mathscr{C}(k,k-L)\mathscr{C}^{\mathrm{T}}(k,k-L) = \mathbf{I}, \qquad \text{for all } k \text{ and } L, \tag{8.51}$$

so that

$$\begin{aligned} \mathscr{C}(k,k-L)\mathscr{C}^{\mathrm{T}}(k,k-L) &= \mathbf{I} - \boldsymbol{\Phi}(k,k-L)\boldsymbol{\Phi}^{\mathrm{T}}(k,k-L) \\ &\leq \mathbf{I} \end{aligned} \tag{8.52}$$

providing automatic satisfaction of the upper bound from (8.49) using $b = 1$.

For the lower bound, we claim:

Result 2 *With $L = N$, the joint-process section is uniformly controllable provided there exists a constant $c > 0$ for which*

$$\cos\theta_i(l) \geq c > 0, \qquad \textit{for all } i \textit{ and } l \geq k_0. \tag{8.53}$$

The proof amounts to calculating the square matrix $\mathscr{C}(k,k-N)$ and observing that it becomes rank deficient if and only if $\cos\theta_i(l) = 0$ for any order index i and any time index $k-L \leq l \leq k$. By continuity arguments, bounding $\cos\theta_i(l)$ away from zero must also bound the smallest singular value of $\mathscr{C}(k,k-L)$ away from zero, proving existence of a constant a fulfilling the lower bound from (8.49). In view of result 1 above, we see that persistence of excitation is sufficient to ensure backward consistency of the full QR algorithm, and that this in turn suffices for bounded error growth.

8.5 Fast QR Algorithms

Perhaps the earliest fast QR decomposition adaptive filtering algorithm (where "fast" means having computational complexity that scales linearly with the filter order N) was devised by Cioffi [11]. Somewhat more coherent developments of two varieties of such fast algorithms were obtained soon thereafter by Proudler et al. [12, 13]: The first featured order recursions in both ascending and descending order and was later rederived in [14], while the second exhibited all order recursions in ascending order, and is better known as the QRD lattice algorithm [15]. A traditional lattice-based derivation of this latter algorithm is found also in Ling [16].

The time recursion of the prediction section of a fast least-squares algorithm is a dynamic system of the form

$$\boldsymbol{\xi}(k) = f\big[\boldsymbol{\xi}(k-1), x(k)\big], \qquad \boldsymbol{\xi}(-1) = \boldsymbol{\xi}_{-1}, \tag{8.54}$$

in which $\boldsymbol{\xi}(\cdot)$ is the state vector (collecting all the quantities that must be stored at each iteration to propagate the solution in time), the map $f[\cdot,\cdot]$ accounts for the update equations, and $\boldsymbol{\xi}_{-1}$ is the initial condition on the state vector. In the full QR algorithm reviewed above, the state is simply $\boldsymbol{\xi}(k) = \mathbf{U}(k)$, and the map $f[\cdot,\cdot]$ performs the orthogonal triangularization to update $\mathbf{U}(k-1)$ to $\mathbf{U}(k)$. Fast least-squares algorithms exploit the shift structure of the data matrix $\mathbf{X}(k)$, allowing a more compact representation of the state and reducing the number of operations in the update equations $f[\cdot,\cdot]$ to a quantity linear in N. Unlike the full QR algorithm, however, the update equations cannot readily be rewritten as a linear recurrence relation, thus complicating considerably any attempt at a direct error analysis.

Backward consistency concepts, on the other hand, are still useful in assessing error propagation properties in such fast least-squares algorithms [17–19]. Introduce the set of reachable states in exact arithmetic, i.e., the set of state vector configurations that may be reached as the input sequence $x(0)$, $x(1)$, …, $x(k)$ varies over $\mathbb{R}^k$, and the initial condition $\boldsymbol{\xi}_{-1}$ varies over all "valid" initial conditions. The set of valid initial conditions is best thought of as the set of state vector orientations $\boldsymbol{\xi}(-1)$ that can be deposited by some past input sequence $x(-1)$, $x(-2)$, $x(-3)$, …, that may potentially extend infinitely into the past.

Now, in finite-precision, the actual prediction section behaves as

$$\tilde{\boldsymbol{\xi}}(k) = f\big[\tilde{\boldsymbol{\xi}}(k-1), x(k)\big] + \delta\boldsymbol{\xi}(k), \tag{8.55}$$

in which $\delta\boldsymbol{\xi}(k)$ accounts for the roundoff errors injected in the state vector at time k. Provided the computed state vector $\tilde{\boldsymbol{\xi}}(k)$ remains within the set of reachable states, it is indistinguishable from the exact state produced by a different input sequence $\tilde{x}(0)$, $\tilde{x}(1)$, …, $\tilde{x}(k)$ using possibly a different (but valid) initial condition $\tilde{\boldsymbol{\xi}}_{-1}$. If we now drive the perturbed system and its exact arithmetic counterpart (8.54) with the same future sequence $x(k+1)$, $x(k+2)$, …, and allow both systems to evolve *without further arithmetic errors*, the perturbed trajectory $\tilde{\boldsymbol{\xi}}(\cdot)$ will return to the true trajectory $\boldsymbol{\xi}(\cdot)$ provided $\lambda < 1$, and the future data is persistently exciting. This

is because, in the absence of arithmetic errors, either system is but a rewriting of the full QR algorithm using a different initial condition $\mathbf{U}(k)$ or $\tilde{\mathbf{U}}(k)$ at time k, and fed with the same input sequence from time k forward. With $\lambda < 1$, the algorithm forgets its initial condition as time evolves. This basic argument shows that, subject to consistency of the state vector, a least-squares algorithms (fast or full) will enjoy stable error propagation [17, 18], in which the error propagation experiment assumes no further arithmetic errors after a perturbation is injected. (This stable error propagation was first observed in [20] from a direct analysis, although without connections to backward consistency concepts). As such, explosive error growth must be preceded by an inconsistent value arising numerically in the state vector, a condition therefore to be avoided if at all possible. We should note that the error propagation experiment described here assumes a single perturbation followed by exact arithmetic calculations. In practice, roundoff errors are injected at each time-step, requiring due attention to error accumulation (as we did for the full QR algorithm above). Although stable propagation of a single error is a necessary condition for bounded error accumulation, it need not be sufficient. Nonetheless, provided the error propagation properties are exponentially stable, which generically holds provided $\lambda < 1$ and the input data are persistently exciting, bounded error growth may be expected to hold, at least for sufficiently fine numerical resolution [21]. Stronger results, in the form of Theorem 2 from Section 8.3.2, are also applicable here ([6] p. 10).

