1

Introduction

The basic problem in *estimation theory* (e. g., [150, 210]), also known as *parameter estimation*, is to recover an unknown signal from a perturbed observation thereof. In the *Bayesian framework* where the unknown signal is assumed to be random, one of the most popular approach is to design the estimator such that the *Mean Square Error* (MSE) between its output and the unknown signal is minimized, resulting in the *Minimum MSE* (MMSE) *estimator*. Besides non-linear MMSE estimators like, e. g., the *Conditional Mean Estimator* (CME) (cf., e. g., [210, 188]), the *Wiener Filter* (WF) is a linear MMSE estimator, i. e., its output is a linear transformation of the observation. If both the unknown signal and the observation are multivariate non-stationary random sequences, the WF is time-varying and matrix-valued, therefore, in the following denoted as the *Matrix WF* (MWF) (e. g., [210]). The design of this time-variant MWF requires the solution of a system of linear equations with multiple right-hand sides, the so-called *Wiener–Hopf equation*, for each time index which is computationally cumbersome especially if the dimension of the observation is very large. There are two fundamental approaches for reducing this computational burden: the first one is to apply *methods of numerical linear algebra* (e. g., [237]) to solve the Wiener–Hopf equation in a computationally efficient way whereas the second approach exploits the *theory of statistical signal processing* (e. g., [210]) to decrease computational complexity by approximating the estimation problem itself.

First, we briefly discuss the methods of numerical linear algebra. Numerical mathematics distinguish between two different types of algorithms for solving systems of linear equations: *direct* and *iterative methods* (see Table 1.1). Contrary to direct methods which do not provide a solution until all steps of the algorithm are processed, iterative methods yield an approximate solution at each iteration step which improves from step to step. The most famous direct method is *Gaussian elimination*[1] [74] or the *Gauss–Jordan*

[1] Note that the principle of Gaussian elimination has been already mentioned in the Chinese book *Jiuzhang Suanshu* (Chinese, The Nine Chapters on the Math-

algorithm[2] [142, 35] both based on the *LU factorization* (e. g., [94]), i. e., the factorization of the system matrix in a lower and an upper triangular matrix. If the system matrix is Hermitian and positive definite as in the case of the Wiener–Hopf equation, the lower and upper triangular matrix are Hermitian versions of each other and the LU factorization can be reformulated as the computationally cheaper *Cholesky factorization* [11, 72, 236, 18]. Two of the first iterative methods are the *Gauss–Seidel* [76, 220] and the *Jacobi* [138, 139] *algorithm* which iteratively update the single entries in the solution assuming that the remaining entries are the values from previous iteration steps. Compared to these methods which need an infinite number of iterations to converge to the optimum solution, iterative *Krylov methods* [204] produce, at least in exact arithmetics, the solution in a finite number of steps. The most general Krylov method is the *Generalized Minimal RESidual* (GMRES) *algorithm* [205] which approximates the exact solution in the Krylov subspace formed by the system matrix and the right-hand side, via the minimization of the residual norm. In each iteration step, the dimension of the Krylov subspace is increased. Whereas the GMRES method can be applied to arbitrarily invertible system matrices, the first Krylov methods, i. e., the *Conjugate Gradient* (CG) [117] and the *Lanczos algorithm* [156, 157] both being mathematically identical, have been derived for Hermitian and positive definite system matrices. Precisely speaking, in [156], the Lanczos algorithm has been also applied to non-Hermitian matrices by using the method of *bi-orthogonalization* but the *Arnoldi algorithm* [4] is the computationally more efficient procedure for arbitrarily invertible system matrices. The extension of Krylov methods to systems of linear equations with multiple right-hand sides result in the *block Krylov methods* like, e. g., the *block Arnoldi algorithm* [208, 207], the *block Lanczos algorithm* [93], and the *Block Conjugate Gradient* (BCG) *algorithm* [175]. For the interested reader, a detailed overview over iterative methods for solving systems of linear equations can be found in [267].

