Conjugate Gradient Implementation of Multi-Stage Nested Wiener Filter for Reduced Dimension Processing

Guido Dietl

Diplomarbeit

Munich University of Technology
Institute for Circuit Theory and Signal Processing
Univ.-Prof. Dr.techn. Josef A. Nossek

Ausgabe des Themas:	2. Oktober 2000
Tag der mündlichen Prüfung:	21. Mai 2001
Betreuer:	Dipl.-Ing. Michael Joham

Guido Dietl
Traglbergstraße 17
93455 Traitsching–Wilting

14. Mai 2001

Contents

Acknowledgments

I would like to thank Prof. Dr.-techn. Josef A. Nossek, Professor Michael D. Zoltowski, Dr.-Ing. Wolfgang Utschick, and Dipl.-Ing. Michael Joham for supporting this work. They gave me a lot of advice and helped me developing the algorithms of this thesis. Moreover, I thank them for the opportunity to stay at Purdue University in West Lafayette, Indiana, USA.

Furthermore, I thank all the people at the Institute for Circuit Theory and Signal Processing. In particular, I thank Dipl.-Ing. Frank Dietrich and Dirk Waldhauser for the helpful discussions I had with them.

Moreover, I'd like to thank the graduate students at Purdue University, especially Samina Chowdhury, Brad Breinholt, and Bill Hillery. They helped me getting used to the new environment, so that my stay at West Lafayette was a great experience for me.

Finally, I thank my parents Gisela and Erich Dietl, my grandmother Elisabeth Gruber, my sister Ulrike Alt, and my girlfriend Anja Reimann. My parents made it possible for me to study Electrical Engineering and especially my father taught me how to like physical and mathematical problems. All of them encouraged me and helped me getting over difficult times.

1. Introduction

The *Wiener filter* [1] estimates an unknown signal $s_0[n]$ from the observation signal $\boldsymbol{y}_0[n]$ in the *Minimum Mean Square Error* (MMSE) sense exploiting only second order statistics. Thus, the Wiener filter is easy to implement and therefore used in many applications. However, the resulting filter needs the inverse of the covariance matrix of the observation. This means observations $\boldsymbol{y}_0[n]$ of high dimensionality imply high computational complexity.

The *Principal Component* (PC) *method* [2] was the first approach to reduce the complexity of the estimation problem. A pre-filter matrix composed of the eigenvectors belonging to the principal eigenvalues of the covariance matrix of the observation is applied to the observation signal to get a transformed signal of lower dimensionality. An alternative approach is the *Cross-Spectral* (CS) *metric* introduced by Goldstein et. al. [3], where the columns of the pre-filter matrix are the eigenvectors which belong to the largest CS metric. Thus, it considers not only the statistics of the observation signal but also the relation to the desired signal. More recently, Goldstein et. al. developed the *Multi-Stage Nested Wiener Filter* (MSNWF) [4] which approximates the Wiener filter in reduced space without the computation of eigenvectors. The MSNWF shows that dimensionality reduction of the observation signal based on eigenvectors is generally suboptimal. Applications like a space-time preprocessor for GPS systems [5] and an equalizer filter for a cdma2000 forward link receiver [6] showed the extraordinary performance of the MSNWF.

Honig et. al. [7] observed that the MSNWF can be seen as the solution of the *Wiener-Hopf equation* in the *Krylov subspace* of the covariance matrix of the observation and the cross-correlation vector between the observation and the desired signal. Finally, in [8] it is shown that the *Arnoldi algorithm* can be replaced by the *Lanczos algorithm* to generate the basis vectors for the Krylov subspace since the covariance matrix is Hermitian. The resulting algorithm is an order-recursive version of the MSNWF which recursively updates the filter and the *Mean Square Error* (MSE) at each step.

The contribution of this thesis is to derive the relationship between the *Conjugate Gradient* (CG) *method* and the Lanczos based implementation of the MSNWF. The CG method was originally introduced by Hestenes and Stiefel [9] in 1952 to solve a system of linear equations. It searches for an approximate solution in the Krylov subspace similar to the Lanczos based MSNWF. We transform the equations of the Lanczos based MSNWF algorithm to yield a formulation of the CG algorithm, and finally present a new implementation of the MSNWF where the weight vector and the *Mean Square Error* (MSE) between the estimated and desired signal is updated at each iteration step.

In the next chapter, we recapitulate the derivation of the Lanczos based MSNWF. Before we show the relationship between the considered algorithms in Chapter 4, we review the basics of the CG algorithm in Chapter 3. Finally, in Chapter 6, we apply the new formulation of the MSNWF algorithm of Section 4.2 as a linear equalizer to an EDGE system and compare its performance to the PC and CS method, which we briefly introduce in Chapter 5. Due to high bit error rates using correlation to estimate the covariance matrix of the observation and the cross-correlation vector between the observation and the desired signal, we derive a least squares method to estimate the statistics in Section 6.2.

2. Multi-Stage Nested Wiener Filter

2.1 Wiener Filter

One general problem in estimation theory is to regain an unknown desired signal $s_0[n] \in \mathbb{C}$ from the known observation signal $\boldsymbol{y}_0[n] \in \mathbb{C}^N$ by using a linear filter $\boldsymbol{w} \in \mathbb{C}^N$ so that the estimate $\hat{s}_0[n] = \boldsymbol{w}^{\mathrm{H}}\boldsymbol{y}_0[n]$. In this thesis, we assume that the desired and the observation signal are zero-mean and multivariate zero-mean Gaussian processes, respectively.

The error between the desired signal $s_0[n]$ and the estimate $\hat{s}_0[n]$ can be written as

$$\varepsilon_0[n] = s_0[n] - \hat{s}_0[n] = s_0[n] - \boldsymbol{w}^{\mathrm{H}}\boldsymbol{y}_0[n], \tag{2.1}$$

whose variance is the *mean square error*

$$\mathrm{MSE}_0 = \mathrm{E}\left\{|\varepsilon_0[n]|^2\right\} = \sigma_{s_0}^2 - \boldsymbol{w}^{\mathrm{H}}\boldsymbol{r}_{\boldsymbol{y}_0,s_0} - \boldsymbol{r}_{\boldsymbol{y}_0,s_0}^{\mathrm{H}}\boldsymbol{w} + \boldsymbol{w}^{\mathrm{H}}\boldsymbol{R}_{\boldsymbol{y}_0}\boldsymbol{w}, \tag{2.2}$$

where

$$\sigma_{s_0}^2 = \mathrm{E}\left\{|s_0[n]|^2\right\} \tag{2.3}$$

is the variance of the desired signal $s_0[n]$,

$$\boldsymbol{r}_{\boldsymbol{y}_0,s_0} = \mathrm{E}\left\{\boldsymbol{y}_0[n]\, s_0^*[n]\right\} \tag{2.4}$$

is the cross-correlation vector between the observation signal $\boldsymbol{y}_0[n]$ and the desired signal $s_0[n]$, and

$$\boldsymbol{R}_{\boldsymbol{y}_0} = \mathrm{E}\left\{\boldsymbol{y}_0[n]\,\boldsymbol{y}_0^{\mathrm{H}}[n]\right\} \tag{2.5}$$

is the covariance matrix of the observation $\boldsymbol{y}_0[n]$.

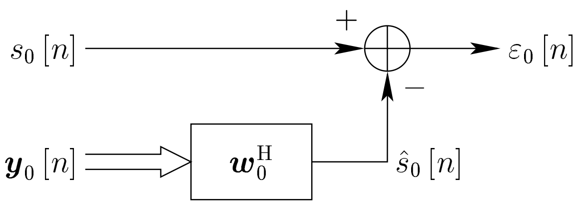

Fig. 2.1. Wiener Filter

The *Wiener filter* $\boldsymbol{w}_0$ whose design structure is shown in Figure 2.1 minimizes the mean square error MSE_0. Thus, we get the design criterion

$$\boldsymbol{w}_0 = \arg\min_{\boldsymbol{w}} \mathrm{MSE}_0, \tag{2.6}$$

which leads to the *Wiener-Hopf equation*

$$\boxed{\boldsymbol{R}_{\boldsymbol{y}_0}\boldsymbol{w}_0 = \boldsymbol{r}_{\boldsymbol{y}_0,s_0}.} \tag{2.7}$$

Its solution, the Wiener filter

$$\boldsymbol{w}_0 = \boldsymbol{R}_{\boldsymbol{y}_0}^{-1}\boldsymbol{r}_{\boldsymbol{y}_0,s_0}, \tag{2.8}$$

achieves the *minimum mean square error*

$$\boxed{\mathrm{MMSE}_0 = \sigma_{s_0}^2 - \boldsymbol{r}_{\boldsymbol{y}_0,s_0}^{\mathrm{H}}\boldsymbol{R}_{\boldsymbol{y}_0}^{-1}\boldsymbol{r}_{\boldsymbol{y}_0,s_0}.} \tag{2.9}$$

2.2 Multi-Stage Nested Wiener Filter (MSNWF)

Goldstein et. al. [4] developed the *Multi-Stage Nested Wiener Filter* (MSNWF) which approximates the solution of the Wiener-Hopf equation (cf. Equation 2.7) without computing the inverse of the covariance matrix. For its derivation, we have to introduce a pre-filter matrix $\boldsymbol{T}_1 \in \mathbb{C}^{N\times N}$ as shown in Figure 2.2 so that the transformed observation signal $\boldsymbol{z}_1[n]$ becomes

$$\boldsymbol{z}_1[n] = \boldsymbol{T}_1^{\mathrm{H}}\boldsymbol{y}_0[n]. \tag{2.10}$$

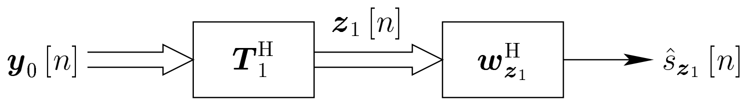

Fig. 2.2. Wiener Filter with Pre-Filter

The new Wiener filter $\boldsymbol{w}_{\boldsymbol{z}_1}$ estimates the desired signal $s_0[n]$ from the transformed observation signal $\boldsymbol{z}_1[n]$, i. e.

$$\boldsymbol{w}_{\boldsymbol{z}_1} = \boldsymbol{R}_{\boldsymbol{z}_1}^{-1}\boldsymbol{r}_{\boldsymbol{z}_1,s_0}, \tag{2.11}$$

where

$$\boldsymbol{R}_{\boldsymbol{z}_1} = \mathrm{E}\left\{\boldsymbol{z}_1[n]\,\boldsymbol{z}_1^{\mathrm{H}}[n]\right\} = \boldsymbol{T}_1^{\mathrm{H}}\boldsymbol{R}_{\boldsymbol{y}_0}\boldsymbol{T}_1 \tag{2.12}$$

is the covariance matrix of the transformed observation signal $\boldsymbol{z}_1[n]$ and

$$\boldsymbol{r}_{\boldsymbol{z}_1,s_0} = \mathrm{E}\left\{\boldsymbol{z}_1[n]\,s_0^*[n]\right\} = \boldsymbol{T}_1^{\mathrm{H}}\boldsymbol{r}_{\boldsymbol{y}_0,s_0} \tag{2.13}$$

is the cross-correlation vector between the transformed observation $\boldsymbol{z}_1[n]$ and the desired signal $s_0[n]$.

Theorem 2.1 *If the pre-filter matrix $\boldsymbol{T}_1$ has full rank, the output $\hat{s}_{\boldsymbol{z}_1}[n] = \boldsymbol{w}_{\boldsymbol{z}_1}^{\mathrm{H}}\boldsymbol{z}_1[n]$ of the new Wiener filter $\boldsymbol{w}_{\boldsymbol{z}_1}$ is the same as the output $\hat{s}_0[n]$ of the Wiener filter $\boldsymbol{w}_0$ without pre-filtering. Thus, the minimum mean square errors of both filters are the same.*

Proof. Replacing $\boldsymbol{R}_{\boldsymbol{z}_1}$ and $\boldsymbol{r}_{\boldsymbol{z}_1,s_0}$ in Equation (2.11) by Equation (2.12) and (2.13), respectively, yields

$$\boldsymbol{w}_{\boldsymbol{z}_1} = \boldsymbol{T}_1^{-1} \boldsymbol{R}_{\boldsymbol{y}_0}^{-1} \boldsymbol{T}_1^{\mathrm{H},-1} \boldsymbol{T}_1^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0}. \tag{2.14}$$

Hence, the output of the Wiener filter $\boldsymbol{w}_{\boldsymbol{z}_1}$ may be written as

$$\hat{s}_{\boldsymbol{z}_1}[n] = \boldsymbol{w}_{\boldsymbol{z}_1}^{\mathrm{H}} \boldsymbol{z}_1[n] = \boldsymbol{r}_{\boldsymbol{y}_0,s_0}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0}^{-1} \boldsymbol{T}_1^{\mathrm{H},-1} \boldsymbol{T}_1^{\mathrm{H}} \boldsymbol{y}_0[n] = \boldsymbol{w}_0^{\mathrm{H}} \boldsymbol{y}_0[n] = \hat{s}_0[n],$$

and the first part of Theorem 2.1 is proven. In order to complete the proof, use Equation (2.14) to compute the minimum mean square error (cf. Equation 2.9) as follows

$$\mathrm{MMSE}_{\boldsymbol{z}_1} = \sigma_{s_0}^2 - \boldsymbol{r}_{\boldsymbol{z}_1,s_0}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{z}_1}^{-1} \boldsymbol{r}_{\boldsymbol{z}_1,s_0} = \sigma_{s_0}^2 - \boldsymbol{r}_{\boldsymbol{y}_0,s_0}^{\mathrm{H}} \boldsymbol{T}_1 \boldsymbol{T}_1^{-1} \boldsymbol{R}_{\boldsymbol{y}_0}^{-1} \boldsymbol{T}_1^{\mathrm{H},-1} \boldsymbol{T}_1^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} = \mathrm{MMSE}_0.$$

□

In order to end up with the MSNWF, we restrict ourselves on the special pre-filter matrix $\boldsymbol{T}_1$ defined as

$$\boldsymbol{T}_1 = \begin{bmatrix} \boldsymbol{m}_1 & \boldsymbol{B}_1 \end{bmatrix}. \tag{2.15}$$

Thus, the transformed observation signal $\boldsymbol{z}_1[n]$ becomes

$$\boldsymbol{z}_1[n] = \begin{bmatrix} \boldsymbol{m}_1^{\mathrm{H}} \boldsymbol{y}_0[n] \\ \boldsymbol{B}_1^{\mathrm{H}} \boldsymbol{y}_0[n] \end{bmatrix} =: \begin{bmatrix} s_1[n] \\ \boldsymbol{y}_1[n] \end{bmatrix}. \tag{2.16}$$

We choose the vector $\boldsymbol{m}_1 \in \mathbb{C}^N$ to have unit norm, i. e. $\|\boldsymbol{m}_1\|_2 = 1$, and the real part of the correlation between the filtered signal $s_1[n] = \boldsymbol{m}_1^{\mathrm{H}} \boldsymbol{y}_0[n] \in \mathbb{C}$ and the desired signal $s_0[n]$ is maximized by $\boldsymbol{m}_1$, hence,

$$\boldsymbol{m}_1 = \arg\max_{\boldsymbol{m}} \mathrm{E}\left\{\mathrm{Re}\left(s_1[n]\, s_0^*[n]\right)\right\} \quad \text{s.t.} \quad \boldsymbol{m}^{\mathrm{H}} \boldsymbol{m} = 1. \tag{2.17}$$

By replacing $s_1[n]$ as defined in Equation (2.16), we get

$$\boldsymbol{m}_1 = \arg\max_{\boldsymbol{m}} \frac{1}{2}\left(\boldsymbol{m}^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} + \boldsymbol{r}_{\boldsymbol{y}_0,s_0}^{\mathrm{H}} \boldsymbol{m}\right) \quad \text{s.t.} \quad \boldsymbol{m}^{\mathrm{H}} \boldsymbol{m} = 1, \tag{2.18}$$

which leads to the normalized matched filter

$$\boldsymbol{m}_1 = \frac{\boldsymbol{r}_{\boldsymbol{y}_0,s_0}}{\left\|\boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right\|_2}. \tag{2.19}$$

Note that the matched filter defined by the optimization criterion in Equation (2.17) does not suppress any interference.

The columns of the matrix $\boldsymbol{B}_1 \in \mathbb{C}^{N\times(N-1)}$ in Equation (2.15) (in the following called the blocking matrix) are chosen to be orthogonal to the matched filter $\boldsymbol{m}_1$. Thus, the blocking matrix $\boldsymbol{B}_1$ satisfies

$$\boldsymbol{B}_1^{\mathrm{H}} \boldsymbol{m}_1 = \boldsymbol{0} \quad \Leftrightarrow \quad \mathrm{span}\left(\boldsymbol{B}_1\right) = \mathrm{null}\left(\boldsymbol{m}_1^{\mathrm{H}}\right). \tag{2.20}$$

The obtained signal $\boldsymbol{y}_1[n] = \boldsymbol{B}_1^{\mathrm{H}} \boldsymbol{y}_0[n] \in \mathbb{C}^{N-1}$ is uncorrelated with the desired signal $s_0[n]$ but includes interference.

With the special pre-filter $\boldsymbol{T}_1$ and the transformed observation $\boldsymbol{z}_1[n]$ defined by Equations (2.15) and (2.16), respectively, the cross-correlation vector $\boldsymbol{r}_{\boldsymbol{z}_1,s_0}$ between $\boldsymbol{z}_1[n]$ and $s_0[n]$ and the covariance matrix $\boldsymbol{R}_{\boldsymbol{z}_1}$ of $\boldsymbol{z}_1[n]$ (cf. Equations 2.13 and 2.12) may be written as

$$\boldsymbol{r}_{\boldsymbol{z}_1,s_0} = \begin{bmatrix} \boldsymbol{m}_1^{\mathrm{H}} \\ \boldsymbol{B}_1^{\mathrm{H}} \end{bmatrix} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} = \left\| \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right\|_2 \boldsymbol{e}_1, \tag{2.21}$$

$$\boldsymbol{R}_{\boldsymbol{z}_1} = \mathrm{E}\left\{ \begin{bmatrix} s_1[n] \\ \boldsymbol{y}_1[n] \end{bmatrix} \begin{bmatrix} s_1^*[n] & \boldsymbol{y}_1^{\mathrm{H}}[n] \end{bmatrix} \right\} = \begin{bmatrix} \sigma_{s_1}^2 & \boldsymbol{r}_{\boldsymbol{y}_1,s_1}^{\mathrm{H}} \\ \boldsymbol{r}_{\boldsymbol{y}_1,s_1} & \boldsymbol{R}_{\boldsymbol{y}_1} \end{bmatrix}, \tag{2.22}$$

with the variance of $s_1[n]$

$$\sigma_{s_1}^2 = \mathrm{E}\left\{ |s_1[n]|^2 \right\} = \boldsymbol{m}_1^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{m}_1, \tag{2.23}$$

the cross-correlation vector between $\boldsymbol{y}_1[n]$ and $s_1[n]$

$$\boldsymbol{r}_{\boldsymbol{y}_1,s_1} = \mathrm{E}\left\{ \boldsymbol{y}_1[n] s_1^*[n] \right\} = \boldsymbol{B}_1^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{m}_1, \tag{2.24}$$

the covariance matrix of $\boldsymbol{y}_1[n]$

$$\boldsymbol{R}_{\boldsymbol{y}_1} = \mathrm{E}\left\{ \boldsymbol{y}_1[n] \boldsymbol{y}_1^{\mathrm{H}}[n] \right\} = \boldsymbol{B}_1^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{B}_1, \tag{2.25}$$

and $\boldsymbol{e}_i$ denotes a unit norm vector with a one at the i-th position. Because of the sparse structure of the cross-correlation vector $\boldsymbol{r}_{\boldsymbol{z}_1,s_0}$, only the first column of the inverse of the covariance matrix $\boldsymbol{R}_{\boldsymbol{z}_1}$ has to be computed. Replacing $\boldsymbol{R}_{\boldsymbol{z}_1}$ and $\boldsymbol{r}_{\boldsymbol{z}_1,s_0}$ in Equation (2.11) by Equation (2.21) and (2.22), respectively, and using the inversion lemma for partitioned matrices (e. g. [1]) leads to

$$\boldsymbol{w}_{\boldsymbol{z}_1} = \boldsymbol{R}_{\boldsymbol{z}_1}^{-1} \boldsymbol{r}_{\boldsymbol{z}_1,s_0} = \alpha_1 \begin{bmatrix} 1 \\ -\boldsymbol{R}_{\boldsymbol{y}_1}^{-1} \boldsymbol{r}_{\boldsymbol{y}_1,s_1} \end{bmatrix} = \alpha_1 \begin{bmatrix} 1 \\ -\boldsymbol{w}_1 \end{bmatrix}, \tag{2.26}$$

with the factor

$$\alpha_1 = \left\| \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right\|_2 \left(\sigma_{s_1}^2 - \boldsymbol{r}_{\boldsymbol{y}_1,s_1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_1}^{-1} \boldsymbol{r}_{\boldsymbol{y}_1,s_1} \right)^{-1}, \tag{2.27}$$

and the reduced dimension Wiener filter

$$\boldsymbol{w}_1 = \boldsymbol{R}_{\boldsymbol{y}_1}^{-1} \boldsymbol{r}_{\boldsymbol{y}_1,s_1} \in \mathbb{C}^{N-1}. \tag{2.28}$$

Applying the new Wiener filter $\boldsymbol{w}_{\boldsymbol{z}_1}$ to the transformed observation $\boldsymbol{z}_1[n]$ yields finally the estimate

$$\hat{s}_0[n] = \boldsymbol{w}_{\boldsymbol{z}_1}^{\mathrm{H}} \begin{bmatrix} s_1[n] \\ \boldsymbol{y}_1[n] \end{bmatrix} = \alpha_1 \begin{bmatrix} 1 & -\boldsymbol{w}_1^{\mathrm{H}} \end{bmatrix} \begin{bmatrix} s_1[n] \\ \boldsymbol{y}_1[n] \end{bmatrix} = \alpha_1 \left(s_1[n] - \boldsymbol{w}_1^{\mathrm{H}} \boldsymbol{y}_1[n] \right). \tag{2.29}$$

Figure 2.3 shows the resulting block diagram, which we derived from the Wiener filter $\boldsymbol{w}_0$ shown in Figure 2.1. The reduced Wiener filter $\boldsymbol{w}_1$ can be replaced in exactly the same way

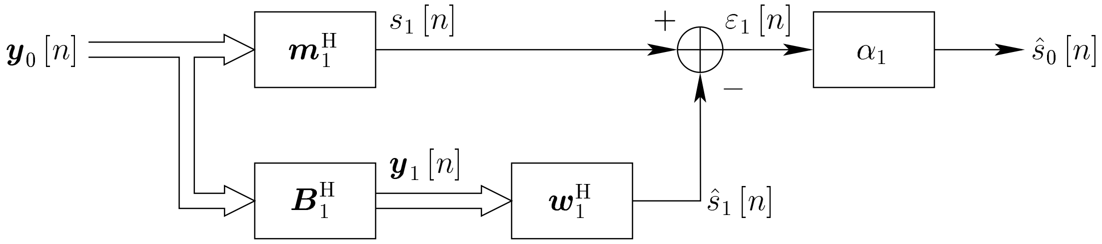

Fig. 2.3. MSNWF after First Step

as the original Wiener filter $\boldsymbol{w}_0$. This procedure can be repeated until the last Wiener filter w_{N-1} is a scalar.