We first review the data structure applicable to fast least-squares algorithms and the data consistency properties which stem from these. For notational simplicity, we first set $\lambda = 1$ and denote the resulting data matrix as $\mathbf{X}_1(k)$ (where the subscript 1 emphasizes that $\lambda = 1$); this assumes a pre-windowed Hankel structure:

$$\mathbf{X}_1(k) = \begin{bmatrix} x(k) & x(k-1) & \cdots & x(k-N) \\ x(k-1) & x(k-2) & \cdots & x(k-N-1) \\ \vdots & \iddots & \iddots & \vdots \\ x(N) & \cdots & x(1) & x(0) \\ \vdots & \iddots & x(0) & 0 \\ x(1) & \iddots & \iddots & \vdots \\ x(0) & 0 & \cdots & 0 \end{bmatrix}. \tag{8.56}$$

If we introduce the correlation lags

$$r_n = \sum_{i=0}^{k-n} x(i)\,x(i+n), \qquad n = 0, 1, \ldots, N, \tag{8.57}$$

and rename the most recent input samples as

$$q_1 = x(k), \quad q_2 = x(k-1), \quad \ldots \quad q_N = x(k-N+1), \tag{8.58}$$

then for any k, the correlation matrix takes the structured form

$$\begin{aligned}\mathbf{R}_1(k) &= \mathbf{X}_1^{\mathrm{T}}(k)\mathbf{X}_1(k) \\ &= \begin{bmatrix} r_0 & r_1 & \cdots & r_N \\ r_1 & r_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & r_1 \\ r_N & \cdots & r_1 & r_0 \end{bmatrix} - \begin{bmatrix} 0 & \cdots & 0 & 0 \\ \vdots & \iddots & 0 & q_1 \\ 0 & \iddots & \iddots & \vdots \\ 0 & q_1 & \cdots & q_N \end{bmatrix}^2 .\end{aligned} \tag{8.59}$$

This is a symmetric Toeplitz matrix minus the square of a triangular Hankel matrix.

For the case $\lambda < 1$, let

$$\boldsymbol{\Lambda}(k) = \mathrm{diag}(1, \lambda^{1/2}, \lambda, \ldots, \lambda^{k/2}) \tag{8.60}$$

so that the data matrix may be written as

$$\mathbf{X}(k) = \boldsymbol{\Lambda}(k)\mathbf{X}_1(k), \tag{8.61}$$

in terms of the (unweighted) data matrix $\mathbf{X}_1(k)$ from (8.56). Let now

$$\mathbf{L} = \mathrm{diag}(1, \lambda^{1/2}, \ldots, \lambda^{N/2}). \tag{8.62}$$

Because $\mathbf{X}_1(k)$ is a Hankel matrix, we may observe the identity

$$\boldsymbol{\Lambda}(k)\mathbf{X}_1(k) = \overline{\mathbf{X}}(k)\mathbf{L}^{-1} \tag{8.63}$$

in which $\overline{\mathbf{X}}(k)$ is a Hankel matrix akin to (8.56) but built from the exponentially weighted sequence

$$\bar{x}(k-n) = \lambda^{n/2} x(k-n), \qquad n = 0, 1, \ldots, k. \tag{8.64}$$

As such, a transformed correlation matrix becomes

$$\mathbf{R}_1(k) \stackrel{\triangle}{=} \mathbf{L}\mathbf{R}(k)\mathbf{L} = \mathbf{L}\mathbf{X}^{\mathrm{T}}(k)\mathbf{X}(k)\mathbf{L} = \overline{\mathbf{X}}^{\mathrm{T}}(k)\overline{\mathbf{X}}(k) \tag{8.65}$$

which is the Gramian of a pre-windowed Hankel matrix $\overline{\mathbf{X}}(k)$. It may thus be written as a structured matrix as in (8.59), in which r_n and q_n are now defined from the (exponentially weighted) sequence $\bar{x}(n)$ from (8.64). This "trick" will allow us to examine the data consistency independently of the value of λ. We should emphasize that the error accumulation effects, however, will still vary with λ, as illustrated in the analysis of the full QR algorithm above.

The basic data consistency question for fast algorithms may thus be summarized as follows:

- Given a correlation matrix $\mathbf{R}_1(k) = \mathbf{L}\mathbf{R}(k)\mathbf{L}$, under what conditions can we find a pre-windowed Hankel matrix, call it $\overline{\mathbf{X}}(n)$, for which $\mathbf{R}(k) = \overline{\mathbf{X}}^{\mathrm{T}}(k)\overline{\mathbf{X}}(k)$?
- Given the stored variables at time k in the prediction section of a fast least-squares algorithm, under what conditions can they be considered the exact values obtained from some input sequence?