Table 1.1. Different algorithms for solving systems of linear equations with system matrix $\boldsymbol{A}$

	Direct methods	Iterative Krylov methods
$\boldsymbol{A} \neq \boldsymbol{A}^{\mathrm{H}}$	Gaussian elimination (LU factorization)	Generalized Minimal RESidual (GMRES), Arnoldi, etc.
$\boldsymbol{A} = \boldsymbol{A}^{\mathrm{H}}$	Cholesky factorization ($\boldsymbol{A}$ positive definite)	Conjugate Gradient (CG, $\boldsymbol{A}$ positive definite), Lanczos

ematical Art) and in the commentary on this book given by Liu Hui in the year 263 [144].

[2] Gaussian elimination and the Gauss–Jordan algorithm differ only in the economy of data storage [133].

Second, in the theory of statistical signal processing, *reduced-rank estimation*, i. e., the estimation of the unknown signal based on a subspace approximation of the observation, decreases computational complexity by reducing the size and the structure of the Wiener–Hopf equation. In fact, if the computation of the subspace basis and the solution of the reduced-size Wiener–Hopf equation is computationally cheaper than solving the original system of linear equations, the resulting-reduced rank method is a candidate for a low-complexity estimator. Note that besides the complexity issue, reduced-rank methods enhance the robustness against estimation errors of second order statistics due to low sample support. First reduced-rank approaches were based on the approximation of the linear MMSE estimator in eigensubspaces spanned by eigenvectors of the system matrix of the Wiener–Hopf equation. One of them is the *Principal Component* (PC) *method* [132] which chooses the subspace spanned by the eigenvectors corresponding to the largest eigenvalues, i. e., the reduced-dimension eigensubspace with the largest signal energy. Nevertheless, in the case of a mixture of signals, the PC procedure does not distinguish between the signal of interest and the interference signal. Hence, the performance of the algorithm degrades drastically in interference dominated scenarios. This drawback of the PC algorithm lead to the invention of the *Cross-Spectral* (CS) *method* [85] which selects the eigenvectors such that the MSE over all eigensubspace based methods with the same rank is minimal. To do so, it considers additionally the cross-covariance between the observation and the unknown signal, thus, being more robust against strong interference. A third reduced-rank method is based on a *multistage decomposition* of the MMSE estimator whose origin lies in the *Generalized Sidelobe Canceler* (GSC) [3, 98]. The resulting reduced-rank *MultiStage WF* (MSWF) [89], or *MultiStage MWF* [88, 34] if the unknown signal is multivariate, is no longer based on eigensubspaces but on a subspace whose first basis vectors are the columns of the right-hand side matrix of the Wiener–Hopf equation.

Michael L. Honig showed in [130] that a special version of the MSWF is the approximation of the WF in a *Krylov subspace*. This fundamental observation connects the methods of numerical linear algebra with the theory of reduced-rank signal processing. Due to the excellent performance of these special MSWF versions, it follows that Krylov methods which are basically used for solving very large and sparse systems of linear equations [204], are good candidates for solving the Wiener–Hopf equation as well. It remains to discuss the relationship between the general MSWF and Krylov methods as well as the connection between the MSMWF and block Krylov methods which involves a multitude of new problems.

Whereas estimation theory deals with the recovery of a continuous-valued parameter, *detection theory* (e. g., [151, 210]) solves the problem of estimating a signal with a finite set of possible values also known as *hypothesis testing*. One example of hypothesis testing is the receiver of a digital communication system detecting the *binary digits* (bits) of a data source where only the perturbed observation thereof after the transmission over the channel is available.

Note that the mapping of the detector from the observation to the estimated bit which is one of the two possible binary values is clearly *non-linear*. However, a detector can be designed based on a linear estimator or *equalizer* followed by a non-linear operation like, e. g., a *hard decision device* or a *soft-input hard-output decoder*. Recently, *iterative* or *Turbo receivers* [61] have been introduced where the non-linear *soft-input soft-output decoder* which follows after linear estimation is used to compute *a priori information* about the transmitted bits. This *a priori* knowledge can be exploited by the linear estimator to improve the quality of its output. After further decoding, the procedure can be repeated for several iterations until the decoder delivers also the detected data bits. Since the use of the *a priori* information leads to a non-zero mean and non-stationary signal model, the linear estimator has to be designed as a time-varying filter which is not strictly linear but *affine*. If the communication system under consideration has multiple antennas, multiple users which are separated based on long signatures, and channels of high orders, the design of the linear estimator is computationally cumbersome. In such cases, low-complexity solutions based on either numerical linear algebra or statistical signal processing as described above play an important role.