The resulting filter structure, the full-rank MSNWF, is shown in Figure 2.4 and can also be seen as a filter bank [8] as depicted in Figure 2.5. Note that both structures have the same behavior as the Wiener filter $\boldsymbol{w}_0$. In the sequel of this section, let $i \in \{1, 2, \ldots, N\}$. It follows from the derivations above that the pre-filter vector $\boldsymbol{t}_i \in \mathbb{C}^N$ may be written as

$$\boxed{\boldsymbol{t}_i = \left(\prod_{k=1}^{i-1} \boldsymbol{B}_k \right) \boldsymbol{m}_i.} \tag{2.30}$$

$\boldsymbol{m}_i \in \mathbb{C}^{N-(i-1)}$ is the normalized matched filter

$$\boxed{\boldsymbol{m}_i = \frac{\boldsymbol{r}_{\boldsymbol{y}_{i-1},s_{i-1}}}{\left\| \boldsymbol{r}_{\boldsymbol{y}_{i-1},s_{i-1}} \right\|_2},} \tag{2.31}$$

which maximizes the real part of the cross-correlation between the new desired signal $s_i[n] = \boldsymbol{m}_i^{\mathrm{H}} \boldsymbol{y}_{i-1}[n] \in \mathbb{C}$ at stage i and the desired signal $s_{i-1}[n]$ at the previous stage $i-1$ of the MSNWF equivalent to $\boldsymbol{m}_1$ in Equation (2.19). $\boldsymbol{r}_{\boldsymbol{y}_{i-1},s_{i-1}}$ denotes the cross-correlation vector between $\boldsymbol{y}_{i-1}[n]$ and $s_{i-1}[n]$, i. e.

$$\begin{aligned} \boldsymbol{r}_{\boldsymbol{y}_{i-1},s_{i-1}} &= \mathrm{E}\left\{ \boldsymbol{y}_{i-1}[n]\, s_{i-1}^*[n] \right\} = \boldsymbol{B}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_{i-2}} \boldsymbol{m}_{i-1} \\ &= \left(\prod_{k=i-1}^{1} \boldsymbol{B}_k^{\mathrm{H}} \right) \boldsymbol{R}_{\boldsymbol{y}_0} \left(\prod_{k=1}^{i-2} \boldsymbol{B}_k \right) \boldsymbol{m}_{i-1}. \end{aligned} \tag{2.32}$$

$\boldsymbol{B}_k \in \mathbb{C}^{(N-(k-1)) \times (N-k)}$, $k \in \{1, 2, \ldots, N-1\}$, is the blocking matrix satisfying the following equation:

$$\boxed{\boldsymbol{B}_k^{\mathrm{H}} \boldsymbol{m}_k = 0 \quad \Leftrightarrow \quad \mathrm{span}\left(\boldsymbol{B}_k\right) = \mathrm{null}\left(\boldsymbol{m}_k^{\mathrm{H}}\right).} \tag{2.33}$$

Thus, similar to $\boldsymbol{B}_1$ above, the new observation signal $\boldsymbol{y}_k[n] = \boldsymbol{B}_k^{\mathrm{H}} \boldsymbol{y}_{k-1}[n] \in \mathbb{C}^{N-k}$ at stage k is uncorrelated with the desired signal $s_{k-1}[n]$ at stage $k-1$. By the way, with the knowledge of $\boldsymbol{B}_k$ and $\boldsymbol{m}_i$ in Equation (2.30), it can easily be shown that the filters $\boldsymbol{t}_2, \boldsymbol{t}_3, \ldots, \boldsymbol{t}_N$ are orthogonal to the filter $\boldsymbol{t}_1$, but the set of vectors $\{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_N\}$ is not necessarily an orthogonal basis of $\mathbb{C}^N$. Finally, α_i is a scalar Wiener filter [8] which estimates the desired signal $s_{i-1}[n]$ from the error $\varepsilon_i[n]$ of the new desired signal $s_i[n]$ to its estimate

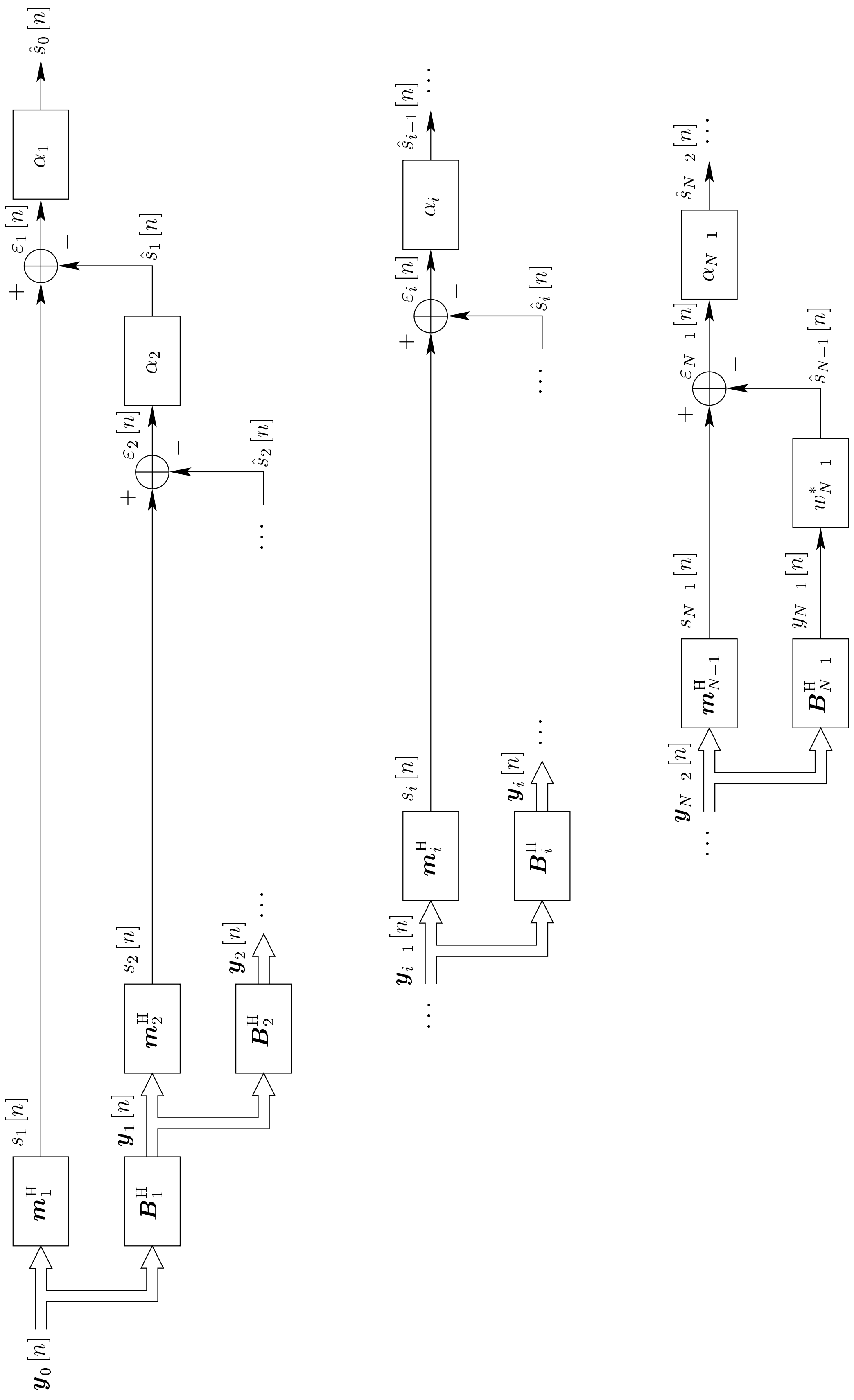

Fig. 2.4. Full-Rank MSNWF

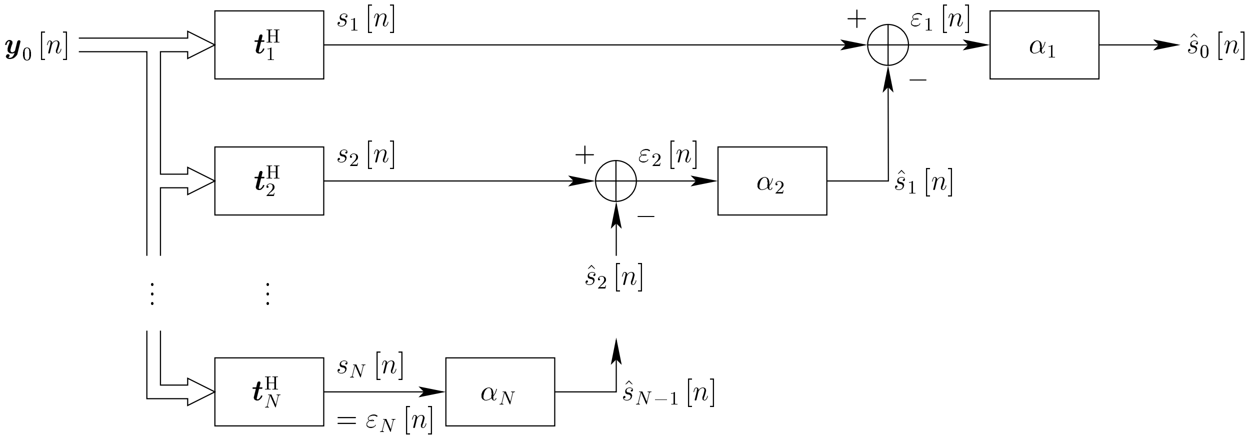

Fig. 2.5. MSNWF as a Filter Bank

$\hat{s}_i[n]$, i. e. $\varepsilon_i[n] = s_i[n] - \hat{s}_i[n]$ for $i < N$, and $\varepsilon_N[n] = s_N[n]$ for $i = N$, respectively. Hence, it holds

$$\boxed{\alpha_i = \sigma_{\varepsilon_i}^{-2} r_{\varepsilon_i, s_{i-1}},} \tag{2.34}$$

with the variance of $\varepsilon_i[n]$ (cf. Equation 2.9)

$$\sigma_{\varepsilon_i}^2 = \mathrm{E}\left\{|\varepsilon_i[n]|^2\right\} = \begin{cases} \sigma_{s_i}^2 - \boldsymbol{r}_{\boldsymbol{y}_i,s_i}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_i}^{-1} \boldsymbol{r}_{\boldsymbol{y}_i,s_i} & \text{for } i < N, \\ \sigma_{s_N}^2, & \text{for } i = N, \end{cases} \tag{2.35}$$

and the cross-correlation between $\varepsilon_i[n]$ and $s_{i-1}[n]$

$$\begin{aligned} r_{\varepsilon_i, s_{i-1}} &= \mathrm{E}\left\{\varepsilon_i[n]\, s_{i-1}^*[n]\right\} \\ &= \begin{cases} \boldsymbol{m}_i^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_{i-1},s_{i-1}} - \boldsymbol{w}_i^{\mathrm{H}} \boldsymbol{B}_i^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_{i-1},s_{i-1}} = \left\|\boldsymbol{r}_{\boldsymbol{y}_{i-1},s_{i-1}}\right\|_2 & \text{for } i < N, \\ m_N^* r_{y_{N-1},s_{N-1}} = \left|r_{y_{N-1},s_{N-1}}\right| & \text{for } i = N. \end{cases} \end{aligned} \tag{2.36}$$

The last equality holds because of Equations (2.31) and (2.33). $\boldsymbol{R}_{\boldsymbol{y}_i}$, $i < N$, is the covariance matrix of $\boldsymbol{y}_i[n]$ which may be written as

$$\boldsymbol{R}_{\boldsymbol{y}_i} = \mathrm{E}\left\{\boldsymbol{y}_i[n]\, \boldsymbol{y}_i^{\mathrm{H}}[n]\right\} = \boldsymbol{B}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_{i-1}} \boldsymbol{B}_i = \left(\prod_{k=i}^{1} \boldsymbol{B}_k^{\mathrm{H}}\right) \boldsymbol{R}_{\boldsymbol{y}_0} \left(\prod_{k=1}^{i} \boldsymbol{B}_k\right), \tag{2.37}$$

and

$$\boldsymbol{w}_i = \begin{cases} \boldsymbol{R}_{\boldsymbol{y}_i}^{-1} \boldsymbol{r}_{\boldsymbol{y}_i,s_i} & \text{for } i < N-1, \\ \sigma_{y_{N-1}}^{-2} r_{y_{N-1},s_{N-1}} = m_N \alpha_N & \text{for } i = N-1, \end{cases} \tag{2.38}$$

is the Wiener filter which estimates $s_i[n]$ from $\boldsymbol{y}_i[n]$. Note that $\alpha_i \in \mathbb{R}$ for all i because $\sigma_{\varepsilon_i}^2 \in \mathbb{R}$ as well as $r_{\varepsilon_i,s_{i-1}} \in \mathbb{R}$.

For later derivations, we need the characteristic of the MSNWF that the pre-filtered observation vector $\boldsymbol{s}[n]$, defined as

$$\boldsymbol{s}[n] = \begin{bmatrix} s_1[n] & s_2[n] & \cdots & s_N[n] \end{bmatrix}^{\mathrm{T}}, \tag{2.39}$$

has a tri-diagonal covariance matrix $\boldsymbol{R}_{\boldsymbol{s}}$ [4, 8]. This is because the matched filters $\boldsymbol{m}_{i-1}$ and $\boldsymbol{m}_i$ of two arbitrary adjoining stages of the MSNWF are designed to maximize the real part of the correlation between $s_{i-1}[n]$ and $s_{i-2}[n]$, and between $s_i[n]$ and $s_{i-1}[n]$, respectively, whereas the blocking matrix $\boldsymbol{B}_{i-1}$ ensures that $s_i[n]$ is uncorrelated with $s_{i-2}[n]$.

In order to obtain a reduced rank MSNWF, we only use the first $D < N$ basis vectors $\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_D$ to build the pre-filter matrix

$$\boldsymbol{T}^{(D)} = \begin{bmatrix} \boldsymbol{t}_1 & \boldsymbol{t}_2 & \cdots & \boldsymbol{t}_D \end{bmatrix} \in \mathbb{C}^{N \times D}, \tag{2.40}$$

which yields the observation vector $\boldsymbol{s}^{(D)}[n] \in \mathbb{C}^D$ of reduced length

$$\boldsymbol{s}^{(D)}[n] = \boldsymbol{T}^{(D),\mathrm{H}} \boldsymbol{y}_0[n]. \tag{2.41}$$

This is equivalent to stopping the replacements, we mentioned in the derivations above, after $D-1$ steps where the remaining Wiener filter $\boldsymbol{w}_{D-1}$ is approximated by the matched filter $\boldsymbol{m}_D$. Next, we have to compute the Wiener filter of length D, $\boldsymbol{w}_{\boldsymbol{s}}^{(D)}$, which estimates $s_0[n]$ from $\boldsymbol{s}^{(D)}[n]$, i. e.

$$\boldsymbol{w}_{\boldsymbol{s}}^{(D)} = \boldsymbol{R}_{\boldsymbol{s}}^{(D),-1} \boldsymbol{r}_{\boldsymbol{s},s_0}^{(D)} = \left(\boldsymbol{T}^{(D),\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{T}^{(D)}\right)^{-1} \boldsymbol{T}^{(D),\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0}, \tag{2.42}$$

in order to get finally an approximation of the Wiener filter $\boldsymbol{w}_0$, the rank D MSNWF

$$\boxed{\boldsymbol{w}_0^{(D)} = \boldsymbol{T}^{(D)} \boldsymbol{w}_{\boldsymbol{s}}^{(D)} = \boldsymbol{T}^{(D)} \left(\boldsymbol{T}^{(D),\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{T}^{(D)}\right)^{-1} \boldsymbol{T}^{(D),\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0},} \tag{2.43}$$

which yields the mean square error

$$\boxed{\mathrm{MSE}^{(D)} = \sigma_{s_0}^2 - \boldsymbol{r}_{\boldsymbol{y}_0,s_0}^{\mathrm{H}} \boldsymbol{T}^{(D)} \left(\boldsymbol{T}^{(D),\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{T}^{(D)}\right)^{-1} \boldsymbol{T}^{(D),\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0}.} \tag{2.44}$$

The covariance matrix $\boldsymbol{R}_{\boldsymbol{s}}^{(D)}$ and the cross-correlation vector $\boldsymbol{r}_{\boldsymbol{s},s_0}^{(D)}$ are defined in Section 2.3.

2.3 Lanczos Based MSNWF

In the following of this thesis, let the singular values of the blocking matrix $\boldsymbol{B}_i$, $i \in \{1, 2, \ldots, N-1\}$, be $\sigma_k = 0$ for $k > N - i$ and otherwise be unequal to 0 and the same, i. e. $\sigma_1 = \ldots = \sigma_{N-i} = \sigma$. In addition, we assume that the pre-filters have unit norm, i. e. $\|\boldsymbol{t}_i\|_2 = 1$ for all $i \in \{1, 2, \ldots, N\}$. Thus, the set of filters $\{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_N\}$ in Figure 2.5 is an orthonormal basis of $\mathbb{C}^N$.

In order to get a simpler formula for the computation of these basis vectors as by Equation (2.30), recall the optimization problem for the first step of the MSNWF development given by Equation (2.17). Applying it to an arbitrary stage $i \in \{1, 2, \ldots, N\}$ of the MSNWF means maximizing the real part of the cross-correlation between the signals $s_i[n] = \boldsymbol{t}_i^{\mathrm{H}} \boldsymbol{y}_0[n]$ and $s_{i-1}[n] = \boldsymbol{t}_{i-1}^{\mathrm{H}} \boldsymbol{y}_0[n]$, i. e.

$$\begin{aligned} \boldsymbol{t}_i &= \arg\max_{\boldsymbol{t}} \mathrm{E}\left\{\mathrm{Re}\left(s_i[n] s_{i-1}^*[n]\right)\right\} \quad \text{or} \\ \boldsymbol{t}_i &= \arg\max_{\boldsymbol{t}} \frac{1}{2}\left(\boldsymbol{t}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} + \boldsymbol{t}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}\right) \quad \text{s.t.:} \quad \boldsymbol{t}^{\mathrm{H}} \boldsymbol{t} = 1 \quad \text{and} \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \boldsymbol{t}^{\mathrm{H}} \boldsymbol{t}_k = 0, \quad k \in \{1, 2, \ldots, i-1\}. \end{aligned} \tag{2.45}$$

We derive the solution of this optimization problem by using the Lagrange function

$$L(\boldsymbol{t}, \mu_1, \mu_2, \ldots, \mu_i) = \frac{1}{2}\left(\boldsymbol{t}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} + \boldsymbol{t}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}\right) - \sum_{k=1}^{i-1} \mu_k \boldsymbol{t}^{\mathrm{H}} \boldsymbol{t}_k - \mu_i \left(\boldsymbol{t}^{\mathrm{H}} \boldsymbol{t} - 1\right). \tag{2.46}$$

Therefore, solving the equation

$$\frac{\partial L(\boldsymbol{t}, \mu_1, \mu_2, \ldots, \mu_i)}{\partial \boldsymbol{t}^*} = \frac{1}{2} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} - \sum_{k=1}^{i-1} \mu_k \boldsymbol{t}_k - \mu_i \boldsymbol{t} = \boldsymbol{0}, \tag{2.47}$$

yields the argument $\boldsymbol{t}$ which optimizes $L(\boldsymbol{t}, \mu_1, \mu_2, \ldots, \mu_i)$. Choose the Lagrange multipliers μ_k for $k \in \{1, 2, \ldots, i-1\}$ as

$$\mu_k = \frac{1}{2} \boldsymbol{t}_k^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1}, \tag{2.48}$$

in order to fulfill the constraint that $\boldsymbol{t}$ is orthogonal to $\boldsymbol{t}_k$ for the given k. Besides, remember that $\boldsymbol{t}_k$ are mutually orthonormal. Finally, we end up with

$$\boldsymbol{t} = \frac{1}{2\mu_i}\left(\mathbf{1}_N - \sum_{k=1}^{i-1} \boldsymbol{t}_k \boldsymbol{t}_k^{\mathrm{H}}\right) \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} = \frac{1}{2\mu_i}\left(\prod_{k=1}^{i-1} \boldsymbol{P}_k\right) \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1}, \tag{2.49}$$

if we substitute the sum with a product of projection matrices defined by

$$\boldsymbol{P}_k = \mathbf{1}_N - \boldsymbol{t}_k \boldsymbol{t}_k^{\mathrm{H}}. \tag{2.50}$$

$\boldsymbol{P}_k$, $k \in \{1, 2, \ldots, N-1\}$, is the projector onto the space orthogonal to $\boldsymbol{t}_k$ and $\mathbf{1}_N$ denotes the $N \times N$ identity matrix. In order to finish the derivation, we have to choose μ_i so that the second constraint, i. e. $\boldsymbol{t}^{\mathrm{H}} \boldsymbol{t} = 1$, holds. This yields the result of the optimization

$$\boxed{\boldsymbol{t}_i = \frac{\left(\prod_{k=1}^{i-1} \boldsymbol{P}_k\right) \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1}}{\left\|\left(\prod_{k=1}^{i-1} \boldsymbol{P}_k\right) \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1}\right\|_2}} \tag{2.51}$$

Note that the recursion formula in Equation (2.51) is the well-known *Arnoldi algorithm* [10, 11]. Hence, if the recursion is initialized with $\boldsymbol{t}_1 = \boldsymbol{m}_1$ and stopped after $D-1$ steps, the produced set of vectors $\{\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_D\}$ is an orthonormal basis of the D-dimensional *Krylov subspace*

$$\mathcal{K}^{(D)}\left(\boldsymbol{R}_{\boldsymbol{y}_0}, \boldsymbol{r}_{\boldsymbol{y}_0, s_0}\right) = \operatorname{span}\left(\begin{bmatrix} \boldsymbol{r}_{\boldsymbol{y}_0, s_0} & \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{r}_{\boldsymbol{y}_0, s_0} & \cdots & \boldsymbol{R}_{\boldsymbol{y}_0}^{D-1} \boldsymbol{r}_{\boldsymbol{y}_0, s_0} \end{bmatrix}\right). \tag{2.52}$$

Consequently, the rank D MSNWF where the filters $\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_D$ are mutually orthogonal, is equivalent [7, 8] to solving the Wiener-Hopf equation in $\mathcal{K}^{(D)}\left(\boldsymbol{R}_{\boldsymbol{y}_0}, \boldsymbol{r}_{\boldsymbol{y}_0, s_0}\right)$. Finally, since the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$ is Hermitian, the basis may also be computed by the *Lanczos algorithm*

$$\boxed{\boldsymbol{t}_i = \frac{\boldsymbol{P}_{i-1} \boldsymbol{P}_{i-2} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1}}{\left\|\boldsymbol{P}_{i-1} \boldsymbol{P}_{i-2} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1}\right\|_2} = \frac{\boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} - \boldsymbol{t}_{i-2}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} \boldsymbol{t}_{i-2} - \boldsymbol{t}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} \boldsymbol{t}_{i-1}}{\left\|\boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} - \boldsymbol{t}_{i-2}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} \boldsymbol{t}_{i-2} - \boldsymbol{t}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} \boldsymbol{t}_{i-1}\right\|_2},} \tag{2.53}$$

which has much less computational complexity and is therefore used in the algorithm presented at the end of this section.