For the first query, clearly $\mathbf{R}_1(k)$ must be positive semi-definite, and assume the structured form illustrated in (8.59) above. Is this sufficient as well? The following result was first obtained in [22] in the context of digital filter design, and developed in greater detail in [23–27]: [2]

Result 3 *The covariance matrix* $\mathbf{R}_1(k) = \mathbf{L}\mathbf{X}^T(k)\mathbf{X}(k)\mathbf{L}$ *may be factored as*

$$\mathbf{R}_1(k) = \overline{\mathbf{X}}^T\overline{\mathbf{X}}, \tag{8.66}$$

with $\overline{\mathbf{X}}$ *a pre-windowed Hankel matrix, provided* $\mathbf{R}_1(k)$ *is positive definite and assumes the "Toeplitz minus squared Hankel" structure illustrated in (8.59).*

In fact, when $\mathbf{R}_1(k)$ is positive definite, there are infinitely many pre-windowed Hankel matrices $\overline{\mathbf{X}}$ fulfilling the factorization; they may all be parameterized via inverse scattering constructions [26, 27]. We should emphasize that the number of rows in any such factor $\overline{\mathbf{X}}$ is not easily controlled in the constructive procedure behind this result [27]; if we constrain the number of rows in $\overline{\mathbf{X}}$, the factorization problem is considerably more difficult. In practice, the number of rows of $\overline{\mathbf{X}}$ is not of immediate concern to the error analysis.

We review next the constructive procedure behind past input reconstruction, adapted from [27], and then examine consistency issues in fast QR adaptive filters.

8.5.1 Past input reconstruction

Here we review the procedure for factoring a structured covariance matrix as in (8.59) into a Hankel data matrix $\overline{\mathbf{X}}$. Let $\mathbf{Z}$ be the shift matrix with ones on the subdiagonal and zeros elsewhere. The displacement structure [28] of the covariance matrix from (8.59) becomes

[2] The original claim from [22] was that such a factorization will exist even when $\mathbf{R}_1(k)$ is positive semi-definite, and singular. This is not true in general [26, 27], unless a certain supplementary condition is satisfied as well. We shall sidestep this technical difficulty by focusing on the positive definite case in what follows.

$$\mathbf{R}_1 - \mathbf{Z}\mathbf{R}_1\mathbf{Z}^{\mathrm{T}} = \begin{bmatrix} r_0 & r_1 & \cdots & r_N \\ r_1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ r_N & 0 & \cdots & 0 \end{bmatrix} - \begin{bmatrix} 0 \\ q_1 \\ \vdots \\ q_N \end{bmatrix} [\cdot]^{\mathrm{T}}$$

$$= \begin{bmatrix} \sqrt{r_0} \\ r_1/\sqrt{r_0} \\ \vdots \\ r_N/\sqrt{r_0} \end{bmatrix} [\cdot]^{\mathrm{T}} - \begin{bmatrix} 0 \\ r_1/\sqrt{r_0} \\ \vdots \\ r_N/\sqrt{r_0} \end{bmatrix} [\cdot]^{\mathrm{T}} - \begin{bmatrix} 0 \\ q_1 \\ \vdots \\ q_N \end{bmatrix} [\cdot]^{\mathrm{T}}, \qquad (8.67)$$

where "$[\cdot]$" means "repeat the previous vector". From the three "generator vectors" [29] so exposed, create the $3 \times (N+1)$ array,

$$\mathbf{G} = \begin{bmatrix} \sqrt{r_0} & r_1/\sqrt{r_0} & r_2/\sqrt{r_0} & \cdots & r_N/\sqrt{r_0} \\ 0 & q_1 & q_2 & \cdots & q_N \\ 0 & r_1/\sqrt{r_0} & r_2/\sqrt{r_0} & \cdots & r_N/\sqrt{r_0} \end{bmatrix}, \qquad (8.68)$$

and iterate the following procedure:

1. Shift the first row of the array one position to the right:

$$\mathbf{G} \xrightarrow{\mathbf{Z}} \begin{bmatrix} 0 & \sqrt{r_0} & r_1/\sqrt{r_0} & \cdots & r_{N-1}/\sqrt{r_0} \\ 0 & q_1 & q_2 & \cdots & q_N \\ 0 & r_1/\sqrt{r_0} & r_2/\sqrt{r_0} & \cdots & r_N/\sqrt{r_0} \end{bmatrix}. \qquad (8.69)$$

2. Choose a hyperbolic rotation to annihilate the second element of the first non-zero column. In the first pass, this appears as

$$\begin{bmatrix} 1/\cos\theta_0 & \sin\theta_0/\cos\theta_0 & 0 \\ \sin\theta_0/\cos\theta_0 & 1/\cos\theta_0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 0 & \sqrt{r_0} & r_1/\sqrt{r_0} & \cdots & r_{N-1}/\sqrt{r_0} \\ 0 & q_1 & q_2 & \cdots & q_N \\ 0 & r_1/\sqrt{r_0} & r_2/\sqrt{r_0} & \cdots & r_N/\sqrt{r_0} \end{bmatrix}$$

$$= \begin{bmatrix} 0 & y_1 & \times & \cdots & \times \\ 0 & 0 & \times & \cdots & \times \\ 0 & r_1/\sqrt{r_0} & r_2/\sqrt{r_0} & \cdots & r_N/\sqrt{r_0} \end{bmatrix}, \qquad (8.70)$$

   in which $y_1 = \sqrt{r_0 - q_1^2}$ and $\sin\theta_0 = -q_1/\sqrt{r_0}$.