1.1 Overview and Contributions

Chapter 2: Efficient Matrix Wiener Filter Implementations

Section 2.1 briefly reviews the *Matrix Wiener Filter* (MWF) (e. g., [210]) for estimating a non-zero mean and non-stationary signal vector sequence based on an observation vector sequence by minimizing the *Mean Square Error* (MSE) of the Euclidean norm between its output and the unknown signal vector sequence. Before presenting suboptimal reduced-complexity approaches in Section 2.3, Section 2.2 introduces the *Cholesky factorization* [11, 72, 236, 18] as a computationally cheap *direct method* to solve the *Wiener–Hopf equation*. Again, the Cholesky method exploits the fact that the system matrix of the Wiener–Hopf equation, i. e., the auto-covariance matrix of the observation vector, is Hermitian and positive definite.[3] Besides, we derive an optimal reduced-complexity method which can be applied to scenarios where the auto-covariance matrix of the observation vector has a specific time-dependent submatrix structure as in the Turbo system of Chapter 5. The resulting algorithm is a generalization of the reduced-complexity method in [240] from a single-user single-antenna to a multiple-user multiple-antenna system. The computational complexities of the Cholesky based and the reduced-complexity MWF are compared based on the exact number of required *FLoating point OPerations* (FLOPs). Finally, in Section 2.3, subspace approximations of the MWF,

[3] In this book, we exclude the special case where the system matrix of the Wiener–Hopf equation is rank-deficient and therefore no longer positive definite but semidefinite.

i. e., *reduced-rank MWFs*, are investigated as alternative low-complexity methods. Compared to the Cholesky factorization and its derivatives, reduced-rank MWFs are no longer optimal even if performed in exact arithmetics. Section 2.3 concludes with the brief review of the eigensubspace based reduced-rank MWFs, i. e., the *Principal Component* (PC) [132] and the MSE optimal *Cross-Spectral* (CS) [85] *method*.

Chapter 3: Block Krylov Methods

Iterative *Krylov methods* [204] are used in numerical linear algebra to solve systems of linear equations as well as eigenvalue problems. If the system of linear equations has several right-hand sides, the most important algorithms are the *block Arnoldi* [208, 207], the *block Lanczos* [93], and the *Block Conjugate Gradient* (BCG) [175] *algorithm*. This chapter derives these *block Krylov methods* in Sections 3.2, 3.3, and 3.4, and investigate their properties. The block Krylov methods iteratively improve the approximation of the exact solution from step to step. The *rate of convergence*, i. e., the number of iterations which are necessary to get a sufficient approximation of the exact solution, is derived for the BCG algorithm in Subsection 3.4.2. Compared to the same result as published in [175], the derivation presented in Subsection 3.4.2 is more elegant. Moreover, we analyze in Subsection 3.4.3 the *regularizing characteristic* of the BCG procedure, i. e., the fact that it produces an improved solution if stopped before convergence in cases where the system matrix and/or the right-hand sides are perturbed. For example, this is the case when these algorithms are applied to the Wiener–Hopf equation where second order statistics are not perfectly known due to estimation errors. We investigate the regularizing effect of the BCG algorithm by deriving its *filter factor representation* [110, 43]. So far, *filter factors* have been only presented (e. g., [110]) for the *Conjugate Gradient* (CG) *algorithm* [117], i. e., the special case of the BCG method for solving a system of linear equations with one right-hand side. Note that filter factors are a well-known tool in the theory of *ill-posed* and *ill-conditioned problems* to describe regularization [110]. Eventually, special versions of the block Lanczos as well as the BCG algorithm are derived where the solution matrix is computed columnwise instead of blockwise in order to achieve more flexibility in stopping the iterations. The resulting *block Lanczos–Ruhe algorithm* [200] and the *dimension-flexible BCG method* [101] (see Subsections 3.3.2 and 3.4.4) can be used to approximate the exact solution in Krylov subspaces with dimensions which are no longer restricted to integer multiples of the dimension of the unknown signal vector.