Next, remember that the rank D MSNWF is $\boldsymbol{w}_0^{(D)} = \boldsymbol{T}^{(D)} \boldsymbol{R}_{\boldsymbol{s}}^{(D),-1} \boldsymbol{r}_{\boldsymbol{s},s_0}^{(D)}$ (cf. Equations 2.42 and 2.43). The tri-diagonal covariance matrix of the pre-filtered observation $\boldsymbol{s}[n]$, $\boldsymbol{R}_{\boldsymbol{s}}^{(D)} \in \mathbb{R}^{D\times D}$, and the cross-correlation vector between $\boldsymbol{s}[n]$ and the desired signal $s_0[n]$, $\boldsymbol{r}_{\boldsymbol{s},s_0}^{(D)} \in \mathbb{R}^D$, may be written as

$$\boldsymbol{R}_{\boldsymbol{s}}^{(D)} = \boldsymbol{T}^{(D),\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{T}^{(D)} = \left[\begin{array}{cc|c} \boldsymbol{T}^{(D-1),\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{T}^{(D-1)} & & \begin{matrix} \boldsymbol{0} \\ r_{D-1,D} \end{matrix} \\ \hline \boldsymbol{0}^{\mathrm{T}} & r_{D-1,D} & r_{D,D} \end{array}\right], \tag{2.54}$$

$$\boldsymbol{r}_{\boldsymbol{s},s_0}^{(D)} = \boldsymbol{T}^{(D)} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} = \begin{bmatrix} \left\| \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right\|_2 \\ \boldsymbol{0} \end{bmatrix}. \tag{2.55}$$

The new entries of $\boldsymbol{R}_{\boldsymbol{s}}^{(D)}$ are simply

$$r_{D-1,D} = \boldsymbol{t}_{D-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_D \quad \text{and} \quad r_{D,D} = \boldsymbol{t}_D^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_D. \tag{2.56}$$

The fact that $\boldsymbol{R}_{\boldsymbol{s}}^{(D)}$ is real-valued, and therefore $\boldsymbol{w}_{\boldsymbol{s}}^{(D)} \in \mathbb{R}^D$ can easily be derived. The covariance matrix is real-valued if its entries $r_{i-1,i} = \boldsymbol{t}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_i$ and $r_{i,i} = \boldsymbol{t}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_i$ (cf. Equation 2.56) are real numbers for all $i \in \{1, 2, \ldots, D\}$. It holds

$$r_{i,i}^* = \boldsymbol{t}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0}^{\mathrm{H}} \boldsymbol{t}_i = \boldsymbol{t}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_i = r_{i,i}, \tag{2.57}$$

because $\boldsymbol{R}_{\boldsymbol{y}_0}$ is Hermitian. Thus, $r_{i,i} \in \mathbb{R}$. In order to show that $r_{i-1,i} \in \mathbb{R}$ use Equations (2.30), (2.37), (2.32) and (2.31). It follows

$$\begin{aligned} r_{i-1,i} = \boldsymbol{t}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_i &= \boldsymbol{m}_{i-1}^{\mathrm{H}} \left(\prod_{k=i-2}^{1} \boldsymbol{B}_k^{\mathrm{H}} \right) \boldsymbol{R}_{\boldsymbol{y}_0} \left(\prod_{k=1}^{i-1} \boldsymbol{B}_k \right) \boldsymbol{m}_i \\ &= \boldsymbol{m}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_{i-2}} \boldsymbol{B}_{i-1} \boldsymbol{m}_i = \boldsymbol{r}_{\boldsymbol{y}_{i-1},s_{i-1}}^{\mathrm{H}} \boldsymbol{m}_i = \left\| \boldsymbol{r}_{\boldsymbol{y}_{i-1},s_{i-1}} \right\|_2 \in \mathbb{R}. \end{aligned}$$

In order to simplify the derivation of the Lanczos based MSNWF which we continue in the following, define

$$\boldsymbol{C}^{(D)} = \boldsymbol{R}_{\boldsymbol{s}}^{(D),-1} = \begin{bmatrix} \boldsymbol{c}_1^{(D)} & \cdots & \boldsymbol{c}_D^{(D)} \end{bmatrix}. \tag{2.58}$$

The inversion lemma for partitioned matrices [1] yields

$$\boldsymbol{C}^{(D)} = \begin{bmatrix} \boldsymbol{C}^{(D-1)} & \boldsymbol{0} \\ \boldsymbol{0}^{\mathrm{T}} & 0 \end{bmatrix} + \beta_D^{-1} \boldsymbol{b}^{(D)} \boldsymbol{b}^{(D),\mathrm{H}}, \tag{2.59}$$

where

$$\boldsymbol{b}^{(D)} = \begin{bmatrix} -r_{D-1,D} \boldsymbol{c}_{D-1}^{(D-1)} \\ 1 \end{bmatrix}, \tag{2.60}$$

and

$$\beta_D = r_{D,D} - r_{D-1,D}^2 c_{D-1,D-1}^{(D-1)}. \tag{2.61}$$

The variable $c_{D-1,D-1}^{(D-1)}$ denotes the last element of the last column $\boldsymbol{c}_{D-1}^{(D-1)}$ of $\boldsymbol{C}^{(D-1)}$ at the previous step. Recall that only the first element of $\boldsymbol{r}_{\boldsymbol{s},s_0}^{(D)}$ is unequal 0 (cf. Equation 2.55), thus, only the first column $\boldsymbol{c}_1^{(D)}$ of the inverse $\boldsymbol{C}^{(D)}$

$$\boldsymbol{c}_1^{(D)} = \begin{bmatrix} \boldsymbol{c}_1^{(D-1)} \\ 0 \end{bmatrix} + \beta_D^{-1} c_{1,D-1}^{(D-1)} \begin{bmatrix} r_{D-1,D}^2 \boldsymbol{c}_{D-1}^{(D-1)} \\ -r_{D-1,D} \end{bmatrix}, \tag{2.62}$$

where $c_{1,D-1}^{(D-1)}$ denotes the first element of $\boldsymbol{c}_{D-1}^{(D-1)}$, is needed for the computation of the Wiener filter $\boldsymbol{w}_{\boldsymbol{s}}^{(D)}$ using Equation (2.42). We observe that the filter at step D depends upon the first column $\boldsymbol{c}_1^{(D-1)}$ at step $D-1$, the new entries of the covariance matrix $r_{D-1,D}$ and $r_{D,D}$, and the previous last column $\boldsymbol{c}_{D-1}^{(D-1)}$. Using again Equation (2.59), we get the following recursion formula for the new last column

$$\boldsymbol{c}_D^{(D)} = \beta_D^{-1} \begin{bmatrix} -r_{D-1,D} \boldsymbol{c}_{D-1}^{(D-1)} \\ 1 \end{bmatrix}, \tag{2.63}$$

which only depends on the previous last column and the new entries of $\boldsymbol{R}_{\boldsymbol{s}}^{(D)}$. Hence, we only have to update the vectors $\boldsymbol{c}_1^{(D)}$ and $\boldsymbol{c}_D^{(D)}$ at each iteration step. Moreover, the mean square error at step D can be updated using the first entry $c_{1,1}^{(D)}$ of $\boldsymbol{c}_1^{(D)}$ as follows (cf. Equation 2.44)

$$\mathrm{MSE}^{(D)} = \sigma_{s_0}^2 - \left\| \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right\|_2^2 c_{1,1}^{(D)}. \tag{2.64}$$

Finally, substituting $\boldsymbol{c}_1^{(i)}$ and $\boldsymbol{c}_i^{(i)}$ by $\boldsymbol{c}_{\text{first}}^{(i)}$ and $\boldsymbol{c}_{\text{last}}^{(i)}$, respectively, and using the last elements of the vectors in Equation (2.63), i. e. $c_{D,D}^{(D)} = \beta_D^{-1}$, in order to replace $c_{D-1,D-1}^{(D-1)}$ in Equation (2.61), yield the resulting Lanczos based implementation of the MSNWF given by Algorithm 2.1 [8].

Algorithm 2.1 Lanczos based MSNWF

$$\boldsymbol{t}_0 = \boldsymbol{0}$$
$$\boldsymbol{t}_1 = \boldsymbol{r}_{\boldsymbol{y}_0,s_0} / \left\| \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right\|_2$$
$$r_{0,1} = 0, \quad r_{1,1} = \beta_1 = \boldsymbol{t}_1^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_1$$
$$c_{\mathrm{first}}^{(1)} = c_{\mathrm{last}}^{(1)} = r_{1,1}^{-1}$$
$$\mathrm{MSE}^{(1)} = \sigma_{d_0}^2 - \left\| \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right\|_2^2 c_{\mathrm{first}}^{(1)}$$

for $i = 2$ to D **do**

$$\boldsymbol{v} = \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_{i-1} - r_{i-1,i-1} \boldsymbol{t}_{i-1} - r_{i-2,i-1} \boldsymbol{t}_{i-2}$$
$$r_{i-1,i} = \|\boldsymbol{v}\|_2$$
$$\boldsymbol{t}_i = \boldsymbol{v} / r_{i-1,i}$$
$$r_{i,i} = \boldsymbol{t}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_i$$
$$\beta_i = r_{i,i} - r_{i-1,i}^2 \beta_{i-1}^{-1}$$
$$\boldsymbol{c}_{\mathrm{first}}^{(i)} = \begin{bmatrix} \boldsymbol{c}_{\mathrm{first}}^{(i-1)} \\ 0 \end{bmatrix} + \beta_i^{-1} c_{\mathrm{last},1}^{(i-1)} \begin{bmatrix} r_{i-1,i}^2 \boldsymbol{c}_{\mathrm{last}}^{(i-1)} \\ -r_{i-1,i} \end{bmatrix}$$
$$\boldsymbol{c}_{\mathrm{last}}^{(i)} = \beta_i^{-1} \begin{bmatrix} -r_{i-1,i} \boldsymbol{c}_{\mathrm{last}}^{(i-1)} \\ 1 \end{bmatrix}$$
$$\mathrm{MSE}^{(i)} = \sigma_{d_0}^2 - \left\| \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right\|_2^2 c_{\mathrm{first},1}^{(i)}$$

end for

$$\boldsymbol{T}^{(D)} = \begin{bmatrix} \boldsymbol{t}_1 & \boldsymbol{t}_2 & \cdots & \boldsymbol{t}_D \end{bmatrix}$$
$$\boldsymbol{w}_0^{(D)} = \left\| \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right\|_2 \boldsymbol{T}^{(D)} \boldsymbol{c}_{\mathrm{first}}^{(D)}$$

3. Conjugate Gradient Method

3.1 Conjugate Gradient (CG) Algorithm

The iterative *Conjugate Gradient* (CG) *algorithm* was first introduced by Hestenes and Stiefel [9] for solving a system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ of N linear equations in N unknowns. The system is assumed to have an exact solution which is given in N steps. Contrary to *direct methods* (e. g. *Gaussian elimination*) where we get no solution until all steps of the algorithm are processed, an *iterative method* yields an approximate solution at each iteration step and the approximation improves from step to step. Hence, only iterative methods can be stopped after $D < N$ steps if we are not interested in the exact solution.

The CG algorithm belongs to the *Conjugate Directions* (CD) *methods* because the approximate solution is searched on mutually $\boldsymbol{A}$-conjugate search directions. Two vectors $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are $\boldsymbol{A}$*-conjugate* if

$$\boldsymbol{x}_1^{\mathrm{H}} \boldsymbol{A} \boldsymbol{x}_2 = \boldsymbol{x}_2^{\mathrm{H}} \boldsymbol{A} \boldsymbol{x}_1 = 0, \tag{3.1}$$

where $\boldsymbol{A}$ is assumed to be Hermitian. Moreover, the CG method is a special case of the CD method because in addition to the $\boldsymbol{A}$-conjugate search directions, the residuals of different iteration steps are mutually orthogonal. In the following, the matrix $\boldsymbol{A} \in \mathbb{C}^{N \times N}$ is assumed to be Hermitian and positive definite.

Algorithm 3.1 First Implementation of CG Algorithm

$\boldsymbol{x}^{(0)} = \boldsymbol{0}$
$\boldsymbol{p}_1 = -\boldsymbol{r}_1 = \boldsymbol{b}$
for $i = 1$ to D **do**
  $\gamma_i = -\left(\boldsymbol{p}_i^{\mathrm{H}} \boldsymbol{r}_i\right) / \left(\boldsymbol{p}_i^{\mathrm{H}} \boldsymbol{A} \boldsymbol{p}_i\right)$
5: $\boldsymbol{x}^{(i)} = \boldsymbol{x}^{(i-1)} + \gamma_i \boldsymbol{p}_i$
  $\boldsymbol{r}_{i+1} = \boldsymbol{r}_i + \gamma_i \boldsymbol{A} \boldsymbol{p}_i$
  $\delta_i = \left(\boldsymbol{p}_i^{\mathrm{H}} \boldsymbol{A} \boldsymbol{r}_{i+1}\right) / \left(\boldsymbol{p}_i^{\mathrm{H}} \boldsymbol{A} \boldsymbol{p}_i\right)$
  $\boldsymbol{p}_{i+1} = -\boldsymbol{r}_{i+1} + \delta_i \boldsymbol{p}_i$
end for

One possible implementation of the CG method is given by Algorithm 3.1 which we use in order to explain its functionality. Note for this chapter that all lines of the algorithm hold for $i \in \{1, 2, \ldots, D\}$. The fundamental recursion formula which updates the approximate solution of the system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ is Line 5 of Algorithm 3.1, where the new approximation $\boldsymbol{x}^{(i)}$ lies on the line through the old approximation $\boldsymbol{x}^{(i-1)}$ in the direction $\boldsymbol{p}_i$, the so-called *search direction*. The *weight factor* γ_i can be understood with its definition in Line 4 of Algorithm 3.1 and the introduction of the *error function* $e(\boldsymbol{x})$ as the $\boldsymbol{A}$-norm of the error of the approximate solution $\boldsymbol{x}$ to the exact solution $\boldsymbol{x}^{(N)}$, i.e.

$$e(\boldsymbol{x}) = \left\|\boldsymbol{x}^{(N)} - \boldsymbol{x}\right\|_{\boldsymbol{A}} = \left(\boldsymbol{x}^{(N)} - \boldsymbol{x}\right)^{\mathrm{H}} \boldsymbol{A} \left(\boldsymbol{x}^{(N)} - \boldsymbol{x}\right) = \boldsymbol{x}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{x} - \boldsymbol{x}^{\mathrm{H}}\boldsymbol{b} - \boldsymbol{b}^{\mathrm{H}}\boldsymbol{x} + \boldsymbol{x}^{(N),\mathrm{H}}\boldsymbol{b}. \quad (3.2)$$

It can easily be shown that the choice of γ_i ensures that the updated approximate solution $\boldsymbol{x}^{(i)}$ minimizes the error function $e(\boldsymbol{x})$ on the line $\boldsymbol{x} = \boldsymbol{x}^{(i-1)} + \gamma \boldsymbol{p}_i$. It holds

$$\gamma_i = \arg\min_{\gamma} e\left(\boldsymbol{x}^{(i-1)} + \gamma \boldsymbol{p}_i\right). \quad (3.3)$$

Line 6 of Algorithm 3.1 updates the *residual* $\boldsymbol{r}_i$ defined as

$$\boldsymbol{r}_i = \boldsymbol{A}\boldsymbol{x}^{(i-1)} - \boldsymbol{b}. \quad (3.4)$$

Note that the residual $\boldsymbol{r}_i$ belongs to the approximation $\boldsymbol{x}^{(i-1)}$. The index mismatch is useful for the derivations we make in Chapter 4. The recursion formula for the residuals can easily be derived as follows

$$\boldsymbol{r}_{i+1} = \boldsymbol{A}\boldsymbol{x}^{(i)} - \boldsymbol{b} = \boldsymbol{A}\left(\boldsymbol{x}^{(i-1)} + \gamma_i \boldsymbol{p}_i\right) - \boldsymbol{b} = \boldsymbol{r}_i + \gamma_i \boldsymbol{A}\boldsymbol{p}_i,$$

by using Line 5 of Algorithm 3.1. Finally, the search direction $\boldsymbol{p}_i$ is updated by the remaining Lines 7 and 8 of Algorithm 3.1 which ensure the $\boldsymbol{A}$-conjugacy of the vector $\boldsymbol{p}_i$ to any vector $\boldsymbol{p}_j$, $i \neq j$.

Next, we derive another realization of the CG algorithm by changing Lines 4 and 7 of Algorithm 3.1. For the derivation we have to use the relations of Theorem 3.1 which will be proven in the next section. Considering Line 8 of Algorithm 3.1 and Equation (3.9) leads to another expression for $-\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{r}_i$ in Line 4 of Algorithm 3.1 as follows

$$-\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{r}_i = \boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{r}_i + \delta_{i-1}\boldsymbol{p}_{i-1}^{\mathrm{H}}\boldsymbol{r}_i = \boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{r}_i. \quad (3.5)$$

Apply Lines 6 and 8 of Algorithm 3.1 in order to replace the expressions $\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{A}$ and $\boldsymbol{p}_i$ in Line 7 of Algorithm 3.1, respectively. We get by using Equations (3.8) and (3.9):

$$\frac{\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{A}\boldsymbol{r}_{i+1}}{\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_i} = \frac{\gamma_i^{-1}\left(\boldsymbol{r}_{i+1}^{\mathrm{H}} - \boldsymbol{r}_i^{\mathrm{H}}\right)\boldsymbol{r}_{i+1}}{\gamma_i^{-1}\left(\boldsymbol{r}_{i+1}^{\mathrm{H}} - \boldsymbol{r}_i^{\mathrm{H}}\right)\left(-\boldsymbol{r}_i + \delta_{i-1}\boldsymbol{p}_{i-1}\right)} = \frac{\boldsymbol{r}_{i+1}^{\mathrm{H}}\boldsymbol{r}_{i+1}}{\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{r}_i}, \quad (3.6)$$

and end up with the CG implementation given by Algorithm 3.2.

Note for later that $\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{r}_i \in \mathbb{R}$ because of Equation (3.5). In addition, $\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_i \in \mathbb{R}$ since $\boldsymbol{A}$ is Hermitian (similar to Equation 2.57). It follows that $\gamma_i \in \mathbb{R}$ and Equation (3.6) implies that $\delta_i \in \mathbb{R}$, too.

Algorithm 3.2 has much less computational complexity as we show in Section 4.2 and should be preferred, therefore. However, it is easier to derive the Lanczos based MSNWF using Algorithm 3.1 (cf. Section 4.1).

Algorithm 3.2 Second Implementation of CG Algorithm

$\boldsymbol{x}^{(0)} = \mathbf{0}$
$\boldsymbol{p}_1 = -\boldsymbol{r}_1 = \boldsymbol{b}$
for $i = 1$ to D **do**
  $\gamma_i = \left(\boldsymbol{r}_i^{\mathrm{H}} \boldsymbol{r}_i\right) / \left(\boldsymbol{p}_i^{\mathrm{H}} \boldsymbol{A} \boldsymbol{p}_i\right)$
  $\boldsymbol{x}^{(i)} = \boldsymbol{x}^{(i-1)} + \gamma_i \boldsymbol{p}_i$
  $\boldsymbol{r}_{i+1} = \boldsymbol{r}_i + \gamma_i \boldsymbol{A} \boldsymbol{p}_i$
  $\delta_i = \left(\boldsymbol{r}_{i+1}^{\mathrm{H}} \boldsymbol{r}_{i+1}\right) / \left(\boldsymbol{r}_i^{\mathrm{H}} \boldsymbol{r}_i\right)$
  $\boldsymbol{p}_{i+1} = -\boldsymbol{r}_{i+1} + \delta_i \boldsymbol{p}_i$
end for

3.2 Basic Relations

In this section, we prove the most important relations of the CG method implemented by Algorithm 3.1. We have already used one of the resulting theorems in Section 3.1 in order to derive a second possible CG realization given by Algorithm 3.2. Moreover, they will help us in order to see the basic relationship between the Lanczos based MSNWF and the CG algorithm in Chapter 4. In the following of this section, assume that $i, j \in \{1, 2, \ldots, D\}$.

Theorem 3.1 *The set of search directions* $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_D\}$ *and the set of residuals* $\{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_D\}$ *generated by Algorithm 3.1 satisfy the relations*

$$\boldsymbol{p}_i^{\mathrm{H}} \boldsymbol{A} \boldsymbol{p}_j = 0, \qquad \forall i \neq j, \tag{3.7}$$

$$\boldsymbol{r}_i^{\mathrm{H}} \boldsymbol{r}_j = 0, \qquad \forall i \neq j, \tag{3.8}$$

$$\boldsymbol{p}_i^{\mathrm{H}} \boldsymbol{r}_j = 0, \qquad \forall i < j. \tag{3.9}$$

Thus, the search directions $\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_D$ *are mutually* $\boldsymbol{A}$*-conjugate and the residuals* $\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_D$ *are mutually orthogonal.*

Proof. The proof will be made by induction. First, the vectors $\boldsymbol{r}_1$, $\boldsymbol{p}_1 = -\boldsymbol{r}_1$, and $\boldsymbol{r}_2$ satisfy the relations in Theorem 3.1 since

$$-\boldsymbol{r}_1^{\mathrm{H}} \boldsymbol{r}_2 = \boldsymbol{p}_1^{\mathrm{H}} \boldsymbol{r}_2 = \boldsymbol{p}_1^{\mathrm{H}} \boldsymbol{r}_1 + \gamma_1 \boldsymbol{p}_1^{\mathrm{H}} \boldsymbol{A} \boldsymbol{p}_1 = 0,$$

where we used Lines 6 and 4 of Algorithm 3.1.

Second, assume that Equations (3.7), (3.8), and (3.9) hold for the set of search directions $\left\{\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_{n-1}\right\}$ and for the set of residuals $\{\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_n\}$. In order to prove Theorem 3.1, we finally have to show that the vectors $\boldsymbol{p}_n$ and $\boldsymbol{r}_{n+1}$ can be added to these sets.