3. Choose a hyperbolic rotation to annihilate the third element of the first non-zero column. In the first pass, this appears as

$$\begin{bmatrix} 1/\cos\phi_0 & 0 & \sin\phi_0/\cos\phi_0 \\ 0 & 1 & 0 \\ \sin\phi_0/\cos\phi_0 & 0 & 1/\cos\phi_0 \end{bmatrix} \times \begin{bmatrix} 0 & y_1 & \times & \cdots & \times \\ 0 & 0 & \times & \cdots & \times \\ 0 & r_1/\sqrt{r_0} & r_2/\sqrt{r_0} & \cdots & r_N/\sqrt{r_0} \end{bmatrix}$$

$$= \begin{bmatrix} 0 & y_2 & \times & \cdots & \times \\ 0 & 0 & \times & \cdots & \times \\ 0 & 0 & \times & \cdots & \times \end{bmatrix}, \tag{8.71}$$

in which $y_2 = \sqrt{y_1^2 - (r_1^2/r_0)}$ and $\sin\phi_0 = -(r_1/\sqrt{r_0})/y_1$.

4. Replace **G** with the resulting array from (8.71), and iterative the above procedure a further $N-1$ times to eliminate all the elements of the second and third rows.

The above procedure will successfully terminate, and yield angles satisfying

$$|\sin\theta_n| < 1 \quad \text{and} \quad |\sin\phi_n| < 1, \qquad \text{for all } n, \tag{8.72}$$

if and only if the matrix $\mathbf{R}_1$ is positive definite [29].

A flow graph of the basic array operations appears as Figure 8.1, in which "z" denotes a right shift operation. Imagine now changing the flow direction of the lower two branches of the flow graph, to obtain Figure 8.2. In doing so, each hyperbolic rotation is converted to a planar (or orthogonal) rotation. We may now terminate the right-hand side of the figure by a lossless load $\mathbf{S}_L(z)$, where lossless here means:

- $\mathbf{S}_L(z)$ is analytic in $|z| < 1$, thus admitting a convergent series expansion

$$\mathbf{S}_L(z) = \sum_{n=0}^{\infty} \mathbf{S}_n z^n, \qquad |z| < 1. \tag{8.73}$$

If z^{-1} is the unit delay operator from digital filter design, then z is the (anti-causal) unit advance operator, and $\mathbf{S}_L(z)$ may be understood as a stable and anti-causal transfer function, with $\{\mathbf{S}_n\}$ its anti-causal impulse response.

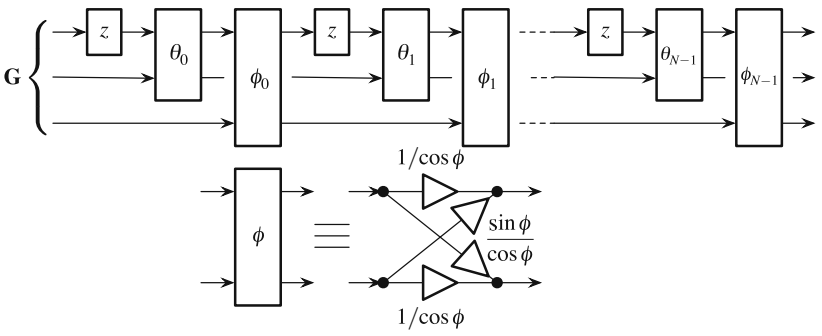

Fig. 8.1 Illustrating the successive annihilation operations applied to the rows of the **G** array.

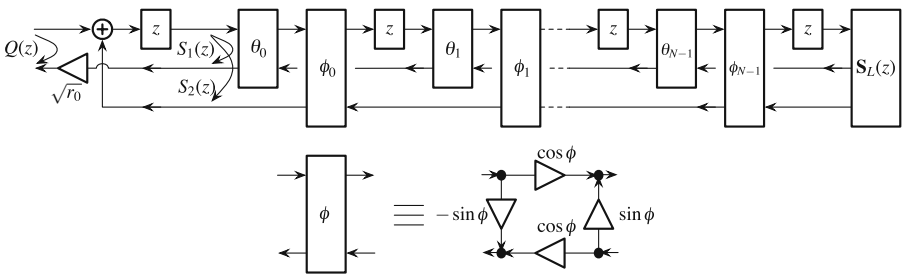

Fig. 8.2 Orthogonal filter, obtained by reversing the flow direction of the lower two branches.

- Upon partitioning $\mathbf{S}_L(z) = \begin{bmatrix} S_{L,1}(z) \\ S_{L,2}(z) \end{bmatrix}$, the radial limits along the unit circle are power complementary:

$$|S_{L,1}(e^{j\omega})|^2 + |S_{L,2}(e^{j\omega})|^2 = 1, \qquad \text{for all } \omega. \tag{8.74}$$

The flow graph with reversed directions on the lower two branches, and incorporating the load $\mathbf{S}_L(z)$, is sketched in Figure 8.2; the curved arrows indicate partial transfer functions $S_1(z)$ and $S_2(z)$ which result at the left-hand-side of the figure. Provided the resulting $S_2(z)$ satisfies the supplementary constraint

$$1 - zS_2(z) \neq 0, \qquad \text{for all } |z| = 1, \tag{8.75}$$

we may close the left-hand side of Figure 8.2 to obtain a stable and anti-causal transfer function

$$Q(z) = \sqrt{r_0}\,\frac{zS_1(z)}{1 - zS_2(z)} = \sum_{n=1}^{\infty} q_n z^n, \tag{8.76}$$

with the following property:

Property 1. Let $\overline{x}(k-n+1) = q_n$, $n = 1, 2, 3, \ldots$. The Hankel matrix $\overline{\mathbf{X}}(k)$ built from this sequence is a factor of the covariance matrix $\mathbf{R}_1$:

$$\mathbf{R}_1 = \overline{\mathbf{X}}^{\mathrm{T}}(k)\overline{\mathbf{X}}(k). \tag{8.77}$$

All such factors may be generated in this way, as $\mathbf{S}_L(z)$ varies over the set of lossless transfer functions.