Chapter 4: Reduced-Rank Matrix Wiener Filters in Krylov Subspaces

The original *MultiStage MWF* (MSMWF) [88, 34, 45] is a reduced-rank approximation of the MWF based on *orthonormal correlator matrices*, so-called

blocking matrices, and *quadratic MWFs* of reduced size. In Section 4.1, we derive the MSMWF in its most general form where the orthonormal correlator matrices have been replaced by arbitrary bases of the corresponding correlated subspaces. Here, a *correlated subspace* is the subspace spanned by the columns of the correlator matrix. Besides, Section 4.1 includes a discussion of the MSMWF's fundamental properties. The general derivation of the MSMWF makes it possible to properly investigate its relationship to the Krylov subspace in Section 4.2. In fact, it reveals the restrictions on the MSMWF which are necessary to end up in one of the block Krylov methods (cf. Subsections 4.2.2 and 4.2.3). Whereas the connection between the *MultiStage Wiener Filter* (MSWF) [89] and Krylov subspaces has been already discussed in [128], the relationship between the MSMWF and the Krylov subspace goes back to [45]. After summarizing the resulting Krylov subspace based MSMWF implementations in Section 4.3, Section 4.4 concludes the chapter by revealing their computational efficiency to the reduced-complexity methods of Chapter 2 with respect to the required number of FLOPs.

Chapter 5: System Model for Iterative Multiuser Detection

This chapter defines the system model of a *Multiple-Access* (MA) *communication system* with iterative multiuser detection (cf., e. g., [254, 127]) which is used in Chapter 6 in order to analyze the performance of the algorithms proposed in the previous chapters. In an iterative detection scenario, the transmitter of each user consists of an encoder, interleaver, and mapper (cf. Section 5.1). Here, the MA scheme is realized using *Direct Sequence Code Division Multiple-Access* (DS-CDMA), i. e., the symbol streams of the different users are spread using *Orthogonal Variable Spreading Factor* (OVSF) *codes* as reviewed in Subsection 5.1.4. After the transmission over a frequency-selective channel as described in Section 5.2, the spread signals of the different users are received by multiple antennas. Then, the observation signal is processed by an *iterative* or *Turbo receiver* [61] exchanging soft information between linear estimator and decoder. Section 5.3 explains how the linear estimators of the previous chapters can be used as linear equalizers in the Turbo receiver, i. e., how they exploit the *a priori information* obtained from the *soft-input soft-output decoder* and how they transform the estimate at the output of the equalizer to an *extrinsic information* which is needed by the soft-input soft-output decoder to compute the *a priori* information required by the linear estimator at the next Turbo iteration (cf. [254, 240, 49]). Besides, the decoder provides the detected data bits after the last Turbo step. The *a priori* information is used to adapt the signal model for the linear estimator design. To do so, the transmitted signal must be assumed to be non-zero mean and non-stationary and the resulting estimator is time-varying. Note that the time-varying statistics can be also approximated via a time-average approach leading to further low-complexity solutions if applied to the already mentioned reduced-complexity algorithms except from the one which exploits the time-dependent

structure of the auto-covariance matrix. Finally, Subsection 5.3.4 reveals the relationship of the proposed equalizers to *iterative soft interference cancellation* [254, 198] or *parallel decision feedback detection* [62, 127].