This holds for $\boldsymbol{p}_n$ if we prove that

$$\boldsymbol{p}_i^{\mathrm{H}} \boldsymbol{A} \boldsymbol{p}_n = 0, \qquad \forall i < n. \tag{3.10}$$

The dot product of Line 6 of Algorithm 3.1 with $\boldsymbol{p}_n$ reads as

$$\boldsymbol{r}_{i+1}^{\mathrm{H}} \boldsymbol{p}_n = \boldsymbol{r}_i^{\mathrm{H}} \boldsymbol{p}_n + \gamma_i \boldsymbol{p}_i^{\mathrm{H}} \boldsymbol{A} \boldsymbol{p}_n.$$

Therefore, if $\boldsymbol{r}_{i+1}^{\mathrm{H}} \boldsymbol{p}_n = \boldsymbol{r}_i^{\mathrm{H}} \boldsymbol{p}_n$, Equation (3.10) is proven. First, let $i < n-1$. In order to get an expression for $\boldsymbol{r}_{i+1}^{\mathrm{H}} \boldsymbol{p}_n$, consider Lines 8 and 6 of Algorithm 3.1:

$$\begin{aligned} \boldsymbol{r}_{i+1}^{\mathrm{H}} \boldsymbol{p}_n &= -\boldsymbol{r}_{i+1}^{\mathrm{H}} \boldsymbol{r}_n + \delta_{n-1} \boldsymbol{r}_{i+1}^{\mathrm{H}} \boldsymbol{p}_{n-1} \\ &= -\boldsymbol{r}_{i+1}^{\mathrm{H}} \boldsymbol{r}_n + \delta_{n-1} \boldsymbol{r}_i^{\mathrm{H}} \boldsymbol{p}_{n-1} + \delta_{n-1} \gamma_i \boldsymbol{p}_i^{\mathrm{H}} \boldsymbol{A} \boldsymbol{p}_{n-1} = \delta_{n-1} \boldsymbol{r}_i^{\mathrm{H}} \boldsymbol{p}_{n-1}. \end{aligned} \tag{3.11}$$

Note that $\boldsymbol{r}_{i+1}^{\mathrm{H}}\boldsymbol{r}_n = 0$ and $\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_{n-1} = 0$ due to the induction assumption. Using again Line 8 of Algorithm 3.1 yields for the expression $\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{p}_n$

$$\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{p}_n = -\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{r}_n + \delta_{n-1}\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{p}_{n-1} = \delta_{n-1}\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{p}_{n-1}. \tag{3.12}$$

If we compare Equations (3.11) and (3.12) we see that $\boldsymbol{r}_{i+1}^{\mathrm{H}}\boldsymbol{p}_n = \boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{p}_n$ and thus, Equation (3.10) holds for $i < n-1$. Now, consider $i = n-1$. Replacing $i+1$ by n in Line 8 of Algorithm 3.1, multiplying it on the left side by $\boldsymbol{p}_{n-1}^{\mathrm{H}}\boldsymbol{A}$ and using Line 7 of Algorithm 3.1 leads to

$$\boldsymbol{p}_{n-1}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_n = -\boldsymbol{p}_{n-1}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{r}_n + \delta_{n-1}\boldsymbol{p}_{n-1}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_{n-1} = 0,$$

which completes the proof of Equation (3.10).

It remains to show that the residual $\boldsymbol{r}_{n+1}$ can be added, too. Therefore, we have to show that

$$\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{r}_{n+1} = 0, \qquad \forall i < n+1, \tag{3.13}$$
$$\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{r}_{n+1} = 0, \qquad \forall i < n+1. \tag{3.14}$$

Set $i = n$ in Line 6 of Algorithm 3.1, multiply it on the left side by $\boldsymbol{r}_i^{\mathrm{H}}$ and use again Line 8 of Algorithm 3.1 in order to replace $\boldsymbol{r}_i^{\mathrm{H}}$ to get

$$\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{r}_{n+1} = \boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{r}_n + \gamma_n\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_n = \boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{r}_n + \gamma_n\delta_{i-1}\boldsymbol{p}_{i-1}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_n - \gamma_n\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_n. \tag{3.15}$$

At once, it can be seen that $\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{r}_{n+1} = 0$ for $i < n$ due to the assumption that the vector $\boldsymbol{r}_n$ is already included in the set of residuals which fulfill the relations in Theorem 3.1, i. e. $\boldsymbol{r}_i^{\mathrm{H}}\boldsymbol{r}_n = 0$, and due to Equation (3.10) which we have already proven above. For $i = n$, Equation 3.15 simplifies by applying again the already established relation $\boldsymbol{p}_{n-1}^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_n = 0$, Lines 4 and 8 of Algorithm 3.1, and the induction assumption that $\boldsymbol{p}_{n-1}^{\mathrm{H}}\boldsymbol{r}_n = 0$ as follows

$$\boldsymbol{r}_n^{\mathrm{H}}\boldsymbol{r}_{n+1} = \boldsymbol{r}_n^{\mathrm{H}}\boldsymbol{r}_n - \gamma_n\boldsymbol{p}_n^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_n = \boldsymbol{r}_n^{\mathrm{H}}\boldsymbol{r}_n + \boldsymbol{p}_n^{\mathrm{H}}\boldsymbol{r}_n = \delta_{n-1}\boldsymbol{p}_{n-1}^{\mathrm{H}}\boldsymbol{r}_n = 0,$$

and Equation (3.13) is established. The remaining Equation (3.14) can be proven in a similar way. Set again $i = n$ in Line 6 of Algorithm 3.1 and compute the dot product with $\boldsymbol{p}_i$. It follows

$$\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{r}_{n+1} = \boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{r}_n + \gamma_n\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_n = 0,$$

where we applied for the case $i = n$, Line 4 of Algorithm 3.1, and for $i < n$, Equation (3.10) and the induction assumption that $\boldsymbol{p}_i^{\mathrm{H}}\boldsymbol{r}_n = 0$. Thus, Equation (3.14) is true and Theorem 3.1 is proven. □

Theorem 3.2 *The following identity of subspaces in $\mathbb{C}^N$ holds at iteration step i*

$$\begin{aligned}\operatorname{span}\left(\begin{bmatrix} \boldsymbol{x}^{(1)} & \boldsymbol{x}^{(2)} & \cdots & \boldsymbol{x}^{(i)} \end{bmatrix}\right) &= \operatorname{span}\left(\begin{bmatrix} \boldsymbol{p}_1 & \boldsymbol{p}_2 & \cdots & \boldsymbol{p}_i \end{bmatrix}\right) \\ &= \operatorname{span}\left(\begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \cdots & \boldsymbol{r}_i \end{bmatrix}\right) \\ &= \operatorname{span}\left(\begin{bmatrix} \boldsymbol{b} & \boldsymbol{A}\boldsymbol{b} & \cdots & \boldsymbol{A}^{i-1}\boldsymbol{b} \end{bmatrix}\right) = \mathcal{K}^{(i)}\left(\boldsymbol{A}, \boldsymbol{b}\right).\end{aligned} \tag{3.16}$$

Hence, after D iteration steps, the approximate solution $\boldsymbol{x}^{(D)}$ of the system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ lies in the D-dimensional Krylov subspace $\mathcal{K}^{(D)}\left(\boldsymbol{A}, \boldsymbol{b}\right)$.

Proof. Again, we prove by induction. Set $i = 1$. Due to the initialization and Line 5 of Algorithm 3.1, it holds the following relations:

$$\boldsymbol{x}^{(1)} = \gamma_1 \boldsymbol{p}_1, \quad \boldsymbol{p}_1 = -\boldsymbol{r}_1, \quad \boldsymbol{r}_1 = -\boldsymbol{b}.$$

Thus, $\boldsymbol{x}^{(1)}$, $\boldsymbol{p}_1$, $\boldsymbol{r}_1$, and $\boldsymbol{b}$ span the same subspace and Equation (3.16) is true for $i = 1$. Now we have to prove that the identity holds for $i = n$ if we assume that it holds for $i = n - 1$.

In order to prove that $\boldsymbol{x}^{(n)}$ lies in the subspace spanned by the vectors $\boldsymbol{p}_1, \boldsymbol{p}_2, \dots, \boldsymbol{p}_n$, consider Line 5 of Algorithm 3.1. With the initialization $\boldsymbol{x}^{(0)} = 0$ and $\gamma_k \in \mathbb{R}$, we get for $\boldsymbol{x}^{(n)}$ the linear combination

$$\boldsymbol{x}^{(n)} = \sum_{k=1}^{n} \gamma_k \boldsymbol{p}_k, \tag{3.17}$$

which establishes the first equality in Equation (3.16).

For the remaining equalities, we need the induction assumption in the form that $\boldsymbol{p}_{n-1}$ and $\boldsymbol{r}_{n-1}$ can be already expressed by the linear combinations

$$\boldsymbol{p}_{n-1} = \sum_{k=1}^{n-1} \varphi_k \boldsymbol{r}_k = \sum_{k=1}^{n-1} \chi_k \boldsymbol{A}^{k-1} \boldsymbol{b} \quad \text{and} \tag{3.18}$$

$$\boldsymbol{r}_{n-1} = \sum_{k=1}^{n-1} \zeta_k \boldsymbol{A}^{k-1} \boldsymbol{b}, \tag{3.19}$$

respectively, where $\varphi_k, \chi_k, \zeta_k \in \mathbb{R}$. Considering Line 8 of Algorithm 3.1 and applying Equation (3.18) leads to the linear combination

$$\boldsymbol{p}_n = -\boldsymbol{r}_n + \delta_{n-1} \sum_{k=1}^{n-1} \varphi_k \boldsymbol{r}_k,$$

which proves that the search direction $\boldsymbol{p}_n$ lies in the subspace spanned by the residuals $\boldsymbol{r}_1, \boldsymbol{r}_2, \dots, \boldsymbol{r}_n$ and the second equality in Equation (3.16) holds.

It remains to show that the residual $\boldsymbol{r}_n$ lies in the Krylov subspace $\mathcal{K}^{(n)}(\boldsymbol{A}, \boldsymbol{b})$. Use Line 6 of Algorithm 3.1 with $i = n - 1$ and replace $\boldsymbol{r}_{n-1}$ and $\boldsymbol{p}_{n-1}$ by Equations (3.19) and (3.18), respectively. It follows the linear combination

$$\boldsymbol{r}_n = \sum_{k=1}^{n-1} \zeta_k \boldsymbol{A}^{k-1} \boldsymbol{b} + \gamma_{n-1} \boldsymbol{A} \sum_{k=1}^{n-1} \chi_k \boldsymbol{A}^{k-1} \boldsymbol{b} = \zeta_1 \boldsymbol{b} + \sum_{k=1}^{n-1} \left(\zeta_{k+1} + \gamma_{n-1} \chi_k\right) \boldsymbol{A}^k \boldsymbol{b},$$

where $\zeta_n := 0$. Therefore, $\boldsymbol{r}_n$ lies in the subspace spanned by $\boldsymbol{b}, \boldsymbol{A}\boldsymbol{b}, \dots, \boldsymbol{A}^{n-1}\boldsymbol{b}$, i. e. the Krylov subspace $\mathcal{K}^{(n)}(\boldsymbol{A}, \boldsymbol{b})$. Finally, the last sentence of Theorem 3.2 is true if $i = D$ and the proof is finished. □

Theorem 3.3 *The fact that the search directions $\boldsymbol{p}_1, \boldsymbol{p}_2, \dots, \boldsymbol{p}_i$ are mutually $\boldsymbol{A}$-conjugate ensures that the choice of γ_i in Line 4 of Algorithm 3.1 minimizes the error function $e\left(\boldsymbol{x}^{(i)}\right)$ not only on the line $\boldsymbol{x}^{(i-1)} + \gamma \boldsymbol{p}_i$ but also in the whole Krylov subspace $\mathcal{K}^{(i)}(\boldsymbol{A}, \boldsymbol{b})$, i. e.*

$$e\left(\boldsymbol{x}^{(i)}\right) = e\left(\sum_{k=1}^{i} \gamma_k \boldsymbol{p}_k\right) = \min_{\gamma_1', \gamma_2', \dots, \gamma_i'} e\left(\sum_{k=1}^{i} \gamma_k' \boldsymbol{p}_k\right). \tag{3.20}$$

Proof. Using the definition of the error function in Equation (3.2) yields

$$e\left(\sum_{k=1}^{i}\gamma_k'\boldsymbol{p}_k\right) = \left(\sum_{k=1}^{i}\gamma_k'\boldsymbol{p}_k^{\mathrm{H}}\right)\boldsymbol{A}\left(\sum_{k=1}^{i}\gamma_k'\boldsymbol{p}_k\right) - \sum_{k=1}^{i}\gamma_k'\left(\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{b} + \boldsymbol{b}^{\mathrm{H}}\boldsymbol{p}_k\right) + \boldsymbol{x}^{(N),\mathrm{H}}\boldsymbol{b}.$$

The first term on the right hand side can be simplified if we remember the $\boldsymbol{A}$-conjugacy of the search directions (cf. Equation 3.7) and thus, all terms may be written under one sum symbol, i. e.

$$e\left(\sum_{k=1}^{i}\gamma_k'\boldsymbol{p}_k\right) = \sum_{k=1}^{i}\left(\gamma_k'^{\,2}\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_k - \gamma_k'\left(\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{b} + \boldsymbol{b}^{\mathrm{H}}\boldsymbol{p}_k\right)\right) + \boldsymbol{x}^{(N),\mathrm{H}}\boldsymbol{b}. \tag{3.21}$$

Hence, for $k \in \{1, 2, \ldots, i\}$, the optimization leads to the system of decoupled linear equations

$$\frac{\partial e\left(\sum\limits_{k=1}^{i}\gamma_k'\boldsymbol{p}_k\right)}{\partial\gamma_k'} = 2\gamma_k'\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_k - \left(\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{b} + \boldsymbol{b}^{\mathrm{H}}\boldsymbol{p}_k\right) = 0, \tag{3.22}$$

and the optimum can be found by considering every γ_k' on its own. Therefore, the $\boldsymbol{A}$-conjugacy of the search directions leads to the fact that the optimization at each iteration step on a line in the search direction is equal to the optimization including the whole Krylov subspace $\mathcal{K}^{(i)}(\boldsymbol{A}, \boldsymbol{b})$. Finally, Equation (3.22) yields the optimum for

$$\gamma_k' = \frac{\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{b} + \boldsymbol{b}^{\mathrm{H}}\boldsymbol{p}_k}{2\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_k}. \tag{3.23}$$

In order to finish the proof, we have to show that $\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{b} = \boldsymbol{b}^{\mathrm{H}}\boldsymbol{p}_k$ or $\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{b} \in \mathbb{R}$, respectively. This can be done by transforming the term $\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{r}_k \in \mathbb{R}$ (cf. Equation 3.5) using Line 6 of Algorithm 3.1 recursively:

$$\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{r}_k = \boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{r}_{k-1} + \gamma_{k-1}\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{A}\boldsymbol{p}_{k-1} = \boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{r}_{k-1} = \ldots = \boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{r}_1 = -\boldsymbol{p}_k^{\mathrm{H}}\boldsymbol{b} \in \mathbb{R}. \tag{3.24}$$

Consequently, $\gamma_k' = \gamma_k$ as given by Line 4 of Algorithm 3.1 and Theorem 3.3 is proven. □

4. Relationship between MSNWF and CG Algorithm

4.1 Transformation of MSNWF to CG

In numerous papers and books (e. g. [8, 11]), it was mentioned that the Lanczos algorithm which is used by the special implementation of the MSNWF derived in Section 2.3, is only a version of the CG algorithm. If we compare the optimization functions in Equations (2.2) and (3.2), we see that they are the same except for a constant which does not change the solution of the optimization. Moreover, with the proof of Theorem 3.3, we have shown that not only the MSNWF but also the CG algorithm finds the approximate solution of the Wiener-Hopf equation (cf. Equation 2.7) in the Krylov subspace $\mathcal{K}^{(D)}\left(\boldsymbol{R}_{\boldsymbol{y}_0}, \boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right)$. Consequently, both methods yield the same result. In order to establish the equivalence of both algorithms we transform [12] the formulas of the Lanczos based MSNWF to those of the CG algorithm in this section. Again, in this chapter, $i \in \{1, 2, \dots, D\}$.

First, we derive some fundamental analogies. Assume that $i \geq 2$. Using Lines 17 and 12 of Algorithm 2.1, and by setting $\boldsymbol{T}^{(i)} = \left[\begin{array}{cc} \boldsymbol{T}^{(i-1)} & \boldsymbol{t}_i \end{array}\right]$, it holds for $\boldsymbol{w}_0^{(i)}$ that

$$\boxed{\boldsymbol{w}_0^{(i)} = \left\|\boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right\|_2 \boldsymbol{T}^{(i)} \boldsymbol{c}_{\text{first}}^{(i)} = \boldsymbol{w}_0^{(i-1)} + \eta_i \boldsymbol{u}_i, \quad \boldsymbol{w}_0^{(1)} = \left\|\boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right\|_2 r_{1,1}^{-1} \boldsymbol{t}_1,} \tag{4.1}$$

where η_i and $\boldsymbol{u}_i$ are defined as

$$\eta_i = \left\|\boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right\|_2 \varrho_i^{-1} \beta_i^{-1} r_{i-1,i} c_{\text{last},1}^{(i-1)}, \tag{4.2}$$

$$\boldsymbol{u}_i = \varrho_i \left(r_{i-1,i} \boldsymbol{T}^{(i-1)} \boldsymbol{c}_{\text{last}}^{(i-1)} - \boldsymbol{t}_i \right), \tag{4.3}$$

and where $\varrho_i \in \mathbb{R} \setminus \{0\}$ is an arbitrary factor whose meaning is explained later. Multiplying Line 13 of Algorithm 2.1 on the left side by $\boldsymbol{T}^{(i)}$ and using Equation (4.3) leads to

$$\boldsymbol{T}^{(i)} \boldsymbol{c}_{\text{last}}^{(i)} = -\beta_i^{-1} \left(r_{i-1,i} \boldsymbol{T}^{(i-1)} \boldsymbol{c}_{\text{last}}^{(i-1)} - \boldsymbol{t}_i \right) = -\varrho_i^{-1} \beta_i^{-1} \boldsymbol{u}_i. \tag{4.4}$$

Substituting i by $i+1$ in Equation (4.3) and replacing $\boldsymbol{T}^{(i)} \boldsymbol{c}_{\text{last}}^{(i)}$ by using Equation (4.4) yields a recursion formula for $\boldsymbol{u}_{i+1}$ and $D > i \geq 1$

$$\boxed{\boldsymbol{u}_{i+1} = \psi_i \boldsymbol{u}_i - \boldsymbol{g}_{i+1}, \quad \boldsymbol{u}_1 = -\varrho_1 \boldsymbol{t}_1,} \tag{4.5}$$

where

$$\psi_i = -\varrho_{i+1} \varrho_i^{-1} \beta_i^{-1} r_{i,i+1}, \tag{4.6}$$

$$\boldsymbol{g}_{i+1} = \varrho_{i+1} \boldsymbol{t}_{i+1}. \tag{4.7}$$

In Equations (4.1) and (4.5), we observe analogous terms if we compare them with Lines 5 and 8 of Algorithm 3.1. The vectors $\boldsymbol{u}_i$ and $\boldsymbol{g}_i$ seem to be equivalent to the search direction $\boldsymbol{p}_i$ and the residual $\boldsymbol{r}_i$ of the CG algorithm, respectively. Besides, remember that in the Lanczos based MSNWF, the linear system to be solved is the Wiener-Hopf equation where the solution is the Wiener filter $\boldsymbol{w}_0$. In the sequel, we prove that the analogous remaining formulas of the CG algorithm can be derived from the MSNWF.