A proof behind this construct may be found in [27]. We should note that the impulse response $\{q_n\}$ will, in general, have infinite duration. The simplest choice for the right-hand-side load $\mathbf{S}_L(z)$ is a constant unit norm vector of the form $\mathbf{S}_L(z) = \begin{bmatrix} \cos\phi_N \\ \sin\phi_N \end{bmatrix}$, where ϕ_N is any convenient value.

This result shows, thus, that there exists a one-to-one correspondence between the set of positive definite structured matrices of the form (8.59), and the set of rotation angles and scale factor fulfilling the inequalities

$$r_0 > 0 \quad \text{and} \quad \cos\theta_i > 0, \quad \cos\phi_i > 0, \qquad i = 0, 1, \ldots, N-1. \tag{8.78}$$

Note that we end up with $2N+1$ parameters, as expected, because the structured matrix $\mathbf{R}_1$ [cf. (8.59)] is specified by $2N+1$ values, namely $r_0, \ldots, r_N$ and $q_1, \ldots, q_N$. We confirm next that $2N+1$ is likewise the minimum state vector dimension of a fast least-squares prediction algorithm.

8.5.2 Reachable states in fast least-squares algorithms

We first consider the fast least-squares algorithm from [12, 14], which is a slightly simpler variant of the original fast QR algorithm from [11], and summarized in Table 8.1. A flow graph of the algorithm appears as Figure 8.3; rotations with a zero at one output represent angle solving steps (steps 2, 4 or 5 in the table), with the angles then copied to the data rotation steps (steps 3 and 6 in the table).

We observe that the state of the algorithm is the collection of $2N+1$ stored variables

$$x_{f,0}(k), \ldots, x_{f,N-1}(k), \sqrt{E_{f,N}(k)}, \varepsilon_{b,0}(k), \ldots, \varepsilon_{b,N-1}(k) \tag{8.79}$$

since all other variables are calculated from these. Thus, the minimum state vector dimension can be no larger than $2N+1$. The minimum state vector dimension can be no smaller, either, since there are $2N+1$ parameters involved in reconstructing past inputs by the procedure of Section 8.5.1, which is to say that $2N+1$ is indeed the minimum state vector dimension of a fast least-squares algorithm.

Suppose we consider now the experiment in which the input sequence $\{x(k)\}$ is allowed to vary over all sequences for which the structured matrix $\mathbf{R}_1(k)$ from (8.59) remains positive definite, and let us keep track of all state orientations that may be reached in the algorithm of Table 8.1 when using exact arithmetic. This defines the set of reachable states for the algorithm, and is characterized by the inequalities [30]

$$\sqrt{E_{f,N}(k)} > 0, \qquad \sum_{i=0}^{N-1} \varepsilon_{b,i}^2(k) < 1. \tag{8.80}$$

These inequalities are necessary and sufficient for the rotation angles determined in steps 2 and 5 of Table 8.1 to satisfy

$$|\sin\theta_i(k)| < 1, \qquad |\sin\phi_i(k)| < 1, \qquad \text{for all } i. \tag{8.81}$$

For any such state, therefore, a valid input sequence may be deduced, based on the following property:

Table 8.1 Minimal fast QR decomposition least-squares prediction algorithm.

FQR_POS_B – Version 2[a]

Available at time k: $x_{f,i}(k-1)$, $i=0,1,\ldots,N-1$;
$\sqrt{E_{f,N}(k-1)}$;
$\varepsilon_{b,i}(k-1)$, $i=0,1,\ldots,N-1$;
New datum: $x(k)$;

1. Obtain conversion factor $\gamma_{N-1}(k)$:

$$\gamma_{N-1}(k)=\sqrt{1-\sum_{i=0}^{N-1}\varepsilon_{b,i}^2(k-1)};$$

2. Solve for $\theta_i(k)$ angles:

$$\begin{bmatrix}\gamma_{i-1}(k)\\ 0\end{bmatrix}=\begin{bmatrix}\cos\theta_i(k) & \sin\theta_i(k)\\ -\sin\theta_i(k) & \cos\theta_i(k)\end{bmatrix}\begin{bmatrix}\gamma_i(k)\\ \varepsilon_{b,i}(k-1)\end{bmatrix},\qquad i=N-1,\ldots,1,0;$$

3. Update forward prediction variables. With $e_{f,0}(k)=x(k)$, run

$$\begin{bmatrix}x_{f,i}(k)\\ e_{f,i+1}(k)\end{bmatrix}=\begin{bmatrix}\cos\theta_i(k) & \sin\theta_i(k)\\ -\sin\theta_i(k) & \cos\theta_i(k)\end{bmatrix}\begin{bmatrix}\sqrt{\lambda}\,x_{f,i}(k-1)\\ e_{f,i}(k)\end{bmatrix},\qquad i=0,1,\ldots,N-1;$$

4. Update square-root forward prediction energy:

$$\begin{bmatrix}\sqrt{E_{f,N}(k)}\\ 0\end{bmatrix}=\begin{bmatrix}\cos\theta_N(k) & \sin\theta_N(k)\\ -\sin\theta_N(k) & \cos\theta_N(k)\end{bmatrix}\begin{bmatrix}\sqrt{\lambda\,E_{f,N}(k-1)}\\ e_{f,N}(k)\end{bmatrix};$$

5. Solve for $\phi_i(k)$ angles:

$$\begin{bmatrix}\sqrt{E_{f,i}(k)}\\ 0\end{bmatrix}=\begin{bmatrix}\cos\phi_i(k) & \sin\phi_i(k)\\ -\sin\phi_i(k) & \cos\phi_i(k)\end{bmatrix}\begin{bmatrix}\sqrt{E_{f,i+1}(k)}\\ x_{f,i}(k)\end{bmatrix},\qquad i=N-1,\ldots,1,0;$$

6. Update backward prediction errors: with $e_{f,N}(k)=\gamma_{N-1}(k-1)\sin\theta_N(k-1)$, run

$$\begin{bmatrix}\varepsilon_{f,i}(k)\\ \varepsilon_{b,i+1}(k)\end{bmatrix}=\begin{bmatrix}\cos\phi_i(k) & \sin\phi_i(k)\\ -\sin\phi_i(k) & \cos\phi_i(k)\end{bmatrix}\begin{bmatrix}\varepsilon_{f,i+1}(k)\\ \varepsilon_{b,i}(k-1)\end{bmatrix},\qquad i=N-1,\ldots,1,0;$$

At the end of the loop, set $\varepsilon_{b,0}(k)=\varepsilon_{f,0}(k)$.