Chapter 6: System Performance

The final chapter analyzes the performance of the coded multiuser DS-CDMA system as introduced in Chapter 5. Besides the comparison of the different equalizers in Section 6.3 based on the *Bit Error Rate* (BER) which is obtained via *Monte Carlo simulations*, we investigate their performance in Section 6.2 using *EXtrinsic Information Transfer* (EXIT) *charts* [19] which are briefly reviewed in Section 6.1. Contrary to the simulated calculation of the equalizer's EXIT characteristic, Subsection 6.1.2 proposes a method for their semianalytical computation which is a generalization of the contributions in [195]. Moreover, Section 6.2 answers the question when rank-reduction is preferable to order-reduction of the equalizer filter in order to obtain computationally cheap implementations by introducing *complexity–performance charts*. Eventually, the regularizing property of the MSMWF's BCG implementation as discussed in Subsection 3.4.3 are investigated when the channel is not perfectly known at the receiver but estimated based on the *Least Squares* (LS) *method* (cf., e. g., [188, 53]).

1.2 Notation

Contrary to mathematical operators which are written based on the standard text font, mathematical symbols are denoted by italic letters where vectors are additionally lower case bold and matrices capital bold. Random variables are written using *sans serif* font and their realizations with the same font but with serifs. Finally, we use calligraphic letters for subspaces and the blackboard notation for sets.

Table 1.2 summarizes frequently used operators and Table 1.3 frequently used symbols. Besides, the matrix

$$\boldsymbol{S}_{(\nu,m,n)} = \left[\mathbf{0}_{m\times\nu}\ \boldsymbol{I}_m\ \mathbf{0}_{m\times(n-\nu)}\right] \in \{0,1\}^{m\times(m+n)}, \tag{1.1}$$

is used to select m rows of a matrix beginning from the $(\nu+1)$th row by applying it to the matrix from the left-hand side. Analogous, if its transpose is applied to a matrix from the right-hand side, m columns are selected beginning from the $(\nu+1)$th column. Besides, the matrix $\boldsymbol{S}_{(\nu,m,n)}$ can be used to describe convolutional matrices.

An alternative access to submatrices is given by the notation $[\boldsymbol{A}]_{m:n,i:j}$ which denotes the $(n-m+1)\times(j-i+1)$ submatrix of $\boldsymbol{A}$ including the elements of $\boldsymbol{A}$ from the $(m+1)$th to the $(n+1)$th row and from the $(i+1)$th to the $(j+1)$th column. Further, if we want access to a single row or column

Table 1.2. Frequently used operators

Operator	Description
$\oplus$, $\diamond$	two general operations defined in a field
$\boxplus$, $\boxdot$	modulo 2 addition and modulo 2 multiplication
$+$, $\cdot$	common addition and multiplication
$\oplus$	direct sum of subspaces
$\oslash$	elementwise division
$\otimes$	Kronecker product
$*$	convolution
$\mathrm{E}\{\cdot\}$	expectation
$(\cdot)^{\circ}$	zero-mean part of a random sequence
$\mathrm{Re}\{\cdot\}$	real part
$\mathrm{Im}\{\cdot\}$	imaginary part
$\lvert\cdot\rvert$	absolute value
$(\cdot)^{*}$	complex conjugate
$(\cdot)^{\mathrm{T}}$	transpose
$(\cdot)^{\mathrm{H}}$	Hermitian, i. e., conjugate transpose
$(\cdot)^{-1}$	inverse
$(\cdot)^{\dagger}$	Moore–Penrose pseudoinverse
$\sqrt{\cdot}$	square root or general square root matrix
$\sqrt[\oplus]{\cdot}$	positive semidefinite square root matrix
$\mathrm{tr}\{\cdot\}$	trace of a matrix
$\mathrm{rank}\{\cdot\}$	rank of a matrix
$\mathrm{range}\{\cdot\}$	range space of a matrix
$\mathrm{null}\{\cdot\}$	null space of a matrix
$\mathrm{vec}\{\cdot\}$	vectorization of a matrix by stacking its columns
$\mathrm{diag}\{\cdot\}$	diagonal matrix with the scalar arguments or the elements of the vector argument on the diagonal
$\mathrm{bdiag}\{\cdot\}$	block diagonal matrix with the matrix arguments on the diagonal
$\lceil\cdot\rceil$	closest integer being larger than or equal to the argument
$\lfloor\cdot\rfloor$	closest integer being smaller than or equal to the argument
$\mathrm{O}(\cdot)$	Landau symbol
$\lVert\cdot\rVert$	general norm of a vector or matrix
$\lVert\cdot\rVert_2$	Euclidean norm of a vector or induced 2-norm of a matrix
$\lVert\cdot\rVert_{\mathrm{F}}$	Hilbert–Schmidt or Frobenius norm of a matrix
$\lVert\cdot\rVert_{\boldsymbol{A}}$	$\boldsymbol{A}$-norm of a matrix