Proposition 4.1 *The vectors $\boldsymbol{g}_i$ can be updated by the recursion formula*

$$\boxed{\boldsymbol{g}_{i+1} = \boldsymbol{g}_i + \eta_i \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i, \quad \boldsymbol{g}_1 = \varrho_1 \boldsymbol{t}_1, \quad \varrho_1 = -\left\| \boldsymbol{r}_{\boldsymbol{y}_0, s_0} \right\|_2,} \tag{4.8}$$

where the definition of η_i is given in Equation (4.2) and where

$$\eta_1 := r_{1,1}^{-1}. \tag{4.9}$$

Proof. We prove by induction. First, set $i = 1$. Recall that $\boldsymbol{u}_1 = -\varrho_1 \boldsymbol{t}_1 = -\boldsymbol{g}_1$ and we get by using the initialization of Equations (4.8) and (4.9), and Lines 7 to 9 of Algorithm 2.1

$$\boldsymbol{g}_1 + \eta_1 \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_1 = \varrho_1 \eta_1 \left(\eta_1^{-1} \boldsymbol{t}_1 - \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_1 \right) = \left\| \boldsymbol{r}_{\boldsymbol{y}_0, s_0} \right\|_2 r_{1,1}^{-1} r_{1,2} \boldsymbol{t}_2 = \varrho_2 \boldsymbol{t}_2 = \boldsymbol{g}_2.$$

The last equality holds if we define ϱ_2 as

$$\varrho_2 = \left\| \boldsymbol{r}_{\boldsymbol{y}_0, s_0} \right\|_2 r_{1,1}^{-1} r_{1,2}. \tag{4.10}$$

Now assume that Equation (4.8) holds for $i = n - 1$. This leads to

$$\boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_{n-1} = \eta_{n-1}^{-1} \left(\boldsymbol{g}_n - \boldsymbol{g}_{n-1} \right). \tag{4.11}$$

In order to prove Equation (4.8) for $i = n$ if it holds for $i = n - 1$, we need a transformation of Equation (4.5). Substituting $i + 1$ by n and multiplying the equation on the left side by $\boldsymbol{R}_{\boldsymbol{y}_0}$ yields

$$\boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_n = \psi_{n-1} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_{n-1} - \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_n. \tag{4.12}$$

Finally, we get by using Equations (4.12), (4.11), and (4.7)

$$\begin{aligned} \boldsymbol{g}_n + \eta_n \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_n &= \boldsymbol{g}_n - \eta_n \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_n + \eta_n \psi_{n-1} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_{n-1} \\ &= \boldsymbol{g}_n - \eta_n \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_n + \eta_{n-1}^{-1} \eta_n \psi_{n-1} \left(\boldsymbol{g}_n - \boldsymbol{g}_{n-1} \right) \\ &= -\varrho_n \eta_n \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_n + \varrho_n \left(1 + \eta_{n-1}^{-1} \eta_n \psi_{n-1} \right) \boldsymbol{t}_n - \varrho_{n-1} \eta_{n-1}^{-1} \eta_n \psi_{n-1} \boldsymbol{t}_{n-1}. \end{aligned} \tag{4.13}$$

On the other hand, setting $i = n + 1$ in Lines 7 to 9 of Algorithm 2.1 and multiplying the recursion formula by ϱ_{n+1} on both sides yields

$$\varrho_{n+1} \boldsymbol{t}_{n+1} = \varrho_{n+1} r_{n,n+1}^{-1} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_n - \varrho_{n+1} r_{n,n+1}^{-1} r_{n,n} \boldsymbol{t}_n - \varrho_{n+1} r_{n,n+1}^{-1} r_{n-1,n} \boldsymbol{t}_{n-1}. \tag{4.14}$$

Recalling Equation (4.7) and comparing Equation (4.13) with (4.14), Proposition 4.1 is proven if the following three equations are true:

$$-\varrho_n \eta_n = \varrho_{n+1} r_{n,n+1}^{-1}, \tag{4.15}$$

$$\varrho_n \left(1 + \eta_{n-1}^{-1} \eta_n \psi_{n-1} \right) = -\varrho_{n+1} r_{n,n+1}^{-1} r_{n,n}, \tag{4.16}$$

$$\varrho_{n-1} \eta_{n-1}^{-1} \eta_n \psi_{n-1} = \varrho_{n+1} r_{n,n+1}^{-1} r_{n-1,n}. \tag{4.17}$$

Using Equations (4.15), (4.2), and Line 13 of Algorithm 2.1 in order to calculate ϱ_{n+1} for $D > n \geq 2$ as follows

$$\varrho_{n+1} = -\varrho_n \eta_n r_{n,n+1} = -\left\|\boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right\|_2 \beta_n^{-1} r_{n,n+1} r_{n-1,n} c_{\text{last},1}^{(n-1)} = \left\|\boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right\|_2 r_{n,n+1} c_{\text{last},1}^{(n)}.$$

If we substitute $n+1$ by i and include Equation (4.10) for $i = 2$, ϱ_i for $i \geq 2$ is given by the formulas

$$\varrho_i = \left\|\boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right\|_2 r_{i-1,i} c_{\text{last},1}^{(i-1)}, \tag{4.18}$$

$$\varrho_i = -\varrho_{i-1} \eta_{i-1} r_{i-1,i}. \tag{4.19}$$

We get simpler formulas for η_i and ψ_i if we plug Equations (4.18) and (4.19) in Equations (4.2) and (4.6), respectively. Consequently, it holds

$$\eta_i = \beta_i^{-1}, \qquad \forall i, \tag{4.20}$$

$$\psi_i = \beta_i^{-2} r_{i,i+1}^2, \qquad \forall i < D. \tag{4.21}$$

In order to prove Equation (4.16) use Equations (4.19), (4.20), (4.21), and Line 11 of Algorithm 2.1. It follows

$$\begin{aligned} \varrho_n \left(1 + \eta_{n-1}^{-1} \eta_n \psi_{n-1}\right) &= -\varrho_{n+1} \eta_n^{-1} r_{n,n+1}^{-1} \left(1 + \beta_n^{-1} \beta_{n-1}^{-1} r_{n-1,n}^2\right) \\ &= -\varrho_{n+1} r_{n,n+1}^{-1} \left(\beta_n + r_{n-1,n}^2 \beta_{n-1}^{-1}\right) = -\varrho_{n+1} r_{n,n+1}^{-1} r_{n,n}. \end{aligned}$$

With the same equations, we get

$$\varrho_{n-1} \eta_{n-1}^{-1} \eta_n \psi_{n-1} = \varrho_{n+1} \eta_{n-1}^{-2} \psi_{n-1} r_{n-1,n}^{-1} r_{n,n+1}^{-1} = \varrho_{n+1} r_{n,n+1}^{-1} r_{n-1,n}.$$

Thus, Equations (4.15), (4.16), and (4.17) hold and Proposition 4.1 is proven. □

Note that with the definition of η_1 by Equation (4.9), Equation (4.1) also holds for $i = 1$, where $\boldsymbol{w}_0^{(0)} := 0$, as well as for $i > 1$.

Proposition 4.2 *The vector $\boldsymbol{g}_i$ is a residual vector for $i \in \{1, 2, \ldots, D+1\}$, i. e.*

$$\boldsymbol{g}_i = \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{w}_0^{(i-1)} - \boldsymbol{r}_{\boldsymbol{y}_0,s_0}, \tag{4.22}$$

and the absolute value of the factor ϱ_i is its length.

Proof. Again, this proposition can be proven by induction. Set $i = 1$. It holds that

$$\boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{w}_0^{(0)} - \boldsymbol{r}_{\boldsymbol{y}_0,s_0} = -\left\|\boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right\|_2 \boldsymbol{t}_1 = \boldsymbol{g}_1,$$

by applying the definition $\boldsymbol{w}_0^{(0)} = 0$, Line 2 of Algorithm 2.1, and Equation (4.8). Assume that Equation (4.22) holds for $i = n-1$. Hence, we get for $i = n$ (cf. Equation 4.1)

$$\boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{w}_0^{(n-1)} - \boldsymbol{r}_{\boldsymbol{y}_0,s_0} = \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{w}_0^{(n-2)} - \boldsymbol{r}_{\boldsymbol{y}_0,s_0} + \eta_{n-1} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_{n-1} = \boldsymbol{g}_{n-1} + \eta_{n-1} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_{n-1} = \boldsymbol{g}_n,$$

which establishes Equation (4.22). The last equality holds because of Proposition 4.1. In order to finish the proof of Proposition 4.2, remember that the vectors $\boldsymbol{t}_i$ have unit length (cf. Lines 7 to 9 of Algorithm 3.1). Hence, Equation (4.7) shows that $|\varrho_i|$ is the length of the residual $\boldsymbol{g}_i$. □

Up to now we showed, in addition to Equations (4.1) and (4.5), that the residuals $\boldsymbol{g}_i$ are updated by a similar formula as the residuals $\boldsymbol{r}_i$ in Line 6 of Algorithm 3.1. It remains to show that the computation of the weight factors ψ_i and η_i is analogous to those in the CG method (cf. Lines 4 and 7 of Algorithm 3.1). These derivations are easier if we first prove the following proposition.

Proposition 4.3 *The vectors $\boldsymbol{u}_j$ and $\boldsymbol{g}_i$ satisfy the relation*

$$\boldsymbol{u}_j^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i = 0, \qquad \forall j \leq i-2. \tag{4.23}$$

Proof. First, let $j=1$ and $i \geq 3$, but fixed. With Equations (4.5) and (4.7) we get

$$\boldsymbol{u}_1^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i = -\varrho_1 \varrho_i \boldsymbol{t}_1^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_i = -\varrho_1 \varrho_i r_{1,i} = 0.$$

The last equality holds, because $r_{j,i}$ are the elements of the tridiagonal matrix $\boldsymbol{R}_{\boldsymbol{s}}^{(D)}$ (cf. Section 2.2 and 2.3) and thus, $r_{j,i} = \boldsymbol{t}_j^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_i = 0$ for all $j \leq i-2$. Assume that Equation (4.23) holds for $j = n-1$. Then, we get for $j = n$

$$\boldsymbol{u}_n^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i = \psi_{n-1} \boldsymbol{u}_{n-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i - \boldsymbol{g}_n^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i = -\varrho_n \varrho_i \boldsymbol{t}_n^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{t}_i = -\varrho_n \varrho_i r_{n,i} = 0,$$

if we use again Equations (4.5) and (4.7). □

Proposition 4.4 *The factor ψ_i in Equation (4.5) satisfies the relation*

$$\boxed{\psi_i = \frac{\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_{i+1}}{\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i},} \tag{4.24}$$

and thus, $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_D\}$ is a set of $\boldsymbol{R}_{\boldsymbol{y}_0}$-conjugate vectors, i. e.

$$\boldsymbol{u}_j^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i = 0, \qquad \forall j \neq i. \tag{4.25}$$

Proof. Replace $i+1$ by $i-1$ in Equation (4.5) and multiply it on the right side with $\boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i$ in order to get

$$\boldsymbol{u}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i = \psi_{i-2} \boldsymbol{u}_{i-2}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i - \boldsymbol{g}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i = -\varrho_{i-1} \varrho_i r_{i-1,i}. \tag{4.26}$$

The last equality holds because of Proposition 4.3 and Equation (4.7).

In order to obtain a similar expression for the denominator on the right side of Equation (4.24), substitute $\boldsymbol{u}_i$ given by Equation (4.5) and use Equations (4.7), (4.19), (4.20), (4.21), (4.26), and Line 10 of Algorithm 2.1. It follows

$$\begin{aligned}\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i &= \boldsymbol{g}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i - \psi_{i-1} \boldsymbol{g}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_{i-1} - \psi_{i-1} \boldsymbol{u}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_i + \psi_{i-1}^2 \boldsymbol{u}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_{i-1} \\ &= \varrho_i^2 \left(r_{i,i} - 2 r_{i-1,i}^2 \beta_{i-1}^{-1} + r_{i-1,i}^2 \beta_{i-1}^{-1} \frac{\boldsymbol{u}_{i-1}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_{i-1}}{\varrho_{i-1}^2 \beta_{i-1}} \right).\end{aligned}$$

Comparing the equation above with Line 11 of Algorithm 2.1 leads to

$$\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i = \varrho_i^2 \beta_i. \tag{4.27}$$

Substituting i by $i+1$ in Equation (4.26), dividing it by the left side of Equation (4.27), and using again Equations (4.19), (4.20), and (4.21), yields finally Equation (4.24). Thus, similar to δ_i in Algorithm 3.1, ψ_i, Equations (4.5), and (4.8) ensure that the vectors $\boldsymbol{u}_i$ are mutually $\boldsymbol{R}_{\boldsymbol{y}_0}$-conjugate (cf. Theorem 3.1). □

Proposition 4.5 *The following equation holds:*

$$\boxed{\eta_i = -\frac{\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{g}_i}{\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i}.} \tag{4.28}$$

Hence, the value of η_i above minimizes the error function

$$e(\boldsymbol{w}) = \boldsymbol{w}^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{w} - \boldsymbol{w}^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} - \boldsymbol{r}_{\boldsymbol{y}_0,s_0}^{\mathrm{H}} \boldsymbol{w} + \boldsymbol{w}_0^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \tag{4.29}$$

on the line $\boldsymbol{w} = \boldsymbol{w}_0^{(i-1)} + \eta_i \boldsymbol{u}_i$.

Proof. Replacing $\boldsymbol{g}_i$ by Equation (4.22), $\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i$ by Equation (4.27) and using recursively Equation (4.1) yields

$$\begin{aligned}
-\frac{\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{g}_i}{\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i} &= -\varrho_i^{-2} \beta_i^{-1} \boldsymbol{u}_i^{\mathrm{H}} \left(\boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{w}_0^{(i-1)} - \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right) \\
&= \varrho_i^{-2} \beta_i^{-1} \boldsymbol{u}_i^{\mathrm{H}} \left(\boldsymbol{r}_{\boldsymbol{y}_0,s_0} - \boldsymbol{R}_{\boldsymbol{y}_0} \sum_{k=1}^{i-1} \eta_k \boldsymbol{u}_k \right) \\
&= \varrho_i^{-2} \beta_i^{-1} \boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0}.
\end{aligned}$$

The last equality holds because the vectors $\boldsymbol{u}_i$ are mutually $\boldsymbol{R}_{\boldsymbol{y}_0}$-conjugate (cf. Equation 4.25). Now, we have to show that $\varrho_i^{-2} \beta_i^{-1} \boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} = \eta_i$. First, set $i = 1$. It holds by using $\boldsymbol{u}_1 = -\varrho_1 \boldsymbol{t}_1$, $\varrho_1 = -\left\| \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right\|_2$, Line 2 of Algorithm 2.1, and Equation (4.20) that

$$\varrho_1^{-2} \beta_1^{-1} \boldsymbol{u}_1^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} = \beta_1^{-1} = \eta_1.$$

Then, set $i \geq 2$, substitute $\boldsymbol{u}_i$ by Equation (4.3), and remember that $\boldsymbol{t}_i^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} = 0$ for all $i \geq 2$. It follows

$$\begin{aligned}
\varrho_i^{-2} \beta_i^{-1} \boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} &= \varrho_i^{-1} \beta_i^{-1} r_{i-1,i} \boldsymbol{c}_{\mathrm{last}}^{(i-1),\mathrm{T}} \boldsymbol{T}^{(i-1),\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} - \varrho_i^{-1} \beta_i^{-1} \boldsymbol{t}_i^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \\
&= \left\| \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right\|_2 \varrho_i^{-1} \beta_i^{-1} r_{i-1,i} c_{\mathrm{last},1}^{(i-1)} = \eta_i,
\end{aligned}$$

if we recall the definition of η_i in Equation (4.2). Therefore, η_i in Equation (4.8) minimizes the error function in the same way as γ_i in Line 5 of Algorithm 3.1 and the proof of Proposition 4.5 is completed. □

We see that Equations (4.28) and (4.24) which compute the weight factors ψ_i and η_i are similar to the remaining Lines 4 and 7 of Algorithm 3.1. To put it in a nutshell, all the formulas, we derived from the Lanczos based MSNWF, are the same as those of the CG algorithm if we make the following equivalences:

$$\text{approximate solution:} \quad \boldsymbol{w}_0^{(i)} \leftrightarrow \boldsymbol{x}^{(i)}, \qquad \eta_i \leftrightarrow \gamma_i, \tag{4.30}$$

$$\text{search directions:} \quad \boldsymbol{u}_i \leftrightarrow \boldsymbol{p}_i, \qquad \psi_i \leftrightarrow \delta_i, \tag{4.31}$$

$$\text{residuals:} \quad \boldsymbol{g}_i \leftrightarrow \boldsymbol{r}_i. \tag{4.32}$$

In the next Section, we use the derived equations to present a CG based implementation of the MSNWF.

4.2 CG Based Implementation of MSNWF

In Section 4.1 we have derived that the Lanczos based MSNWF can be expressed by the formulas of the CG algorithm. Thus, Equations (4.28) and (4.24) can be replaced in the same manner as Lines 4 and 7 of Algorithm 3.1 by Lines 4 and 7 of Algorithm 3.2, respectively. It follows if we remember the equivalences we found in Equations (4.30), (4.31), and (4.32)

$$\eta_i = \frac{\boldsymbol{g}_i^{\mathrm{H}} \boldsymbol{g}_i}{\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i}, \tag{4.33}$$

$$\psi_i = \frac{\boldsymbol{g}_{i+1}^{\mathrm{H}} \boldsymbol{g}_{i+1}}{\boldsymbol{g}_i^{\mathrm{H}} \boldsymbol{g}_i}. \tag{4.34}$$

This replacement reduces computational complexity because the matrix vector multiplication $\boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{g}_{i+1}$ in Equation (4.24) is avoided. Besides, the only matrix vector product left, $\boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i$, which is needed in Equation (4.8) has already been computed in Equation (4.33) before. Therefore, the resulting computational complexity for a rank D MSNWF is $O\left(N^2 D\right)$, since only one matrix vector multiplication with $O\left(N^2\right)$ has to be performed at each step.

Compared to the Lanczos implementation of the MSNWF, the CG algorithm does not compute the mean square error $\mathrm{MSE}^{(i)}$ at each step. To get such a recursion formula for $\mathrm{MSE}^{(i)}$ in the CG implementation consider the first elements in Line 12 of Algorithm 2.1. It holds for $i \geq 2$ that

$$c_{\text{first},1}^{(i)} = c_{\text{first},1}^{(i-1)} + \beta_i^{-1} r_{i-1,i}^2 c_{\text{last},1}^{(i-1),2}. \tag{4.35}$$

Apply this equation to replace $c_{\text{first},1}^{(i)}$ in Line 14 of Algorithm 2.1

$$\mathrm{MSE}^{(i)} = \mathrm{MSE}^{(i-1)} - \left\| \boldsymbol{r}_{\boldsymbol{y}_0, s_0} \right\|_2^2 \beta_i^{-1} r_{i-1,i}^2 c_{\text{last},1}^{(i-1),2} = \mathrm{MSE}^{(i-1)} - \varrho_i^2 \eta_i, \tag{4.36}$$

where $\mathrm{MSE}^{(1)}$ is defined by Line 4 of Algorithm 2.1. The last equality holds because of Equations (4.18) and (4.20). The fact that $|\varrho_i|$ is the length of the residual $\boldsymbol{g}_i$ and that $\mathrm{MSE}^{(1)} = \sigma_{s_0}^2 - \varrho_1^2 \eta_1$, yields the recursion formula

$$\boxed{\mathrm{MSE}^{(i)} = \mathrm{MSE}^{(i-1)} - \eta_i \boldsymbol{g}_i^{\mathrm{H}} \boldsymbol{g}_i, \quad \mathrm{MSE}^{(0)} = \sigma_{s_0}^2.} \tag{4.37}$$

Finally, summarizing Equations (4.33), (4.1), (4.37), (4.8), (4.34), and (4.5) leads to a CG implementation of the MSNWF which is given by Algorithm 4.1. Note that P. S. Chang and A. N. Willson, Jr., presented a similar algorithm to solve the Wiener-Hopf equation in [13]. However, we derived the CG algorithm from the MSNWF and in addition, our implementation computes the mean square error at each iteration step. In the following section, the resulting CG MSNWF algorithm is employed as a linear equalizer to an EDGE system.

Algorithm 4.1 CG based MSNWF

$\boldsymbol{w}_0^{(0)} = \boldsymbol{0}$
$\boldsymbol{u}_1 = -\boldsymbol{g}_1 = \boldsymbol{r}_{\boldsymbol{y}_0, s_0}$
$l_1 = \boldsymbol{g}_1^{\mathrm{H}} \boldsymbol{g}_1$
$\mathrm{MSE}^{(0)} = \sigma_{s_0}^2$
for $i = 1$ to D **do**
  $\boldsymbol{v} = \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{u}_i$
  $\eta_i = l_i / \left(\boldsymbol{u}_i^{\mathrm{H}} \boldsymbol{v} \right)$
  $\boldsymbol{w}_0^{(i)} = \boldsymbol{w}_0^{(i-1)} + \eta_i \boldsymbol{u}_i$
  $\mathrm{MSE}^{(i)} = \mathrm{MSE}^{(i-1)} - \eta_i l_i$
  $\boldsymbol{g}_{i+1} = \boldsymbol{g}_i + \eta_i \boldsymbol{v}$
  $l_{i+1} = \boldsymbol{g}_{i+1}^{\mathrm{H}} \boldsymbol{g}_{i+1}$
  $\psi_i = l_{i+1} / l_i$
  $\boldsymbol{u}_{i+1} = -\boldsymbol{g}_{i+1} + \psi_i \boldsymbol{u}_i$
end for

5. Further Methods for Dimension Reduction

5.1 Principal Component Method

The *Principal Component* (PC) *method* [2] is another way to decrease computational complexity by considering a transformed observation signal of reduced dimension. The original Wiener filter $\boldsymbol{w}_0 \in \mathbb{C}^N$ is replaced by the pre-filter $\boldsymbol{T}_{\mathrm{PC}}^{(D)} \in \mathbb{C}^{N \times D}$, $D < N$, followed by a Wiener filter of reduced dimension, $\boldsymbol{w}_{\boldsymbol{s}_{\mathrm{PC}}}^{(D)} \in \mathbb{C}^D$, which estimates the desired signal $s_0[n]$ from the transformed observation $\boldsymbol{s}_{\mathrm{PC}}^{(D)}[n] = \boldsymbol{T}_{\mathrm{PC}}^{(D),\mathrm{H}} \boldsymbol{y}_0[n]$. The resulting filter structure is shown in Figure 5.1. Due to the dimension reduction caused by the pre-filter, the result yields only an approximate solution of the Wiener-Hopf equation (cf. Equation 2.7).

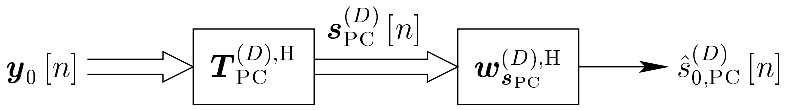

Fig. 5.1. PC Filter

In order to build the pre-filter matrix of the PC method, we need the eigenvector decomposition of the covariance matrix of the observation signal $\boldsymbol{y}_0[n]$, i. e.

$$\boldsymbol{R}_{\boldsymbol{y}_0} = \boldsymbol{Q}\boldsymbol{\Lambda}\boldsymbol{Q}^{\mathrm{H}}, \tag{5.1}$$

where

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{q}_1 & \boldsymbol{q}_2 & \dots & \boldsymbol{q}_N \end{bmatrix} \tag{5.2}$$

is the orthonormal modal matrix and

$$\boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1, \lambda_2, \dots, \lambda_N) \tag{5.3}$$

is a diagonal matrix of the eigenvalues of $\boldsymbol{R}_{\boldsymbol{y}_0}$. Without loss of generality, we assume that $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_N$. Then, the pre-filter matrix $\boldsymbol{T}_{\mathrm{PC}}^{(D)}$ is obtained by using the first D eigenvectors corresponding to the largest eigenvalues, i. e.

$$\boldsymbol{T}_{\mathrm{PC}}^{(D)} = \begin{bmatrix} \boldsymbol{q}_1 & \boldsymbol{q}_2 & \dots & \boldsymbol{q}_D \end{bmatrix}. \tag{5.4}$$

In Section 2.2, we have derived the reduced rank MSNWF which can be seen as a pre-filter $\boldsymbol{T}^{(D)}$ followed by a Wiener filter $\boldsymbol{w}_{\boldsymbol{s}}^{(D)}$, too. Thus, similar to Equation (2.43), the PC approximation of the Wiener filter $\boldsymbol{w}_0$ may be written as

$$\begin{aligned}\boldsymbol{w}_{0,\mathrm{PC}}^{(D)} = \boldsymbol{T}_{\mathrm{PC}}^{(D)} \boldsymbol{w}_{\boldsymbol{s}_{\mathrm{PC}}}^{(D)} &= \boldsymbol{T}_{\mathrm{PC}}^{(D)} \left(\boldsymbol{T}_{\mathrm{PC}}^{(D),\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{T}_{\mathrm{PC}}^{(D)} \right)^{-1} \boldsymbol{T}_{\mathrm{PC}}^{(D),\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \\ &= \boldsymbol{T}_{\mathrm{PC}}^{(D)} \boldsymbol{\Lambda}_{\mathrm{PC}}^{(D),-1} \boldsymbol{T}_{\mathrm{PC}}^{(D),\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0}\end{aligned} \tag{5.5}$$

with the diagonal matrix

$$\boldsymbol{\Lambda}_{\mathrm{PC}}^{(D)} = \operatorname{diag}\left(\lambda_1, \lambda_2, \ldots, \lambda_D\right) \in \mathbb{C}^{D \times D}. \tag{5.6}$$

Because of the reduced dimension of the pre-filtered observation $\boldsymbol{s}_{\mathrm{PC}}^{(D)}[n]$, the inversion of its covariance matrix $\boldsymbol{R}_{\boldsymbol{s}_{\mathrm{PC}}}^{(D)} = \boldsymbol{T}_{\mathrm{PC}}^{(D),\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{T}_{\mathrm{PC}}^{(D)} = \boldsymbol{\Lambda}_{\mathrm{PC}}^{(D)}$ has less computational complexity compared to the inversion of the covariance matrix of the original observation signal, $\boldsymbol{R}_{\boldsymbol{y}_0}$, but the used eigenvector decomposition has a complexity of $O\left(N^2 D\right)$ if we only compute the eigenvectors which correspond to the largest eigenvalues of $\boldsymbol{R}_{\boldsymbol{y}_0}$.