[a] This algorithm corresponds to the one introduced in [14] and named FQR_POS_B–Version 2 in Chapter 4.

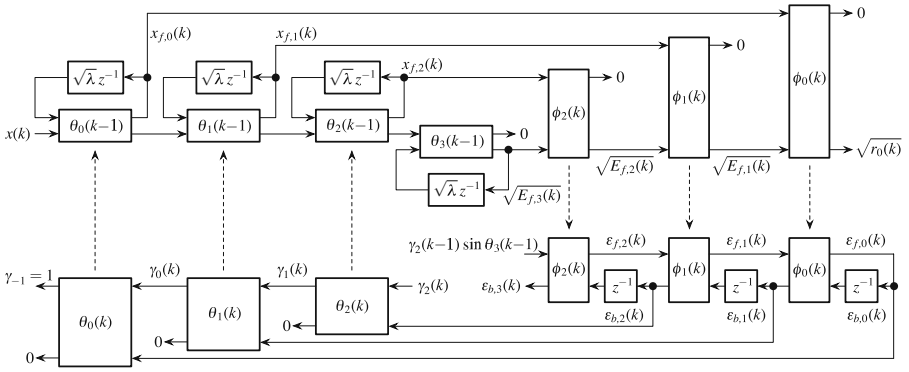

Fig. 8.3 Flow graph of minimal fast QR decomposition least-squares algorithm.

Property 2. Let $\theta_0(k), \ldots, \theta_{N-1}(k)$ and $\phi_0(k), \ldots, \phi_{N-1}(k)$ be the angles computed in exact arithmetic from the algorithm of Table 8.1, for a given input sequence $\{x(\cdot)\}$. Let $\mathbf{R}_1(k)$ be the structured matrix from (8.59) built using this same input sequence. The angles $\theta_0(k), \ldots, \theta_{N-1}(k)$ and $\phi_0(k), \ldots, \phi_{N-1}(k)$ are precisely those which achieve the annihilation of the generator vectors in the algorithm of Section 8.5.1.

The proof of this property involves more advanced notions of displacement generators [31], and may be found in [27, Appendix A]. The upshot is that, as long as the computed state variables in the algorithm satisfy the inequalities (8.80), they may be considered the exact values associated with a perturbed input sequence: From the computed state variables (or the rotation angles which annihilate them in steps 2 and 5 of Table 8.1), a valid past input sequence that would give rise to the computed values may be placed in evidence from the anti-causal filter of Figure 8.2. Backward consistency is thus straightforward to ensure in this version of the algorithm.

8.5.3 QR decomposition lattice algorithm

A noted oddity of the previous fast algorithm is that the order recursions run in both ascending and descending order, which can impede pipelined implementations. An alternate fast QR decomposition algorithm, obtained by Proudler et al. [12, 13], and Ling [16], runs all recursions in ascending order, and consists of the cascade of basic sections illustrated in Figure 8.4; the resulting algorithm is summarized in Table 8.2.

We observe that the total storage is $5N$ variables in the state vector, which is greater than the minimal number $2N+1$. Thus, some redundancy necessarily exists

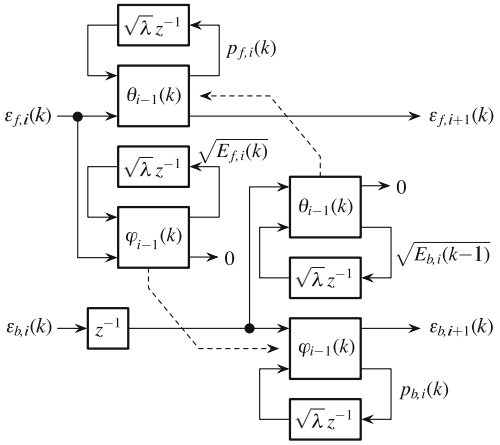

Fig. 8.4 Basic section to be cascaded to obtain the QRD-lattice algorithm.

among the elements of the state vector. For example, one may show [15] that, in exact arithmetic,

$$p_{f,i}(k) = \frac{\sum_{j=0}^{k} \varepsilon_{f,i}(j)\,\varepsilon_{b,i}(j-1)}{\sqrt{E_{b,i}(k-1)}}, \text{ and} \tag{8.82}$$

$$p_{b,i}(k) = \frac{\sum_{j=0}^{k} \varepsilon_{f,i}(j)\,\varepsilon_{b,i}(j-1)}{\sqrt{E_{f,i}(k)}}. \tag{8.83}$$

From this follows easily the equality

$$p_{f,i}(k)\sqrt{E_{b,i}(k-1)} - p_{b,i}(k)\sqrt{E_{f,i}(k)} = 0, \qquad \text{for each } k \text{ and } i, \tag{8.84}$$

inducing one constraint on four of the state variables from any section, so that one variable may be deduced given the remaining three. In the same manner, redundancy between sections may be deduced from the equality

$$\frac{\sqrt{E_{f,i+1}(k)}}{\sqrt{E_{f,i}(k)}} - \frac{\sqrt{E_{b,i+1}(k)}}{\sqrt{E_{b,i}(k-1)}} = 0, \qquad \text{for each } k \text{ and } i, \tag{8.85}$$

since either ratio relates to the reflection coefficient from section i (e.g., [14, 29]). The values calculated in finite precision, however, will not generally satisfy these constraints, which is to say that consistency is not inherited by the QRD-lattice algorithm (a generic problem with non-minimal realizations [17]). For this algorithm, however, the error accumulation properties will nonetheless remain bounded provided $\lambda < 1$ and the input sequence $\{x(k)\}$ is persistently exciting. This is because the algorithm consists of rotations, and variables reinjected through the feedback loop are all attenuated by $\sqrt{\lambda}$; the proof of bounded error growth mimics that of Section 8.4.1 for the full QR adaptive filtering algorithm. Alternatively, a minimal

Table 8.2 QR decomposition lattice least-squares prediction algorithm.