of a matrix, respectively, we use the abbreviation 'n' instead of writing '$n : n$'. Note that ':' is the abbreviation for '$0 : z$' where z is the maximum value of the index under consideration. Eventually, $[\boldsymbol{a}]_{m:n}$ denotes the $(n - m + 1)$-dimensional subvector of $\boldsymbol{a}$ including the $(m + 1)$th to the $(n + 1)$th element.

The *probability* $\mathrm{P}(\mathsf{d} = d)$ denotes the likelihood that a realization of the discrete random variable $\mathsf{d} \in \mathbb{C}$ is equal to $d \in \mathbb{C}$ and $\mathrm{p}_{\mathsf{x}}(x)$ is the *probability density function* of the continuous random variable $\mathsf{x} \in \mathbb{C}$. The soft

Table 1.3. Frequently used symbols

Symbol	Description
$:=$	definition
$\perp \mathcal{S}$	orthogonal subspace complement of $\mathcal{S}$
$\boldsymbol{P}_{\mathcal{S}}$	projector matrix projecting on subspace $\mathcal{S}$
$\hat{(\cdot)}$	estimated value
$\tilde{(\cdot)}$	detected value
$\delta_{i,j}$	Kronecker delta
$\delta[n]$	unit impulse
$\boldsymbol{I}_n$	$n \times n$ identity matrix
$\boldsymbol{e}_i$	ith column of the $n \times n$ identity matrix[4]
$\mathbf{1}_n$	n-dimensional all-ones vector
$\mathbf{0}_{n \times m}$	$n \times m$ zero matrix
$\mathbf{0}_n$	n-dimensional zero vector
$\mathbb{C}$	set of complex numbers
$\mathbb{R}$	set of real numbers
$\mathbb{R}_{0,+}$	set of non-negative real numbers
$\mathbb{R}_{+}$	set of positive real numbers
$\mathbb{Z}$	set of integers
$\mathbb{N}_0$	set of non-negative integers
$\mathbb{N}$	set of positive integers

information of a binary random variable $\mathsf{b} \in \{0, 1\}$ is represented by the *Log-Likelihood Ratio* (LLR) [107]

$$l = \ln \frac{\mathrm{P}\left(\mathsf{b} = 0\right)}{\mathrm{P}\left(\mathsf{b} = 1\right)} \in \mathbb{R}. \tag{1.2}$$

Table 1.4 depicts the first and second order statistical moments of the scalar random sequences $\mathsf{x}[n] \in \mathbb{C}$ and $\mathsf{y}[n] \in \mathbb{C}$, and the vector random sequences $\boldsymbol{\mathsf{u}}[n] \in \mathbb{C}^{\ell}$ and $\boldsymbol{\mathsf{v}}[n] \in \mathbb{C}^{m}$ at time index n. Note that the cross-correlations and cross-covariances are functions of the index n and the latency time ν. Since we choose a fixed ν throughout this book, an additional index is omitted in the given notation. Besides, here, the auto-correlations and auto-covariances denote the values of the corresponding auto-correlation and auto-covariance functions evaluated at zero, i. e., for a zero-shift. For example, the auto-correlation $r_{\mathsf{x}}[n]$ of $\mathsf{x}[n]$ is equal to the auto-correlation function $r_{\mathsf{x}}[n, \nu] = \mathrm{E}\{\mathsf{x}[n]\mathsf{x}^*[n - \nu]\}$ at $\nu = 0$, i. e., $r_{\mathsf{x}}[n] = r_{\mathsf{x}}[n, 0]$. The same holds for $c_{\mathsf{x}}[n]$, $\boldsymbol{R}_{\boldsymbol{\mathsf{u}}}[n]$, and $\boldsymbol{C}_{\boldsymbol{\mathsf{u}}}[n]$.