Contrary to the MSNWF and the CG algorithm (cf. Chapters 2, 3, and 4), the PC method searches for the approximate solution of the Wiener-Hopf equation in a subspace spanned by the eigenvectors corresponding to the largest eigenvalues of the covariance matrix, instead of searching in the Krylov subspace $\mathcal{K}^{(D)}\left(\boldsymbol{R}_{\boldsymbol{y}_0}, \boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right)$. Hence, the PC algorithm does not consider the cross-correlation between the observation $\boldsymbol{y}_0[n]$ and the desired signal $s_0[n]$, i. e. it uses the signal components with the largest power without distinguishing between the desired signal and interference.

5.2 Cross Spectral Method

Goldstein et. al. [14] introduced the *Cross Spectral* (CS) *metric* in order to include the information of the cross-correlation between the observation $\boldsymbol{y}_0[n]$ and the desired signal $s_0[n]$ in the choice of the eigenvectors of the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$ for the composition of the pre-filter matrix. The result is an improved approximate solution of the Wiener-Hopf equation compared to the PC method.

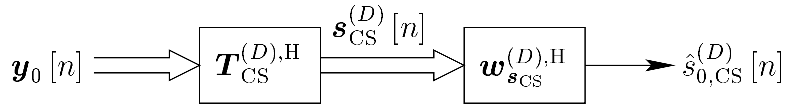

Fig. 5.2. CS Filter

Figure 5.2 shows the CS filter structure. The columns of the pre-filter matrix $\boldsymbol{T}_{\mathrm{CS}}^{(D)} \in \mathbb{C}^{N \times D}$, $D < N$, are the eigenvectors of $\boldsymbol{R}_{\boldsymbol{y}_0}$ corresponding to the largest CS metric which is explained in the next paragraph, and the Wiener filter $\boldsymbol{w}_{\boldsymbol{s}_{\mathrm{CS}}}^{(D)} \in \mathbb{C}^D$ estimates $s_0[n]$ from $\boldsymbol{s}_{\mathrm{CS}}^{(D)}[n]$. The resulting CS filter

$$\boldsymbol{w}_{0,\mathrm{CS}}^{(D)} = \boldsymbol{T}_{\mathrm{CS}}^{(D)} \boldsymbol{\Lambda}_{\mathrm{CS}}^{(D),-1} \boldsymbol{T}_{\mathrm{CS}}^{(D),\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0}, \tag{5.7}$$

if we make the corresponding substitutions in Equation (5.5).

It remains to derive the CS metric. First, recall the eigenvector decomposition of $\boldsymbol{R}_{\boldsymbol{y}_0}$ in Equation (5.1). Then, replace the matrix $\boldsymbol{T}^{(D)}$ in Equation (2.44) by the CS pre-filter $\boldsymbol{T}_{\mathrm{CS}}^{(D)}$.

It follows for the mean square error

$$
\begin{aligned}
\mathrm{MSE}_{\mathrm{CS}}^{(D)} &= \sigma_{s_0}^2 - \boldsymbol{r}_{\boldsymbol{y}_0,s_0}^{\mathrm{H}} \boldsymbol{T}_{\mathrm{CS}}^{(D)} \left(\boldsymbol{T}_{\mathrm{CS}}^{(D),\mathrm{H}} \boldsymbol{R}_{\boldsymbol{y}_0} \boldsymbol{T}_{\mathrm{CS}}^{(D)} \right)^{-1} \boldsymbol{T}_{\mathrm{CS}}^{(D),\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \\
&= \sigma_{s_0}^2 - \boldsymbol{r}_{\boldsymbol{y}_0,s_0}^{\mathrm{H}} \boldsymbol{T}_{\mathrm{CS}}^{(D)} \boldsymbol{\Lambda}_{\mathrm{CS}}^{(D),-1} \boldsymbol{T}_{\mathrm{CS}}^{(D),\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \\
&= \sigma_{s_0}^2 - \sum_{i \in \mathcal{M}} \frac{\left| \boldsymbol{q}_i^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right|^2}{\lambda_i},
\end{aligned}
\tag{5.8}
$$

where the diagonal matrix $\boldsymbol{\Lambda}_{\mathrm{CS}}^{(D)}$ contains the eigenvalues λ_i, $i \in \mathcal{M}$, of the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$, and the pre-filter matrix $\boldsymbol{T}_{\mathrm{CS}}^{(D)}$ the corresponding eigenvectors $\boldsymbol{q}_i$. The set $\mathcal{M}$ is chosen to ensure that the mean square error $\mathrm{MSE}_{\mathrm{CS}}^{(D)}$ is minimum. Thus, the following optimization problem has to be solved

$$
\mathcal{M} = \arg \max_{\substack{\mathcal{M}' \subset \{1,2,\ldots,N\} \\ |\mathcal{M}'| = D}} \sum_{i \in \mathcal{M}'} \frac{\left| \boldsymbol{q}_i^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right|^2}{\lambda_i},
\tag{5.9}
$$

and the CS metric which decides whether a eigenvector $\boldsymbol{q}_i$ is chosen as a column of $\boldsymbol{T}_{\mathrm{CS}}^{(D)}$ or not, is the term $\left| \boldsymbol{q}_i^{\mathrm{H}} \boldsymbol{r}_{\boldsymbol{y}_0,s_0} \right|^2 / \lambda_i$.

The CS method approximates the Wiener filter $\boldsymbol{w}_0$ in a subspace spanned by the set of eigenvectors $\{\boldsymbol{q}_i | i \in \mathcal{M}\}$, i. e. the eigenvectors of $\boldsymbol{R}_{\boldsymbol{y}_0}$ with the largest CS metric. Therefore, the selection criterion considers the cross-correlation vector $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$, but note that the spanned subspace is not the Krylov subspace $\mathcal{K}^{(D)}\left(\boldsymbol{R}_{\boldsymbol{y}_0}, \boldsymbol{r}_{\boldsymbol{y}_0,s_0}\right)$ used by the MSNWF. Unfortunately, the complexity for the CS algorithm is $O\left(N^3\right)$ because all eigenvectors of the eigenvector decomposition have to be computed before we can select the vectors with the largest CS metrics. In Section 6.3 we will see the performance of the PC and the CS method compared to the CG implementation of the MSNWF.

6. Application to an EDGE System

6.1 Transmission System

In the following of this thesis, we apply the derived linear equalizer filters to an *Enhanced Data rates for GSM Evolution* (EDGE) *system* depicted in Figure 6.1. The EDGE radio interface can be used in existing GSM systems in order to increase data rates while reusing the same frequency bands and burst structure. EDGE was standardized [15] as Phase 2+ of GSM by the *European Telecommunications Standards Institute* (ETSI).

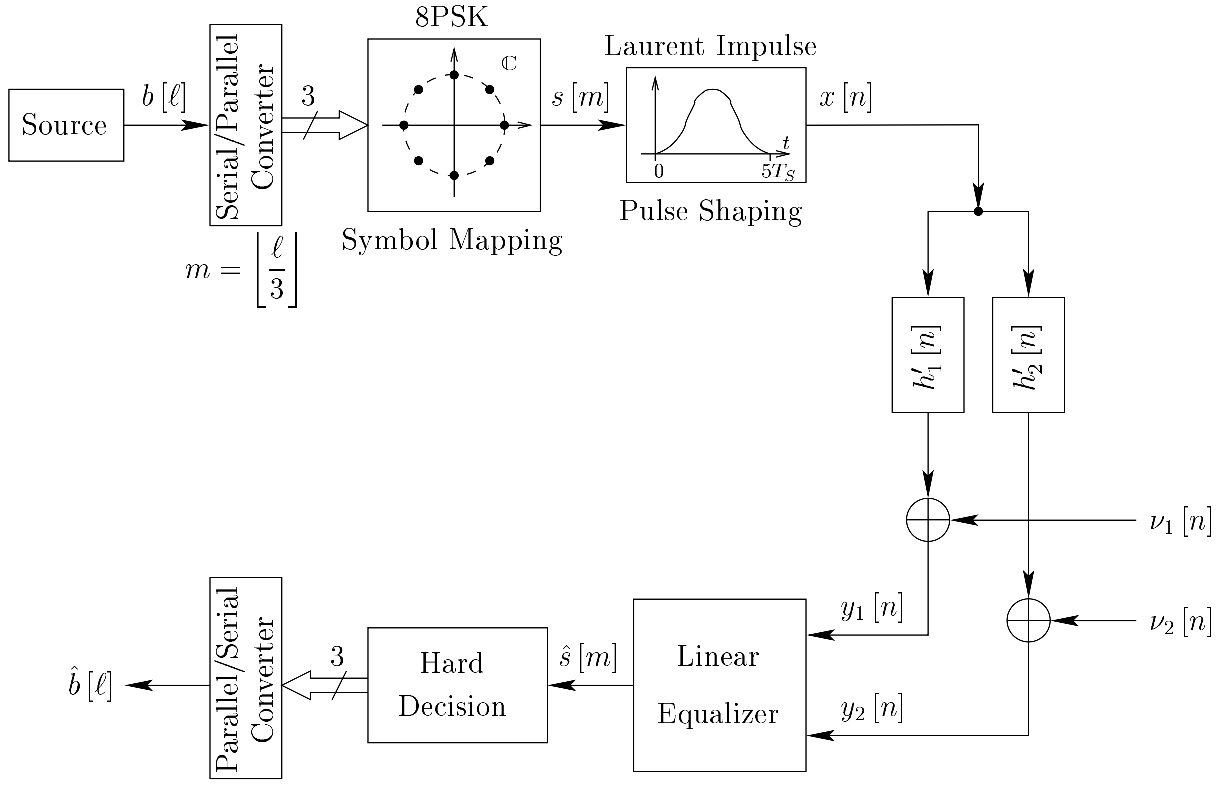

Fig. 6.1. EDGE Transmission System

First, the data bits $b[\ell] \in \{0, 1\}$ of the source are mapped to the 8PSK symbols $s[m] \in \mathbb{C}$ in the way given by Table 6.1. Due to the serial/parallel-converter, the bit clock is three times the symbol clock which is expressed by the different arguments ℓ and m. The symbols are grouped in the burst structure given by Figure 6.2. The duration of an EDGE burst is approximately 577 μs and it contains 116 data symbols, 26 known training symbols, and 3 tail symbols at the beginning and the end, respectively. The guard period at the end of the burst lasts 8.25 symbol times.

Data Bits			Symbol
$b[3m]$	$b[3m+1]$	$b[3m+2]$	$s[m]$
1	1	1	1
0	1	1	$1/\sqrt{2}+\mathrm{j}/\sqrt{2}$
0	1	0	j
0	0	0	$-1/\sqrt{2}+\mathrm{j}/\sqrt{2}$
0	0	1	-1
1	0	1	$-1/\sqrt{2}-\mathrm{j}/\sqrt{2}$
1	0	0	$-\mathrm{j}$
1	1	0	$1/\sqrt{2}-\mathrm{j}/\sqrt{2}$

Table 6.1. 8PSK Symbol Mapping

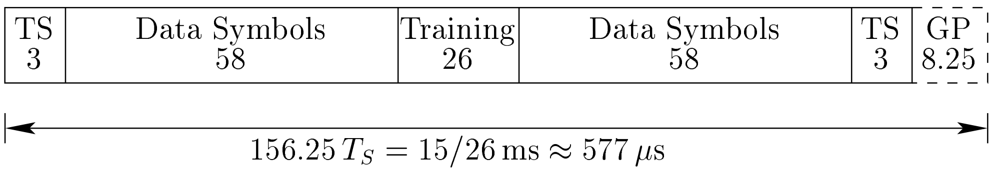

Fig. 6.2. Normal EDGE Burst (TS: Tail Symbols, GP: Guard Period)

Then, the 8PSK symbols modulate the linear pulse shaping filter which is the linearized GMSK impulse

$$g(t) = \begin{cases} \prod_{i=0}^{3} R(t+iT_\mathrm{S}) & \text{for } 0 \le t \le 5T_\mathrm{S}, \\ 0 & \text{else,} \end{cases} \tag{6.1}$$

where

$$R(t) = \begin{cases} \sin\left(\pi \int_0^t S(t')\,dt'\right) & \text{for } 0 \le t \le 4T_\mathrm{S}, \\ \sin\left(\frac{\pi}{2} - \pi \int_0^{t-4T_\mathrm{S}} S(t')\,dt'\right) & \text{for } 4T_\mathrm{S} \le t \le 8T_\mathrm{S}, \\ 0 & \text{else,} \end{cases} \tag{6.2}$$

with

$$S(t) = \frac{1}{2T_\mathrm{S}}\left(Q\left(0.6\,\pi\frac{t-\frac{3}{2}T_\mathrm{S}}{T_\mathrm{S}\sqrt{\ln 2}}\right) - Q\left(0.6\,\pi\frac{t-\frac{5}{2}T_\mathrm{S}}{T_\mathrm{S}\sqrt{\ln 2}}\right)\right). \tag{6.3}$$

The function $Q(t)$ is the Gaussian error integral

$$Q(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{t} e^{-\frac{t'^2}{2}}\,dt', \tag{6.4}$$

and $T_S = 48/13\,\mu s \approx 3.69\,\mu s$ is the symbol duration. Figure 6.3 shows the impulse $g(t)$ for $t \in [0, 5T_S]$, which is the main component in a Laurent decomposition [16] of GMSK modulation and contains more than 99.5% of signal energy. It can be seen that $g(t)$ is unequal zero for five symbol times. Thus, we have severe inter-symbol interference even for flat fading channels.

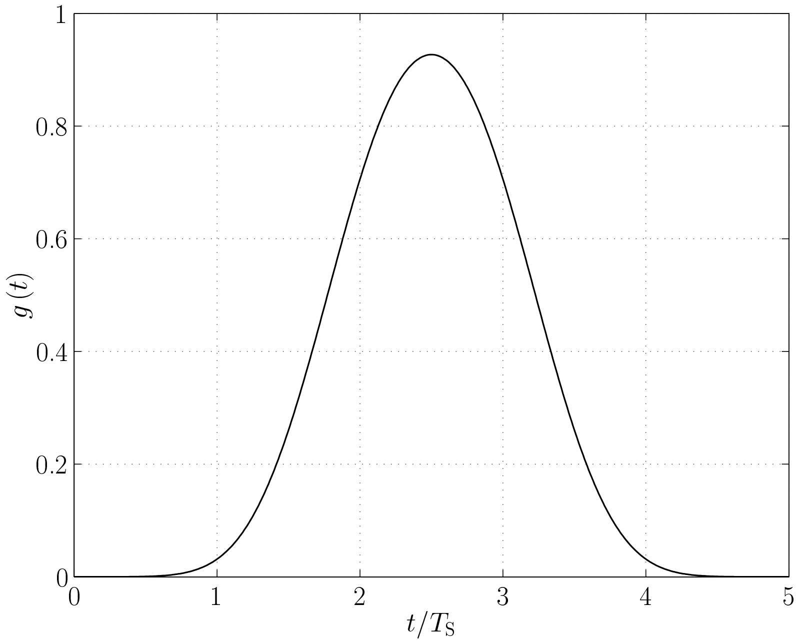

Fig. 6.3. Linearized GMSK impulse

The base band signal

$$x[n] = \sum_m s[m]\, g\left(\left(\frac{n}{\kappa} - m\right) T_S\right), \tag{6.5}$$

where κ is the oversampling factor, propagates over Rayleigh multi-path fading channels (cf. Figure 6.1) with the impulse responses of length L', $h_1'[n]$ and $h_2'[n]$, respectively. Besides, $\nu_1[n]$ and $\nu_2[n]$ denote additive white Gaussian noise.

The signals $y_1[n]$ and $y_2[n]$, received from two antennas, are processed by a linear equalizer filter to get an estimation $\hat{s}[m]$ of the transmitted symbol $s[m]$. In the simulations of Section 6.3, we will use the MMSE, PC, and CS equalizer to have a comparison to the CG implementation of the MSNWF. Note that the linear equalizer block in Figure 6.1 includes an estimation of the statistics which is explained in Section 6.2. Finally, a hard decision followed by a parallel/serial-converter yields an estimation $\hat{b}[\ell]$ of the data bits $b[\ell]$ which are compared to the transmitted data bits to compute the *bit error rate* (BER).

6.2 Estimation Techniques

All the equalizer filters mentioned above require an estimation of the covariance matrix of the observation $\boldsymbol{y}_0[n]$ and some of them need in addition an estimate of the cross-correlation vector between the observation $\boldsymbol{y}_0[n]$ and the desired signal $s_0[n]$. In this section, we present two possible techniques to estimate these second order statistics.

The first way to estimate the statistics is *correlation*. Therefore, it is necessary to define the structure of the observation vector $\boldsymbol{y}_0[n] \in \mathbb{C}^N$. If we set

$$\begin{aligned}
\boldsymbol{y}_{0,1}[n] &= \begin{bmatrix} y_1[n] & y_1[n+1] & \cdots & y_1\left[n+\frac{N}{2}-1\right] \end{bmatrix}^{\mathrm{T}}, \\
\boldsymbol{y}_{0,2}[n] &= \begin{bmatrix} y_2[n] & y_2[n+1] & \cdots & y_2\left[n+\frac{N}{2}-1\right] \end{bmatrix}^{\mathrm{T}}, \\
\boldsymbol{y}_0[n] &= \begin{bmatrix} \boldsymbol{y}_{0,1}[n] \\ \boldsymbol{y}_{0,2}[n] \end{bmatrix},
\end{aligned} \tag{6.6}$$

the cross-correlation vector $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$ is estimated by

$$\boxed{\hat{\boldsymbol{r}}^{(\mathrm{C})}_{\boldsymbol{y}_0,s_0} = \frac{1}{K_r}\sum_{k=0}^{K_r-1} \boldsymbol{y}_0[k]\, s_0^*[k],} \tag{6.7}$$

and the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$ by

$$\boxed{\hat{\boldsymbol{R}}^{(\mathrm{C})}_{\boldsymbol{y}_0} = \frac{1}{K_R}\sum_{k=0}^{K_R-1} \boldsymbol{y}_0[k]\, \boldsymbol{y}_0^{\mathrm{H}}[k].} \tag{6.8}$$

K_R is the number of samples available to estimate $\boldsymbol{R}_{\boldsymbol{y}_0}$ and K_r is the number of samples for the estimation of $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$. Note that K_r is also the number of training symbols because the knowledge of the desired signal $s_0[n]$ is necessary (cf. Equation 6.7) in order to compute $\hat{\boldsymbol{r}}^{(\mathrm{C})}_{\boldsymbol{y}_0,s_0}$. In general, $K_R > K_r$ since the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$ can also be estimated without the knowledge of training symbols.

The second technique estimates the channel via *least squares method* and uses the result to compute an estimate of the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$ and the cross-correlation vector $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$. Before we apply the least squares method, it is useful to derive a matrix-vector representation of the transmission system. In the sequel of this section, the index $i \in \{1,2\}$ indicates the antenna to which the term corresponds. Note that the following formulas can easily be generalized for an arbitrary number of antennas.

First, we define the channel vector $\boldsymbol{h}'_i$ as

$$\boldsymbol{h}'_i = \begin{bmatrix} h'_i[0] & h'_i[1] & \cdots & h'_i[L'-1] \end{bmatrix}^{\mathrm{T}}. \tag{6.9}$$

Then, we introduce a new channel vector $\boldsymbol{h}_i \in \mathbb{C}^L$, $L = L' + 5\kappa$, which also includes the sampled pulse shaping filter $g(t)$, i. e.

$$\boldsymbol{h}_i = \boldsymbol{\mathcal{G}}\boldsymbol{h}'_i. \tag{6.10}$$

The pulse shaping matrix $\boldsymbol{\mathcal{G}} \in \mathbb{C}^{L\times L'}$ is the convolution (Toeplitz) matrix

$$\boldsymbol{\mathcal{G}} = \begin{bmatrix} g[0] & & & \\ g[1] & g[0] & & \\ \vdots & g[1] & \ddots & \\ g[5\kappa] & \vdots & \ddots & g[0] \\ & g[5\kappa] & & g[1] \\ & & \ddots & \vdots \\ & & & g[5\kappa] \end{bmatrix}, \quad g[k] = g\left(\frac{k}{\kappa}T_{\mathrm{S}}\right), \quad k \in \{0,1,\ldots,5\kappa\}. \tag{6.11}$$

Another representation of $\boldsymbol{\mathcal{G}}$ may be obtained by defining the $L \times L$ nilpotent shift matrix

$$\boldsymbol{\Gamma}_L = \begin{bmatrix} \mathbf{0}_{L-1}^{\mathrm{T}} & 0 \\ \mathbf{1}_{L-1} & \mathbf{0}_{L-1} \end{bmatrix}, \tag{6.12}$$

where $\mathbf{0}_{L-1}$ is the $(L-1) \times 1$ zero vector and $\mathbf{1}_{L-1}$ is the $(L-1) \times (L-1)$ identity matrix. Defining $\boldsymbol{\Gamma}_L^0 = \mathbf{1}_L$ and the inverse $\boldsymbol{\Gamma}_L^{-1} = \boldsymbol{\Gamma}_L^{\mathrm{T}}$, yields finally

$$\boldsymbol{\mathcal{G}} = \left(\sum_{k=0}^{5\kappa} g\left[k\right] \boldsymbol{\Gamma}_L^k \right) \begin{bmatrix} \mathbf{1}_{L'} \\ \mathbf{0}_{5\kappa \times L'} \end{bmatrix}, \tag{6.13}$$

with the $5\kappa \times L'$ zero matrix $\mathbf{0}_{5\kappa \times L'}$.