FQRD–Lattice [12, 13, 16]

Available at time k: $p_{f,i}(k-1)$, $i=0,1,\ldots,N-1$; (forward prediction variables)
$p_{b,i}(k-1)$, $i=0,1,\ldots,N-1$; (backward prediction variables)
$\sqrt{E_{f,i}(k-1)}$, $i=0,1,\ldots,N-1$; $\sqrt{\text{forward prediction energies}}$
$\sqrt{E_{b,i}(k-1)}$, $i=0,1,\ldots,N-1$; $\sqrt{\text{backward prediction energies}}$
$\varepsilon_{b,i}(k-1)$, $i=0,1,\ldots,N-1$; (backward prediction residuals)

New datum: $\varepsilon_{f,0}(k)=\varepsilon_{b,0}(k)=x(k)$;

For $i=0, 1, \ldots, N-1$, do:

1. Solve for $\theta_i(k)$ angle:

$$\begin{bmatrix}\sqrt{E_{b,i}(k-1)}\\ 0\end{bmatrix}=\begin{bmatrix}\cos\theta_i(k) & \sin\theta_i(k)\\ -\sin\theta_i(k) & \cos\theta_i(k)\end{bmatrix}\begin{bmatrix}\sqrt{\lambda}\,E_{b,i}(k-2)\\ \varepsilon_{b,i}(k-1)\end{bmatrix};$$

2. Update forward prediction variables:

$$\begin{bmatrix}p_{f,i}(k)\\ \varepsilon_{f,i+1}(k)\end{bmatrix}=\begin{bmatrix}\cos\theta_i(k) & \sin\theta_i(k)\\ -\sin\theta_i(k) & \cos\theta_i(k)\end{bmatrix}\begin{bmatrix}\sqrt{\lambda}\,p_{f,i}(k-1)\\ \varepsilon_{f,i}(k)\end{bmatrix};$$

3. Solve for $\varphi_i(k)$ angle:

$$\begin{bmatrix}\sqrt{E_{f,i}(k)}\\ 0\end{bmatrix}=\begin{bmatrix}\cos\varphi_i(k) & \sin\varphi_i(k)\\ -\sin\varphi_i(k) & \cos\varphi_i(k)\end{bmatrix}\begin{bmatrix}\sqrt{\lambda}\,E_{f,i}(k-1)\\ \varepsilon_{f,i}(k)\end{bmatrix};$$

4. Update backward prediction variables:

$$\begin{bmatrix}p_{b,i}(k)\\ \varepsilon_{b,i+1}(k)\end{bmatrix}=\begin{bmatrix}\cos\varphi_i(k) & \sin\varphi_i(k)\\ -\sin\varphi_i(k) & \cos\varphi_i(k)\end{bmatrix}\begin{bmatrix}\sqrt{\lambda}\,p_{b,i}(k-1)\\ \varepsilon_{b,i}(k-1)\end{bmatrix};$$

variant of the rotation-based lattice algorithm may also be developed [32], using principles of spherical trigonometry. The resulting algorithm inherits the backward consistency property of other minimal algorithms, with the resulting stable error propagation.

8.6 Conclusion

We have reviewed how standard concepts of backward consistency and conditioning intervene in the analysis of recursive least-squares algorithms, and focusing in particular how these concepts play out in QR decomposition recursive

least-squares adaptive filtering algorithms. The notion of persistence of excitation on the input sequence proves equivalent to that of a condition number remaining uniformly bounded in time.

The full QR decomposition update equations admit a reasonably direct analysis showing bounded error accumulation, owing to the orthogonal nature of the calculations combined with past data being attenuated via the forgetting factor λ. The conditions for such bounded error growth may also be explained via backward consistency: If numerical errors are indistinguishable from perturbation on the past input data, then the effects of such numerical errors must be forgotten at the same rate as the past input data. This concept proves convenient when studying fast least-squares algorithms, which tend not to admit tractable error analyses through other methods. Two fast QR decomposition algorithms were highlighted. The first is minimal in its storage requirements, and admits simple checks for backward consistency. The rotation angles of that algorithm were shown to have a direct connection to past input reconstruction procedure, which serves to validate the simple description of the set of reachable states. The QR decomposition lattice algorithm was also reviewed, as it offers a modular, pipelineable structure. Although the algorithm is not minimal and thus does not offer backward consistency in general, bounded error growth can be shown in a manner similar to that for the full QR decomposition algorithm. This represents a noted improvement over some other fast least-squares algorithms, notably certain fast transversal filters, which can exhibit explosive error growth once the backward consistency conditions are violated [15, 17, 18].