[4] The dimension of the unit vector $\boldsymbol{e}_i$ is defined implicitly via the context.

Table 1.4. Statistical moments at time index n

Moment	Description
$m_{\mathsf{x}}[n] = \mathrm{E}\{\mathsf{x}[n]\}$	mean of $\mathsf{x}[n]$
$\boldsymbol{m}_{\boldsymbol{u}}[n] = \mathrm{E}\{\boldsymbol{u}[n]\}$	mean of $\boldsymbol{u}[n]$
$r_{\mathsf{x}}[n] = \mathrm{E}\{\lvert\mathsf{x}[n]\rvert^2\}$	auto-correlation of $\mathsf{x}[n]$
$c_{\mathsf{x}}[n] = r_{\mathsf{x}}[n] - \lvert m_{\mathsf{x}}[n]\rvert^2$	auto-covariance or variance of $\mathsf{x}[n]$
$\sigma_{\mathsf{x}}[n] = \sqrt{c_{\mathsf{x}}[n]}$	standard deviation of $\mathsf{x}[n]$
$\boldsymbol{R}_{\boldsymbol{u}}[n] = \mathrm{E}\{\boldsymbol{u}[n]\boldsymbol{u}^{\mathrm{H}}[n]\}$	auto-correlation matrix of $\boldsymbol{u}[n]$
$\boldsymbol{C}_{\boldsymbol{u}}[n] = \boldsymbol{R}_{\boldsymbol{u}}[n] - \boldsymbol{m}_{\boldsymbol{u}}[n]\boldsymbol{m}_{\boldsymbol{u}}^{\mathrm{H}}[n]$	auto-covariance matrix of $\boldsymbol{u}[n]$
$r_{\mathsf{x},\mathsf{y}}[n] = \mathrm{E}\{\mathsf{x}[n]\mathsf{y}^*[n-\nu]\}$	cross-correlation between $\mathsf{x}[n]$ and $\mathsf{y}[n]$
$c_{\mathsf{x},\mathsf{y}}[n] = r_{\mathsf{x},\mathsf{y}}[n] - m_{\mathsf{x}}[n]m_{\mathsf{y}}^*[n-\nu]$	cross-covariance between $\mathsf{x}[n]$ and $\mathsf{y}[n]$
$\boldsymbol{r}_{\boldsymbol{u},\mathsf{x}}[n] = \mathrm{E}\{\boldsymbol{u}[n]\mathsf{x}^*[n-\nu]\}$	cross-correlation vector between $\boldsymbol{u}[n]$ and $\mathsf{x}[n]$
$\boldsymbol{c}_{\boldsymbol{u},\mathsf{x}}[n] = \boldsymbol{r}_{\boldsymbol{u},\mathsf{x}}[n] - \boldsymbol{m}_{\boldsymbol{u}}[n]m_{\mathsf{x}}^*[n-\nu]$	cross-covariance vector between $\boldsymbol{u}[n]$ and $\mathsf{x}[n]$
$\boldsymbol{R}_{\boldsymbol{u},\boldsymbol{v}}[n] = \mathrm{E}\{\boldsymbol{u}[n]\boldsymbol{v}^{\mathrm{H}}[n-\nu]\}$	cross-correlation matrix between $\boldsymbol{u}[n]$ and $\boldsymbol{v}[n]$
$\boldsymbol{C}_{\boldsymbol{u},\boldsymbol{v}}[n] = \boldsymbol{R}_{\boldsymbol{u},\boldsymbol{v}}[n] - \boldsymbol{m}_{\boldsymbol{u}}[n]\boldsymbol{m}_{\boldsymbol{v}}^{\mathrm{H}}[n-\nu]$	cross-covariance matrix between $\boldsymbol{u}[n]$ and $\boldsymbol{v}[n]$