The transmission of the training sequence over the system given by Figure 6.1 can be expressed by the following equation:

$$\boldsymbol{y}_i = \boldsymbol{H}_i^{(K_r)} \boldsymbol{s} + \boldsymbol{\nu}_i, \qquad i \in \{1, 2\}. \tag{6.14}$$

The vector $\boldsymbol{s} \in \mathbb{C}^{K_r}$ contains all the training symbols of one burst, i. e.

$$\boldsymbol{s} = \begin{bmatrix} s\left[n_{\mathrm{t}}\right] & s\left[n_{\mathrm{t}}+1\right] & \cdots & s\left[n_{\mathrm{t}}+K_r-1\right] \end{bmatrix}^{\mathrm{T}}, \tag{6.15}$$

where the sequence begins at the $(n_{\mathrm{t}}+1)$-th symbol of the burst. The noise vector $\boldsymbol{\nu}_i \in \mathbb{C}^{\kappa(K_r-1)+1}$, added to branch i, is defined as

$$\boldsymbol{\nu}_i = \begin{bmatrix} \nu_i\left[n_{\mathrm{t}}\right] & \nu_i\left[n_{\mathrm{t}}+1\right] & \cdots & \nu_i\left[n_{\mathrm{t}}+\kappa\left(K_r-1\right)\right] \end{bmatrix}^{\mathrm{T}}, \tag{6.16}$$

and the received signal vector $\boldsymbol{y}_i \in \mathbb{C}^{\kappa(K_r-1)+1}$ at the i-th antenna as

$$\boldsymbol{y}_i = \begin{bmatrix} y\left[n_{\mathrm{t}}\right] & y\left[n_{\mathrm{t}}+1\right] & \cdots & y\left[n_{\mathrm{t}}+\kappa\left(K_r-1\right)\right] \end{bmatrix}^{\mathrm{T}}. \tag{6.17}$$

Finally, the channel matrix $\boldsymbol{H}_i^{(K_r)} \in \mathbb{C}^{(\kappa(K_r-1)+1)\times K_r}$ may be written as

$$\boldsymbol{H}_i^{(K_r)} = \begin{bmatrix} \boldsymbol{h}_i^{(1)} & \boldsymbol{h}_i^{(2)} & \cdots & \boldsymbol{h}_i^{(K_r)} \end{bmatrix}, \quad \boldsymbol{h}_i^{(j)} = \boldsymbol{\mathcal{H}}_i^{(K_r)} \boldsymbol{e}_{\kappa(j-1)+1}, \quad j \in \{1, 2, \ldots, K_r\}, \tag{6.18}$$

where

$$\boldsymbol{\mathcal{H}}_i^{(K_r)} = \sum_{k=0}^{L-1} \boldsymbol{h}_i^{\mathrm{T}} \boldsymbol{e}_k \boldsymbol{\Gamma}_{\kappa(K_r-1)+1}^{k-(L-1)} \in \mathbb{C}^{(\kappa(K_r-1)+1)\times(\kappa(K_r-1)+1)} \tag{6.19}$$

is a convolution matrix similar to $\boldsymbol{\mathcal{G}}$ (cf. Equation 6.11), and $\boldsymbol{e}_{\kappa(j-1)+1}$ or $\boldsymbol{e}_k$ is a unit norm vector with a one at the $(\kappa\left(j-1\right)+1)$-th or k-th position, respectively. Compared to the matrix $\boldsymbol{\mathcal{H}}_i^{(K_r)}$, the channel matrix $\boldsymbol{H}_i^{(K_r)}$ considers the fact that the received signals $y_1\left[n\right]$ and $y_2\left[n\right]$ in Figure 6.1 are oversampled with the factor κ while the symbols $s\left[m\right]$ are only sampled in times of the symbol duration T_{S}.

If we use the training symbols of one burst in order to define the symbol matrix $\boldsymbol{S} \in \mathbb{C}^{(\kappa(K_r-1)+1)\times L}$ as

$$\boldsymbol{S} = \left(\sum_{k=0}^{K_r-1} s\left[n_{\mathrm{t}}+k\right] \boldsymbol{\Gamma}_{\kappa(K_r-1)+1}^{\kappa k+1-L} \right) \begin{bmatrix} \mathbf{1}_L \\ \mathbf{0}_{(\kappa(K_r-1)+1-L)\times L} \end{bmatrix}, \tag{6.20}$$

and remember Equation (6.10), Equation (6.14) may be transformed as follows

$$\boldsymbol{y}_i = \boldsymbol{H}_i^{(K_r)} \boldsymbol{s} + \boldsymbol{\nu}_i = \boldsymbol{S}\boldsymbol{h}_i + \boldsymbol{\nu}_i = \boldsymbol{S}\boldsymbol{\mathcal{G}}\boldsymbol{h}_i' + \boldsymbol{\nu}_i. \tag{6.21}$$

Thus, the channel vector $\boldsymbol{h}_i$ can be estimated applying the least squares method [1], i. e.

$$\hat{\boldsymbol{h}}_i = \boldsymbol{\mathcal{G}}\hat{\boldsymbol{h}}_i' = \boldsymbol{\mathcal{G}} \left(\boldsymbol{S}\boldsymbol{\mathcal{G}}\right)^{\dagger} \boldsymbol{y}_i. \tag{6.22}$$

In order to compute finally $\hat{\boldsymbol{R}}_{\boldsymbol{y}_0}^{(\mathrm{LS})}$ and $\hat{\boldsymbol{r}}_{\boldsymbol{y}_0,s_0}^{(\mathrm{LS})}$ from the channel vector $\hat{\boldsymbol{h}}_i$, it is necessary to calculate the channel matrix $\hat{\boldsymbol{H}}_i^{(M)} \in \mathbb{C}^{N\times M}$, $M = \lfloor (N-1)/\kappa \rfloor + 1$, first. Note that $\boldsymbol{H}_i^{(M)}$ generates the observation vector $\boldsymbol{y}_0[n]$ and not the received training sequence as $\boldsymbol{H}_i^{(K_r)}$ does. To this end, we replace $\boldsymbol{h}_i$ and K_r in Equation (6.19) by $\hat{\boldsymbol{h}}_i$ and M, respectively, and plug the result $\hat{\boldsymbol{\mathcal{H}}}_i^{(M)}$ in Equation (6.18). If we build the matrix $\hat{\boldsymbol{H}}^{(M)}$ by using the resulting matrices $\hat{\boldsymbol{H}}_1^{(M)}$ and $\hat{\boldsymbol{H}}_2^{(M)}$

$$\hat{\boldsymbol{H}}^{(M)} = \begin{bmatrix} \hat{\boldsymbol{H}}_1^{(M)} \\ \hat{\boldsymbol{H}}_2^{(M)} \end{bmatrix}, \tag{6.23}$$

the estimation of the cross-correlation vector $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$ is simply the d-th column of $\hat{\boldsymbol{H}}^{(M)}$, i. e.

$$\boxed{\hat{\boldsymbol{r}}_{\boldsymbol{y}_0,s_0}^{(\mathrm{LS})} = \hat{\boldsymbol{H}}^{(M)} \boldsymbol{e}_d,} \tag{6.24}$$

where d is a integer symbol delay. Assuming white Gaussian noise, uncorrelated transmitted symbols and a signal power of one yields the estimate of the covariance matrix

$$\boxed{\hat{\boldsymbol{R}}_{\boldsymbol{y}_0}^{(\mathrm{LS})} = \frac{1}{\widehat{\mathrm{SNR}}} \mathbf{1}_N + \hat{\boldsymbol{H}}^{(M)} \hat{\boldsymbol{H}}^{(M),\mathrm{H}}.} \tag{6.25}$$

In Equation (6.25), we see that it remains to estimate the SNR at the receiver. The estimation of the noise vector $\boldsymbol{\nu}_i$ is the error of the least squares method in Equation (6.22). It holds for $i \in \{1, 2\}$

$$\hat{\boldsymbol{\nu}}_i = \boldsymbol{y}_i - \boldsymbol{S}\hat{\boldsymbol{h}}_i, \tag{6.26}$$

and thus, the remaining term in Equation (6.25), $\widehat{\mathrm{SNR}}$, may be written as

$$\widehat{\mathrm{SNR}} = \frac{2\kappa \left(K_r - 1\right) + 1}{\hat{\boldsymbol{\nu}}_1^{\mathrm{H}} \hat{\boldsymbol{\nu}}_1 + \hat{\boldsymbol{\nu}}_2^{\mathrm{H}} \hat{\boldsymbol{\nu}}_2}. \tag{6.27}$$

In the next section, we apply the equalizers derived in Chapters 2, 4, and 5 to an EDGE system using the estimates given by Equations (6.7) and (6.8), or Equations (6.24) and (6.25), respectively.

6.3 Simulation Results

Recall the EDGE transmission system defined in Section 6.1. We sample two times during one symbol duration, i. e. $\kappa = 2$, and take 20 samples of the received signal at each antenna to build the space-time observation vector $\boldsymbol{y}_0[n]$, thus, its dimension $N = 40$. The length of the channel vector $L' = 7$ if we assume a delay spread of $\tau_{\max} = 10\,\mu$s or approximately three symbol times, hence, the channel vector which includes the pulse shaping filter, $\boldsymbol{h}_i$, has length $L = 17$. Besides, the channel is constant during one burst.

Simulations show that the CG implementation of the MSNWF given by Algorithm 4.1 yields exactly the same results as the Lanczos based MSNWF. This confirms the transformation we made in Section 4.1 and in the following simulations, we may restrict ourselves on the investigation of the CG based MSNWF.

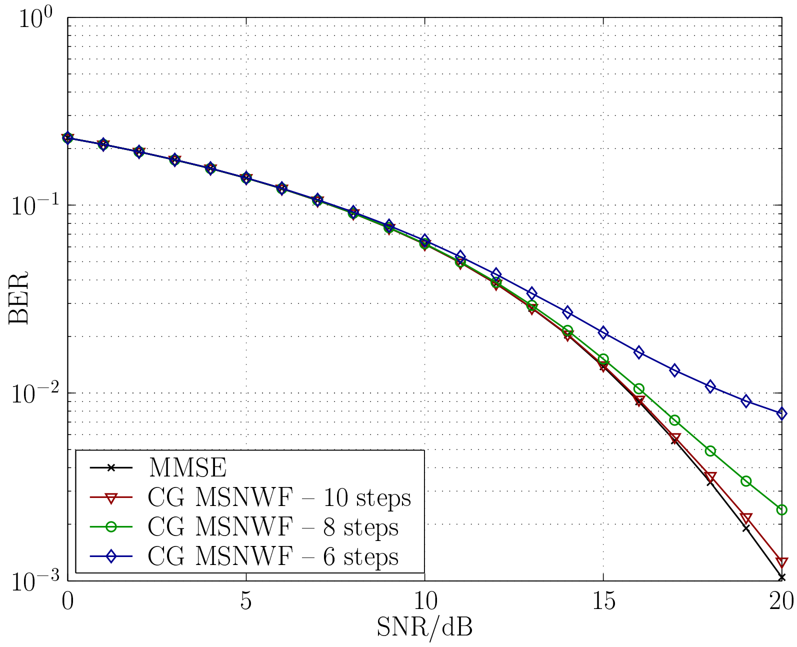

Fig. 6.4. BER for known statistics using CG MSNWF equalizer

For comparison purposes, we first analyze a known channel. Thus, the exact statistics, i. e. the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$ and the cross-correlation vector $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$ are available. Figure 6.4 shows the resulting BER using the CG based MSNWF with $D \in \{6, 8, 10\}$ steps compared to the MMSE equalizer or Wiener filter which corresponds to the MSNWF with $D = 40$ steps. We observe that the MSNWF with $D = 10$ steps is very close to the MMSE equalizer even for high SNR values.

In order to verify the dependence of the BER on the number of iteration steps D, we computed Figure 6.5, where we used a SNR of 15 dB. Again, for $D > 9$, the CG implementation of the MSNWF yields almost the same result as the MMSE equalizer, however, the MMSE solution is always better than every rank D MSNWF.

In Figure 6.6, the CG based MSNWF is compared to the CS and PC equalizer which we explained in Chapter 5. Except for the MMSE filter, all equalizers in Figure 6.6 find the approximate solution of the Wiener-Hopf equation in a subspace of dimension $D = 8$. We see that the MSNWF yields the best approximation of the MMSE equalizer or the Wiener filter.

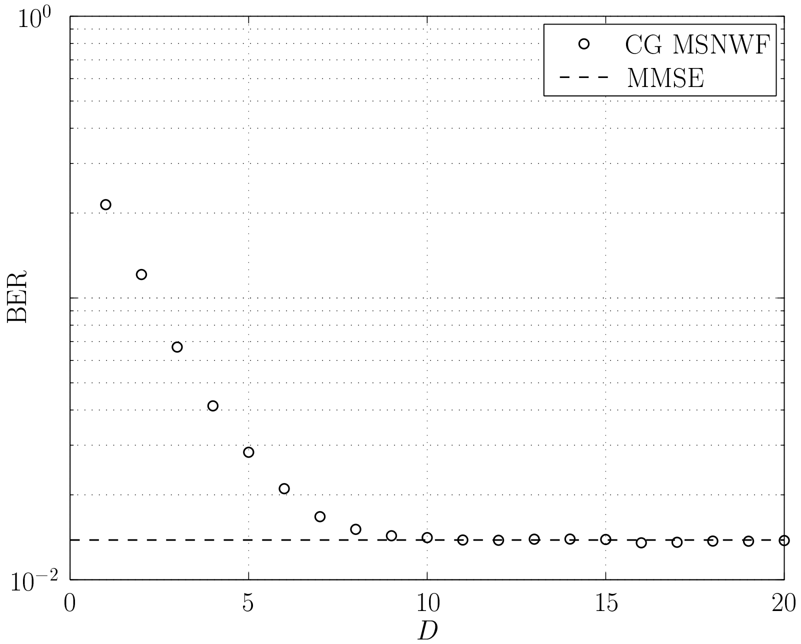

Fig. 6.5. BER for known statistics using CG MSNWF equalizer (SNR = 15 dB)

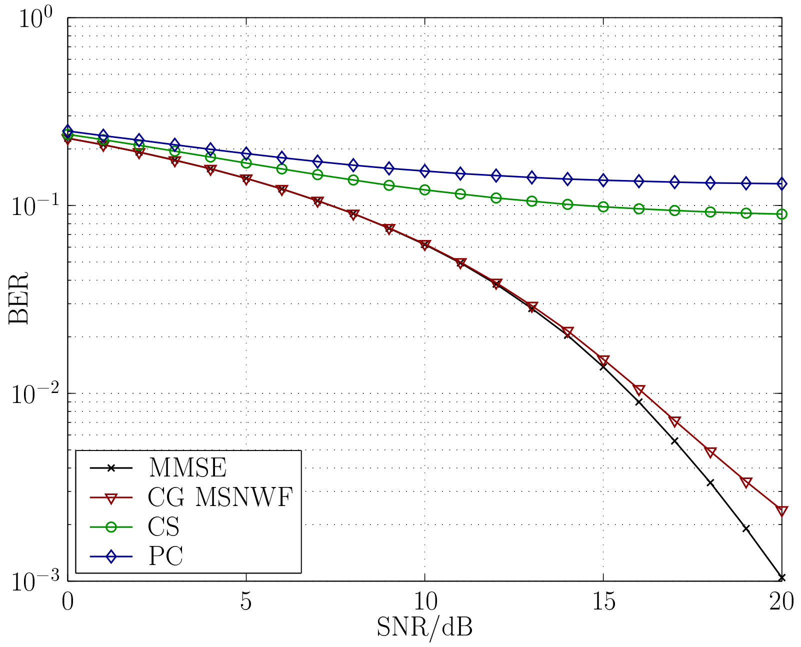

Fig. 6.6. BER for known statistics using different equalizers ($D = 8$)

Next, we estimate the statistics with correlation, take $K_r = 26$ training symbols beginning at the 62-nd symbol of a burst, i. e. $n_t = 61$, and use Equation (6.7) to compute the estimate of the cross-correlation vector, $\hat{\boldsymbol{r}}^{(C)}_{\boldsymbol{y}_0,s_0}$. Equation (6.8) yields an estimate of the covariance matrix, $\hat{\boldsymbol{R}}^{(C)}_{\boldsymbol{y}_0}$, if we set $K_R = 148$ symbols. We get the simulation result depicted in Figure 6.7. Compared to the simulation where we assumed a known channel (cf. Figure 6.4), the CG based MSNWF with $D \in \{2, 4, 6\}$ steps is now always better than the MMSE filter. This confirms also Figure 6.8 where the BER depending on the number of iteration steps D is

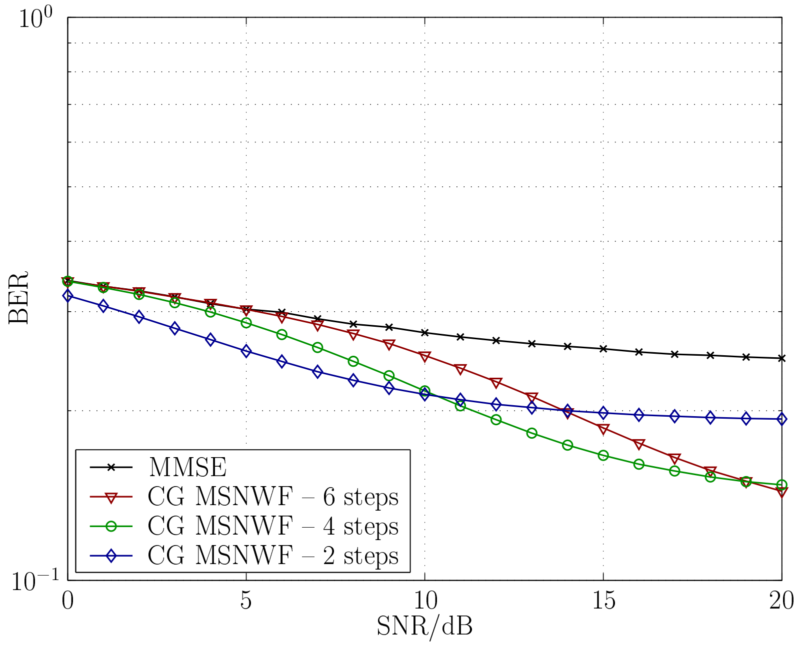

Fig. 6.7. BER for estimated statistics (correlation) using CG MSNWF equalizer

shown (SNR $= 15\,\mathrm{dB}$). Moreover, we observe that every rank D MSNWF performs better than the MMSE equalizer and that the best solution is obtained for $D = 4$. Unfortunately, there exists no criterion to stop the algorithm at the iteration step with the optimum solution. It remains to explain the fact that the BER increases again for $D > 4$. Recall the eigenvector decomposition of $\boldsymbol{R}_{\boldsymbol{y}_0}$ and note that the small eigenvalues of $\boldsymbol{R}_{\boldsymbol{y}_0}$ have the largest estimation errors. Due to the inversion of $\boldsymbol{R}_{\boldsymbol{y}_0}$ in the Wiener filter, especially these eigenvalues determine the filter coefficients and cause the bad behavior.

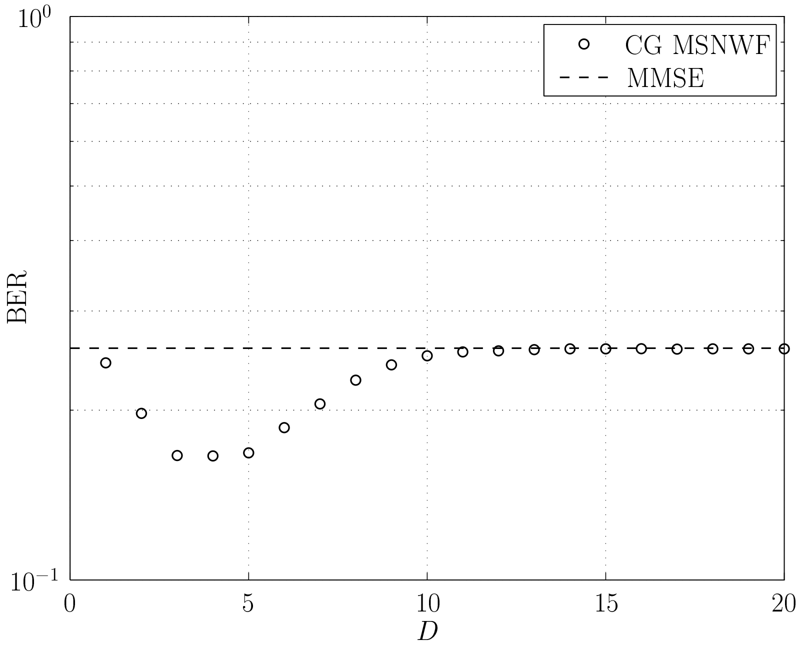

Fig. 6.8. BER for estimated statistics (correlation) using CG MSNWF equalizer (SNR $= 15\,\mathrm{dB}$)

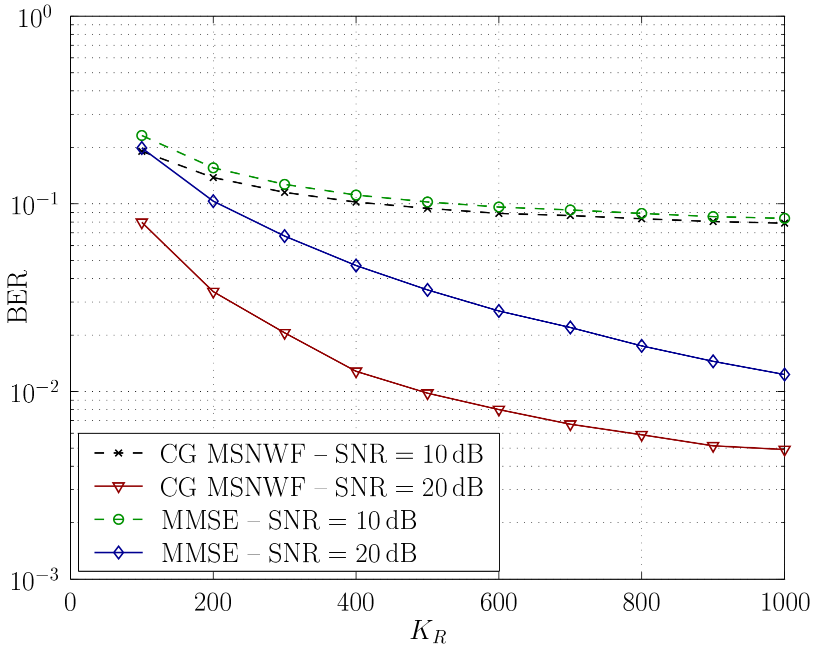

Fig. 6.9. BER for known $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$ and estimated $\boldsymbol{R}_{\boldsymbol{y}_0}$ (correlation) using MMSE or CG MSNWF equalizer with $D = 8$ steps, respectively

The BER in Figure 6.7 is much higher compared to Figure 6.4 where we knew the statistics. Figure 6.9, where $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$ is assumed to be known and only the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$ is estimated by correlation, shows that especially the estimation of the covariance matrix causes this problem. The BER of the CG MSNWF with $D = 8$ steps and SNR $= 15\,\mathrm{dB}$ in Figure 6.8 is only a little bit higher than the BER of the CG MSNWF for $K_R = 148$ and SNR $= 15\,\mathrm{dB}$ in Figure 6.9. Figure 6.9 shows also that the number of samples K_R used for the estimation of $\boldsymbol{R}_{\boldsymbol{y}_0}$, has to be much larger than 148 to get a low BER. However, the CG based MSNWF achieves the same BER as the MMSE with much less samples K_R.

In order to improve the result shown in Figure 6.7, we apply the least squares method to estimate the channel and from it the statistics as described in Section 6.2. Figure 6.10 shows the results for the CG based MSNWF with $D \in \{6, 8, 10\}$ steps and the MMSE equalizer. We see that the least squares method yields much lower bit error rates than using correlation. As in Figure 6.4, the MSNWF with $D = 10$ steps behaves almost the same as the MMSE filter.