References

1. J. H. Wilkinson, Error analysis of direct methods of matrix inversion. Journal of the Association for Computing Machinery (JACM), vol. 8, no. 3, pp. 281–330 (July 1961)
2. G. W. Stewart and J. Sun, Matrix Perturbation Theory. Academic Press, San Diego, CA, USA (1990)
3. N. J. Higham, Accuracy and Stability of Numerical Algorithms. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, USA (1996)
4. G. H. Golub and J. H. Wilkinson, Note on the iterative refinement of least squares solution. Numerische Mathematik, vol. 9, pp. 139–148 (1966)
5. Å. Björck, Numerical Methods for Least Squares Problems. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, USA (1996)
6. F. Chaitin-Chatelin and V. Frayssé, Lectures on Finite Precision Computations. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, USA (1996)
7. J. H. Wilkinson, A priori error analysis of algebraic processes. Proceedings of the International Congress of Mathematicians, Izdat. MIE, pp. 629–639, Moscow, USSR (1968)
8. T. Kailath, Linear Systems. Prentice-Hall, Englewood Cliffs, NJ, USA (1980)
9. R. R. Bitmead and B. D. O. Anderson, Lyapunov techniques for the exponential stability of linear difference equations with random coefficients. IEEE Transactions on Automatic Control, vol. 25, no. 4, pp. 782–787 (August 1980)
10. P. Dewilde and A.-J. van der Veen, Time-Varying Systems and Computations. Kluwer Academic Publishers, Boston, MA, USA (1998)
11. J. M. Cioffi, The fast adaptive ROTOR's RLS algorithm. IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 38, no. 4, pp. 631–653 (April 1990)

12. I. K. Proudler, J. G. McWhirter, and T. J. Shepherd, Fast QRD-based algorithms for least squares linear prediction. Mathematics in Signal Processing II (J. G. McWhirter, ed.), Institute of Mathematics and its Applications (IMA) Conference Series, Clarendon Press, no. 26, pp. 465–488, Oxford, UK (September 1990)
13. I. K. Proudler, J. G. McWhirter, and T. J. Shepherd, Computationally efficient QR decomposition approach to least squares adaptive filtering. IEE Proceedings-Part F, vol. 138, pp. 341–353 (August 1991)
14. P. A. Regalia and M. G. Bellanger, On the duality between fast QR methods and lattice methods in least squares adaptive filtering. IEEE Transactions on Signal Processing, vol. 39, no. 4, pp. 879–891 (April 1991)
15. S. Haykin, Adaptive Filter Theory. 3rd edition Prentice-Hall, Upper Saddle River, NJ, USA (1996)
16. F. Ling, Givens rotation based least-squares lattice and related algorithms. IEEE Transactions on Signal Processing, vol. 39, no, 7, pp. 1541–1551 (July 1991)
17. P. A. Regalia, Numerical stability issues in fast least-squares adaptation algorithms. Optical Engineering, vol. 31, pp. 1144–1152 (June 1992)
18. D. T. M. Slock, The backward consistency concept and round-off error propagation dynamics in recursive least-squares algorithms. Optical Engineering, vol. 31, pp. 1153–1169 (June 1992)
19. J. R. Bunch, R. C. Le Borne, and I. K. Proudler, A conceptual framework for consistency, conditioning, and stability issues in signal processing. IEEE Transactions on Signal Processing, vol. 49, no. 9, pp. 1971–1981 (September 2001)
20. S. Ljung and L. Ljung, Error propagation properties of recursive least-squares adaptation algorithms. Automatica, vol. 21, no. 2, pp. 157–167 (March 1985)
21. A. P. Liavas and P. A. Regalia, On the numerical stability and accuracy of the conventional recursive least-squares algorithm. IEEE Transactions on Signal Processing, vol. 47, no. 1, pp. 88–96 (January 1999)
22. C. T. Mullis and R. A. Roberts, The use of second-order information in the approximation of discrete-time linear systems. IEEE Transactions on Acoustics, Speech and Signal Processing, vol. ASSP-24, no. 3, pp. 226–238 (June 1976)
23. Y. Inouye, Approximation of multivariable linear systems with impulse response and autocorrelation sequences. Automatica, vol. 19, no. 2, pp. 265–277 (May 1983)
24. A. M. King, U. B. Desai, and R. E. Skelton, A generalized approach to q-Markov covariance equivalent realizations for discrete systems. Automatica, vol. 24, no. 4, pp. 507–515 (July 1988)
25. P. A. Regalia, M. Mboup, and M. Ashari, A class of first- and second-order interpolation problems in model reduction. Archiv für Elektronik und Ubertragungstechnik, vol. 49, no. 5/6, pp. 332–343 (September 1995)
26. D. Alpay, V. Bolotnikov, and Ph. Loubaton, On tangential H_2 interpolation with second order norm constraints. Integral Equations and Operator Theory (Birkhäuser Basel), vol. 24, no. 2, pp. 156–178 (June 1996)
27. P. A. Regalia, Past input reconstruction in fast least-squares algorithms. IEEE Transactions on Signal Processing, vol. 45, no. 9, pp. 2231–2240 (September 1997)
28. T. Kailath and A. H. Sayed, Displacement structure: Theory and applications. SIAM Review, vol. 37, no. 3, pp. 297–386 (September 1995)
29. H. Lev-Ari and T. Kailath, Lattice filter parametrization and modeling of nonstationary processes. IEEE Transactions on Information Theory, vol. IT-30, no. 1, pp. 2–16 (January 1984)
30. P. A. Regalia, Numerical stability properties of a *QR*-based fast least squares algorithm. IEEE Transactions on Signal Processing, vol. 41, no. 6, pp. 2096–109 (June 1993)
31. J. Chun, T. Kailath, and H. Lev-Ari, Fast parallel algorithms for QR and triangular factorization. SIAM Journal on Scientific and Statistical Computing, vol. 8, no. 6, pp. 899–913 (November 1987)
32. F. Desbouvries and P. A. Regalia, A minimal rotation-based FRLS lattice algorithm. IEEE Transactions on Signal Processing, vol. 45, no. 5, pp. 1371–1374 (May 1997)