Finally, Figure 6.11 shows a comparison of the CG MSNWF algorithm to the PC and CS equalizers. Again, the MSNWF is much better compared to the other reduced dimension methods.

Note that all figures show the raw BER. Due to channel coding, an uncoded BER of 10^{-1} results in acceptable speech transmission.

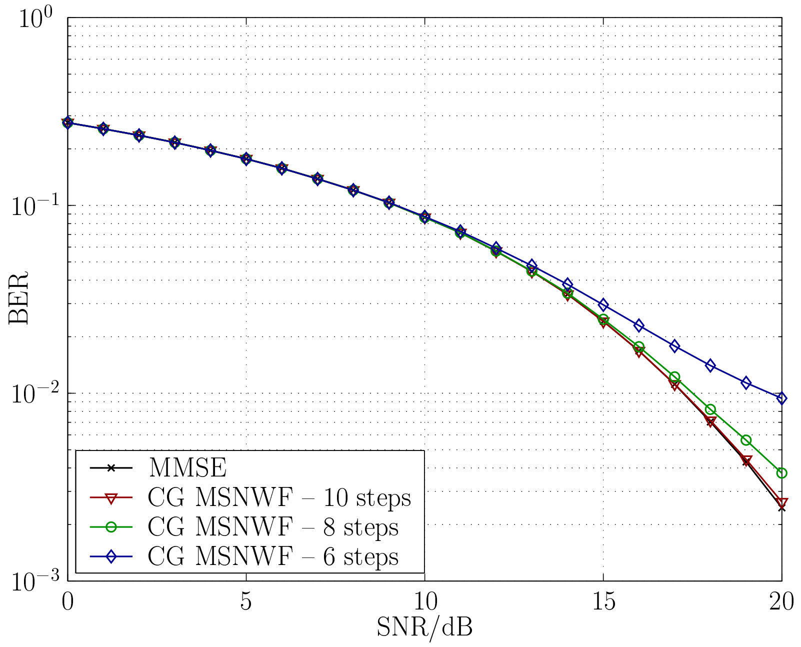

Fig. 6.10. BER for estimated statistics (least squares method) using CG MSNWF equalizer

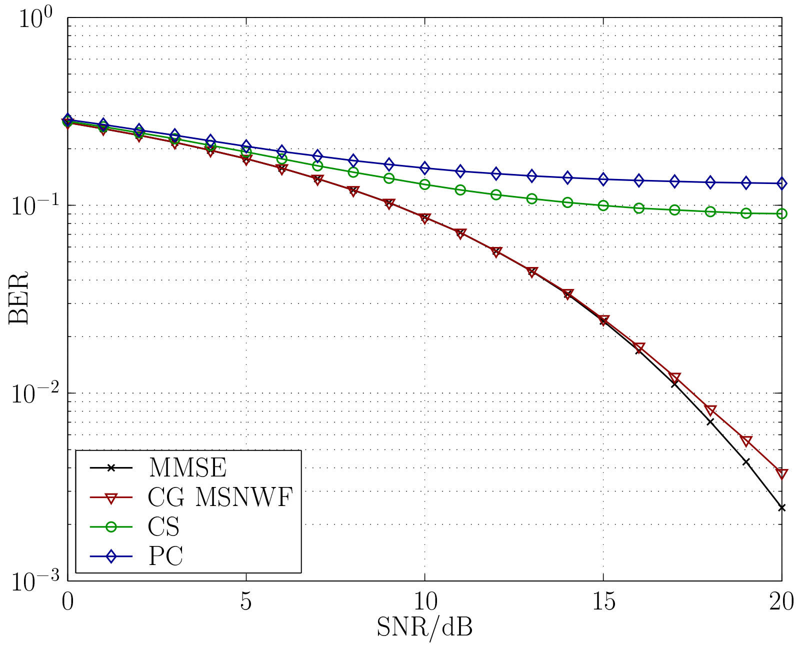

Fig. 6.11. BER for estimated statistics (least squares method) using different equalizers ($D = 8$)

7. Conclusion and Outlook

In this thesis, we derived the relationship between the Lanczos based implementation of the MSNWF and the CG method. Both are methods to approximate a solution of a linear equation system, which is in our case the Wiener-Hopf equation, searching in a subspace of reduced dimension. A new implementation of the MSNWF is obtained by transforming its formulas into those of the CG algorithm.

Then, we applied the obtained algorithm to an EDGE transmission system and compared it to other reduced dimension methods, i. e. the PC and CS algorithm. If we assumed known statistics, simulation results showed that despite the reduced computational complexity, the CG based MSNWF yields almost the same result as the MMSE equalizer and moreover, it performs much better than the PC or CS filter. Although the results deteriorated if we estimated the statistics using correlation, we observed that the CG MSNWF performs better than the MMSE equalizer for every number of iteration steps D. Finally, we used the least squares method to estimate the channel and the statistics. The simulation results could be improved and again, the CG implementation of the MSNWF is a good approximation of the Wiener filter.

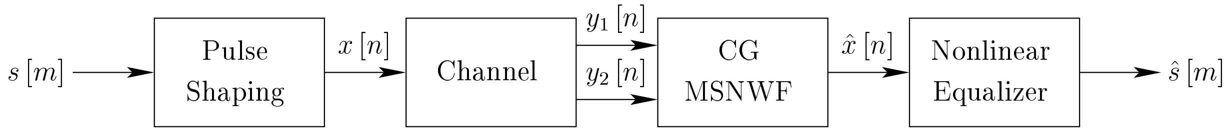

Fig. 7.1. Nonlinear Equalizer with CG based MSNWF as Pre-Filter

Another way to improve the BER in an EDGE system is depicted in Figure 7.1 [17]. We propose to use the CG MSNWF as a pre-filter in order to estimate the base band signal $x[n]$ from the received signals $y_1[n]$ and $y_2[n]$, respectively, using the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$ and the cross-correlation vector $\boldsymbol{r}_{\boldsymbol{y}_0,x_0}$. Note that the cross-correlation vector can be computed using the training symbols and the known pulse shaping filter. Then, the estimated base band signal, $\hat{x}[n]$, is processed by a nonlinear equalizer, e. g. a Viterbi equalizer to remove the severe inter-symbol interference due to pulse shaping, which causes the high BER in the case of linear processing. The nonlinear filter estimates the desired symbol $s[m]$ from the estimated base band signal. Because of 8PSK-modulation and a pulse shaping filter containing $4\kappa+1$ samples, the number of states of a Viterbi equalizer is $8^{4\kappa}$. This high number of states causes a huge trellis structure and is therefore difficult to implement. Applying a suboptimal reduced state Viterbi equalizer seems to be an acceptable alternative.

Appendix

A1 Abbreviations

BER	Bit Error Rate
C	Correlation
CD	Conjugate Directions
CDMA	Code Division Multiple Access
CG	Conjugate Gradients
CS	Cross Spectral
EDGE	Enhanced Data rates for GSM Evolution
ETSI	European Telecommunications Standards Institute
GMSK	Gaussian Minimum Shift Keying
GP	Guard Period
GPS	Global Positioning System
GSM	Global System for Mobile communication
LS	Least Squares
MMSE	Minimum Mean Square Error
MSE	Mean Square Error
MSNWF	Multi-Stage Nested Wiener Filter
PSK	Phase Shift Keying
TS	Tail Symbols

A2 Symbols

$\lvert\cdot\rvert$	absolute value
$\lVert\cdot\rVert_2$	Euclidean norm
$\lVert\cdot\rVert_{\boldsymbol{A}}$	$\boldsymbol{A}$-norm
$(\cdot)^*$	conjugate complex
$(\cdot)^{\mathrm{T}}$	transpose
$(\cdot)^{\mathrm{H}}$	conjugate transpose
$(\cdot)^{\dagger}$	pseudo inverse
$(\cdot)^{(D)}$	term belongs to reduced dimension D
$\mathbf{1}_i$	$i \times i$ identity matrix

$\mathbf{0}_i$	$i \times 1$ zero vector
$\mathbf{0}_{i \times j}$	$i \times j$ zero matrix
α_i	real valued factor at stage i of MSNWF
β_i	real valued variable used at iteration step i of the Lanczos based MSNWF
γ_i	real valued step length at iteration step i of CG algorithm
γ_i'	arbitrary real valued step length at iteration step i
$\mathbf{\Gamma}_i$	$i \times i$ nilpotent matrix
δ_i	real valued step improvement at iteration step i of CG algorithm
$\varepsilon_0 [n]$	error between desired signal and its estimate
$\zeta_i, \varphi_i, \chi_i$	real valued factors of linear combinations
η_i	real valued step length at iteration step i of the CG based MSNWF
κ	oversampling factor
λ_i	eigenvalue of the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$
$\mathbf{\Lambda}$	diagonal matrix of the eigenvalues of $\boldsymbol{R}_{\boldsymbol{y}_0}$
$\mathbf{\Lambda}_{\mathrm{CS}}^{(D)}$	diagonal matrix of the D eigenvalues of $\boldsymbol{R}_{\boldsymbol{y}_0}$ which fulfill the CS criterion
$\mathbf{\Lambda}_{\mathrm{PC}}^{(D)}$	diagonal matrix of the D largest eigenvalues of $\boldsymbol{R}_{\boldsymbol{y}_0}$
μ_i	Lagrange multiplier
$\nu_i [n]$	additive white Gaussian noise signal added at the i-th antenna
$\boldsymbol{\nu}_i$	additive white Gaussian noise vector added at the i-th antenna
$\hat{\boldsymbol{\nu}}_i$	estimated noise vector added at the i-th antenna
ϱ_i	real valued variable whose absolute value is the length of the residual $\boldsymbol{g}_i$ at iteration step $i-1$ of the CG based MSNWF
σ_i	singular value of blocking matrix
σ_x^2	variance of a signal $x [n]$
ψ_i	real valued step improvement at iteration step i of the CG based MSNWF
$\boldsymbol{A}$	matrix of a linear equation system
$b [n]$	data bit
$\hat{b} [n]$	estimated data bit
$\boldsymbol{b}$	right side of a linear equation system
$\boldsymbol{b}^{(D)}$	vector used for the derivation of the backward recursion
$\boldsymbol{B}_i$	blocking matrix at stage i of MSNWF
$c_{i,j}^{(D)}$	real valued matrix element at the i-th row and the j-th column of $\boldsymbol{C}^{(D)}$
$\boldsymbol{c}_i^{(D)}$	real valued i-th column of $\boldsymbol{C}^{(D)}$
$\boldsymbol{c}_{\mathrm{first}}^{(i)}$	real valued first column of $\boldsymbol{C}^{(i)}$
$\boldsymbol{c}_{\mathrm{last}}^{(i)}$	real valued last column of $\boldsymbol{C}^{(i)}$
$\boldsymbol{C}^{(D)}$	real valued inverse of $\boldsymbol{R}_{\boldsymbol{s}}^{(D)}$
D	reduced dimension

$e(\boldsymbol{x})$	error function
$\boldsymbol{e}_i$	unit norm vector with a one at the i-th position
$\mathrm{E}\{\cdot\}$	expected value
$g(t)$	linear Laurent pulse shaping filter
$g[n]$	sampled Laurent impulse
$\boldsymbol{g}_i$	residual at iteration step $i-1$ of the CG based MSNWF
$\boldsymbol{\mathcal{G}}$	pulse shaping matrix
$h_i'[n]$	channel impulse response of i-th antenna (pulse shaping excluded)
$\boldsymbol{h}_i$	channel vector of i-th antenna (pulse shaping included)
$\hat{\boldsymbol{h}}_i$	estimated channel vector of i-th antenna (pulse shaping included)
$\boldsymbol{h}_i'$	channel vector of i-th antenna (pulse shaping excluded)
$\hat{\boldsymbol{h}}_i'$	estimated channel vector of i-th antenna (pulse shaping excluded)
$\boldsymbol{h}_i^{(m)}$	m-th column of the channel matrix $\boldsymbol{H}_i^{(j)}$ of i-th antenna
$\boldsymbol{H}_i^{(j)}$	$(\kappa(j-1)+1)\times j$ channel matrix of i-th antenna
$\hat{\boldsymbol{H}}_i^{(j)}$	estimated $(\kappa(j-1)+1)\times j$ channel matrix of i-th antenna
$\boldsymbol{\mathcal{H}}_i^{(j)}$	$(\kappa(j-1)+1)\times(\kappa(j-1)+1)$ channel matrix of i-th antenna for oversampled symbol sequence
$\hat{\boldsymbol{\mathcal{H}}}_i^{(j)}$	estimated $(\kappa(j-1)+1)\times(\kappa(j-1)+1)$ channel matrix of i-th antenna for oversampled symbol sequence
i,j,k,ℓ,m,n	integer variables
K_r	number of samples used to estimate the cross-correlation vector $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$
K_R	number of samples used to estimate the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$
$\mathcal{K}^{(i)}(\boldsymbol{A},\boldsymbol{x})$	i-dimensional Krylov subspace of a matrix $\boldsymbol{A}$ and a vector $\boldsymbol{x}$
l_i	absolute value of ϱ_i used at iteration step $i-1$ of the CG based MSNWF
L	dimension of channel vector $\boldsymbol{h}_i$ which includes pulse shaping
L'	length of $h_i'[n]$ or dimension of channel vector $\boldsymbol{h}_i'$, respectively (pulse shaping excluded)
$L(\boldsymbol{t},\mu_1,\ldots)$	Lagrange function
$\boldsymbol{m}_i$	matched filter at stage i of MSNWF
M	number of symbols which produce the oversampled observation vector
MSE_0	mean square error of the linear filter $\boldsymbol{w}$
$\mathrm{MSE}_0^{(D)}$	mean square error of rank D MSNWF
$\mathrm{MSE}_{\mathrm{CS}}^{(D)}$	mean square error of CS filter
MMSE_0	minimum mean square error achieved by the Wiener filter $\boldsymbol{w}_0$
$\mathrm{MMSE}_{\boldsymbol{z}_1}$	minimum mean square error achieved by the Wiener filter $\boldsymbol{w}_{\boldsymbol{z}_1}$
$\mathcal{M}$	set of integers corresponding to the eigenvalues of $\boldsymbol{R}_{\boldsymbol{y}_0}$ which fulfill the CS criterion
$\mathcal{M}'$	arbitrary subset of $\{1,2,\ldots,N\}$
n_{t}	time index of the first symbol of the training sequence within a burst
N	dimension of observation

$O(\cdot)$	Landau symbol
$\boldsymbol{p}_i$	$\boldsymbol{A}$-conjugate vectors at iteration step i of CG algorithm
$\boldsymbol{P}_i$	projector onto the space orthogonal to the pre-filter $\boldsymbol{t}_i$
$Q(t)$	Gaussian error integral
$\boldsymbol{Q}$	modal matrix of the covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$
$r_{i,j}$	matrix element at the i-th row and the j-th column of $\boldsymbol{R}_{\boldsymbol{s}}^{(D)}$
$r_{x,y}$	cross-correlation between a signal $x[n]$ and $y[n]$
$\boldsymbol{r}_i$	residual vector at iteration step $i-1$ of CG algorithm
$\boldsymbol{r}_{\boldsymbol{x},y}$	cross-correlation vector between a vector $\boldsymbol{x}[n]$ and a signal $y[n]$
$\hat{\boldsymbol{r}}_{\boldsymbol{y}_0,s_0}^{(\mathrm{C})}$	estimated cross-correlation vector $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$ (correlation method)
$\hat{\boldsymbol{r}}_{\boldsymbol{y}_0,s_0}^{(\mathrm{LS})}$	estimated cross-correlation vector $\boldsymbol{r}_{\boldsymbol{y}_0,s_0}$ (least squares method)
$R(t)$	function used for the definition of the pulse shaping filter $g(t)$
$\boldsymbol{R}_{\boldsymbol{x}}$	covariance matrix of a vector $\boldsymbol{x}[n]$
$\hat{\boldsymbol{R}}_{\boldsymbol{y}_0}^{(\mathrm{C})}$	estimated covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$ (correlation method)
$\hat{\boldsymbol{R}}_{\boldsymbol{y}_0}^{(\mathrm{LS})}$	estimated covariance matrix $\boldsymbol{R}_{\boldsymbol{y}_0}$ (least squares method)
$s[m]$	data symbol
$\hat{s}[m]$	estimated symbol
$s_0[n]$	desired signal
$\hat{s}_0[n]$	estimate of desired signal
$s_i[n]$	desired signal at stage i of MSNWF
$\hat{s}_i[n]$	estimate of the desired signal at stage i of MSNWF
$\hat{s}_{0,\mathrm{CS}}^{(D)}[n]$	CS estimate of desired signal using a D-dimensional subspace
$\hat{s}_{0,\mathrm{PC}}^{(D)}[n]$	PC estimate of desired signal using a D-dimensional subspace
$\hat{s}_{\boldsymbol{z}_1}[n]$	estimate of the desired signal using the filter $\boldsymbol{w}_{\boldsymbol{z}_1}$
$\boldsymbol{s}$	data symbol vector
$\boldsymbol{s}[n]$	pre-filtered observation vector
$\boldsymbol{s}_{\mathrm{CS}}^{(D)}[n]$	CS pre-filtered $D\times 1$ observation vector
$\boldsymbol{s}_{\mathrm{PC}}^{(D)}[n]$	PC pre-filtered $D\times 1$ observation vector
$S(t)$	function used for the definition of the pulse shaping filter $g(t)$
$\widehat{\mathrm{SNR}}$	estimated SNR
$\boldsymbol{S}$	symbol matrix
t	time variable
$\boldsymbol{t}_i$	i-th pre-filter vector of MSNWF
T_{S}	symbol time
$\boldsymbol{T}_1$	pre-filter matrix
$\boldsymbol{T}^{(D)}$	pre-filter matrix of rank D MSNWF
$\boldsymbol{T}_{\mathrm{CS}}^{(D)}$	CS pre-filter matrix using a D-dimensional subspace
$\boldsymbol{T}_{\mathrm{PC}}^{(D)}$	PC pre-filter matrix using a D-dimensional subspace
$\boldsymbol{u}_i$	$\boldsymbol{R}_{\boldsymbol{y}_0}$-conjugate vectors at iteration step i of CG based MSNWF
$\boldsymbol{v}$	vector used by the Lanczos based MSNWF or the CG based MSNWF
$\boldsymbol{w}$	linear filter

$\boldsymbol{w}_0$	Wiener filter
$\boldsymbol{w}_i$	Wiener filter at stage i of MSNWF
$\boldsymbol{w}_{\boldsymbol{x}}$	Wiener filter which estimates the desired signal from a vector $\boldsymbol{x}[n]$
$\boldsymbol{w}_0^{(D)}$	rank D MSNWF
$\boldsymbol{w}_{0,\mathrm{CS}}^{(D)}$	CS filter using a D-dimensional subspace
$\boldsymbol{w}_{0,\mathrm{PC}}^{(D)}$	PC filter using a D-dimensional subspace
$x[n]$	transmitted base band signal
$\hat{x}[n]$	estimated base band signal
$\boldsymbol{x}$	solution of a linear equation system
$\boldsymbol{x}^{(i)}$	approximate solution of a linear equation system at iteration step i of CG algorithm
$y_i[n]$	received signal at the i-th antenna
$\boldsymbol{y}_i$	received signal vector at the i-th antenna
$\boldsymbol{y}_0[n]$	observation signal
$\boldsymbol{y}_i[n]$	observation signal at stage i of MSNWF
$\boldsymbol{y}_{0,i}[n]$	observation signal at the i-th antenna
$\boldsymbol{z}_1[n]$	pre-filtered observation signal

Bibliography

[1] Scharf, L. L.: Statistical Signal Processing. Addison-Wesley, 1991.

[2] Hotelling, H.: Analysis of a Complex of Statistical Variables into Principal Components. – In: Journal of Educational Psychology **24** (September/October 1933), pp. 417–441, 498–520.

[3] Goldstein, J. S.; Reed, I. S.: Subspace Selection for Partially Adaptive Sensor Array Processing. – In: IEEE Transactions on Aerospace and Electronic Systems **33** (April 1997), pp. 539–543.

[4] Goldstein, J. S.; Reed, I. S.; Scharf, L. L.: A Multistage Representation of the Wiener Filter Based on Orthogonal Projections. – In: IEEE Transactions on Information Theory **44** (November 1998), pp. 2943–2959.

[5] Myrick, W. L.; Zoltowski, M. D.; Goldstein, J. S.: GPS Jammer Suppression with Low-Sample Support Using Reduced-Rank Power Minimization. – In: Proc. SSAP 2000, August 2000. pp. 514–518.

[6] Chowdhury, S.; Zoltowski, M. D.; Goldstein, J. S.: Reduced-Rank Adaptive MMSE Equalization for High-Speed CDMA Forward Link with Sparse Multipath Channels. – In: Proc. 34th Asilomar Conference on Signals, Systems, and Computers, October 2000.

[7] Honig, M. L.; Xiao, W.: Performance of Reduced-Rank Linear Interference Suppression for DS-CDMA. December 1999. submitted to *IEEE Transactions on Information Theory*.

[8] Joham, M.; Zoltowski, M. D.: Interpretation of the Multi-Stage Nested Wiener Filter in the Krylov Subspace Framework. Tech. Rept. TUM-LNS-TR-00-6. Munich University of Technology, November, November 2000. Also: Technical Report TR-ECE-00-51, Purdue University.

[9] Hestenes, M. R.; Stiefel, E.: Methods of Conjugate Gradients for Solving Linear Systems. – In: Journal of Research of the National Bureau of Standards **49** (December 1952), pp. 409–436.

[10] Trefethen, L. N.; III, D. B.: Numerical Linear Algebra. SIAM, 1997.

[11] Golub, G.; Loan, C. V.: Matrix Computations. Johns Hopkins University Press, 1996.

[12] Dietl, G.; Zoltowski, M. D.; Joham, M.: Recursive Reduced-Rank Adaptive Equalization for Wireless Communications. – In: Proc. of SPIE **4395**, April 2001.

[13] Chang, P. S.; Willson, Jr., A. N.: Analysis of Conjugate Gradient Algorithms for Adaptive Filtering. – In: IEEE Transactions on Signal Processing **48** (February 2000), pp. 409–418.

[14] Goldstein, J. S.; Reed, I. S.: Reduced-Rank Adaptive Filtering. – In: IEEE Transactions on Signal Processing **45** (February 1997), pp. 492–496.

[15] European Telecommunications Standards Institute: Digital cellular telecommunications system (Phase 2+); Modulation. EN 300 959 V7.1.1, ETSI, http://www.etsi.org, June 2000.

[16] Laurent, P. A.: Exact and Approximate Construction of Digital Phase Modulations by Superposition of Amplitude Modulated Pulses (AMP). – In: IEEE Transactions on Communications **COM-34** (February 1986), pp. 150–160.

[17] Ariyavisitakul, S. L.; Winters, J. H.; Sollenberger, N. R.: Joint Equalization and Interference Suppression for High Data Rate Wireless Systems. – In: IEEE Journal on Selected Areas in Communications **18** (July 2000), pp. 1214–1220.